Securing Software Reconfigurable Communications Devices

Approved Document WINNF-08-P-0013

Version V1.0.0

20 July 2010

TERMS, CONDITIONS & NOTICES

Table of Contents

List of Figures

List of Tables

List of Appendices

Preface

This document is intended to provide guidance, key considerations and recommendations for SDR developers and manufacturers regarding the design and manufacturing processes essential to producing appropriate security solutions for software reconfigurable radio platforms. It spans a comprehensive range of security topics such as considerations for stakeholders and other roles and their security needs as well as potential vulnerabilities, threats, attacks/exploits, and associated risk analyses. It delves into the role of the radio platform security policy in enforcing higher level security policies and essential considerations for developing a viable security architecture that ensures all necessary security services and mechanisms are present and implemented in a manner which is consistent with the defined policy. Considerations are also given to the security, application and use of the emerging technology of machine interpretable downloadable policies. Guidance is provided regarding the application and implementation of services and mechanisms for traditional as well as emerging radio security services, security mechanisms and security critical system processes. The document includes within the text many representative requirements as well as references to other standards and references which manufacturers can consider for use in their products. Part two of this document, which is planned for publication later this year, will have a consolidated listing of requirements by topic.

Contributors

William T. Scott PhD
Principal Systems Engineer
General Dynamics C4 Systems
8220 E Roosevelt St Scottsdale, AZ 85257 USA
William.T.Scott@gdc4s.com

John J. Fitton,
Harris Fellow, Senior Scientist
Harris Corporation
RF Communications Division
1680 University Avenue
Rochester, NY 14610 USA
John.fitton@harris.com

Tony Martin SCA Technica
Software and Security Architect
P.O. Box 3148
Nashua NH 03061-3148 USA
Tony.Martin@SCATechnica.com

Bernard Eydt
Booz Allen Hamilton
eydt_bernie@bah.com

This document is a substantial revision to an earlier draft version of this document which was prepared by previous members of the Forum's Security Working Group. Their contributions were the foundation of this document and are gratefully acknowledged. These contributors are:

Eimear Gallery
Royal Holloway, University of London
e.m.gallery@rhul.ac.uk

Ed Greenwood,
Independent consultant
edgreenwood@cox.net

Steve Muir,
Vanu, Inc.
steve@vanu.com

Tony Walters,
Independent consultant
twalters@rogers.com

List of Abbreviations

ACL	Access Control List
AES-CBC	Advanced Encryption Standard – Cipher Block Chaining
AES- GCM	Advanced Encryption Standard – Galois Counter Mode
ASIC	Application Specific Integrated Circuit
BIT	Built in Test
CA	Certification Authority
CCEVS	Common Criteria Evaluation and Validation Scheme
CCRA	Common Criteria Recognition Arrangement
CDMA	Code Division Multiple Access
CKL	Compromised Key List
COMPUSEC	Computer Security
COMSEC	Communications Security
CPU	Central Processing Unit
CR	Cognitive Radio
CRL	Certificate Revocation List
DAA	Download Authorization Authority
DAC	Discretionary Access Control
DARPA	Defense Advanced Research Projects Agency
DES	Data Encryption Standard

DMA	Direct Memory Access
DoD	Department of Defense (USA)
DoS	Denial of Service
DRBG	deterministic random bit generator
DSII	Download, Storage, Installation, and Instantiation
DSP	Digital signal processors
DSA	Dynamic Spectrum Access
EAL	Evaluated Assurance Level
EAP	Extensible Authentication Protocol (RFC 3748)
ERPSP	Explicit Radio Platform Security Policy
(E)PROM	(Electrically) Programmable Read Only Memory
FCC	Federal Communication Commission
FIPS	Federal Information Processing Standard
FPGA	Field Programmable Gate Arrays
GPP	General Purpose Processors
GPRS	General Packet Radio Service
HMAC	Keyed-hash Message Authentication Codes
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force

IMEI	International Mobile Equipment Identifier
INFOSEC	Information Security
IP	Internet Protocol
ISO/IEC	International Standards Organization/Internet Electro-Technical Commission
IT	Information Technology
ITU	International Telecommunications Union
JTRS	Joint Tactical Radio System
KEK	Key Encryption Key
KGD	key generation function
KMI	Key Management Infrastructure
KMF	Key Management Function
KMS	Key Management System
KMS	Key and Credential Management Service
LPP	Least Privilege Principle
MAC	Mandatory Access Control
MILS	Multiple Independent Levels of Security
MMU	Memory Management Unit

NIST	National Institute of Standards and Technology
NOp	Network Operator
NRBG	Nondeterministic Random Bit Generator
NVM	Non-Volatile Memory
NVS	Non-Volatile Storage
OS	Operating System
OSP	Organizational Security Policy
PAC	Physical Access Control
PCS	Personal Communication System
PIN	Personal Identification Number
PKE	Public Key Encryption
PKI	Public Key Infrastructure
RAM	Random Access Memory
RBAC	Role Based Access Control
RFC	Request for Comment
RFID	Radio Frequency Identification
RM	Reference Monitor
RNG	Random number generation
ROM	Read-only Memory
RPA	Radio Platform (software) Applications

RPOE	Radio Platform Operating (software) Environment
RPSA	Radio Platform Security Architecture
RPSDR	Radio Platform Security Design Requirements
RPSOR	Radio Platform Security Operational Requirements
RPSP	Radio Platform Security Policy
RPSR	Radio Platform Security Requirements
RSS	Radio Security Services
RTOS	Real Time Operating System
SAC	Spectrum access control
SAP	Spectrum Access Policy
SCA	Software Communications Architecture
SCC	Standards Coordinating Committee
SCP	Software Content Provider
SD	Software Distributor
SDMSP	SDR Device Meta Security Policy
SDR	Software Defined Radio
SDRD	Software Defined/Software Reconfigurable Radio Device
SHA	Secure Hash Algorithms
SNMP	Simple Network Management Protocol
SoD	Separation of Duties
SPA	Service Provider (software) Applications
SPPA	Security Policy Primary Authority

SPEE	Security Policy Enforcement Engine
SSL	Secure Socket Layer
SSP	System Security Policy
TCB	Trusted Computing Base
TCSEC	Trusted Computer System Evaluation Criteria
TLS	Transport Layer Security
TPM	Trusted Platform Module
TRANSEC	Transmission Security
TSF	Trusted Security Function
UA	User (software) Applications
UMTS	Universal Mobile Telecommunications System
USB	Universal Serial Bus
WAP	Wireless Application Protocol
WEP	Wired Equivalent Privacy
WINNF	Wireless Innovation Forum (SDR Forum V2.0)
XG	neXt Generation (a DARPA Project)

Securing Software Reconfigurable Communications Devices

1 Introduction

1.1 Scope

Viruses, trojans and other forms of malicious software based attacks are commonplace today in computational environments, including those of software reconfigurable radios. In this hostile environment, security functions should be at the top of the list defining essential capabilities and features for any Software Definable Radio (SDR), but especially for software reconfigurable and/or cognitive radios.

To achieve that end, this document is intended to assist the Software Defined and Software Reconfigurable radio industry to understand more fully the security concerns involved with these radio technologies in order to build a more secure device. For reference simplification purposes we shall refer to the full range of software reconfigurable radios as simply SDRs. This document presents a set of threats common to an SDR Device (SDRD) and a set of functional requirements for security services and mechanisms which can be used to serve as counter measures to mitigate these threats.

The focus of the work is to present an approach for the development of an SDRD security architecture and processes that address the threats and security issues an SDRD is likely to face in its development, manufacturing and operational environments. The approach:

- Identifies the important assets of a prototypical SDRD;
- Identifies threats to the base platform as well as those associated with the development and manufacturing processes;
- Describes the process of risk assessment and the development of security policy which addresses the risks,
- Discusses security services that help the platform provide functions in support of the security policy and,
- Identifies and describes Security Critical Processes necessary to conform to the Organizational, System and Platform security policies;
- Identifies security design and architectural features that address the overall platform security policy needs.

Vulnerabilities which are exploited outside the platform core resources, such as those introduced by waveforms (i.e. air interface), are generally excluded from this discussion. They are beyond the scope of this document and are more properly addressed by those standards bodies responsible for defining and maintaining the governing standards. However, this document does address security design elements and mitigation strategies which are intended to limit the scope of any exploit which may occur via these waveforms.

SDRDs may be integrated into other platforms such as laptop computers, automobiles or remote sensors. This document is focused only on the SDRD component and not on the host laptop or other embedment environment. Security for laptops and other operating environments are being addressed by other entities such as the Trusted Computing Group initiative which has defined a Trusted Platform Module. (http://www.trustedcomputinggroup.org). This organization has also undertaken an initiative to apply their concepts to mobile platforms and have developed a reference architecture applicable to a mobile computing platform. However the current activities appear to focus solely on mobile telephone applications, whereas this document addresses a much broader range of radio communication devices.

Given the complexity of the environment, the range of products and the evolving technology, only a subset of potential threats to SDRD are discussed. The set of threats addressed should not be considered definitive nor should the requirements be considered exhaustive or the only set that address the threats listed in this document. It is meant as a guide and starting point, and a basis for commonality. It is designed to assist in the security analysis for specific, deployable devices, but should not be considered complete or definitive. This is an area where outside sources of expertise may be available for consultation and this is a recommended approach to consider.

While the topic of this specification is focused primarily on commercial/ and civil government (e.g., public safety) applications, there is much that can be derived from an understanding of requirements applied to military radios. To that end this document recommends reviewing the Joint Tactical Radio System (JTRS) Software Communications Architecture (SCA) Version 2.2 Security Supplement as an example of security requirements with a military focus. This document is still available on the JTRS website as a historical reference even though it was later withdrawn as a component of the SCA when SCA 2.2.2 was published. Another set of security criteria, referenced in earlier documents in this series is the Common Criteria. (http://www.niap-ccevs.org/cc-scheme/)

These criteria encompass a broad set of requirements applicable to many types of software controlled devices and include formal certification of conformance. (See for example Appendix D to Document 4 referenced in section 1.2 below.)

While a relevant and essential topic of discussion in this document, the Forum does not recommend or specify a minimum level of assurance and robustness for the security measures or for the security design associated with an SDRD. It is the Forum's view that these elements are beyond the scope of this organization. These are design aspects which the manufacturer of the device, in consideration for its customers' interests, must determine, with due consideration to the nature of the threats and an assessment of their associated exploitation risk while considering the nature and consequences of a successful exploitation of a vulnerability.

1.2 Background

This document is one of a series of documents which have been published by the Wireless Innovation Forum addressing the topic of Security associated with the design of Software Defined and/or reconfigurable communication devices. The term SDRD is used throughout this document since it is a more general term than Software Defined Radio whose definitions varies among the different organizations concerned with this technology.

Listed below are those documents which have been published prior to this document with explanatory comments about each.

1) SDRF-02-A-0003-V0.00, *Report on Issues and Activity in the Area of Security for Software Defined Radio, 1 September 2002 (121 pages)*

Comment: This report was prepared for consideration by the FCC and other regulatory bodies and addressed the broad issues relating to security for wireless devices employing SDR technology.

2) SDRF-02-S-007-V1.0.0. *Requirements for Radio Software Download for RF Reconfiguration 13 November 2002 (41 Pages)*

Comment: Also known as DL-REQ, this document presents specific requirements for protocols for downloading software to a software-defined radio (SDR) device for its reconfiguration. The SDR device could be a small handheld communication device such as a cell phone, a slightly larger wearable device such as a manpack, or an immobile networked device such as a wireless base station.

3) SDRF-02-P-0006-V1.0.0 *SDR System Security, November 2002*

Comment: Also known as S&A –SEC, this document provides a brief introduction to some of the security aspects related to the introduction of Software Defined Radio Technology to wireless personal communication systems (PCS). To do that, we establish a taxonomy of security elements, and examine the more important ones. The element structure enables development of a *Threat Vector*, and facilitates examination of new security considerations.

4) SDRF-04-P- 0010-V1.0.0 *Security Considerations for Operational Software for Software Defined Radio Devices in a Commercial Wireless Domain, 27 October 2004*

Comment: Also known as DL-SIN this document provides detailed security requirements for operational software provisioning and configuration including download, storage, installation, and instantiation (DSII). The document also provides an introduction to some promising technologies that have been proposed for meeting those requirements.

5) SDRF-06-S-0002-V1.0.0 *High-Level SDR Security Requirements January 2006*

Comment: This document was an interim document and is superseded by the current document.

2 Software Reconfigurable Communication Device Scenario Example

This section presents an example operational scenario for a SDRD. It provides a brief introduction to the types of operations that reconfigurable radios are expected to perform. More detailed descriptions of these operations will be discussed later in the document. The purpose of the operational scenario is informative. It is not intended to be a formal use case on which the security functional requirements are based.

SDRDs can take many forms. They can be terminals or infrastructure devices. They can be in fixed locations or mobile, or they may be embedded in some manner in other devices (e.g., a laptop computer). They can serve commercial, military, or public safety applications. No single scenario can account for this diversity. Nevertheless, to best capture the range of concepts relevant to SDRD security, the SDRD used in the following operational scenario is a commercial wireless handset, hereafter termed a *mobile phone*. The general principle behind this choice is that mobile phones face a particularly challenging reconfiguration environment relative to many other types of SDRDs and therefore generally involve an equivalent or broader set of supporting operations. The reasons why mobile phones are considered to face a challenging reconfiguration environment include:

- Mobile phones are, as their name suggests, highly mobile, often connecting to different networks and moving across regulatory jurisdictional boundaries.
- Even if these capabilities are not offered today, many mobile phone users would like to be able to connect to multiple service providers, using a variety of communications protocols to support applications.
- Mobile phones are subject to significant constraints, such as size, weight and power.
- The owners of mobile phones typically have different interests than the applications developers and network operators that enable the phone's capabilities (e.g., they do not have incentives to protect vendor revenue streams or intellectual property).
- While mobile phones may be a lower priority in the world of threat mongers, the user's data stored on the phone is likely a desirable target, and the network infrastructure is likely a high priority target.

The exact nature of the reconfiguration and related operations will vary depending on the extent to which these characteristics are present, but a central position of this document is that many of the same operational and security framework elements can be used for all SDRDs. What varies by application is the subset of operations performed and the level of assurance at which these operations are performed. All SDRDs applications are expected to involve some aspect of the general types of operations described in the operational scenario for mobile phones.

2.1 Sample Mobile Phone Scenario

A user owns a mobile phone that has software reconfigurable radio functions. Just before the user's plane takes off on an overseas flight, the user is requested to turn off the handset. Once the plane lands, the handset is turned back on. During the phone's start up process, the phone acquires information about its radio environment. The phone is programmed to obtain and enforce the rules of the regulatory jurisdiction in which it is currently located. These rules, which express the local regulatory policies, may already exist within the phone, or they might be downloaded over the air as part of the overall start up and location exploratory processes.

In addition, as part of the start up process, the phone determines that its primary network operator does not offer service in the area. It detects that several other communication service providers are providing voice and data service using a variety of frequencies and communications protocols. Finally, the phone ascertains that it is now in a new regulatory jurisdiction with respect to spectrum access regulations.

The phone is programmed to obtain and enforce the rules of the regulatory jurisdiction in which it is currently located. In this case, the new jurisdiction reserves a subset of the frequencies for different services that were available for commercial services in the home country. In accordance with the local regulatory policy, the phone configures itself not to transmit on the newly prohibited frequencies.

The display on the phone shows the user communication service options. The user selects a service offered by a local wireless telecommunications company. When the user attempts to connect to that company's network, the network first queries the phone to learn its capabilities and assess its trustworthiness. As a result of the exchange, the network determines that the phone requires an update to the radio software to interoperate fully with the network. In addition, the network operator has optional value added services that require an additional software update. The mandatory updated software is downloaded to the phone and installed while the user is informed of the value added service and its cost. The user is then prompted to accept the mandatory portion of the download and is then queried regarding acceptance of the fee associated with use of the value added services. The user accepts the offer of the service and provides credit card information to the network operator so that appropriate charges can be made when the value added services are used. The value added service software is then downloaded and installed into the mobile phone.

2.2 Integrating Security into the Scenario

The phone has technical security controls to provide assurance that the operations perform as intended. This section provides a brief introduction to the types of controls that are necessary.

When Jane turned on her phone, there is an assumption that the startup (boot) process will correctly load and execute code that assesses the phone's RF environment as well as obtain and enforce local regulatory policy and the device's own security policy. Note that the start up process may be subject to corruption. To prevent any unauthorized modification of the start up process, this phone has a mechanism that ensures that the boot code and the operating code have not been modified since the SDRD was last used.

To provide assurance that this mechanism has not been corrupted, software on the device is bound to hardware using cryptography and unique and immutable key material which has been stored in tamper resistant storage. To avoid corruption of the binding requires a trusted manufacturing process that mitigates the risk of cloning and tampering, and the ability to detect tampering when it does occur.

When the phone obtains or selects (if pre-stored within the device) the local regulatory policy, it needs a mechanism to determine if the policy is valid. To do this the phone authenticates the policy and ensures that it was obtained from an authorized source. The policy is written using a standard syntax for proper interpretation. [In the IEEE Standards Coordinating Committee (SCC) 41 (formerly P1900) work is underway to develop standards applicable to downloadable policies.]

In the initial exchange between the user's phone and the new network, the network needs to trust the information the phone provides about its configuration. This helps the network identify needed software and to determine if the phone represents a threat. Therefore, the phone provides irrefutable evidence of its configuration and the network is able to authenticate the device and its credentials.

When the user's phone downloads the new software needed, it needs assurance that the code is not malicious and that it will do what it purports. To accomplish this, relevant stakeholders (such as the developer of the software and the communications service provider, and perhaps others) make claims about the code they distribute that counter repudiation claims. For the phone to authenticate these claims, the phone and the stakeholders have a prior trust relationship with one or more third-parties that provides assurance to the true identities of the entities involved in the transaction.

In the scenario, the network operator supplies proprietary software and receives fees for usage. The operator has an interest in preventing the proprietary software from being used elsewhere or from being copied for analysis purposes. (The analysis might identify vulnerabilities and methods of exploiting them.) For these reasons the operator might desire confidentiality of its code both in transit and at rest. Confidentiality in transit could be supplied by the use of a download protocol that included a confidentiality service or the code package could be encrypted in the public key of the phone prior to being transmitted. Confidentiality at rest could be based on security controls that prevented users of the phone from copying confidentiality protected software stored on the device.

To prevent rogue software from adversely impacting other applications on the phone, the phone's design should provide a means to ensure that active applications are properly isolated from each other as well as from all applications which enforce security functionality. This should not preclude security functions from carrying out their responsibilities, whatever they may be. This isolation should prevent corruption/alteration of code and data associated with these other processes. One method to accomplish this would be to include an isolation layer and associated access control mechanisms and rules/policies. Another might be to have a separation kernel, and rigorous memory management capability regulated by an internal trusted security mechanism. These and other aspects will be explored further in later sections of this document.

3 A Security Analysis for Reconfigurable Communication Devices

To develop a set of requirements that secure the platform of a reconfigurable radio, it is necessary to understand:

1) What is being protected (assets),
2) What are the potential weaknesses (vulnerabilities)
3) What are the threats
4) How can the threats be exploited and
5) What is the likelihood a particular threat will be exploited against the device of interest?

From this chain, an analysis can be performed to develop a security architecture with its associated set of security design features, processes and mechanisms than can be applied to protect the assets from the various threats at an acceptable level of risk.

3.1 Asset Classes

An identification of assets associated with an SDRD is an essential component of the security analysis because it defines that which needs to be protected and assists in the identification of the stakeholders involved with security. The following sections identify and discuss the assets which are at risk in or associated with an SDRD.

3.1.1 Communications Service/ Network

A communication service delivers content to an SDRD. The delivery is by means of either point-to-point or point-to-multipoint connections. Usually, but not always, a service provider supplies OSI levels 1, 2 and 3 (Physical, Data Link and Network) and third parties supply content. Both usually charge for the services creating value that must be protected. This can also allow a device to access a user's local or personal area network, interfacing on unknown or uncontrolled networks.

3.1.2 Electromagnetic Spectrum

The electromagnetic spectrum is a managed resource. In order to allow the operation of multiple devices at the same time the transmissions of each radio emitter are defined and controlled. Spectrum is assigned by regional/national governing authorities through licenses to service providers, often at considerable cost to the providers. Access to and use of spectrum is an asset of value to both the governing authorities and license holders. An SDRD is capable of transmitting a range of frequencies. The licensor’s policy and the local regulatory agency constrain the range of operation.

3.1.3 Health and Safety

The health and safety of users and those nearby need to be protected from action that could result in injury or loss of life. For example, regulatory authorities set permissible power limits that SDRD must conform to. Effective enforcement mechanisms have value to the SDRD owner, network operator and others.

3.1.4 Radio Hardware/ Physical Device

The SDRD itself is an asset to the user/owner of the device. Loss or theft of the device could allow its misuse such as theft of service, mischief or when public safety/Government owned assets are involved, even acts terrorism could result. Loss or theft of the device could also place data resident in the device at risk.

3.1.5 Radio Software/Firmware

Radio platform software/firmware consists of any and all software that utilizes the computational resources available on an SDRD. It is the primary target of threats directed at an SDRD. Software developers, manufacturers and operators should bear in mind, because we are dealing with SDRDs, the primary threats can only be realized through the loading, installation and instantiation (execution) of software. Clearly the main threat mitigation for SDR security is to prevent this from occurring for unauthorized or unproven software. To that end, it is useful to define the different classes of software/firmware that can exist on an SDRD so that we may better define the scope of applicable security functions required to mitigate the threats to each.

3.1.5.1 Radio Platform Applications (RPA)

An RPA is the software that controls the behavior of the radio as a radio. It includes software that defines the air, interface and the modulation and communication protocols. In the context of the SCA, this includes software defined as being part of a waveform as well as software installed on the platform that is used to manage or control the radio in a radio network environment.

3.1.5.2 Service Provider Applications (SPA)

An SPA is software used to support service provider services, such as network access, for the user of the radio. This might include special messaging services, video services, etc. The SPA interacts with the two preceding application classes, obtaining computational environment and communications and security services (RPOE) and communication services (RPA) needed to support the SPA provided service.

3.1.5.3 User Applications (UA)

Users are likely to purchase and download applications that reside and run on an SDRD. Damage to or modification of the application, such as the insertion of viruses or other malware into a UA reduces or eliminates the application’s value to the user and may threaten other elements of the SDRD. A user application is any software that does not fall into any of the preceding three categories. Examples are games, word processing, address and contact management software.

3.1.5.4 Radio Platform Operating Environment (RPOE)

In the context of the Wireless Innovation Forum endorsed SCA[1] this software consists of the Core Framework, the operating system software, devices, drivers, middleware, and services such as a downloader and installer. It includes any other software fundamental to the operation of the radio platform with the exception of the RPA, SPA and UA defined above. This might include a voice Codec, and other software components (e.g., interleavers, Viterbi encoders/decoders etc.) which can be used by the RPA or any other software. Of particular import to this document is the fundamental point that the RPOE includes all of the radio security services. For a policy based radio this class would also include policy enforcement mechanisms and/or services.

3.1.6 Reputation

Service provider networks, government agencies and others may monitor network activity for suspicious activities. When suspicious activities are detected, monitoring agents respond by recording related information such as the source SDRD, the duration of the activity and type of activity. When a device acts inappropriately, it damages the device's credibility, the owner's credibility and perhaps the manufacturer's credibility. This reputation has value that must be protected from the loss of credibility. Securing the device provides assurance that is does what it is supposed to do.

3.1.7 User Data

As a device's capabilities grow, the user's personal information stored on the device is likely to grow with it. Examples of such data include: credit card numbers, pins, user log-in names, home addresses, account numbers, address books, date books and other personal information. Disclosure of that information to hostile parties through attacks on the SDRD can cause considerable damage, financial and otherwise, to the owner of that data.

3.1.8 Platform Configuration and Operating Data

A device needs configuration information for proper operation. Configuration information includes information on allowed operations and operating parameters, and information related to platform security such as digital certificates used by the platform and it users. For public safety and military organizations it could include operating parameters such as definition of the specific frequency channels to be used. This information or data can control the behavior of the operating platform applications as well as specific behaviors of the SDRD and would be a desirable and potentially vulnerable target for malicious purposes. For example, when policy based behavior is used to control the functionality of devices, unwanted modification of the

[1] http://groups.sdrforum.org/download.php?sid=780

configuration or policy information can adversely impact a device or the networks in which it operates.

3.2 Stakeholders & Roles

A suggested concept of operations involves several roles, each with its own objectives, and each with authority over SDRD resources and assets. An entity that has an asset associated with the SDRD and consequently a stake in the resulting behavior of the SDRD is termed a stakeholder. As we shall explore later, a role is an entity who is either a specific stakeholder or someone representing a stakeholder, and who is involved in some aspect of the use, operation, management, control, deployment, maintenance, and/or security of the device and the network in which it operates. The number of roles and stakeholders is flexible and is dependent upon the SDRD's type and user community

3.2.1 Device User

The user role corresponds to the individual or entity that uses the communications device to access communication based services. The user may or may not have property rights in the device, e.g. a cell phone provided to the user by his employer. The user may have only limited property rights, for example, the user of a commercial handset generally owns the device, but its use may be limited to certain networks, and software on the unit is only licensed. However a public safety user is not the owner but is granted the right of access and use to the device in order to fulfill the user's role in the public safety organization in which the user serves.

3.2.2 Device Administrator/Owner

The Administrator/Owner role corresponds to the individual or an entity that desires to control which of the set of authorized or permitted communication services are enabled by the device. Administrator/Owner will have objectives different from the user. Administrators may wish to limit the sources, distribution, content and time of downloadable software and policies. For instance, a parent might purchase a reconfigurable device and want to restrict behavior when used by children. Similarly, enterprises might provide devices to their employees but want to enforce corporate policies. In these cases, the device owner may serve as the stakeholder for the user role.

3.2.3 Regulator

The regulator is the legal authority that assigns spectrum rights to communication service providers and establishes limits for safe operation of radio equipment. In some jurisdictions multiple stakeholders may fill the role (e.g., FCC and the National Telecommunications and Information Administration in the US). A few of the objectives of the regulator stakeholders include avoiding radio interference and ensuring that electromagnetic radiation does not exceed specified thresholds.

3.2.4 Communications Service Provider

The Communications Service Provider is whatever entity is providing the device with a radio communications service, which includes commercial wireless network operators, broadcast satellite subscription services, and peers in an ad hoc network. Communication Service Provider stakeholders may have a variety of objectives such as limiting network access to those with particular versions of radio software or ensuring that only those who have paid for proprietary software can receive a connection. A reconfigurable device may have multiple communication service provider stakeholders if the user of the device subscribes to multiple services.

3.2.5 Manufacturer

In most current regulation, the radio manufacturer is held responsible for the behavior of the radio. So long as this continues, the manufacturer stakeholder will want to continue to restrict behavior on the device throughout its life cycle. However, identifying a single manufacturer may be difficult for reconfigurable communications because devices may involve the integration of several hardware and software components, potentially in a plug-and-play manner. For this reason, the manufacturer role may be filled by several stakeholders in certain environments. In the end, the manufacturer is the entity that assumes liability for the performance of the device, which in most cases is an integrator of hardware and software components to create a platform for radio software. In other instances governing authorities may define the responsible entity.

3.2.6 Software/Content Provider (SCP)

The software content provider is the entity that takes responsibility for the performance of the radio software, in most cases the entity that wrote the code. The provider may have a number of objectives, including that its code is only executed on known good platform configurations, is used in conjunction with communications service providers familiar with its operation or, is limited for use to those that have paid licensing fees. Content providers are concerned about privacy (protection of Intellectual Property) and being paid for use of their IP.

3.2.7 Download Authorization Authority (DAA)

In the context of this document, the DAA is an entity with the authority to approve the download of software/firmware of a designated type (per definitions in section 3.1.5). The approval mechanism may be any of several types including a downloadable machine interpretable policy, a required digital signature on a download file, or perhaps even a tabular listing of approved software and their providers etc. This role is defined solely within the context of enforcing security policy.

3.2.8 Software Distributor (SD)

In the context of this document the SD is any entity who is an approved distribution point for any software which is authorized to be downloaded onto the SDRD. It may be a network operator's server, a software vendor or service provider's server, or an individual who is authorized to connect storage media to the SDRD for download purposes. This role is also defined solely within the context of enforcing security policy.

3.2.9 Policy Distributor

This role parallels the Software Distributor role, because it characterize those entities which are designated as being authorized to distribute policies of a designated type to an SDRD and to components of the network in which the SDRD operates. As with the preceding role, this role is defined solely within the context of enforcing security policy.

3.2.10 Policy Issuer

This is a broad class of roles each defined by the type and the nature of the policy being issued. Examples of such policies are regulatory policies, network security policies, network management policies, as well as individual SDRD security policies. From an SDRD security policy enforcement perspective, a Policy Issuer is an entity who is authorized to issue a corresponding type of policy. There many variations possible and are of course SDRD system and network design dependent. As with the preceding roles, this role is defined solely within the context of enforcing security policy.

3.3 Vulnerability Classes

Vulnerabilities are weaknesses (which may have been intentionally included/ inserted) that exist in processes, protocols, hardware and/or software design that provide the means or opportunity for exploitation. As part of a system security design process it is essential to consider potential sources of vulnerabilities as well as the means to avoid the risk of their successful exploitation. Systems in the design phase only have potential vulnerabilities, but these vulnerabilities for analysis purposes can be derived from actual examples of real world vulnerabilities. The following sections address various real world aspects for consideration as part of a security design vulnerability assessment. Our focus is primarily on software aspects since it is via software that most vulnerabilities can be exploited

3.3.1 Design Process Vulnerabilities

Having a well defined and structured software design process that includes such things as coding standards, peer reviews, code reviews and other similar modern techniques, is an important aspect to avoid either intentional and unintentional vulnerabilities that are introduced during the software design process. It is essential that such reviews specifically address security issues. When reviews ignore security, vulnerabilities may be unintentionally introduced

In the past software designers have put into their code features and capabilities that exist only to simplify their task of testing and debugging the code for which they are responsible. This may include a “back door” into the operating environment that bypasses standard access mechanisms. Unfortunately, all too often this code is still resident in the final product and available for exploitation either by others or by the original author. A software design process that precluded their use or, as a minimum ensures such residuals are removed in the final product, can deny any opportunity for exploitation.

Another potential source of vulnerabilities is the result of undocumented features that are not part of the original requirements for the product but which the programmer thought were “a good idea”. Not only do these increase the development cost of a product, but they are often undocumented (except in the source code) and few are aware of their existence. Because of this they will likely not be considered by the security design team in their threat analysis and could thus be a source of an actual vulnerability in the end product. Here again, a structured design process that includes requirement traceability, both forward and backward with appropriate reviews will identify code features which are not supported by any defined requirement. If such are found and they are worthwhile ideas or features, then they can be included in the requirement base and considered as part of the overall security design. In this way they become documented features.

Care should also be taken in the overall functionality and capabilities of the device operating environment. As we are witnessing in the personal computing environment, e.g., PCs, PDA’s cell-phones etc., an overriding desire to include a broad spectrum of flexible capabilities and features within the OS can result in a constant need for “patches” to address the seemingly unending security holes. With the addition of greater functionality, a system has a greater potential for vulnerabilities unless proper security features are embedded in the devices security architecture and design to preclude exploitation.

3.3.2 Manufacturing Process Vulnerabilities

Even when a robust security design exists, the manner in which the end product is manufactured and tested may afford opportunities for the design security features to be bypassed. A few examples will illustrate this point.

3.3.2.1 Firmware Substitution

SDRs typically use some type of firmware for essential code used to boot/start-up the radio. This may consist of a simple loader program or it may also contain larger parts of the overall radio operating environment. Even if this code has been thorough vetted and tested, such that it is known to be “trusted”, opportunities to substitute or add to the “trusted” code can exist in the manufacturing process. This does imply that an “insider” would deliberately be making this substitution/addition to the code package but depending upon the intended application of the SDR such an attack could have huge economic or political benefits to those responsible. Having appropriate controls and processes in the manufacturing process can practically eliminate the potentiality for such a threat in the factory environment, but this is a kind of vulnerability and threat that can extend beyond the factory and this aspect will be addressed later.

3.3.2.2 Key /Credential Compromise

Security features in the design will require that the radio possess a Public Key Infrastructure (PKI) Device Certificate that can be used for authentication and other purposes. This is in addition to any PKI certificate(s) issued to users of the device. Device access to the private key for the certificate is obtained via a “PIN” (Personal Identification Number) which serves as the decryption key for the private key. When there are large numbers of devices being manufactured each of which requires a Device Certificate, it is impractical to have the device generate its own Public and Private Key pair as is typically done for user certificates since this would require each device to be connected on-line to the Certification Authority (CA). A more likely situation is the use of a dedicated computer system to generate key pairs and to conduct the exchange with the CA to obtain the signed certificates. This computer system could then be used to load the certificates, private keys and associated PINs into the each device.

In addition to the Device Certificate the radio will need at least one certificate chain from the issuing Certification Authority (CA) to the root Certification Authority. This Root certificate is the basis for the root of trust. It is imperative that only a valid root certificate be loaded and that no substitutions or additions can occur.

Any operations loading these certificates and keys should be done under controlled conditions with appropriately vetted individuals to ensure that the process is not somehow compromised to either expose the PINs and private keys or permit falsified credentials to be installed. For example the theft of the PIN and Private Key would allow another device to be programmed with the same credentials which could then “impersonate” the original device. PIN theft alone might allow a malicious process unauthorized use of the private key for a variety of purposes. Finally, additional credentials or substitute credentials could be inserted into the device for later use. As noted above, consideration to using a dedicated protected computer to load this material is one way to reduce some of the exposure risks.

Of course, unless the design has appropriate safeguards to protect this sensitive key material/information within the SDR, then the best process controls will not suffice either. The overall system and process design must be complementary regarding security.

3.3.2.3 Undocumented Device Functionality

Devices and components may have undocumented/induced behavior allowing for backdoors or other mechanisms which allow unauthorized exploitations to occur. These “features” could potentially bypass or disable security mechanisms as they may reside in chips or processing engines. Even thorough examination of chips from trusted vendors which are manufactured in “untrusted” locations/factories can result in this type of vulnerability. This vulnerability can also be introduced when Application Specific Integrated Circuit (ASIC) masks are sent to a foundry where illicit modifications may be made. Device suppliers must implement appropriate controls and processes to prevent these types of modifications from being exploited at any point in the manufacturing process.

3.3.3 Communication Protocols

Communication protocols are also potential sources of vulnerabilities. They might exist at any layer of the stack, or via a device interfaces, whether the interface is a direct physical interface (e.g., Ethernet) or an air interface such as Blue Tooth. The Internet Protocol (IP) world is the source of numerous “buffer overflow” and other attacks exploiting protocol weaknesses which in many instances, if successful, permit the perpetrator to assume effective control of the device. In these instances the device is usually a computer of some sort (e.g., server, end user personal computer, etc.) which can then be used for a broad variety of purposes, all unbeknownst to its owner. While an SDRD implementing a standard protocol is not necessarily able to prevent the exploit of a real vulnerability in a protocol, it can ensure that, other than potential short term denial of service, no additional exploits will be able to modify or access either the software or data resident on the platform. As noted in the introduction to this document, methods to prevent any exploits of the protocols are not addressed in this document, but methods and design elements to prevent further exploits of software and data are discussed.

3.3.4 Open Source and Third Party Software

Another potential area of vulnerability involves the incorporation and / or use of open source or third party software. This includes both integrated applications and subroutines (e.g., a protocol stack) as part of the operating software, as well as tools such as compilers and assemblers.

For open source and third party code which is being integrated into the design, potential vulnerabilities can only be avoided if the source code is provided and is scrutinized and tested to the same degree as code being developed by the design team. These potential vulnerabilities are the same as those discussed earlier and require the same rigorous process to ensure that none of these kinds of vulnerabilities are present.

Tools such as compilers and assemblers are another possible source of unwanted code. Atypical comments embedded in the source code, or seemingly innocent code which appears to have no real defined function can be triggers to corrupted compilers and assemblers to insert malicious code into the object code package. This vulnerability is particularly insidious because almost nobody ever reviews object code. The only protection available is to use tools from reliable vendors and to ensure that all comments and code in the source code package is relevant and understood by members of the design team.

3.3.5 Open Standards Software Design/API’s

Software based on open standards and APIs also are a potential vulnerability since they allow adversaries insight into the design and operation of this form of software. One of the vulnerabilities is that open standards/APIs have the potential to expose real vulnerabilities (if they exist) in the standard or API which might be exploited in some way. However, we must remember that to exploit this type of vulnerability the malicious code required would have to be capable of bypassing all of the security safeguards provided by an SDRD which are designed to ensure that only valid authorized code is downloaded, installed or instantiated on a SDR platform. Of course careful analysis and evaluation of the open standards and API design would hopefully identify any such vulnerability.

3.3.6 Policy Based Operations

It has long been recognized that the processes for downloading software or policy into SDR platforms requires integrity and authentication mechanisms to minimize the threat of hacking SDR software. Even in an SDR software environment based on open standards, the barriers to hacking are fairly high because detailed knowledge of the terminal is required in order to produce code which can change terminal behavior. However, the use of a standard format to define operational policies regarding security, regulatory or other aspects of a radio network operations could significantly lower the barrier of knowledge required and could be viewed as increasing terminal vulnerability.

The key to avoiding security weakness due to use of public policy standards is the employment of robust security mechanisms for encryption and authentication. It is also essential to constrain who may actually author policy statements while also ensuring that the terminal receiving a policy statement will only accept it from an authorized source for the specific policy type. The definition of authorized sources for a given set of policies might itself be defined within another policy loaded by a service provider/operator. Included with the definition of the list of authorized sources and the policy types which each is authorized to author and distribute, should be the associated Digital Certificate containing their public keys. Each certificate must be able to be authenticated down the chain of trust to the Root Certificate. Note however, because of global roaming, and very different areas of interest, a given terminal might actually have to support multiple different chains of trust, each relying on a different root certificate.

3.3.7 Software and Configuration Data Download/Distribution Operations

One of the most powerful features of a SDRD is the ability to be reconfigured “on the fly”. This reconfiguration might involve improved/updated air interface or other types of software/firmware, or it might involve operating configuration or network configuration data used to govern operation of the SDRD in the communications networks. Regardless whether the download/distribution occurs via an air interface or another physical interface on the device such as USB, Ethernet, or other such physical interfaces, the ability to distribute and the action of distribution create a potential vulnerability that can be exploited by a variety of threats. These might involve attempts to alter, replace or substitute data, software and firmware during distribution, or it could involve resending legitimate but obsolete code or data that had been previously copied by the perpetrator during an earlier distribution event. Another type of attack might be to obtain access to the software and firmware code during distribution for purposes of copying and modifying it to include malicious code for later insertion. It is because of attacks like these that standards bodies have addressed the topic of secure downloads for SDRDs for various air interfaces. The Forum’s view is that security measures for download/distribution must be agnostic of the particular interface used and should be based on a common set of security services and mechanisms.

3.3.8 External Interfaces

External interfaces to devices may provide vulnerabilities that can be exploited. An example of this can be found in DMA enabled ports through which the SDRD’s active memory can be modified, altering applications and loaded configuration data. Examples of such external interfaces include USB and Firewire ports that are used in digital forensics to read memory without the operating system being aware of the activity. Other attacks have been demonstrated, modifying the base operating system through DMA devices such as Firewire.

3.3.9 Cognitive/Smart Radio Behavior

The introduction of any new technology necessitates that security concerns be considered to determine whether the technology raises any new security issues that may be either unique or more vulnerable than current technologies. Security issues have indeed been raised about radios whose behavior is a result of learning (e.g., a cognitive or smart radio) or those which employ sensing of the RF environment for purposes of optimizing communications capability for the existing environment or for spectrum sharing purposes employing Dynamic Spectrum Access (DSA). These issues are the subject of ongoing research some of which has already been published.

One area of concern deals with those Cognitive Radio systems which employ a pilot channel. In 2008 the Cognitive Radio Working Group of the SDR Forum, in a document prepared for submission to the ITU, identified potential vulnerabilities involving "jamming" and "spoofing" associated with the use of a cognitive radio system "Pilot Channel" (in Annex 3 to Annex 10 of the referenced document) and discussed several security aspects relating to vulnerabilities which may be introduced with the use of machine interpretable policy based operations. This later capability is often associated with Cognitive Radio applications.

Much of current academic research in this area evolves from the recognition that since a CR learns from its local environment and alters its behavior based on what it learns, that it would be possible for an attacker, who is knowledgeable of the externally accessible parameters and factors which can influence a given CR systems behavior, to influence these factors by altering the external environment is such a way as to direct the behavior of the CR system. The objectives of these attacks may, for example, be a denial of service or directing a cognitive radio into a sub-optimal communications mode.

This is an emerging area of research in academic and Government research environments as well as in private industry. Considering that new standards are being developed for application of this technology area, it is imperative that the related standards bodies consider the range of new vulnerabilities and threats as they work to create the necessary standards.

3.4 Threat Classes

NCSC-TG-004 Glossary of Computer Security Terms created in the 1980's defines a threat as: "Any circumstance or event with the potential to cause harm to a system in the form of destruction, disclosure, modification of data, and/or denial of service" (see Appendix B). RFC 2828[2] lists a threat as: "A potential for violation of security, which exists when there is a

[2] http://www.faqs.org/rfcs/rfc2828.html

circumstance, capability, action, or event that could breach security and cause harm." In simple language a threat is a potential danger that might be used to exploit an actual vulnerability.

All threats can be classified into a common or general set which allows for the high level identification of security services which can be employed to thwart or mitigate the threat. It is important to understand the difference between a threat and an attack. An attack is an action that leverages a threat to exploit vulnerabilities of a system. Threats fall into the following classifications:

3.4.1 Denial of Service (DoS)

Denial of Service (DoS) threats use behavior to deny, delay or disrupt a service(s). It can range from disrupting a particular service on a device, to shutting down an entire network. Examples of a DoS attack includes a device broadcasting on a channel, utilizing 100% of the bandwidth, preventing access, i.e., jamming, or flooding a network access point with traffic to overload the servers. DoS attacks can be difficult if not impossible to prevent.

3.4.2 Unauthorized Access

There are two forms of unauthorized access and each requires different security methods. The first form is through physical access to devices. Opportunities include during manufacture, device installation in the infrastructure, and unattended operations. A variation on unauthorized access through physical access is the theft of a user device that is then modified and returned to the user. This threat aspect requires either physical access controls (as a preventative measure for infrastructure components) and tamper detection mechanisms that detect unauthorized physical access to the internals of a device. Measures should also be included to protect the programs and data resident within the device from access or modification.

The second form is unauthorized access to the internal data of the device via through the control and/or user interface. Relevant security services include access control, "user" identification and authentication as well as encryption services.

3.4.3 Eavesdropping

Eavesdropping, also known as wiretapping, involves various means of purloining and examining the contents of data in transit to gain access to information. This might involve an unauthorized person listening to an unprotected conversation without at least one party's knowledge or consent. Attacks can exploit this threat by observing information traversing wired or wireless communication mediums, or it may include using an electrical device (or software) to gather keystrokes from a keyboard to steal access information such as user account data, identifiers and passwords. Depending upon the specific attack, hardware design measures coupled with either encryption and/or robust access control measures are the common methods of protecting the information.

3.4.4 Masquerade

Masquerade, also known as spoofing, is where one entity represents itself as another. It might involve a rogue radio station coming up on the operating channel of a legitimate station and broadcasting false information. It might even involve electronic transactions which present credentials belonging to a legitimate entity to make the recipient believe that it is that entity. Such an attack may be as simple as using another person's user name and password to access a system. The ideal solution is to employ credentials that cannot be falsified and means such as authentication mechanisms which ensure that the entity presenting the credentials is the one to which the credentials belong.

3.4.5 Modification

Modification involves unauthorized alteration of information, including leveraging backdoors for this purpose. This can include alteration of data in storage, processing or transmission. Other examples of attacks of this type are a buffer overflow to insert and run code and SQL injection, modifying parameters to mine data. One type of defense against this type of threat is to add information to the data or alter data patterns in a way that permits any changes to be detected and, when necessary, corrected. The applications of digital signatures to data in transit and secure hashes to data in storage are two examples of methods to detect modifications. There are also encryption methods that can reveal attempts to alter the underlying encrypted data. Depending upon where the data is stored it may also be necessary to consider other means to ensure the entity (both physical resource and stored code) which detects modifications is also protected from being modified in an undetectable way. When the threat environment warrants it, this may involve protecting that entity with physical anti-tamper mechanisms.

3.4.6 Repudiation

Repudiation is the denial of responsibility for a reported incident or responsibility for the occurrence of a transaction or activity. Claiming information was not downloaded when it was, claiming a call was not made when it was, or that a business transaction did not occur when it did, are all examples of repudiation. Current technology involving the use of digital signatures and authentication methods, coupled with recording relevant data for each transaction can document data useful in instances of repudiation. The collection and performance of these functions for this purpose is known as a non-repudiation service. Often this record has legal ramifications and will be required to meet established standards.

3.4.7 Replay

Replay involves copying legitimate information and then resending this information at a later time. It is another form of Masquerade. In this case, the threat involves resending a message in an attempt to illicit improper behavior. Examples of this threat might involve resending a bank deposit message multiple times in an attempt to increase the balance of an account. Attempts to disrupt network operations might involve sending out of date (but otherwise legitimate) network management or control data. Another might involve distributing older (obsolete) software versions of radio code to mobile and/or infrastructure devices. Depending upon the specific threat exploit, protected time based information may be employed in conjunction with real time authentication mechanisms which ensure that the sender legitimate. Mathematical based series (sequential) of non-repeating values such as those found in the IPSEC packet sequence numbers, is one example of a method that can help to deter attacks leveraging this type of threat, but this only is applicable to certain attacks and may not be useful in other forms of attack. Another method is to only accept authenticated data from authenticated and known sources.

3.4.8 Traffic Analysis

Examination of information flows to gain insight as to the identity of the parties involved in an otherwise secure communication, or monitoring the communications and the behavior of entities responding to an outside stimulus. This type of threat is typically applicable only to governmental and/or military communications and as such will not be further addressed in this document.

3.5 Attack/Exploit Classes

The generalized threats can be realized by various classes of attacks on a system. An attack is an action that leverages a threat to exploit vulnerabilities of a system. An exploit is the specific form of attack employed by the perpetrator. The following paragraphs introduce a generalized list of attack classes applicable to SDRDs with mappings to their associated threats. This discussion can only be at a very high level as there are a great many different types of attacks and an even greater number of exploits for each type of attack, all of which are correlated with the perceived or hoped for vulnerability. There is a great deal of information available about these attacks and exploits available on the Internet, however caution is needed and only known reputable sites should be consulted. A partial listing of resources is included in Appendix C

3.5.1 Malicious Software Installation

Introduction of software that alters the proper functioning of the SDRD or a component of the SDRD in a manner that degrades the security or functionality it provides. As explained earlier in the discussion on vulnerabilities, there are many ways for malicious code to be introduced. This attack may exploit a form of Masquerade by claiming to be a privileged or trusted entity or may leverage the threat of modification by exploiting a buffer overflow attack to insert and execute code on a system, or it may be embedded in the boot memory devices during manufacturing or later by gaining physical access to an installed SDRD which is part of the network infrastructure.

3.5.2 Software Misuse

Use of software in such a way that the use:

1) Alters the proper functioning of the SRDD or a component of the SDRD in a manner that degrades the security or functionality it provides; or
2) Discloses information in violating the information's owner's rights; or
3) Violates the software license agreement;

This attack can fall under the threats of Modification or Masquerade, depending on which exploit method is employed. [**Note**: It is not generally considered the responsibility of the SDRD to enforce software license agreements. That is the responsibility of the users or licensees of the software and in some instances, user applications provide this functionality (e.g., Digital Rights Management).]

3.5.3 Spectrum Misuse

Spectrum may be used outside of the allowed parameters and can effect a device's reputation and that of the network as a whole; leveraging the threat Denial of Service. Alternatively this might be considered a theft of service.

3.5.4 Tampering

If physical access to a radio can be gained, running and stored information is at risk. This attack can be used to eavesdrop or to modify information, including software or configuration data for an SDRD. Destruction of hardware can cause a Denial of Service.

3.5.5 Spoofing

If an intermediary is able to modify information in transit, replay information (capture a copy of the information and retransmit the copy as a later date) or present oneself as an authorized stakeholder of the SDR then the intermediary can uses these techniques to gain access to alter the behavior or function of a system. Use of these techniques comes under the label of Spoofing.

3.5.6 Unauthorized Modification of Data or Software

An unauthorized change to software/firmware/hardware (loss of integrity) that alters the proper functioning of the SDRD or a component of the SDRD in a manner that degrades the security or functionality it provides. This is a threat of Modification without having physical access to the device.

3.5.7 Accessing Information Without Authorization

Information resident on the SDRD has value to the information's owner. Unauthorized reading of the information may allow the reader opportunities to misuse the information read. For example, the owner of an SDRD may store financial information such as credit card numbers, bank account numbers or passwords to bank accounts. Upon access an unauthorized agent may use that information to access the owner's accounts and subvert the funds in those accounts. This attack can leverage the threats of Masquerade, by presenting false credentials to gain access, or Eavesdropping and reading the information.

3.6 Risk Assessment

For a threat to be effective, the attacker must be able to exploit a threat to a device. Not all radios are equally vulnerable to a specific threat, nor are the environments an SDRD operates in equally enabling of a threat. The Risk Assessment thus must consider the potential threat environment.

Risk assessment is a probabilistic rating assessment of the likelihood that a hostile entity will devote the required effort (and cost) to exploit a potential or real vulnerability, combined with the probabilistic likelihood of the success of exploiting that vulnerability given the planned protective mechanisms. This evaluation must be done in conjunction with an estimate of the cost/impact of any damage or loss associated with the targeted assets as well as the perceived value of the asset(s) to the potential perpetrator/attacker. This latter valuation also is directly related to the attacker's motivation and reason for devoting resources necessary to successfully exploit a vulnerability.

Consequently the analysis must include identification of those entities who might desire to exploit a vulnerability (threat identification) and for each, provide a probabilistic assessment of their capability (resource evaluation) and willingness to devote the required resources, both financial and otherwise, necessary to successfully exploit the vulnerability. This assessment must also consider the perceived value of the targeted assets to the various stakeholders. Some stakeholders (e.g., users) may not even be aware of an asset class just as some attackers may have little or no interest in an asset.

Independent of the preceding, the risk assessment must consider a technical analysis of the SDRD security architecture which identifies potential vulnerabilities and the means by which the vulnerability might be exploited. An SDRD's susceptibility to vulnerabilities may vary depending on the operational scenario in which the SDRD is used as well as the underlying robustness of the planned design and implementation of the security mechanisms. For example, a satellite in orbit is much less susceptible to physical access than an unattended base station located in a rural area. Consequently the level of a threat should be calculated for each vulnerability to produce a composite threat assessment. For each perceived exploit a threat mitigation strategy must be developed and the implementation cost estimated. Since not all mitigation strategies will be 100% effective an assessment of the resources necessary to overcome the exploit mitigation method should also be considered and provided as data back into the first part of the risk assessment analysis. Clearly this may require several iterations to resolve high risk exploits against high value assets before an affordable mitigation strategy is developed.

Armed with the information derived from these activities, a cost effective Security Architecture with its associated set of security mechanisms, can be selected for implementation with the appropriate degree of design robustness commensurate with the evaluated threat environment.

The ISO/IEC 27002:2005 Code of practice for information security management recommends the following topics be examined during a risk assessment:

- security policy,
- organization of information security,
- asset management,
- human resources security,

- physical and environmental security,
- communications and operations management,
- access control,
- information systems acquisition, development and maintenance,
- information security incident management,
- business continuity management, and
- regulatory compliance.

Risk assessment is a continually ongoing process since the threat environment is constantly changing. For this reason SDRD using organizations and SDRD manufacturers need to consider this throughout the life cycle of a product

A couple of examples will serve to illustrate several of the key points addressed in the preceding discussion. Table 1 lists the threat classes discussed earlier in Section 3.4. Table 2 provides for a scenario of possible attacks against a mobile cell phone.

Examination of the rows in Tables 2 and 3 will show that not every threat class applies to every asset class. In some instances the stakeholder perspective will influence which threat classes apply to an asset class. Inherent to creation of a table like this are assumptions about which threat classes apply to a given asset and which do not apply. Influencing these assumptions are presumptions about the capabilities and motivations of the attacker. Our examples aren't any different, but for this document they are only examples to illustrate the process.

Table 1: Threat Classes
• Denial of Service
• Unauthorized Access
• Eavesdropping
• Masquerade
• Modification
• Repudiation
• Replay

The first column in Table 2 lists the Target Asset classes addressed earlier in Section 3.1 while the second column lists applicable threat classes (Section 3.4) that the attacker might desire to exploit.

The third column heading identifies the **Stakeholder**, which is the mobile phone user/owner, and the data below indicates a presumed valuation of the asset as perceived by that stakeholder. In some instance the stakeholder is unaware of the asset and consequently would place no value on that asset except for how its exploitation might affect other assets such as communication services.

The fourth column heading identifies the **Attacker**, which in this case is presumed to be a technically untrained but technically skilled/internet savvy individual. This attacker is presumed to have limited resources (e.g., technical skills/knowhow, time, financial, manpower, etc.), to utilize in creating an attempted exploit. The data in the column below represents a presumed evaluation by the attacker of the interest (motivation) and willingness to apply the necessary resources to the exploit. Note we have presumed that this attacker has either no or little interest in some of the threat classes (e.g., denial of service) and this is reflected in the indicated evaluation.

The fifth column is an assessment of the general level of resources (same as above) needed to be able to create an exploit (which still may or may not be successful) while the last column represent an evaluation of the risk that, given the available resources to attacker, and, considering the resources required to create an exploit, the defined attacker would devote the necessary resources to the task.

Table 3 provides a second example scenario. In this scenario the stakeholder is now the Network operator rather than the cell phone user/owner. While the stakeholder may have changed, the target device under attack is still the user's mobile phone and the attacker and his interests and resources are unchanged. The network operator perspective brings into play additional threat classes for consideration.

A formal analysis would bring into play many more quantifiable parameters, but these would be based on a baseline design approach for the communications device which would allow highly objective risk analyses and cost data to be formulated. Such analysis is beyond the purpose or scope of this document.

Table 2 Example Mobile Cell Phone Risk Assessment for User/Owner -Scenario 1

Target Asset Class	Threat Class	Stakeholder: Mobile Phone User/Owner	Attacker: Amateur Internet Savvy Tech Resources: Low	Resources needed to exploit Threat	Assessed Likelihood of Threat Attack
		Stakeholder Valuation of Asset/Threat	Attacker Valuation of Asset/Threat		
Communication Services	Denial of Service	High	Low	Low	Low
	Unauthorized Access	High	High	Low	High
Electromagnetic Spectrum	Denial of Service	Unaware	-	High	Zero
	Eavesdropping	High	Low	Low	Low
Health and Safety	Modification	Unaware	-	High	Zero
Radio Hardware	Unauthorized Access	High	High	Low	High
	Modification	High	-	High	Zero
Radio Software/Firmware: RPOE RPA SPA	Modification	Unaware	Low	High	Low
Radio Software/Firmware: User Applications	Modification	Medium	High	Medium	Medium
Reputation	Masquerade	High	Low	Low	Low
User Data	Unauthorized Access	High	Medium-High	Medium-High	Low
Platform Configuration And Operation Data	Modification	Unaware	Medium	High	Low
	Unauthorized Access	Unaware	Medium	High	Low
	Denial of Service	High	Medium	High	Low

Table 3 Example: Mobile Cell Phone Risk Assessment for Network Operator Scenario 2

Target Asset Class	Threat Class	Stakeholder: Network Operator	Attacker: Amateur Internet Savvy Tech Resources: Low	Resources Needed to Exploit Threat	Assessed Likelihood of Threat Attack
		Stakeholder Valuation of Asset/Threat	Attacker Valuation of Asset/Threat		
Communication Services	DOS	High	Low	Low	Low
	Unauthorized Access	High	High	Low	High
	Masquerade/repudiation	High	High	Low	High
Electromagnetic Spectrum	Denial of service	High	-	High	Zero
	Unauthorized Access	High	High	Medium-High	Low-Medium
	Eavesdropping	Low	Low	Low	Low
Health and Safety	Modification	Low	-	High	Zero
Radio Hardware	Unauthorized Access	High	High	Low	High
	Modification	High	-	High	Zero
Radio Software/Firmware: RPOE RPA SPA	Modification	High	Low	High	Low
Radio Software/Firmware: User Aplications	Modification	Medium	High	Medium	Medium
Reputation	Unauthorized Access	High	Low	Medium-High	Low
	Masquerade	High	Low	Low	Low
User Data	Unauthorized Access	Low-Medium	Medium-High	Medium-High	Low
Platform Configuration and Operating Data	Modification	High	Medium	High	Medium
	Unauthorized Access	High	Medium	High	Low-Medium
	Denial of service	High	Medium	High	High

4 Security Policy and the SDRD

The preceding chapter has outlined an approach to understanding and establishing the vulnerabilities and threat environment associated with any particular class of SDRD. The completion of this essential step is used to establish the foundation for developing security policies that can be applied to the SDRD and the system within which it operates. There are other important factors that contribute to these policies, mainly stakeholder interests and of course balancing the costs of implementing and maintaining these policies against those costs associated with threats causing a violation of the policy. As part of this process we must first define what is meant by a security policy.

4.1 Security Policy Definitions

It is relevant to understand what is meant by the term “security policy” in the context of this document since security policy is a term that has several meanings depending upon perspective and context. To avoid ambiguity and misinterpretation, the following definitions are provided to clarify these terms as they are used in this document.

4.1.1 Organizational Security Policy

The **Organizational Security Policy (OSP**) is the broadest and most general of the security policies. The OSP is a formal statement of the rules by which people who are given access to an organization's technology and information assets must abide. It is a document intended to guide humans rather than equipment. The OSP is enforced by individuals in the organization. It is not directly part of an SDR but it does include a definition of the assets of the system and the individual components that must be protected and the assurance level of protection mechanisms to be applied during the design and development phase of the system and it component parts.

4.1.2 System Security Policy

The **System Security Policy (SSP)** is a set of rules, requirements and practices that specify or regulate how a system (e.g., the networked hardware components, the SDRDs, as well as software and physical plant elements of the system) provides security services to protect resources. The SSP is therefore a component of the System Security Architecture and Design that implement the relevant aspects of the Organization Security policy. It supplies the technical goals and objectives against which the System Security Architecture and Design are evaluated. The SSP is one element of the decomposition of the OSP. Other elements are not relevant to the SDRD and thus beyond the scope of this document.

Significant portions of the SSP are implemented via security services that may employ security policy expressions and/or are integrated into the system architecture and design. The security services may either employ re-configurable application interpretable expressions of relevant portions of the SSP and/or they may be implemented as an inherent aspect built into the system hardware and software design.

4.1.3 Radio Platform Security Policy

The **Radio Platform Security Policy (RPSP)** is the portion of the SSP relevant to the SDRD. The RPSP is a set of rules, whose enforcement is either implicit in the design and/or explicit via machine interpretable expressions. In either case, these rules:

1) Define and constrain the application of security services, and

2) Govern or restrain a system's possible actions as defined by the SSP.

The RPSP is a critical required component contributing to the overall SDRD Security Architecture definition, as well as the detailed design and implementation of the entire SDRD**.**

Examples of behaviors that may be governed by RPSP include, but are not limited to the following:

- Expressions of security policy regarding specific types of security services or functions
- Expressions to another entity delineating the minimal security standards and mechanisms required for interoperable operations with the device.
- Operational rules that restrict device operation to behaviors that comply with those required or allowed by the network's SSP.
- Air interface specific security policy regarding waveform or other application specific functions. (E.g., from which specific PKI certification authorities does the device accept certificates?)

The discussion above describes the RPSP as derived from the System Security Policy which in turn derives from the Organizational Security Policy. Considerable risk is induced if this process of decomposition is not followed. A Radio Platform Security Policy that fails to have trace-back to the System Security Policy and the Organizational Security policy is likely to have holes in it that attackers can and, given the opportunity, will exploit.

The authors of this document strongly advise that the developers of the Radio Platform Security Policy, System Security Policy and Organizational Security Policy expend the efforts necessary to assure that System Security Policy is consistent with the Organizational Security Policy and address all the relevant Organizational Security Policy security goals. Similar efforts should be expended to assure that RPSP is consistent with and address the relevant System Security Policy security goals.

To be able to effectively apply the RPSP it must be expressed and transformed into language that the designers of the SDRD can use and apply to its design. To that end, the RPSP is transformed into **The Radio Platform Security Requirements (RPSR).** The RPSR captures the Radio Platform Security Policy as a set of requirements that guide the development, deployment, provision, and operation of the SDRD that together give assurance that the SDRD RPSP will be enforced by the SDRD. These are further broken down into Radio Platform Security Design Requirements and the Radio Platform Security Operational Requirements.

Radio Platform Security Design Requirements (RPSDR) are those requirements that impose constraints on the functions and the overall architecture as well as the Radio Platform Security Architecture of the SDRD.

Radio Platform Security Operational Requirements (RPSOR) are those requirements that impose constraints on the production, distribution and operation of an SDRD within the intended network environment.

The **Radio Platform Security Architecture** is the framework upon which all of the security functions, services and mechanisms are implemented on the SDRD. It defines the design of the system from a security perspective and allows for the proper selection and implementation of security services and mechanisms. The purpose of the security architecture is to implement the RPSDR assuring that that the SDRD enforces the RPSP.

4.1.4 Assurance Levels

As described in latter sections, the RPSP is enforced by a security architecture that dictates how and when any given security mechanism is applied. But not all enforcement mechanisms are equal in the degree to which they are able to protect the SDRD. As indicated previously, the RPSP must also address the level of assurance required in the implementation of the security mechanisms being employed so that other elements of the RPSP are being enforced. This document thus defines **assurance level** as the grounds for confidence or trust that a SDR Platform meets the Radio Platform Security Policy (RPSP) and that the security mechanisms are not being circumvented. The required assurance level is explicitly stated within the **RPSDR.** These requirement expressions of the **assurance level** include both quality and robustness requirements which state and quantify what is necessary to ensure that the protection mechanisms are properly implemented to counter the threats identified by the risk assessment. Additional discussion of assurance level is presented later in section 7.2.7.

4.2 The Essential Need for the Radio Platform Security Policy

In light of the preceding, it should be apparent that the existence of the RPSP and the higher level policies from which it was derived are considered as being essential elements in the process of developing an SDRD. These serve as the foundation for trust by the stakeholders of the SDRD that their security needs and concerns are being provided.

4.2.1 Trusting the SDRD

Trust is both a vague and valued attribute of an SDRD. There are two kinds of trust involved in an SDRD. The first derives from a formal definition of "Trust" that comes from a standard security design perspective (see glossary). The second is the meaning of trust from a user's perspective. For example, users have a sense of trust that the SDRD will meet their operational needs and that it will be available for use when they need to use it. They trust and through that trust they rely on their SDR to supply a level of expected behavior. They rely on the SDRD preventing others from eavesdropping on their conversations or accessing their text messages. They might rely on their SDR to prevent others from accessing SDR resident information that the user considers personal. They might rely on the SDR to prevent others from tracking certain attributes of their use of the SDR. Of course, before these aspects can be relied upon, they have to be defined components within the RPSP. The consequences from misplaced trust can be considerable with the result that the user finds the value of the SDR greatly reduced or perhaps of no value at all.

Reconfigurable attributes of SDRD technology complicate the issues associated with trust. Prior technologies without this capability had well defined and constrained limits of operation which either worked or didn't. While SDRD technology doesn't remove these constraints entirely (e.g., a radio cannot operate in a frequency band which is not supported the hardware design), it considerably broadens the span of capabilities which can be offered.

One value of reconfigurable SDR technology is that devices can support a much greater variety of communication methods than the legacy hardware radios and can be upgraded with new features and capabilities. With this variety comes the problem that not all of the newly supported capability may be desirable from either the user's or another stakeholder's perspective. For example, users do not want their devices to be reconfigured surreptitiously which might result in their being connected to an expensive network without their knowledge. Network Operators do not want handset devices that are attached to their network to act as gateways for interlopers (e.g., exploiting a Bluetooth interface on a device to place calls via the device), and, regulators do not want devices to be reconfigured to utilize frequencies in violation of regulatory policy.

Justification for stakeholders trust in the SDRD requires that the device operations be constrained to those that conform to that established by the stakeholders. This means the SDRD must enforce restrictions on access to and use of platform resources such the SDRD behaves in the way expected for the intended purpose. These restrictions must be explicitly stated to be understood. The statement of these restrictions is contained within the RPSP. The security policy thus bounds the behavior of the SDR supplying a basis for trust in the SDR, and the level of expected trust a user should have in his SDR is bound by the security policy and the quality of its implementation. As is often stated in the world of information assurance, security must be built in, not added on.

4.2.2 Security Policy and Trust: an Example

Returning to the example of above and an SDR supplying a phone service, the RPSP might state in some manner that all calls need to have a confidentiality service, e.g., only the called and caller have access to the conversation. Unlike landlines, SDR calls go over the air and could be monitored by commercially available devices. While the monitoring cannot be prevented, by utilizing proper encryption techniques the information gleaned from that monitoring can be reduced down to virtually zero. The RPSP may also dictate that as part of call setup the peers must establish a confidentiality service using a particular grade of cryptography, (more on cryptography in the services section) and if the confidentiality service cannot be set up, the parties in the call could be alerted to the situation that the conversation is subject to being monitored. This example has been restricted to an operational example of call set-up. There are many other possible examples applicable to all aspects of the SDRD operation and design.

4.3 Security Policy and SDRD Design

Users would have little trust in an SDRD with a weak security policy or poor implementation if they were aware of such a situation. They trust the manufacture to design a device that does not violate this trust. As such a security policy is a set of critical requirement components contributing to the overall SDRD architecture and design and most particularly to the SDRD security architecture and design.

Having the security policy as a principle component of the design and implementation of radio architecture allows for:

1) Inclusion of the required security services into the Architecture.

2) Placement of the security services such that other aspects of the architecture cannot circumvent the security architecture (robustness or level of assurance), and

3) Enforcement of practices that assure the design reflects the Security Architecture.

It is these latter two aspects that are often overlooked. Even the best security architecture is of little use if the implementers fail to produce a design that reflects the all of the requirements for the architecture. Worse, users, relying on the purported security design, put undeserved trust into the SDR.

4.4 Explicit Security Policy

Most current generation commercial radios have implicit implementations of the SSP; thus the policy is expressed as an integrated element of the SDRD design as a result of hardware and or software design features and implementation. A few of the current generation SDRDs might have a portion of the RPSP expressed in a machine interpretable digital form. In such a form, the parameters can be expressed in a manner that supports the ability to accommodate policy variations within defined limits that are needed to meet either application specific or operational situation specific security needs. An example of this might be to use access control mechanisms as discussed in Section 5.1.1

Many future generations of SDRDs will advance this trend by providing the capability to express individual stakeholder discretionary policy in areas which fall into the stakeholders' domain. They may include portions intended to be modified and updated as a downloaded object to meet stakeholder's changing/evolving needs.

Explicit Radio Platform Security Policy

When an SDRD uses a subset of the RPSP rules expressed in digital form this statement of rules is, for the purposes of this document, called the **Explicit Radio Platform Security Policy (ERPSP).** ERPSPs provide a stakeholder the ability to manage security policy elements which might require some degree of flexibility. For example, in the next chapter discussing security services, access control mechanisms typically require the flexibility to identify who may access the SDRD for any given purpose and define their access privileges. Likewise, network operators may need to add new network components which can effect network security operations and operating parameters. A policy statement which defines these privileges and roles is an example of an ERPSP, and may be expressed by a machine interpretable and downloadable policy statement, via a capability provided by the human machine interface, or both. The allowed latitude governing the variables associated with an ERPSP should be consistent with the RPSP and its higher level predecessor the SSP. Determination of what portions of the RPSP will be expressible in this manner and how they will be expressed and interpreted by the SDRD will depend on many factors, including the functional use/application of the SDRD, the vulnerability and threat analysis results and decisions made while formulating the system and security architecture of the SDRD and finally, the requirements of the stakeholders.

When the SDRD supports the use of an ERPSP it will need a functional element that, for purposes of this document will be identified as **Security Policy Enforcement Engine (SPEE).**

A given system design may support only a single ERPSP or it may require a variety, each of which specifies policy variables for a specific security aspect. Likewise the system design may provide for one or several SPEE functions with ERPSP enforcement split amongst them. Thus this functional element may be singular or multiple distributed among various elements of the system. The SPEE is the mechanism that interprets the downloaded policy and implements the enforcement of the rules expressed by policy. Particular care should be given to the design, implementation and testing of the SPEE because it is essential that the policies defined by the ERPSP are enforced at all times and cannot be circumvented. As such any SPEE represents a type of security critical process, (security critical processes are addressed later in this document) and the assurance levels associated with the SPEE implementation as well as the distribution/download and storage of any ERPSP should be correspondingly high.

As the practice of distributing security policy elements in the form of a downloadable ERPSP expands, network operators may require the need to express this policy in a standard language and form. This would allow them to establish management systems which can accommodate SDRDs manufactured by many suppliers. In fact such work is already underway in the IEEE SCC41/P1900 working groups as well as the WINNF Modeling Language for Mobility Work Group. These activities address a broader range of communications policy managed behavior than just security, and include regulatory and cognitive radio behavioral policy.

4.5 Stakeholders and Security Policy Contributions

Developers of the RPSP must consider the possibility that conflicts could arise when multiple stakeholders can issue ERPSPs since stakeholders' interests may come into conflict. For example, service providers may wish to offers services that the subscriber wishes disabled or the user may wish to access websites that the Network Operator has identified as sources of viruses or other malicious activities. The RPSP developers must recognize the potential for conflicts in the design and either adopt ways to avoid or resolve the conflicts. As part of that conflict resolution, it is the RPSP developers who determine whose interests dominate and/or how policy conflicts can be resolved. As you can see by our examples there may not be one predominant stakeholder, and the design must consider how to partition the various components of policy so that conflicts can be resolved. The rest of this section discusses resolution of stakeholder conflicts.

For the discussion on policy, this document takes the position that for more capable SDR Devices:

1) An SDR Device may have more than one stakeholder
2) Each stakeholder has one or more "objects" be it software, hardware or firmware resident on the SDR Device.
3) The stakeholder requires the SDR Device to exercise a level of control over that object.
4) The required level of control is indicated by a stakeholder policy for a given object.
5) The policy is software that is downloaded to the SDR Device.
6) Before granting or denying access the stakeholder needs to know that the execution environment will maintain the protections and integrity of the object. Informed decisions require stakeholders having some knowledge about the execution environment in which their objects will run.

For this document, a RPSP that allows stakeholder contributions is a policy that not only expresses the what, when and how security functions are applied, but also dictates the rights and privileges granted to a stakeholder over an SDR Device object.

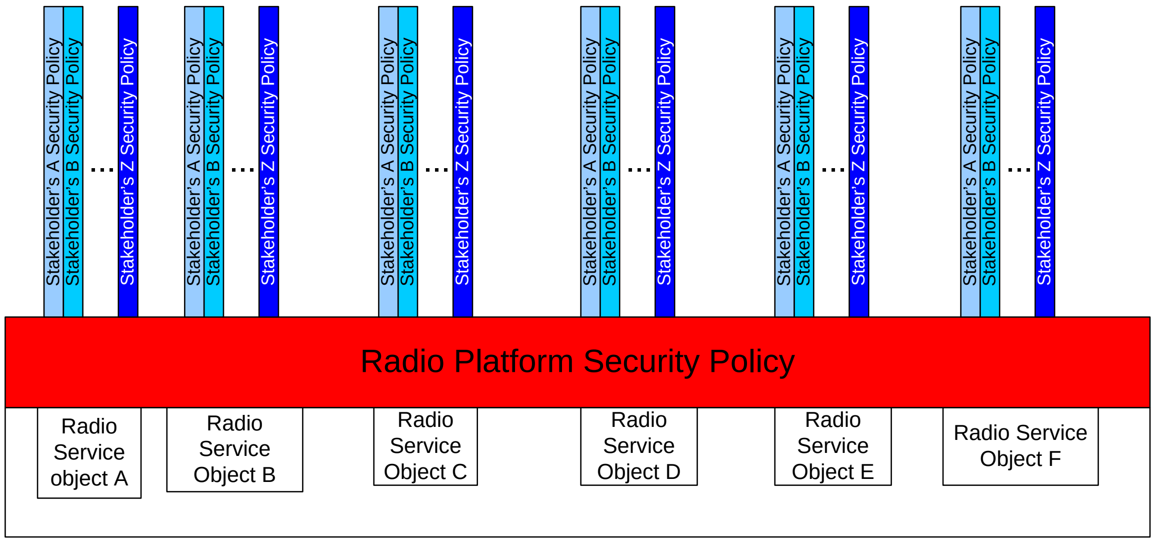

Figure 1 Security Policy Stakeholders and Assets

4.5.1 Conflicts in Stakeholders' Security Policy

Security policies are never perfect, especially when they integrate the positions of multiple stakeholders. When multiple stakeholders are involved, mechanisms are needed that either prevent or resolve conflicts between stakeholders' security policy.

To address these and other concerns, limits must be placed on the set of radio policy configuration variations. Configurations that permit undesired behaviors or create policy conflicts should be excluded from the set. This document will treat the rule(s) that decide which configurations are permitted and which are not allowed as the **SDR Device Meta Security Policy (SDMSP)**. Given that SDMSP limits and controls acceptable configurations, the SDMSP will be part of the hardware and firmware of the Device.

As a practical matter there may be multiple entities that supply components of an SDR Platform security policy but there will be one entity that defines the overall policy including the authority given to other entities. It is incumbent on that entity to ensure the delegated authority does not lead to conflicts among subordinate policies or that when such conflicts arise it provides a means to resolve that conflict. This document identifies that entity as the **Security Policy Primary Authority (SPPA)**. The SPPA defines the authority given to other entities who can express their needs via an Explicit Security Policy. It defines which entities are stakeholders, contributors to RPSP, and the span of control granted to each.

As a practical matter the SPPA is typically the entity responsible for the operation of the SDRD and the networks in which it operates. Only in rare circumstances would another entity possess this role.

4.5.1.1 Span of Control

The authority the SPPA grants to the various stakeholders is called the span of control. The span of control defines the rights of control over security related software, hardware and configuration elements granted to others by the SPPA. In contemporary radio devices and systems one of the most common applications for span of control concerns the application of role based access control (See section 5.1.1 for a detailed discussion of access control to the SDRD's Human Machine Interface (HMI) and to users and interfaces used to remotely manage the SDRD. Two examples can be used to illustrate the span of control aspect.

4.5.1.1.1 Span of Control - Keys and Credentials

As discussed throughout this document, contemporary communications make extensive use of public key based cryptographic techniques. One example is during authentication of the participants in a communication exchange. The peers exchange certificate based credentials and use the exchange to generate the material needed by the cryptographic servers to protect the communication channel. The platform security functions access the keys and credentials, and perform the functions required by the protocols, but, in a sense, they do so on behalf of the user. The user's actions initiate the platform process that requires access to the key database. User's actions may also involve adding or updating keys and credentials to the key database. This access must only be permitted if the platform policy defines ownership of these data parameters or if the user has been granted access to use the keys, e.g., the SPPA has placed certain of the keys and credentials under the user's Span of Control.

4.5.1.1.2 Span of Control - Configuration Parameters

As described earlier configuration parameters are used to specify a broad variety of information used by various functions in a platform. For example, one class of data may address radio air interface operating parameters, while another could define network configuration parameters, and yet another class of data might involve user data. Policy span of control would thus specify for any given stakeholder those portions of the data on the platform he is able to access or manage and any specific limitations within that area. Role based access control (RBAC) is a model to which the span of control concept and principal is directly applied. The RBAC model is discussed in more detail within paragraph 5.1.1.2

4.5.1.2 Authority Precedence

When there are overlapping areas of control there must be a means provided to resolve conflicts. One method is to define a strict priority scheme in which the policy of the entity with the highest priority dominates. In this case the SPPA's authority would always dominate over the authority granted any other stakeholder. For instance a user might be granted authority to change a configuration value but the change may need to be executed by the SPPA which may deny the change. Alternatively the current parameter value may have an ownership attribute. If the priority of the parameter current value was set by an entity with a higher priority, then the lower priority would be denied the change.

The priority scheme might employ another variant where the higher authority establish bounds on parameters and lower priority users may be permitted to change the parameter value so long as the new value does not violate the established limits for the parameter.

Another more elaborate method may employ a hierarchical scheme. In such a scheme a higher authority may delegate authority to lower levels. Once delegated authority over an object, stakeholders can exercise their own permissive/restrictive control over the object and even grant authority to lower level entities. For example in a RBAC model, the role of an entity may have a span of control for several configuration data areas. This entity may then be able to delegate authority to several entities each of which would then have the ability to access and manage only the specific data area to which it was granted or it might limit access to only be able to view/read the information, but not to change it. An example is a user sets up a visitor account on his cell phone granting the user limited access to the services and applications the user is permitted. Thus in effect this process is the equivalent of having a new contribution component to the ERPSP supplied by the higher level user that dictates the additional constraints imposed upon the subordinate role.

Particular care should be taken to avoid implementing a conflict resolution scheme that would result in a denial of service vulnerability. For instance, a user may wish that the SDRD add anti-eavesdropping protection to every conversation but a jurisdiction may insist that all key material used by a device while in the jurisdiction be archived with the jurisdiction’s authority. Revealing the session key used to protect a phone call would violate the user’s anti-eavesdropping policy but not revealing the key violates the jurisdictional policy. The result may be a denial of phone call services, not the intention of either party. Specific resolution mechanism to prevent such a conflict is the responsibility of the platform manufacturer and starts with the definition of the OSP, and continues with definitions for the SSP and the RPSP. However, as discussed in the next section, situations may arise when a denial of service may not be avoidable.

4.5.1.3 Prohibit vs. Allowable Precedence

Practical security policies may in some instance prohibit some behaviors that in the strictest sense would otherwise be deemed allowable, but which the security policy in effect prohibits. For example, an authentication mechanism is being applied to some exchange with an external network component and that component requests the use of a process or algorithm that is not permitted by the current RPSP. It may also be that the RPSP defines that certificates issued by certain Certification Authorities are not to be accepted in relationship to any related security service. In both of these instance, while the possible outcome may a denial of a specific service, the purpose of the RPSP in these instances is to prevent an even more dangerous threat from being able to exploit a potentially more damaging vulnerability. This illustrates the importance of assuring that disallowed behavior is prohibited even at the expense of denying otherwise allowable behaviors.

5 Radio Security Services and Mechanisms

This section discusses the set of core Radio Security Service (RSS) classes that provide the security mechanisms which are used to support and provide security operations and processes for the SDRD. These services are particularly important to those security critical system processes expected to be deployed on software reconfigurable radio platforms. Addressed are the security service classes with examples of mechanisms which can be employed by these services while illustrating how these services can support established system and platform security policies and processes.

5.1 Radio Security Services and Mechanisms

Listed in Table 4 below are a set of core security service classes which are consistent with earlier SDRF security documents. To these we have added new service types addressing security services related to software reconfigurable radio and the emerging technologies associated with cognitive radios. Functions provided by a combination of the mechanisms used by these services comprise basic security functions as well as security critical system processes such as those addressed in the next chapter.

For example, a secure download process could involve the Confidentiality Service, Information Integrity Service, Identification, Authentication and Non-repudiation, Access Control (Authorization) Service, Logging and the Memory Management Enforcement Services. The Key and Certificate Management service might also be employed by these other services as part of the download process to obtain access to the keys and certificates necessary to support these other functions. Encryption and decryption mechanisms could be used to protect the privacy of the information being downloaded. These mechanisms might use symmetric key (e.g., AES) encryption and decryption. Public key cryptography based functions, besides supporting the creation and distribution of the common key used for encryption, are also employed to create a secure hash (message digest), which when encrypted is known as a digital signature. These are used by the security service functions for Integrity, Authentication, Non-repudiation, as well as Access Control (authorization), and Auditing service mechanisms. These security services classes are simply a means of classifying high level security functions and are typically implementation independent.

Note that the grouping of the security services in the table and subsequent discussion is perhaps somewhat non-traditional. The rationale for this grouping will become evident in the discussions which follow as we address each of the various Security Service Classes.

Table 4 SDRD Security Service Classes

Access Control Service including authorization for:	
o Human-SDRD Interface interactions o Software Downloads/Updates	o Policy Downloads & Updates o Configuration Data downloads/Updates o Access/use of platform resources
Identification, Authentication and Non-repudiation Services for:	
o Users o User Devices o Network Devices	o Software content providers o Network Operators o Service Providers
Information Integrity Services for :	
o All resident user data o All resident radio & network configuration Data o All resident software and firmware	o Any downloadable data or software o Over the Air Control and configuration commands
Information Security (INFOSEC) Confidentiality Services including encryption and decryption services for:	
o User communications including Network Control communications o Device Uploads to networks (e.g., Log data, configuration data) o Policy (security, regulatory, etc.) downloads	o Configuration Data downloads o Software Downloads o User data Storage o Configuration Data Storage o Key Material Storage
Transmission Security (TRANSEC) Services for waveform/air interface related security functions such as:	
o Spread spectrum applications o Frequency hopping applications	o Cover for waveform control information o Cover for waveform data
Key and Credential Management Services for:	
o Users Certificates and private keys o PINs, Passwords, Biometric access and other electronic credential data	o Device Certificates and private keys o Root & intermediate Certification Authority Certificates
Platform Resource Security Management Services for :	
o Memory Management Enforcement	o Platform Software Configuration Management • Radio Platform Operating Environment • Radio Platform Applications
Logging, Auditing and Security Alarm Services	
o Usage logs o Security Event logs o Cognitive/DSA Operations logs	o Non-repudiation logs o Security Related Alarm services o Audit log preparation

Policy Enforcement and Management Security Services for:	
o The Platform security policy o Waveform/application security policies	o SDRD Behavioral control (cognitive/learning radio) o Regulatory Policies o Other downloadable policies (e.g., Network Management, Network Security

5.1.1 Access Control Service

5.1.1.1 Introduction

In the world of computer security, before the internet and the extensive wireless networks of today, traditional access control to computing resources focused on defining who had access to machines or devices and managing the PINs, passwords and the profiles defining individual user access rights/privileges. As illustrated by the entries in Table 4, for SDRD's, including cognitive radios and other system/network components, access control has a much broader context.

Access to an SDRD can occur remotely via the wired or wireless networks as well as through direct physical interfaces. Because of the threats witnessed on almost a daily basis, the Forum recommends that an expanded role be given to access control measures; it is believed that these measures are necessary to maintain the security and integrity of the device.

In recent years, the introduction of SDRD technology brought awareness of downloading software and firmware updates and providing security for these processes. Further technological advances have broadened the scope of download operations to SDRDs to include download and/or update operations involving machine interpretable policies (e.g., network management, regulatory, radio platform or waveform security, platform security elements etc.), as well as radio and/or network configuration data, radio environment maps, and other information that may reside within and be used by an SDRD.

Thus, loadable information via any physical interface, including by air should be subject to appropriate access control restrictions. Such access restrictions should define who is authorized to perform the associated function, what credentials are required, identification of the appropriate authenticating authorities, the scope of access authority, who may authorize the distribution/download and who may distribute. Downloadable security policies might define the access restriction sets in terms of the types and classes of information/software/firmware being downloaded. For security sensitive operations such as policy downloads, the security policy definitions should consider allowing only preloaded digital credentials of authorized individuals (e.g., similar to pre-loaded of root certificate), while for other types of download operations (e.g., software), the authorizations might be specified in downloadable security policies which can be created by external systems and downloaded themselves into the SDRD. These types of access control restrictions can significantly reduce the vulnerabilities and threats associated with any type of download.

Access control measures should also be considered for application to components (e.g., routers, servers, management systems) of the larger communications system in which the SDRD operates. Compromise of any part of the system could lead to the corruption or compromise of the SDRD. While not specifically within the scope of this document, many of the recommendations and outlined concerns also apply to these other system components.

In addition, access control security measures existing within either the broader system or within the SDRD itself could prevent lost or stolen radios (e.g., those used by public safety officials) from being used either to monitor communications or to prevent attempts to spoof operations by false communications. Access control of this type could essentially lock out these radios while in the hands of non-authorized individuals but become fully functional in the hands of an authorized user. Likewise, access control applied to network services and operations could help prevent misuse or fraudulent activities involving these assets.

Mission critical radio devices (e.g., law enforcement, homeland security and other federal agency applications) have a higher sensitivity to the risk of malicious software code and attacks; access control-like measures may apply to internal software operations by defining what applications can use specific services and have data read and or write access rights. This topic will be expanded upon later in this document since it is really an important component of the security architecture and design of the SDRD.

5.1.1.2 Applicable Access Control Models

There are four types of Access Control models that may be applicable to systems employing SDRD technology. These four types are:

- Physical Access Control (PAC)
- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role Based Access Control (RBAC)

Physical Access Control, as its name implies, restricts access to an object by means of physical barriers and requires that the person desiring access have the authorized capability of passing through or around the physical barriers. There are of course a wide range of methods for providing these barriers ranging from armed guards to electronic locks on doors accessible via keypads PINS or biometric devices. These types of controls are applicable to control facilities such as Public Safety dispatch centers or cellular network management centers as well as other fixed infrastructure such as base stations. The use of this type of access control is not within the scope of this document since it does not apply directly to the SDRD core components, but it is appropriate for consideration to the larger system in which the SDRD operates. As will be discussed in the Security Architecture section, physical constraints and associated access control mechanisms might also be applicable to the internal design elements of an SDRD when the threat environment indicates that such measures are warranted.

The other three types of access control have evolved from the world of computer security. Both discretionary and mandatory access control were defined in a US DoD developed criteria known as the Trusted Computer System Evaluation Criteria [DoD 5200.28-STD]. These criteria consisted of a number of volumes each of which had a cover of a different color, hence this criteria collection was also known as the “Rainbow Series”. These definitions are from the “Orange Book”.

In the Orange Book **discretionary access control** (**DAC**) was defined as "a means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject (unless restrained by mandatory access control)".

The Orange Book defines **mandatory access control (MAC)** as "*a means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (i.e., clearance) of subjects to access information of such sensitivity*". However, in non-DoD computer applications the term has broadened to encompass “*a type of access control by which the operating system constrains the ability of a subject or initiator to access or generally perform some sort of operation on an object or target. In practice, a subject is usually a process or thread; objects are constructs such as files, directories, TCP/UDP ports, shared memory segments, etc. Subjects and objects each have a set of security attributes. Whenever a subject attempts to access an object, an authorization rule enforced by the operating system kernel examines these security attributes and decides whether the access can take place. Any operation by any subject on any object will be tested against the set of authorization rules (aka policy) to determine if the operation is allowed.*” [2] – (From Wikipedia)

Role Based Access Control (**RBAC**) is an alternative approach to restricting system access to authorized users and was derived more recently than DAC or MAC.

Wikipedia has a fairly comprehensive treatment of RBAC and the following highlighted text is an adapted and edited extract of that article which characterizes and clarifies different aspects of the RBAC. The reader is encouraged to go to Wikipedia and read the full content as well as related topics referenced in the Wikipedia article.

RBAC is sometimes referred to as role-based security in which the role of a user defines a set of allowed operations allocated to the role. Thus by being assigned a specific role, the user is granted permission to perform the functions associated with the role. Thus unlike MAC, RBAC is policy neutral and provides a highly flexible access control technology sufficiently powerful to actually simulate DAC and MAC. In 2000 NIST initiated an activity that eventually resulted in a national standard for RBAC. In 2004, the standard received ballot approval and was adopted as ANSI/INCITS 359-2004.

Prior to the development of RBAC, MAC and DAC were considered to be the only known models for access control: if a model was not MAC, it was considered to be a DAC model, and vice versa.

Within an organization, roles are created for various job functions. The permissions to perform certain operations are assigned to specific roles. Members of staff (or other system users) are assigned particular roles, and through those role assignments acquire the permissions to perform particular system functions.

Since users are not assigned permissions directly, but only acquire them through their role (or roles), management of individual user rights becomes a matter of simply assigning appropriate roles to the user; this simplifies common operations, such as adding a user, or changing a user's department.

RBAC differs from access control lists (ACLs) used in traditional discretionary access control systems in that it assigns permissions to specific operations with meaning in the organization, rather than to low level data objects. For example, an access control list could be used to grant or deny write access to a particular system file, but it would not dictate how that file could be changed. In an RBAC-based system, an operation might be to create a 'credit account' transaction in a financial application or to populate a 'blood sugar level test' record in a medical application. The assignment of permission to perform a particular operation is meaningful, because the operations are granular with meaning within the application. RBAC has been shown to be particularly well suited to separation of duties (SoD) requirements, which ensure that two or more people must be involved in authorizing critical operations. Necessary and sufficient conditions for safety of SoD in RBAC have been analyzed. An underlying principle of SoD is that no individual should be able to effect a breach of security through dual privilege. By extension, no person may hold a role that exercises audit, control or review authority over another, concurrently held role.

However, in large organizations with a heterogeneous IT infrastructure and requirements that span dozens or perhaps hundreds of systems and applications, using RBAC to manage sufficient roles and assign adequate role memberships becomes extremely complex without hierarchical creation of roles and privilege assignments. This latter topic is a recent expansion on the ANSI/INCITS standard model.

There are other access control models, more specialized in nature and limited in application scope and thus will not be addressed in this document. Because RBAC has been shown to be capable of fulfilling the requirements of either a DAC or MAC model, the Forum recommends that the RBAC model be the basis of access control mechanisms applicable to SDRDs and the systems within which they operate. It can be tailored to fulfill a broad spectrum of access control requirements ranging from the simple to the complex. These aspects will be illustrated in the next section.

5.1.1.3 SDRD Operations Subject to Access Control

As indicated in the information below extracted from Table 4, there are five general areas in which access control security mechanisms should be considered for application within an SDRD. We shall now address each of the five application areas listed in the table and discuss specific considerations for each.

Access Control Services, including authorization for:	
o Human-SDRD Interface interactions	o Policy Downloads & Updates
o Software Downloads/Updates	o Configuration Data downloads/Updates
	o Access/use of platform resources

5.1.1.3.1 Human-SDRD Interface Interactions

This topic considers all types of interactions between performed via interfaces on the device designed for use by a human. This may be, for example a physical keypad and/or, touch-screen on the SDRD or it may be virtual interface accessible via any of the device's wired or wireless interfaces.

Before security sensitive interactions with an SDRD are allowed, the individual should identify him/herself to the system and provide some form of identification that can be used by the system to "confirm" the identity. For example, a log in name and password/pin might serve for these two functions. Note, this form of log in doesn't really identify the individual; it simply states that the person logging in has the right PIN or password associated with that identity. In some instances this may be adequate (cell phone user access control), in others a more positive identification of the individual may be needed.

More positive identification might be done with some form of electronic credentials (e.g., a token or smart card) which can be read directly via a physical interface or via a wireless connection. If wireless transactions are employed they should be encrypted in some manner, for example the credentials may be stored in an encrypted form or a secure session is used to transmit the data. For security sensitive or mission critical SDRD applications, keypad entry for PINS/passwords should be performed via a trusted path or a protected channel so that keystroke monitoring of the entered data by a malicious process is not feasible.

Using an RBAC model, the individual also may have to select the desired role unless the role is unambiguously identified by the initial log-in (Note: an RBAC model permits individuals to have more than one role subject only to the "separation of duties" principal). No individual should be allowed to have more than one role "active" at a time and no more than one individual should be allowed to log in at any given time for a given SDRD interface unless the association between user and portal can be unambiguously ascertained for all interactions. An example of a permissive dual log-in is where a user/communication is logged in to the SDRD via the standard user interface and a remote log-in occurs via the air-interface for some administrative function. Another acceptable situation would be when independent secure tunnels can be established via a common interface. It is essential that remote log-in operations should only be permitted over a properly and bi-directionally secured channel.

One advantage of the RBAC model is that it can be scaled to encompass as many or as few roles needed for any given system application. For example, the military might have a role structure consisting of an operator, user/communicator, maintenance technician, security officer and system administrator. Regardless of how many individuals exist for each of these roles for a given SDRD, the capabilities of each role are defined and common to each individual in a given role. Some system functions may be accessible to two or more roles (e.g., maintenance logs), while others are strictly limited to a single role (e.g., only the security officer role may have access to read and erase security audit logs). It is even allowed for a given individual to be authorized access in multiple roles. However, as mentioned in the Wikipedia extract, the Separation of Duties (SoD) principle would not permit any one individual to be authorized the ability to serve for example as a security officer and an administrator since there is a direct conflict of interest. This is because the administrator is the one who established user accounts (including that for a security officer) and authorizes individuals to be assigned to specific roles, and the security officer has the security oversight to monitor the audit logs (which record the creation and assignment of user accounts). This dual role combination could result in a security compromise if permitted. Thus, in this example the system would not (and should not) accept any user account which included both roles.

In commercial cellular applications, roles such as a user and maintenance technician are clearly evident but additional roles also appear to be warranted as we shall explore shortly.

The Public Safety SIG of the Forum has published Use Case Study reports which have examined the application of cognitive radio (CR) technology to public safety radio systems[3,4]. Role based operations have emerged as an area in which CR is highly relevant. In these studies it became evident that a broader range of roles would likely be needed.

For instance, public safety organizations include the broad categories of medical assistance personnel, fire fighters, and law enforcement, all of which are communicators/users of the radio systems. However within any of these categories, expansion of the role definition can be important for CR as well as security operations. For example, Medical roles might be expanded by identifying Emergency Medical Technician, (EMT), Nurse, and physician roles. For physicians, area of specialty may also be relevant. Thus when defining roles and their associated access control functions, it is important to consider how these roles might be involved in other aspects of SDRD design and operation since a single set of roles applicable to both access control and CR functions will provide a less complicated design overall.

5.1.1.3.2 Software Downloads/Updates

Software downloads can occur via any physical device interface or air interface designed to support the functionality; a Bluetooth interface might also be used for this function if the design supports it. Regardless of the interface's electrical or physical properties, there are certain logical restrictions which should be considered to ensure the security and integrity of the SDRD during software/firmware downloads.

The download should be of a type designated for the interface being used. For example, an interface designated to load only Radio Platform Operating Environment software would not be allowed to be used for loading user applications. Firmware updates might require a direct

[3] http://groups.sdrforum.org/download.php?sid=769
[4] http://groups.sdrforum.org/download.php?sid=1460

physical connection by a specific interface. A cellular base station would probably not permit over the air software/firmware updates. As with other aspects of access control, a machine interpretable security policy might define the lists of allowed download interfaces for each class of software. This allows flexibility to effect policy changes if and when they are needed.

It is appropriate at this point to examine additional security considerations for software/firmware downloads. Earlier in Chapter 5, four categories of software resident in the platform were identified as follows:

- Radio Platform Operating Environment (RPOE
- Radio Platform Applications (RPA)
- Service Provider Applications (SPA)
- User Applications (UA)

RPOE and RPA determine how the SDRD operates and may derive from many different sources. Typically the network operator (NOp), be it for a commercial cellular network, a public safety network, or a military network, will want to make decisions concerning by whom, how, when, and where these categories of software will be updated; since ultimately, the NOp is viewed as and is responsible for the service and overall configuration management of his network and is viewed in that way by the users of the service.

The NOp fulfills the role of the Download Authorization Authority (DAA) and only software downloads in these two categories which are so authorized should be accepted by the SDRD. This authorization might be granted by a DAA’s digital signature on the downloaded package, or it might be contained in a list of authorized software providers and approved versions loaded as part of a security policy.

Identification of both the software package and the version of the software is essential for proper security. Specifying the version prevents downloading and installation of obsolete/superseded code packages unless rollback to a specific version is necessary. Identifying the software package prevents downloads that are not applicable to a given SDRD model (i.e. it is intended for a different hardware version) which could result in device failure. These restrictive attributes, in combination, can aid in preventing an improper download rendering a device either unusable or vulnerable to attacks fixed by later versions. These attributes may also apply to the SPA and UA software download categories but are of lesser importance since errors and vulnerabilities in these should not affect basic radio operations.

The Software Distributor (SD) role identifies an entity authorized to download the software to the SDRD. SDs may be actual software/content providers, the NOp or an independent third party providing downloads as a service. Identification of these entities to the terminal can be another effective security measure because it restricts the potential list to known and generally trusted sources or entities. This role is considered essential for the first two software categories, important for the SPA class and desirable for the UA class.

Another role common to all four software categories listed above is software/content provider (SCP). These entities create and integrate the software into usable applications or provide other content/information such a media files, data bases etc used by applications already installed on the platform. Including a list of authorized content providers can be an effective security measure, especially for the UA class. However, given today's smart phones and the number and variety of applications and sources, it might be difficult to maintain such a listing. The requirement for a DAA signature on any UA is perhaps equally effective and places the burden on the content provider to obtain signed versions from the DAA. This would also imply that the distribution points would be able to identify which DAA version is needed by any given SDRD. The Forum views having the DAA signature on any downloadable software or configuration data as an effective security measure to guard against spoofing and its consequences.

5.1.1.3.3 Policy Downloads & Updates

Policies have the potential to significantly alter the behavior of an SDRD if they are corrupted or misused, thereby introducing a new set of vulnerabilities. While there may be any number of different policy types in terms of what kind of SDRD behavior they govern, they generally fall into one of two classes. One class consists of policies which are created, managed and controlled by the NOp and the other is created and controlled by entities other than the NOp. Regulatory policies are a specific example of the latter.

The former class of policies may be of a proprietary and non-standardized form although standardization would be desirable from the NOp point of view. For the regulatory policy class a standard form is highly recommended and is viewed as being essential in order to support global roaming. In either event, a malicious entity with the necessary resources could create counterfeit policies and standardization simplifies this task. Thus security measures applied to policies should include additional security provisions.

Policies managed and controlled by the NOp should be subject to all of the security provisions identified for software downloads. In addition, in order to avoid spoofing (counterfeit) or replacement of a current policy by an obsolete policy and barring the existence of other suitable measures, it is recommended that copies of the credentials of the DAA and the SD be loaded into the SDRD as part of a trusted process similar to Root Certificate. Since there may potentially be multiple DAA or SD entities applicable to an SDRD, copies of each would be needed. It is not envisioned this would be an extensive list of entities thus the credential storage and management should be practicable.

The same cannot be said to be true for Regulatory policies. There are many regulatory agencies around the globe who will want to be assured their policies are being enforced. Much like the world of PKI, which has hundreds of Certification Authorities (one or more per nation), a standardized system and means to authorize, authenticate and distribute these policies is needed. Such a system should address the issues such as cross-certification of authorizing agencies; further consideration on this topic is beyond the scope of this document. However, for regulatory policies, at a minimum the same security provisions should apply to this class as does apply to the other policy class. The Forum recommends and urges the regulatory agencies around the globe to initiate and foster global standards in this area.

5.1.1.3.4 Configuration Data, Downloads and Updates

Configuration data has similar vulnerability as found for proprietary policies; form and content has to be ascertained if SDRD behavior is to be controlled or manipulated. If known, form and content can be exploited to alter SDRD behavior in significant and potentially harmful ways. Configuration data, like radio channel operating parameters and key material may need to be created by one entity and distributed by multiple entities, thus the security mechanisms for this class are the same as for policy downloads

5.1.1.3.5 Access and Use of Platform Resources

This class of access operations is different than the preceding because it focuses on SDRD processes accessing objects internal to the system. The desired object may be a data object or some other resource of the system such as use of a security service. Access control as applicable to these types of operations is absolutely essential to maintain a secure operating environment. In this instance the Principle of Least Privilege is especially germane and requires each subject in a system be granted the most restrictive set of privileges needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use. This is a form of access control and involves the choice of operating system and other system architectural elements. For that reason, this subject will be addressed later in this document in the Security Architecture section.

5.1.2 Identification, Authentication and Non-Repudiation Services

Access control functions discussed in the previous section are predicated on the ability of the SDRD to identify a user or entity (e.g. Network server) and to have some degree of confidence that the user/entity is who they claim to be. As noted in the extract from Table 4 below, identification is necessary for the users, their devices (e.g., the SDR), network devices such as servers, as well as organizational entities such as software providers, network operators and the entire range of service/content providers accessible via wireless devices.

Identification, Authentication and Non-repudiation Services for:	
o Users	o Software content providers
o User Devices	o Network Operators
o Network Devices	o Service Providers

In simple systems users identify themselves with a user log-on name and enter a PIN or password. A PIN/password sometimes (e.g., typical cell phone lockout code) is the only thing entered and serves both as an identifier and “authenticator”. At somewhat higher levels of security, digital devices containing electronic credentials can be used. Among these are included RFID devices, smartcards and other forms of digital token devices (e.g., a USB thumb drive). In very high security applications, ergonometric information, such a fingerprints, handprints or retinal scans, can serve as the identification information and the user may or may not be required to enter a pin/password. However, these forms of identification don’t apply to the myriad of other types of entities who need to access SDRD's resources internally, by external connections, via the internet or the air interface.

As we have discussed in the preceding section (Access Control), there are various entities that must have access to the SDRD for various download operations and system management functions such as uploading security. These entities must be able to identify themselves to the SDRD and in some cases to each other; they must also be able to produce electronic credentials which can be verified as being authentic. For digital forms of identification, the current technology involves the use of digital certificates which have been prepared, issued and digitally signed with a secure digital signature by a Certification Authority (CA) or a designated sub-authority. This technology uses an asymmetric encryption technology known as Public Key Encryption (PKE). There are user certificates as well as device certificates and both are needed to support the security functions of an SDRD.

For instance, when a security log is uploaded through the network, the device prepares and transmits the log, not the user. The user may not be aware that such an event is occurring. Thus the device's certificate associated with the SDRD would be used as part of the identification process. In another example, the user might wish to perform some transaction requiring positive identification to the other entity. The user's certificate would be used as part of the identification and authentication process and the user should be required to enter the PIN to allow his private key to be used as part of the process. This latter aspect ensures the user is aware that his certificate is being used for this purpose.

This sidebar provides a brief summary of the technology and how it is used to provide the identification and authentication functions.

Required for each PKE Certificate issued to a user, a user device or other entity is a Public Key and a password/PIN encrypted Private Key. The public key is embedded into the certificate contents and is thus integrity protected by the secure digital signature placed on the certificate by the issuing authority. The public and private keys are mathematically related in that information encrypted in the private key can be decrypted by the public key and vice versa. A signature generated using the private key of the originator allows the recipient to be sure that the document came for the originator and that it has not been altered in transit across the network. This of course assumes that the private key or the PIN which protects the private key has not been compromised.

The digital signature forms the basis of authentication. To be sure that the sender is who they claim to be, the recipient can authenticate the certificate of the originator since it must have been signed by a CA or designated sub-authority using their respective private keys. This can be accomplished by using the Public Key of the CA to verify the signature on the certificate. This requires the authenticating device to have copies of the certificates issued to the CA and any other sub authority. The CA certificate is known as the "root" certificate since it is the anchor of "the chain of trust" in the PKE hierarchy.

When it is important to document the occurrence of an event (e.g., logging into a database containing law enforcement or perhaps sensitive health data) the device processing the request [which we shall call the Authenticating Entity (AE) can record the event along with the entire digital certificate of the Requesting Entity (RE) as well as the document and/or token signed by the RE.

The token is a piece of data (e.g., random number) produced by the AE and sent to the RE. It is signed in real time by the RE and returned to the AE which can then authenticate the signature using the RE's public key. This method prevents someone from impersonating the entity logging in because it is done at the time of the transaction and the originator of the transaction (RE) must have access to its private key.

This transaction record can then be used to counter any claim by the RE (who may attempt to repudiate the event) that the event didn't occur, or that even if it did, that the entity was not responsible. This results from the simple fact that only that entity could have signed the document/token when the transaction occurred unless his private key was stolen in some manner. This is known as a non-repudiation service and can be important in many types of transactions such as downloads into SDR devices or in data base access transactions involving the user of an SDR or other computational platform.

The use of digital PKE credentials contained in a token of some form, (e.g., RF ID) can also be used to prevent the use of stolen equipment by rendering it in-operable without the "presence" of proper credentials. This might also be used to temporarily and automatically lock out a vehicular mounted radio from local use when an officer must quickly depart from a vehicle to pursue a suspect on foot, while still allowing it to function as a repeater (assuming the radio is capable of repeater operation). In this instance loss of the RF signal response from the RFID carried by the officer would cause a temporary lockout of any local microphone usage or other local physical access to the radio until that officer or another officer returned to the vehicle.

As can be seen, compromise of a private key invalidates the identification, authentication and non-repudiation services in regard to the owner the compromised key. Thus protection of the private key is imperative. The key should be encrypted until it is needed for use; the PIN or other authentication information used to access (decrypt) the key must also be protected. If stolen, the authentication information would permit application/use of the private key by a malicious process thereby making it seem as though the malicious process were legitimate.

While it is the user's responsibility to protect the PIN until it is entered, the system/SDRD receiving the PIN must ensure the path from the point of entry of the PIN to the point of use is not subject to any monitoring in order to prevent the PIN from being stolen. This applies both to physical PIN entry via a keypad or via a wireless interface between the SDRD and an external token. If a wireless interface is used to enter the PIN, the PIN itself must be encrypted during transmission via this interface at a minimum. Furthermore, unless the token has computational ability to encrypt and decrypt keys, that would mean the PIN had been stored and transmitted in encrypted form. In such a situation, stealing the PIN while it is being transmitted and simply copying the encrypted PIN into another token would allow credential forging, unless other measures are taken. Generally speaking, the use of simple RFID technology for this purpose is not recommended because it is very susceptible to such exploits. The use of RFID devices with integral encryption/decryption capability to protect the credential data during storage and transmission is recommended.

It is also important the PIN not be retained in any internal storage after it has been used and is no longer needed by the security process. Thus the PIN and any other unencrypted key should be securely erased after they have been used.

In the case of device certificates, the PIN must be stored internally in a secure non-volatile storage module within the device and not be accessible to external users or to any software which is not authorized to access and use the information. The use of the PIN allows the private key to always be stored and protected by a central security module. Other means are required to ensure that only authorized processes are granted access to the PIN so that they may request the application of the private key to authentication and other types of security services requiring its use. For example any process which may need to request such a cryptographic service, could be provided a copy of the PIN when it is instantiated. Alternatively access control mechanisms could allow the process access to or use of the PIN when it is needed.

The use of PKE to validate electronic credentials is viewed as essential to any process which must include authentication or non-repudiation. The forum recommends the use of standardized methods and certificates (e.g., X.509) where they exist.

5.1.3 Information Integrity Service

A digital signature, besides permitting authentication of the source, also allows the recipient to verify the document has not been altered; the digital signature process involves computing a mathematical function known as a secure hash (see for example FIPS publication 180-3).

The hash is computed using the contents of the document/token. The result is signed/encrypted using the private key of the originator. Thus only the public key of the originator can be used to decrypt the value. The hash algorithm creates a mathematical result solely dependent on the data being protected and is not feasibly reproducible by any other data combination. There are a variety of hash algorithms; the US government has published a standard (FIPS Publication 180-3) which contains five different algorithms, where at least part of the basis for selection is determined by the amount of data to be protected.

Per FIPS PUB 180-3:

> *"The five hash algorithms specified in this Standard are called secure because, for a given algorithm, it is computationally infeasible*
> *1) to find a message that corresponds to a given message digest, or*
> *2) to find two different messages that produce the same message digest.*
>
> *Any change to a message will, with a very high probability, result in a different message digest. This will result in a verification failure when the secure hash algorithm is used with a digital signature algorithm or a keyed-hash message authentication algorithm."*

The keyed-hash message authentication function is addressed in a different FIPS standard (FIPS PUB 198). Per FIPS PUB 198:

> *This standard specifies an algorithm for applications requiring message authentication. Message authentication is achieved via the construction of a message authentication code (MAC). MACs based on cryptographic hash functions are known as HMACs. The purpose of a MAC is to authenticate both the source of a message and its integrity without the use of any additional mechanisms. HMACs have two functionally distinct parameters, a message input and a secret key known only to the message originator and intended receiver(s). Additional applications of keyed-hash functions include their use in*

challenge-response identification protocols for computing responses, which are a function of both a secret key and a challenge message.

As one can see the HMAC function not only provides assurance about the originator, but also about the recipient as well.

These functions provide assurances that data cannot be altered without detection, and that it is not possible to substitute any other data or document for the original. The recipient thus is able to determine if any alterations have occurred. It does not identify or otherwise indicate which element(s) in the data have been altered or if the alterations were deliberate or simply errors in transmission, so it is important to realize that integrity mechanisms are not error detection and correction mechanisms. If this latter functional set is required then separate mechanisms must be used.

The extract from Table 4 presented below summarizes examples of elements where the integrity mechanisms should be considered for application for any given SDRD

Information Integrity Services for :	
o resident user data	o Any downloadable data or software
o resident radio & network configuration Data	o Over the Air Control and configuration commands
o resident software and firmware	

The use of secure hashes in a digital signature for integrity purposes has particular application where the information/data/software is being sent as an exchange and assurance of the source of the information is needed. While there are other error detection mechanisms including simple parity, checksums and hashes, these methods do not provide any assurance the data and hash/checksum pair have not been altered.

It is important to note that the use of the identified data integrity mechanisms may be burdensome if used to protect any data or other information placed in storage. The hashing and signing verification process is computationally complex and could impact on system processing time. Usage should be limited to critical system files.

Certain encryption methods are designed to reliably detect any errors in the received cipher text data stream and can be used for some integrity purposes. Thus when confidentiality is required for data in storage, use of one of these types of algorithms (e.g., AES–CBC, AES-GCM) can be used. This type of integrity mechanism does not provide a means of data recovery if errors are detected.

Integrity measurements should be applied to a platform’s firmware, critical operational software, explicit policy statements and critical configuration data stored to ensure the information has not been tampered or altered. The ability to detect such tampering may deter insider attacks on sensitive information since it will be quickly spotted. Any software protected in this manner can be checked during a secure boot or instantiation process and while slowing the processes, would not impact real time functions. Storing this information in tamper resistant storage is also an alternative, but requires corresponding tamper detection mechanisms be implemented.

Integrity measures should also be considered for communications sent over a remote channel used to control and manage the SDRD. These measures may include a signed hash mechanism or encryption methods such as those cited above. The degree of protection implemented is of course dependent upon the threat analysis.

5.1.4 Information Security (INFOSEC) Services

The Information Security service, otherwise known as confidentiality, is often one of the first that comes to mind when one thinks about security (see extract from Table 4 below). Many of today's commercial wireless systems employ some form of privacy protection, but are limited in only protecting the air interface portion of the larger communications path. Some over the air links have exploitable weaknesses, most notable the WEP encryption initially used on Wi-Fi links which has been superseded by WAP and WAP2. Likewise the Public Safety standard known as P25 provides for confidentiality of user communications for the public safety community and there are numerous radios providing integrated communications security for armed forces around the globe.

Information Security (INFOSEC) Confidentiality Services including encryption and decryption services for	
o User communications including Network Control communications	o Configuration Data downloads
o Device Uploads to networks (e.g., Log data, configuration data)	o Software Downloads
o Policy (security, regulatory etc.) downloads	o User data Storage
	o Configuration Data Storage
	o Key Material Storage

Internet transactions typically employ a protocol known as Secure Sockets Layer (SSL) uses the same PKE algorithms and certificates discussed earlier. Other protocols rely on another form of encryption known as symmetric key encryption using algorithms such a DES, triple DES and AES among others.

In symmetric key encryption, all parties involved share a common private key. There is no public key in this system, only the private key and this key must be possessed by all parties to the communications. One presumed disadvantage of symmetric key encryption is the fact that the private key has to be distributed to all of the parties involved and it has to be done in a way that does not allow the value of the key to be compromised. In this instance PKE presents one solution which allows the private key to be securely distributed. In fact many internet transactions use this capability of PKE unbeknownst to the users.

The reason symmetric encryption is used is a result of the fact that PKE algorithms are computationally intense and time consuming, while symmetric key encryption is faster and more efficient . Using PKE to encrypt a full message or document intended for one or more recipients can be very time consuming, particularly when there are multiple recipients. This is because using PKE asymmetric encryption the message would have to be encrypted separately in the public key of each intended recipient, and each recipient would then have to decrypt its own copy of the message using its private key.

Instead, by using symmetric encryption, the originator generates a temporary private symmetric key and uses it to encrypt the message. Then it encrypts just the private key in each of the recipients' public keys. It then sends the one encrypted copy of the message and the encrypted private keys for each recipient to all intended recipients as tokens attached to the message. Each recipient then decrypts its copy of the private key, and uses it to decrypt the message. For this case PKE is being used to provide a form of key management.

There are of course systems which use symmetric key encryption and pre-distribute keys to users of the system.

The use of encryption to protect the privacy communications in transit is perhaps the most commonly viewed use of this service. The service may be used to encrypt voice communications or sensitive data (e.g., law enforcement data base, medical, legal etc.) or to protect the intellectual property. For some information, encryption is necessary not only while it is being transmitted but also when it is in storage. For example, commercial users may wish to store passwords and other personal/financial data on their communications device while public safety user devices may contain sensitive operational data.

It is important to emphasize true end to end encryption is not always used in the commercial telecommunications world. If the data is truly sensitive and needs to be protected, it should be encrypted only at the source and decrypted only by the intended recipient (end to end encryption). Furthermore, if the data needs to be protected in storage, then the entity storing the data may encrypt it using either a locally or centrally generated (symmetric encryption) secret key. These keys can then be managed and protected by the Key Management Service described later in this document.

As we have mentioned before, the private key is itself encrypted using a password or PIN created at the same time the public/private key pair was generated and the password/PIN is a form of an encryption key. When humans are involved, it is the responsibility of the human to protect this password. However, there are many computational entities, including SDRs which use PKE, and they must possess their own digital credentials, including a PIN which permits access to the private key in the certificate.

For these devices, the manufacturer (actually the security architect) for the device must provide an answer to the question- "How is the password for the device's private key to be securely protected?" One recognized approach is to build into the device a tamper resistant secure storage area only accessible by the security processes (e.g., Key Management) when power is applied. Another, perhaps less desirable, method might require the user of the device to enter this information via the device's keypad. A third alternative might be to store a part of the password in a removable storage device kept by the user while retaining part in the device itself. Only when the two parts are placed together would access to the PIN be available.

So as we have seen, this service is employed both to support external interfaces and internal system functions and services. Standards applicable to the external interfaces will specify the specific services and associated mechanisms that apply to that interface, but they do not specify how those services are to be provided within the design and architecture of the SDRD. This is the role of the security architecture since it will specify which aspects of this service, as well as the other services, will be applied at the application level versus those that employ centralized common services.

Other examples describing methods of applying this service may be found though out this document.

5.1.5 Transmission Security (TRANSEC) Services

Transmission Security Services are a lesser known security service. It is similar to the INFOSEC service in that it is a service providing a form of confidentiality. However this confidentiality is generally intended to apply to some operational aspect of the air interface of the SDRD. The principle situations for which this service is used are listed below in the extract from Table 4.

Transmission Security (TRANSEC) Services for waveform/air interface related security functions such as:	
o Spread spectrum applications	o Cover for waveform control information
o Frequency hopping applications	o Cover for waveform data

Spread spectrum and frequency hopping are among the earlier applications of TRANSEC. Both technologies were initially developed for military applications and some implementations are being applied to commercial uses. For example Code Division Multiple Access (CDMA), originally developed by the US DoD, has been applied to commercial cellular use for almost 20 years since it allows a single wideband channel to be shared by multiple subscribers. Frequency Hopping has been applied to some household cordless phones as a privacy measure between the handset and the base unit. The first spreads a narrowband channel over a wide frequency band and the other transmits over many different, narrowband frequencies sequentially, dwelling only on each only long enough to send a small portion of the information.

In both instances, mathematical processes are applied to specified algorithms using in some instances a specific key to generate a stream of data which is then applied in a defined manner to produce a result which is used to either modulate (spread) the signal over the wideband frequency set or to rapidly change the frequency of transmission.

The applications of providing ‘cover’ for waveform control and/or data are simply use of another confidentiality service which happens to be employed at the waveform application processing level rather than by a centralized security service. These functions also possess military origins but are currently known to be used in both UMTS and GPRS applications.

Because these types of processes occur at the waveform processing application level, there is little choice as to where to use the keystream generated by the cryptographic algorithm. It must be applied at the application level. However, there are security architecture considerations concerning where and how to generate the keystream and where to store the keys used by this process. Once again the perceived threats, risk analysis and the RPSP must influence this decision, but overall system performance may also be a highly weighted factor.

5.1.6 Key and Credential Management Services (KMS)

As we saw earlier in the discussion concerning the use of PINS to protect the private keys contained in PKE certificates, the failure to protect a key or its application to a secure process can result in a security compromise. Protection of key materials is paramount from the point of origin of the key to its entry as well as during its storage and use within the SDRD. The Key and Credential Management Service (KMS) within an SDRD deals with this latter aspect of storage and use, but does not address specifically the broader distribution security issues although it may have some provisions in its design which support the distribution functions.

There a variety of different types of data serving the function of either being a key and/or a credential (See extract from Table 4 below,) and can simply be referred to as key materials. As discussed previously, disclosure or loss of these key materials would render the associated security service as ineffective. Thus the loading, handling, storage, and transfer of key material require the highest level of protective measures in order to ensure that the RPSP can be enforced.

Key and Credential Management Services for:	
o Users Certificates and private keys	o Device Certificates and private keys
o PINs, Passwords, Biometric access and other electronic credential data	o Root & Intermediate Certification Authority Certificates

In the following sections we shall discuss various aspects and considerations associated with implementing Key Management Services and the underlying support mechanisms.

5.1.6.1 Key and Random Number Generation

A cryptographic algorithm is a set of mathematical transforms that turn plain text into cipher texts and cipher text into plain text. One part of the transformation process is the key. The key is used in the cryptographic process to vary the transformation at each point in the process. Good cryptographic algorithms have very large sets with the results of each transform being unique within the mathematical bounds of the algorithm. For any given algorithm, the bounds are established by the length of the key. The bigger (more bits) the key possesses the stronger the cryptographic process. But not all key lengths are equally strong. Comparing public key cryptography key size to those of conventional secret key based cryptography illustrates this difference.

For example, a conventional cryptographic key of 80 bits has the equivalent strength of a 1024 bit key used for public key cryptography (RSA) so it is always important to understand the mathematics of the cryptographic process in determining appropriate key lengths. For those who may be interested, Wikipedia, under the topic of “asymmetric algorithm key lengths”, has additional information on this topic.

Selection of the key to be used in for a particular instance should be such that

1) It is virtually impossible to guess from by anyone else, and
2) It is never reused.

Good practices generate the key using a random number generator. For details see: http://csrc.nist.gov/groups/ST/toolkit/random_number.html

5.1.6.2 Key Management Infrastructure (KMI)

Any system which requires the use of key material in support of security services also requires a supporting key management infrastructure (KMI). The KMI is responsible for producing key material used by the components of the system and ensuring that each component of the system obtains all of the key material that it needs. As we shall see the components of the system may also be capable of creating some, but not all of the required key material.

There are many commercial companies which provide KMI services to the public and business sectors. Governments around the globe have also established their own KMI to meet their security needs. The authors of this document are not aware of any specific organizations which have as yet been established to service any specialized needs for regulatory or public safety usage since the existing commercial KMI establishment is probably viewed as being adequate to meet these needs. Regardless of whether this is true or not, the key factor in the decision whether to rely on existing commercial establishments or to require the use of a purpose built infrastructure component depends on the degree to which the entity providing the commercial KMI can be trusted to protect the security critical components of the Key Material in a manner which is consistent with the Organizational Security Policy. In the case of PKE certificates, this relates to the PINs which encrypt the Private Keys, and the Private Keys themselves as well as the manner in which this material is safeguarded in the process of delivering the material from the point of origin to the point of use. This decision will be predicated on the mission of the using organization and the nature and magnitude of the threats which this organization will face.

Public Key Cryptography is only one method of providing the required services. Some algorithms require the use of symmetric keys, and security policy or practical design considerations would usually require that these keys must be pre-placed in each using device. In this case the keys must be generated at one point and distributed to the devices. In a large system involving multiple devices and one to many keys per device, a centralized system which includes key generation components may be an effective solution. In smaller systems, one of the devices may provide the key generation functions and initiate the distribution to the others.

5.1.6.3 Key Material Distribution and Receipt

This topic, like that in the preceding section deals with matters which are primarily external to the SDRD. None-the-less, some discussion of this topic is necessary since it ultimately has impact on the security architecture and design of the SDRD.

There are fundamentally two classes of key material distribution. The first class involves ways to distribute key material in plain text (unencrypted) and the second involves distribution of encrypted keys. A technical term for the second method is known as key wrapping.

Key wrapping uses specialized algorithms (and modes) suitable for the encryption of keys considering such factors as the relatively short length, entropic characteristics of the data and considerations regarding the systems which may generate them. Key wrapping algorithms typically use symmetric key encryption which employs the same key to encrypt and decrypt the information. Alternatively, an asymmetric encryption/ decryption process which uses two different mathematically related keys may be selected. There are a number of specialized security considerations when selecting a proper algorithm. Because it is beyond the scope of this document, further research is recommended to properly understand the threat environment including the forms of attacks which may be employed before making a decision. This will allow for the selection of an appropriate form and methodology applicable to the SDRD's operational and threat environments.

A key used to wrap another key is known as a Key Encryption Key (KEK). In the case of symmetric key wrapping, the originator and the recipients of the key material must all possess the same KEK and therefore must be distributed to all SDRDs ahead of time. This distribution may be a physical process which transports and loads the key into each recipient or it might employ an electronic process which encrypts the symmetric KEK with another key using asymmetric key wrapping. Of course, this latter process requires that the receiving unit have the appropriate decryption key which would be provided in an associated PKE certificate which also must be distributed ahead of time. So in either case, a key distribution system always seems to be presented with a "chicken and egg" situation until the initial keying has occurred.

If there is a likelihood of an SDRD being lost or stolen and it could be used in a harmful manner to the system, then it may be important to consider providing each SDRD a different KEK. In this case, if any key or radio is compromised, then new key material can be distributed throughout the system and easily exclude the lost/stolen device.

The use of asymmetric wrapping supports this notion. The example below illustrates this concept more clearly:

- **Example:**

  A key distribution message may need to update a significant amount of key material for one or more SDRDs in a system. Since there is considerable amount of data involved, a symmetric key encryption process is most efficient. To distribute this KEK, the key generation system component then encrypts the KEK using the public key of each intended recipient and send these key tokens to each SDRD prior to distributing the set of wrapped keys.

Of course distribution of any key material need not be an overly complicated matter. Public safety organization's handheld and sometime vehicular radios are cycled on a daily basis by being returned to some facility such as a police station or an ambulance dispatch center. Before the radios are issued for their next use cycle, they could be keyed with relevant key material, including KEKs. Including KEKs allows them to be updated while in use in the field via over-the-air-rekeying methods. For other applications, such as military tactical radios, similar methods can be employed. Fixed infrastructure radios would have to be keyed when installed and provided the means to be updated by wired network infrastructure or over the air. Consideration is also needed to deal with instances where the keys become corrupted due to hardware failures or other causes.

Cryptographic devices need to be initialized with an initial set of key material; key material needs to be updated periodically due either to key compromise or key expiration.

In some instances where accountability is necessary, the KMI and Key Distribution process may require an assurance the keys have been delivered and an acknowledgement receipt of some form may be required. This may either be an external accounting process or inherent to the key exchange process. Whatever process is used, considerations should be given to ensure the integrity of the process and to eliminate/reduce spoofing attempts.

The Key Management Infrastructure, for any given system, should be designed to meet the operational needs and constraints of the communication system supported and the SDRDs in the system provide the corresponding KMI services supporting the SDRD's end of the KMI and key distribution process.

5.1.6.4 Key Material Identification and Expiration

The core security functions of an SDRD must ensure keys are properly and unambiguously identified; the processes requiring key based security service can specify the proper key to use for any given security service mechanism. Furthermore, since key material has a limited time use governed by several security factors, the SDRD must have the ability to ensure that keys will not be used beyond their useful life.

5.1.6.4.1 Key Material Identification

Because a SDRD may require multiple forms of key material, the associated KMI and/or distribution process will need to support a means of identifying each individual key material item, its intended use (i.e. function and other usage information such as which radio net or channel it is used for) and for which SDRD(s) it is intended. Additionally, there may be multiple keys with the same use and target SDRD; the identification method must also be able to distinguish among these different keys which provide the same functional use.

For example, a patrolman might only need a single traffic encryption key since operations may be restricted to a single channel. A police captain may have this same key but also a key used for executive level communications on either the same or a different radio channel as is used by the patrolman. In this example, the key types and usages are the same but their purpose is different. The COMSEC service needs to be able to identify, store and use each key properly and when necessary, allow the user to identify which is needed for the current operational use.

Some systems employ key tags or labels to provide identification with the tags being created by the key generation source. Other systems may add the labels at the device through operator/user actions. Yet another may have designated selectable storage locations into which the appropriate key is loaded and the storage location is associated with a particular usage by design. These all may be acceptable methods of providing the key identification function.

Key material may be used at the waveform level or by centralized security functions. These processes are likely to require the requesting process identify the specific key to be used. The identifiers used must be unique to the SDRD processing environment and may either be centrally generated by the KMI or the SDRD may create and assign a locally unique identifier used only for internal processes.

When any local identifier is used, the process requesting the usage of such a key must have a way to discover the identifier as well as the other functionality and usage information associated with a key. However this activity is implemented, appropriate security measures should be used to prevent malicious processes from altering any identification or functional usage information or any associations between the two such that the wrong key ends up being used for a process which it was not intended. Integrity measures previously described can be applied to each key, various labels and identifiers to ensure that no such errors can occur.

5.1.6.4.2 Key Material Expiration

Any key, including those associated with PKE have a limited lifetime of use known as key expiration. This applies to symmetric keys, PKE user certificate keys, device certificate keys and root certificate keys. Certificates (conforming to X.509) have explicit expiration and the key expiration is generally considered to be this same date. This practice reduces the risk of compromise as well as providing protective measures helping to protect the confidentiality of the information from a long term, cryptanalytic attack. For this reason, it is essential that the key's useful lifetime be much shorter than the expected time for cryptanalysis. The key length must be long enough to reduce the risk probability of compromise due to cryptanalytic techniques. The validity period for a key pair may also depend on the circumstances in which the key is used. The more frequently it is used and/or thus the more information it is used to protect increases the cryptanalysis risk probability.

As a consequence and as a practical matter, the appropriate key size is determined by the validity period considered together with the value of the information being protected by the key and the estimated strength/resources of an expected attacker and constraints of the cryptographic algorithm which may only support a limited set of key lengths. In a certificate, the expiration date of a key is typically the same as the expiration date of the certificate, though it need not be.

A key may be retained for some period of time after the expiration date in order to decrypt information sent prior to the expiration of the key. This would of course only apply to the device which originally owned the key. After a key has expired, no cryptographic process or service should be allowed to use it to initiate a new security process since other devices will not accept the expired certificate or the key. Control and management of key expiration is thus an important part of the SDRD KMS.

Because keys and certificates can expire, the KMI, and perhaps the SDRD KMS, will need to include the ability to anticipate expiration of key material and provide replenishment prior to the expiration date. For example, new keys could be created and pushed down by the KMI to a SDRD. Alternatively, the SDRD's KMS might generate a replenishment request prior to the expiration date so that new material can be provided by the KMI.

5.1.6.5 Key Material Storage and Protection

Key material, regardless of what its application function is within the security services, must be provided protected storage. The degree of protection afforded and the methods or means used to provide the protection depends on several factors:

1. Where is the key used within the security architecture and can it be provided protected storage at its point of use?
2. Where is the key stored when not being used?
3. How does the key get from its storage point to its point of use?
4. Does the key have any required or inherent protection mechanisms?

The answer to these questions is part of the process of defining the Security Architecture, Security Policy and explicit system design requirements. Once the answers are known, it is possible to determine specific mechanisms required to protect the integrity and confidentiality of the keys as well as ensure proper access to the key material.

While key wrapping algorithms can be used to protect the confidentiality and integrity of key material either in distribution or in storage, wrapping does not provide any capability to control or restrict access to the key, only integrity and confidentiality. If wrapping is used, the key must be unwrapped and checked for integrity and source authenticated each time it is used to ensure that it is the proper key associated with its label and that no errors (intentional or otherwise) have occurred while in storage or during the unwrap process.

Even if integrity and confidentiality can be provided, controlling access to key material is still a concern related to the SDRD's threat environment. Unrestricted access could allow a malicious process to erase or substitute keys. If the key were simply erased then a successful denial of service attack has occurred; a substitution allows an attacker access to material protected with the key and placed in an accessible area (over the air or on shared storage). Neither of these attacks is desirable.

Access to the actual unwrapped key so that it can be used in a cryptographic process must be carefully guarded. Failure to control access to the unwrapped key could allow it to be used improperly (e.g., for the wrong purpose) or read by malicious code for purposes of impersonation/spoofing or falsification of data.

If the point of storage and the point of use are the same then access control may be of lesser concern so long as the method of storage protection prevents access to the key except by the designated application. For example, the use of a dedicated processor (with its own data memory) to provide centralized security services meets this criteria. Alternatively memory protection mechanisms provided by a memory management unit (MMU) under the control of memory partitioning operating system could also satisfy the need. An FPGA can dedicate gates to specialized and protected storage, accessible only to the application. Tradeoffs made during the definition of the SDRD security architecture will determine the best choice for any given SDRD.

If physical access to specific key material is restricted, it may be possible to satisfy the confidentiality and integrity measures in a simpler way. The key material could be stored in an unencrypted form and integrity might involve either simple parity checks or checksums. Anti-tamper mechanisms might be a design consideration if tamper is a concern in the threat environment to protect unencrypted keys.

Some systems may require the use of a centralized key storage managed by a dedicated Key Management Function (KMF) within the SDRD. Such a KMF might provide protected key material storage for an associated centralized set of cryptographic based services. It might provide the same type of service for keys used at application level based security services taking place in other processing spaces that may not have the same degree of protection. In this latter instance the key material may pass from the centralized KMF to the applications through spaces that may have a significantly lower level of protection unless the key material is otherwise safeguarded in some manner

In either case, keys must be delivered from the KMF to the requesting services; transfers require the same functional security needs but the degree of protection will be different because of different threat environments. Both delivery methods require integrity features and means to assure that the desired end processes have received their respective key, but in the centralized process, key confidentiality may not be a concern, while it certainly will be a concern when delivering it to a process through an unprotected space.

Depending upon the security architecture choices, the transfer process might occur over a data bus protected by access control mechanisms which limits access to only authorized processes. Alternatively a dedicated key material distribution bus may be provided accessible only to those entities authorized to receive key material. Another alternative might be to establish a secure (encrypted) communications channel via system buses between the KMF and the using process. These examples serve to illustrate just a few of any number of possibilities. Ultimately, the SDRD security architecture must resolve all of the design and security policy requirement issues to achieve a cost effective solution.

5.1.6.6 Key Material Erasure/Zeroization

Earlier we discussed the topic of key material expiration. When key materials expire, they must be destroyed because if they persist in the system, it elevates their potential to be compromised. Any key material comprise allows any information protected by the key(s) to be compromised as well. Key compromise may also occur as a result of malicious practices such as key logging programs which capture PINs entered via a keypad. Access to such a PIN allows impersonation and spoofing and other exploits to occur. When it is recognized that any key has been compromised, it is imperative that all instances of that key be destroyed to prevent further use. PKI methods employ a compromised key list (CKL) and a Certificate Revocation List (CRL) to aid in the distribution of information about such instances. Military radios with integrated cryptographic capabilities include a “zeroize-all” control and some have a selective key zeroize function. The first is a safeguard to prevent hostile entities from capturing a radio with active keys and the second allows a user to remove a compromised or expired key. Newer systems are developing an over the air zeroization capability intended to target a lost or stolen radio.

Keys can also be compromised by failure of internal processes to erase keys after they have been used in some application space security process. Failure to erase (or prevent unauthorized access to) the storage or memory locations containing active key material can certainly result in a compromised key. Thus, any application which uses or accesses key material must include appropriate provisions to ensure key material erasure

For many SDRD applications, simple erasure of expired key material is adequate. At the extreme end, higher assurance key erasure methods will be needed if there is a risk that a stolen radio could be used by an adversary whose technical resources might allow key recovery. These methods employ multiple write and read operations to further obscure any residual information which might reveal a key or portion thereof. For any SDRD requiring such methods, it will likely also require external physical controls such as the “zeroize” control mentioned above, in order to ensure that any and all sensitive data is erased if a designated physical threat environment exists.

5.1.7 Platform Resource Security Management Services

With the advent of software reconfigurable radio devices, the need for a new security service class has arisen. Examples of the types of services that are needed are illustrated in the extract from Table 4 below. Let’s examine this class of service and how it applies to an SDRD.

Platform Resource Security Management Services for :	
o Memory Management Enforcement	o Platform Software Configuration Management • Radio Platform Operating Environment • Radio Platform Applications

5.1.7.1 Memory Management Enforcement

Memory management as an element of the system security architecture for process separation is discussed later in Section 7.3.1. When using an operating system (OS) secure or otherwise, the OS manages the real time allocation and use of memory via memory management processes and/or a memory management unit (MMU) function if available. In current technology the MMU is usually an integral part of the CPU integrated circuit device. If the OS is secure or trusted and supports partitioning (see Section 9 for more discussion of this topic) as well as providing support mechanisms for the Least Privilege Principle, it provides many of the important security management functions related to memory management. There are however other functions which may be applicable to an SDRD depending upon the threat environment.

We have all perhaps heard of so called "buffer overflow" attacks. According to Wikipedia the term is defined as follows:

In computer security and programming, a **buffer overflow**, or **buffer overrun**, is an anomaly where a process stores data in a buffer outside the memory the programmer set aside for it. The extra data overwrites adjacent memory, which may contain other data, including program variables and program flow control data. This may result in erratic program behavior, including memory access errors, incorrect results, program termination (a crash), or a breach of system security.

Not all memory can necessarily or efficiently be managed by a CPU, and not all CPUs or DSPs possess these capabilities. There are several possible security mechanisms which may be considered to deter the threat of these types of attacks.

One mechanism involves monitoring the buffer for the overflow condition using dedicated hardware (e.g., an FPGA based "monitor"). When the overflow event is detected, the "monitor" could cause either a hard or soft reboot of the CPU (or software partition) involved, or perhaps a vectored interrupt to either trusted ROM based code or another OS managed software partition which could then restore the legitimate processes and attempt to determine the cause of the overflow. Depending up the SDRD application these methods may or may not be either allowable or practical options.

Consideration should also be given, when permitted by the memory technology being used, to define data buffer space separately from code space. If this can be done, data space or specific portions there-of could be execute protected as well as write protected, and software code space could be write protected to prevent unauthorized changes. In fact it may be feasible to design an FPGA based MMU function which can allow establishing multiple read protect, write protect and/or execution protected boundaries. With such boundaries established then the normally prohibited operations for a given block of memory could be unlocked for use only after other appropriate security processed (access control, authentication etc.) have occurred and thereby prevent these types of exploits as well as other memory based attacks.

Under the control of a trusted OS or a centralized security module, such an implementation could be a powerful security safeguard and enforcement tool. This type of execute protect and read/write control can also be used to manage the secure instantiation of software as well as regulate and control access to the file system even when the main OS is not fully secure or trusted. Even for platforms which employ a trusted OS, such an external MMU can be used to provide another level of protection by preventing access to sensitive data (e.g., private keys) used at the application levels for either confidentiality or integrity purposes.

There may also be commercially available software solutions suitable for larger SDRD platforms, but these are generally designed for a more significant computer-like environment than is typically available in an SDRD. This situation could however change as SDRD technology becomes more prevalent since there may be financial incentives to develop this class of applications for SDRDs.

5.1.7.2 Platform Software Configuration Management

Earlier during the discussions in section 5.1.1.3.2 it was stated that version control was recommended to be another mechanism provided by platform security services management. In particular this recommendation is made for the RPOE and RPA classes of software. There are security and other operational reasons why this is recommended.

Given the rapidity which SDRDs can evolve in both hardware and software configuration throughout an SDRD model’s lifetime, a mismatch between allowable software versions for a given hardware variant could result in mis-operation or failure of the device if the wrong software version were used. Likewise hostile entities could very well attempt to intentionally load obsolete or a mislabeled code version with the intent of instigating a denial of service attack.

The use of an explicit downloadable security policy which defines the allowable range of versions for each downloadable software package can be an effective tool to allow the platform to determine if a version which may otherwise pass integrity and authentication checks is allowed to be loaded onto the platform. Even when these packages come from an authorized source the wrong package may be sent because of human error, or perhaps because the hostile source has an “insider” in the organization. Thus version control is a highly recommended tool.

If the SDRDs operational environment is such that physical access by a hostile entity to an SDRD platform is feasible, the application of version control during application start-up/instantiation may be an important security mechanism to ensure that the unit has not been tampered in some way. In such a case a tamper resistant security function could maintain a protected record of “the current version” which could then be used as a comparison reference during the secure boot or application instantiation process.

Note that this is recommended only for the RPOE and RPA software classes since these are the only ones that can be effectively managed by the platform using an explicit machine interpretable security policy. For the remaining two classes (User Applications and Service Provider applications), if and when version control is deemed necessary, then, rather than allowing unrestricted direct downloads from the respective websites, it is recommended that only approved versions be downloadable from the Network Operator's website. In this way the NOp's website could present for download only those versions which are compatible with the user's specific device configuration.

5.1.8 Logging, Auditing and Security Alarm Services

Auditing is a security service which records events to permit subsequent analysis of the events (see extract from Table 4 below). From a security perspective its primary function is manage a security event log to be able to detect when the RPSP has been violated and a compromise has occurred. This log as well as others can also be used to conduct behavioral studies.

Logging, Auditing and Security Alarm Services for:	
o Usage logs	o Non-repudiation logs
o Security Event logs	o Security Related Alarm services
o Cognitive/DSA Operations logs	o Audit log preparation

Some of the events which may be recorded relate to security such as user log-ins (who and when they log in or log out), others events may relate to radio usage (start time, duration and channel used), radio management (e.g., download events, configuration changes). For a CR a record of cognitive operations could be maintained, which might include recording of location, channel usage and parameters which effect cognitive decisions. Several of these logs may have also legal implications, thus the security (privacy), authenticity and integrity/accuracy of the log records is paramount because of accountability and non-repudiation concerns.

Auditing is considered a security service because the integrity and accuracy of the events logged must be maintained and protected from accidental or intentional attempts to alter the log, and the log record can be used to detect violations of the RPSP. Furthermore, access to the log needs to be restricted to authorized individuals, for user privacy as well security concerns and perhaps even legal concerns as mentioned previously. This is where the separation of duties aspect of access control is paramount, since no individual who can access the log or cause the log to be erased should be able to possess any other capability which could result in any form of security compromise.

Because of limited storage for many, if not most, wireless devices, frequent uploading of the log to a central repository would be warranted. For public safety (e.g.) law enforcement uses, this might occur at the end of the work shift when the radio is returned for battery charging and any required updates. For other types an over the air upload process might be employed.

For any of these uploads off-line encryption can be used to encrypt the information before it is transmitted. There are several methods which can be used to determine the key to be used to encrypt the log for transmission purposes. If PKE is used, the public key of the device controlling the central repository could be used to encrypt the log and only that entity would be able to decrypt the log. However as we noted earlier this could be a time and CPU resource consuming activity. The originating node could also generate a key for use as a symmetric encryption key and then transmit this key as a PKE encrypted token using the public key of the receiving central repository. Of course this assumes the SDRD has good random number generator process and a pre-placed copy of the repository's public key. A third alternative is to employ a mutual key creation exchange process to generate a key for symmetric key encryption of the log contents. This could occur via any of the device interfaces and only the repository device and originating devices would have access to this key.

The wireless device would also be required to provide a secure digital signature and integrity hash for the log contents prior to uploading. This validates the source and the integrity of the log contents and should be maintained along with the original log content in the central repository as long as the log data is retained. That may then be used to provide non-repudiation services.

Of course, the request for an upload or erasure of a log by any entity is itself a recordable event but not in the log which is being uploaded. That entry occurs in the next log sequence.

Depending upon the specific type of SDRD it might also be relevant to encrypt the audit log records while in local storage in the wireless device to ensure that privacy is not compromised should the radio be lost or stolen. For this purpose the device can use a locally generated symmetric key or its own public key although the former is likely to be the most efficient, but the log could be rendered useless if the self-generated key is damaged or lost in some manner.

For an SDR there may be value in being able to configure via a data or policy download, which events are to be recorded in the audit log and the frequency or times when the log is to be uploaded. Being able to select the specific events allows not only for controlling storage needs and uploading frequency, but it also permits focused study of events with subsequent off-line analysis, or tailoring the log to the specific application for which the radio is being used.

A final consideration for logging is in recognition of the possibility that the storage allocated for the log can become consumed. What then happens to the next events which must be logged? One strategy is to anticipate this event by monitoring the % of log space consumed and uploading the log via an air interface when a threshold is reached. This may not always be possible. Another strategy is to alert a central log authority and wait for it to request an upload. If the request doesn't arrive, then the log would rollover, overwriting the oldest event. This results in a circular log.

Making these determinations is an important security design consideration since failure to record events enables an attack which strives to overflow the log so that a crucial event is either not recorded, or if it were recorded that by creating enough log events, a crucial record could be overwritten. This latter aspect can be made more difficult if crucial log events are given a higher priority such that all logged events would have to be of equal priority before one of these are overwritten. As with any of the security mechanisms we have discussed, we must always consider the threat environment and conduct an appropriate risk analysis to determine the most appropriate and cost effect implementation.

Because logs can contain voluminous information and many different types of events, it is often useful for the organization which processes log to have an application which provides for "filtered" searches using event type, user, time windows and other parameters contained in the log as the filter and sort parameters for the log search.

5.1.9 Policy Enforcement and Management Security Services

Elsewhere in this document we have addressed the special vulnerability concerns about various types of policy and we have described recommendations for the application of security services during download operations. In Section 4 we address the general topic of policy based operations and described security and other design considerations for implementation of the policy enforcement mechanisms including considerations to ensure that irresolvable conflicts do not exist amongst various sources and types of policy. In this section we shall focus on recommendations for the necessary security services during storage and for ensuring that any policy is properly placed with the correct enforcement mechanism. These types of services will ensure that if a policy is needed for a certain process or by a specific application that the correct policy, whose integrity has been maintained, is delivered to the responsible enforcement mechanism. This is the "enforcement" aspect of this service.

It is important to recognize that malicious code may attempt to masquerade as the "enforcement process" as part of a spoofing effort, making it appear that a policy is being enforced when it is not. Various other threats exist.

The actual entity which enforces the policy should be a trusted process and may be either a separate application/process or it may be incorporated within a centralized security service. If the former is the case, then the centralized security services may play an assisting role in the enforcement.

On the "management" side of this service, it is essential that the authentication and integrity checked version that was received, is maintained such that unauthorized changes or even deletion or substitution of the policy occurs. Implicit to this aspect of the service is version control.

The specifics of these services should be defined by the RPSP and may be inherent to the security and other processes regarding download and file storage or they may be called out by a downloadable machine interpretable explicit policy statement. The RPSP should describe the characteristics of the Policy Management and Enforcement Service. While those portions of the platform policies such as waveform/application policy, regulatory policies, and behavioral control policy may not be part of the RPSP, nonetheless the RPSP should describe the protections afforded these elements (as noted in the extract from Table 4 below).

Policy Enforcement and Management Security Services for:

- The Platform security policy
- Waveform/application security policies
- SDRD Behavioral control (cognitive/learning radio)
- Regulatory Policies
- Other downloadable policies (e.g., Network Management, Network Security

Therefore the Policy Management and Enforcement Security Service is responsible for assuring that other security services are woven together to assure that the appropriate policy component is available to its associated enforcement engine. More specifically this service is responsible for:

1) Insuring that after successful download or installation on the platform that policy components are placed in secure storage in a manner in which their integrity can be assured. (e.g., This service could apply its own signature to the policy statement).
2) When applicable, periodically checking for expiration of policy statements using platform time resources and alerting appropriate entities of the need to update policy statements.
3) Employing integrity services on secure storage to protect both the policy and the policy enforcement engine while in storage (at rest).
4) Invoking integrity and authentication services to validate that the policy has not been modified when it is retrieved from storage.
5) Authenticating that the policy enforcement executable code (policy enforcement engine) appropriate for the particular policy is active. (This code would be instantiated as a security critical process which may be a separate process or provided as part of this service)
6) Establishing a trusted channel/protected channel between the policy enforcement engine and the storage location of the authenticated policy to allow policy retrieval.
7) Supplying the policy enforcement engine with the appropriate policy
8) Ensuring that the policy enforcement engine has been provided with all necessary platform support resources such as date and time, geo-location etc.
9) Ensuring that the specific policy enforcement engine is granted the necessary access to other platform resources required to perform its responsibilities.
10) Periodically running integrity checks on the image of the policy being enforced.

These needs are met using one or more platform security services defined in other sections.

In the end, it is the responsibility of this service to ensure that all required policies are in place and being enforced.

6 Security Critical Processes

Security critical processes are the next topic to be considered in the process of establishing the security design and architecture of an SDRD. The processes discussed in this section are those which are above and beyond the SDRD's Security Services, all of which are typically considered as security critical.

As discussed earlier in Chapter 3 the security vulnerability and threat analysis must consider many aspects of SDRD design, manufacture and operation. During this analysis process a few will be identified as being security critical. **Security Critical** processes are those processes which, if compromised, could prevent the enforcement of the platform's security policy.

Particular care should be given to the identification, design and incorporation of these security critical processes into the SDRD. The process for creating policies for the Organization (OSP), System (SSP) and Radio Platform (RPSP) will have considered the results of the vulnerability and threat analysis and should have identified those processes that are security critical. The Radio Platform Security Requirements (RPSR) should give special attention to these processes to assure that the resulting implementation can be trusted to carryout intended responsibilities while enforcing the platform's security policy. Moreover, security critical processes should be given particular focus during the entire design, development, testing and production processes to ensure they conform to the applicable policy.

While this document cannot identify and fully address every possible security critical process, there are a number of areas which are readily identifiable. Therefore, at a minimum the OSP, SSP and RPSP should include requirements for security critical processes that apply to each of the following areas in the lifecycle of the SDRD:

- Design and Development
- Manufacture/Provisioning
- SDRD Operations:
  - Secure Boot Process (for core platform services software).
  - Secure Instantiation/Execution of software:
    - Radio Platform Operating Environment
    - Radio Platform Applications
    - Service Provider Applications
    - User Applications
  - Download/Installation Software
    - Radio Platform Operating Environment
    - Radio Platform Applications
    - Service Provider Applications

        - User Applications

    - Policy Downloads

        - Radio Security Policies
        - Regulatory Policies
        - Other Policies

- SDRD Local and Remote Management Operations
- Platform Decommissioning and disposal

Brief descriptions applicable to the security critical processes for Design and Development, Manufacture, Initial provisioning and Decommission of the SDRD are provided, However, a comprehensive treatment of these topics is beyond the scope of this document.

6.1 Design and Development

There is an expression that "Security must be baked in not added in". This is a reflection on the realization that threats to an SDRD begin early in the overall lifecycle and in fact starts with the design and development processes. As stated before, the design and development process should be included as part of a thorough threat analysis, leading to the development security policies commensurate with the operational use and threat environment of the SDRD. Then the security policy must be reduced to specific platform security requirements and allocate them to the various stages of the SDRD's lifecycle. Failure to follow this process is likely to prevent not only the identification of needed security functions or security architecture features required by the target SDRD for it to enforce the platform security policy, but it may introduce vulnerabilities in the actual design and development process as described in Section 3.3.1 Thus the design and development stage includes several practices and procedures that must be considered as being security critical processes.

6.2 Manufacture/Provisioning

A goal of manufacturing is to produce an SDRD that meets the customer's needs. This includes ensuring that the security architecture and security functions designed into the device are actually present and fully functional. Failure to construct a device and load it with the software, firmware and other security related items elements in accordance with the established OSP can significantly reduce or even destroy the SDRD's ability to enforce the RPSP. As described in 3.3.2 the manufacturing process affords opportunities for threats to exploit vulnerabilities that would negatively impact or eliminate the effectiveness of the RPSP. Thus the OSP must contain procedures and checks that are intended to maintain the integrity, and preclude the exploitation of manufacturing processes.

As mentioned previously, examples of security critical processes relevant to this area are the programming of firmware, the loading/installation of Platform software (most particularly the platform operating environment), and the initial loading of device key materials such as the private keys as well as root and device certificates. Depending upon the SDRD's use, and the specific manufacturing processes employed, there may be other operations that fall into the security critical category. These should be identifiable during the Vulnerability and Threat analysis phase allowing the OSP to include appropriate security measures to counter potential threats.

6.3 Platform Operations

At all times of operations the SDRD should enforce the platform's security policy. Meeting this requirement entails several security critical processes.

6.3.1 Secure Boot

SDRDs operate in many states. At a minimum the platform begins operations from a shut down state. It must boot up the SDRD into a state that 1) enforces the platform security policy and 2) insures that platform operations transforms only to operational states that enforce the portions of the security policy that a apply to that state. The process of transiting from a shutdown state to a secure startup state is called a secure boot. In the context of this document this would typically complete when the Radio Platform Operating Environment (RPOE) has achieved a fully operational state and that all services provided by the RPOE are available for use. Specific start-up processes and associated security measures needed to ensure that the secure boot process is successful are highly dependent upon the overall architecture and functionality of any given SDRD. Notwithstanding this situation, it is reasonable to state that the process will (and should) include some or all of the measures listed in the following:

1. Validating the integrity of boot strap code.
2. Validating the integrity of all RPOE code extracted from the platform file system.
3. Activating and initializing the Operating System, including any security kernel. Operating System Kernel functions are a subset of Operating System functions. Process separation and memory management are just two possibilities, but not all of the OS services are or should be security critical processes and those that are should be a part of the OS kernel and loaded in as part of the secure boot. As indicated above the secure boot will check the integrity and authenticity of the OS kernel before loading. Providing the authentication function for the OS Kernel implies that the boot process either has the capability to perform this function or that a separate centralized security functional component is independently booted and available to provide this service.
4. Activating all required security services provided by the RPOE and testing to ensure that each is functional and capable of providing the required service.
5. Applying these security services as defined in the RPSP to ensure the integrity and validity of all platform code instantiated on the platform.
6. Testing and validating that all hardware based security mechanisms are active and functional. This includes alarms, guards and other hardware based security elements.
7. Testing and validating that all elements of the RPOE are installed and in their proper state.

8. Providing the means to detect critical failures in the boot process and the capability to initiate that enact and enforce the RPSP that dictates SDRD behavior under these conditions.

6.3.2 Secure Instantiation/Execution of software

Having achieved a successful completion of the secure boot process, the RPOE is ready to instantiate and initiate the execution of other software essential to the functioning of the SDRD, which we have previously categorized into three classes of applications. Much, if not most of this software is likely to have been obtained from other development sources and consequently may not have the same degree of security robustness as the RPOE. As such these applications may present vulnerabilities subject to exploitation. While exploitation of these vulnerabilities may not be able to be fully prevented, the SDRD can and should include measures that constrain the extent to which the exploit can affect the SDRD. Some of these measures are addressed in the next chapter discussing the Security Architecture.

There are however, security functions beyond architectural considerations that are applicable to this security critical process.

When an application needs to be instantiated there are a number of security functions which may be specified by the RPSP. The WINNF recommends the following as a minimum:

1) The application code is subjected to an integrity check immediately prior to instantiation to ensure that it has not been modified since it was last checked. The threat environment will determine whether this check is based on simple checksum mechanisms or whether stronger mechanisms (e.g., secure hash) are required.
2) For SDRDs which are at higher risk, the digital signature on the code may need to be re-authenticated as part of the previous step.
3) It may be necessary to ensure that the code package is protected in its transit from the file storage system to the processing space within the SDRD.
4) When the code is installed and initialized, an application specific security policy may need to be put into effect. This policy may dictate additional security services that are mandated before the application is allowed to begin normal operations. For example, as part of the application initialization, BIT operations employing security services may be dictated by the policy. Only after these are completed will the application be allowed access to all of the platform data and services to which it has been permitted to access.

6.3.3 Download/Installation Software

There are few distinctions between a download operation and installation of Software onto an SDRD since in both instances the end result is that an authenticated, integrity checked and authorized code package has been placed into the SDRD’s file system (or equivalent). Depending upon the type of SDRD involved and the various interfaces which it provides, downloads may occur via the air interface or via a separate physical interface on the device (e.g., USB port). The interface used will determine of course the specific sequence and type of services which may be used. For example an air interface download may employ encryption to protect the privacy of the code while it may not be deemed necessary for a direct physical interface such as USB.

Air interface downloads may also need to conform to a specified standard (e.g., those established by 3GPP[5]). However, it is important that during the creation of the OSP, SSP and RPSP, policy is created defining required operations irrespective of whether or not other standards have been employed. This is necessary to avoid vulnerabilities that may be inherent to a standardized interface and/or the protocols used to provide the download.

It is the view and recommendation of the Wireless Innovation Forum that as a minimum:

1) Integrity checks employing robust and standardized mechanisms are applied to the entire download.
2) That all code packages bear at least one digital signature and the source of the signature is authenticated.
3) That the RPSP's download/installation policies include provisions that restrict downloads to sources that are approvied or authorized in some manner.
4) That appropriate authentication protocols are employed to authenticate the entity providing the download distribution service.

These recommendations apply to all classes of software defined in this document. These are:

- Radio Platform Operating Environment
- Radio Platform Applications
- Service Provider Applications
- User Applications

6.3.4 Policy Downloads

Until such time as standards are established, the scope, form and structure of downloadable policies cannot be defined. Earlier in the document several possible classes were identified. These included radio security policies, application specific security policies, regulatory security policies, cognitive radio behavioral control policies and others such as network management related policies. Special security considerations for policy downloads have previously been discussed in the chapter dealing with vulnerabilities and threats and policy downloads have been used as examples in the discussions dealing with radio platform security services. Presented below is a summary of the Forum's recommendations concerning minimum standards for policy downloads.

It is the view and recommendation of the Wireless Innovation Forum that as a minimum:

[5] http://www.3gpp.org/

1) Integrity checks employing robust and standardized mechanisms are applied to the entire policy download.
2) That all policy packages bear at least one digital signature and that the source of the signature is authenticated.
3) That the RPSP ensure that the SDRD only accept policy sources which are defined in some manner to be authorized sources/creators of that class of policy.
4) That the RPSP's download/installation policies include provisions that restrict downloads to distribution sources that are approved or authorized in some manner.
5) That appropriate authentication protocols are employed to authenticate the entity providing the download distribution service.

6.4 SDRD Local and Remote Management Operations

Local and remote management of an SDRD platform is viewed as a security critical process because these types of operations provide the capability to significantly control the behavior of the device being managed. For example, as part of operations in an extended network environment the network management function may have the ability to download software, change policies, alter configuration data as well provide near real time management of networking operations in which the SDRD participates. Some may be loaded through a physical port but the industry trends are toward over the air re-configuration and for some component applications the use of remote management protocols such as SNMP.

Other types of radio, for example tactical radio systems, provide an HMI capability that permits operating configurations to be changed by switching between predefined configurations as well as by directly altering the configuration data parameters that control any given configuration. These local operations can be viewed as a front panel implementation of the capabilities that may be managed remotely using SNMP. Other local operations may be the direct loading of configuration data including radio operating parameters and cryptographic key material.

The procedures and processes which support local and remote operations are considered security critical.

In the preceding sections (above) we have addressed the actual download aspects of these types of operations, while earlier in the discussions of Access Control (Section 5.1.1)and Authentication (Section 5.1.2), the essential security elements associated with the local or remote access aspects were covered. These discussions made the recommendation for the use of a role based access control method as well as some recommendations on potential applications of such a method. The need for protecting user entered log-in parameters was also addressed. We mention these here simply to highlight the fact that they are security critical processes.

6.5 Platform Decommissioning and Disposal

SDRDs have an end of life. At the end of life the SDR will likely still have sensitive information, such as keys and credentials and possible other private information that belonged to the user. If we were to examine a sampling of the myriad of cell phones and smart phones that are disposed annually it would not be surprising to find only a small percentage of which would have had this information erased.

Whether or not disposal of an SDRD represents a security critical process is very much dependent on the views of the stakeholders. One advantage of the application of OSP, SSP and RPSP principles to any given radio system is that consideration for stakeholder interests is inherent. When SDRDs are used in sensitive applications such as public safety or military radios, evidence that the "decommissioning" aspect has been well considered is readily evident by the availability of dedicated controls to cause total erasure of sensitive information. In these cases the RPSP is not the governing policy, since it is the user/owner of the device that must activate these controls. Thus it is an element of either the OSP or the SSP which defines this process. Commercial cell phones certainly provide the users with the capability to erase individual elements of the data but the process can be tedious and many users fail to employ this capability. What perhaps may be needed is for commercial operators to provide a service to their users which would allow the service provider to erase all sensitive data within the phone when it is replaced.

For military and public safety radios, the process of erasing sensitive data is likely to be a security critical process.

7 Security Architecture

The preceding chapters have prepared the way to address the topic of the SDRD Security Architecture. There are many possible images that may come to mind when one thinks about security. One is a bank with its vault containing safe deposit boxes, thick vault doors and security guards. Another is a prison with its high walls, armed guards and cell blocks within which prisoners are isolated from each another and the rest of the prison facilities. Lastly we can envisage a highly sensitive military facility with electrified and barbed wire fences, guard dogs, armed sentries, all protecting buildings with separate rooms which in turn may be protected by electronic locks. In some instances access to a room might require a minimum of two individuals to be present whenever the room is occupied to ensure that unauthorized actions or communications do not occur.

Typically all of these example systems would include sensors designed to detect unauthorized entry into a protected space and alarms that trigger when security is compromised in some manner. Protected areas might also include video surveillance so that security personnel could visually monitor highly sensitive areas. And of course, logs would be created that document entry into restricted spaces and other security related events.

These examples all illustrate several broadly recognized security architecture concepts:

- Isolation/Separation
- Restricted Access and Access Control
- Information Flow Control
- Active/Passive Surveillance
- Defense in Depth
- Audit logs

The Security Architecture for an SDRD defines security measures and features which are analogous to these examples and we shall explore each of these security architecture concepts as well as several others shortly. However, at this point it is appropriate to clarify just what is meant by the term Security Architecture as it is used in this document.

At the highest level, the Security Architecture of an SDRD can be viewed as the framework and foundation upon which all of the security functions, services and mechanisms of a system or platform are implemented. It defines and constrains the design of the system/platform from a security perspective and must allow for the proper selection and implementation of security services and mechanisms in conjunction with all of the other platform functional requirements. As such it determines the means by which security policy is enforced throughout all aspects of the SDRD design. Consequently the security architecture must be integral to the overall system/platform architecture and design. For reference simplification we shall refer to the SDRD's security architecture as the **Radio Platform Security Architecture (RPSA)**.

7.1 Objectives and Purpose of the Radio Platform Security Architecture

Radio Platform Security Policy (RPSP) governs and restrains the platform's RPSA. Stated more strongly, RPSP elements that impact communication and processing must be enforced by the RPSA. The RPSA enables the system design to conform to the requirements and constraints called out by the RPSP and therefore defines the design of the system from a security perspective and determines not only how the security architectural tools, services and mechanisms are used but also it drives their selection, implementation and interaction with each other and the rest of the system.

Practical constraints, such as cost, power consumption, size and weight considerations, impact the choices and decisions leading to the RPSA definition. The RPSP governs the degree of protection while the radio platform functional requirements govern the platforms usage and functions. The RPSA must support both sets of requirements. The RPSA must be such that the security objectives are met while still meeting the intended functional usage of the platform. It must ensure that the platform is capable of providing the intended services to the users and to the overall system within which it resides, all bounded by the target cost of the platform. These constraints force a balance between security enforcement, system functions, environmental threats against platform costs and usability. Added security can increase platform development and production costs and has the potential to constrain intended usage if not carefully balanced; some necessary security measures are seldom liked by users. A completely secure system that delivers useful services may be viewed as impractical if not impossible; the best that can be achieved is to build in cost effective security services that address the most serious threats as indicated by the risk assessment.

Trade studies on the usage and placement of services and supporting security tools on the platform are used to create the balance between the multiple and sometimes conflicting objectives. For example, a required function may be to use Secure Socket Layer (SSL) to support applications such as web browsers. If this functionality is governed by the security policy (as it should be), a choice must be made deciding if the functionality will be supported at the application level or some by a centralized security service located within an isolated and secure processing space. At the application level, protecting sensitive information such as session keys, encryption/decryption and authentication functions can be more difficult if not impossible. Thus the security architect must ask if a function is within a security domain as defined by the policy and if so, what are the various security architectural tools available to secure this function and then assess the outcomes of various cost, performance and risk tradeoffs.

As the platform development transits through the various design stages, processes and procedures should be in place to ensure compliance with the RPSA. To assist in this maintenance, the results of the trade studies and design analyses conducted during development of the RPSA should be documented as a set of Radio Platform Security Design Requirements (RPSDRs). The purpose of the RPSDRs is to express how the design is to implement the RPSA security architecture and enforce the RPSP. These requirements can be tracked throughout the SDRD design and form the basis of assurance that the SDRD design enforces the Radio Platform Security Policy (RPSP).The RPSDR reflects the over all goals of the RPSP as a set of architectural and design requirements and is the foremost driver on the platform architecture regarding security.

7.2 Security Architecture Concepts

From our preceding examples we can see that isolation/separation, restricted access, access control and active surveillance are obvious elements of the security architecture. Less obvious perhaps are concepts of information flow control and defense in depth (where more than one security measure needs to be overcome to compromise the integrity of the system and the information it holds.). All but the latter are an outgrowth of an even more fundamental computer/information security principal known as the Principal of Least Privilege. Before exploring these and other related concepts in more detail, it is important to put them into their proper historical and technical context.

7.2.1 Background

Many if not most of the security concepts which apply to SDRDs have their genesis in the world of computer/software security. In the nineteen sixties and seventies, as the use of computers escalated in the government and military organizations, the need to provide security measures to protect the information on these systems became paramount. In this time period before the internet and world-wide-web, the threats to these systems were primarily viewed in terms of users gaining access to information to which they were not authorized. In essence, these computer systems were managing large data bases of information at different levels of sensitivity and the security measures were defined to control user access to information. There were generally characterized as data processing systems or automation information processing systems.

Other applications of computers during this period were in the area of computer based store and forward message processing and communications systems. These highly specialized computer systems were connected into global networks using both wireline and satellite communications to provide the required connectivity. These dedicated communication channels were secured using external cryptographic units to provide the require security for the communications while in transit. Ignoring the communications aspects, these message processing systems were also data base repositories for messages being prepared, transmitted or stored at their final destination. Thus, these too were similar to the other data base management and information processing system regarding access to the often highly sensitive messages.

To address the emerging need for secure computing systems, various government organizations, in particular the military services of the Department of Defense (DoD) conducted studies and sought recommendations from various sources. The first formal criteria was published as part of what became known as the "Rainbow Series" (See Appendix B for a listing of this series). This series was a set of standards and guides related to implementing computer security measures. The primary volume in the series was known as the Orange Book whose title is "Trusted Computer System Evaluation Criteria" The most recent published version of this standard is "DoD 5200.28-STD". Other volumes either defined criteria for other aspects (e.g., networking) or provided interpretation guidance on various aspects of the specified criteria.

With the growth of the internet and world-wide-web, many other different types of computer systems evolved to fill the roles that this networking environment demanded. Consequently in the 1990's the criteria contained in the Orange Book and the Red Book [*Trusted Network Interpretation]* were updated and have been superseded by the Common Criteria Evaluation and Validation Scheme (CCEVS). For background and further information, see the CCEVS web site (http://www.niap-ccevs.org/cc-scheme/).

Thus it is important to realize that the vast majority of software and computer security information generated is based upon a processing environment which, while similar in many regards to that of an SDRD, is also different in many aspects. These differences present a challenge to those involved in creating relevant security architectures for this new technology and it is a factor that those involved in defining the security architecture of an SDRD must always bear in mind. Therefore when applying these earlier security concepts and criteria, the SDRD security architect must consider the context in which the requirement originated and was stated, and then consider if and how it has subsequently evolved. Only then can the architect apply it to the current SDRD architectural context. One method that can be used to do this is to answer the following questions for each criterion being addressed.

1) How did the criterion apply in its earlier context?
2) What has changed in that context concerning application to the SDRD and how has it changed?
3) How must the criterion be applied now to adapt to the changes?

In the remainder of this section we shall introduce a number of fundamental principles and concepts that are relevant when defining an appropriate RPSA for any given SDRD.

7.2.2 Least Privilege Principle

The Least Privilege Principal (LPP) is the perhaps the primary fundamental idea that is the basis for two of our earlier identified concepts:

- Isolation/Separation, and
- Restricted Access and Access Control

The Orange Book defines the Least Privilege Principal as follows:

> *"**Least Privilege** - This principle requires that each subject in a system be granted the most restrictive set of privileges (or lowest clearance) needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use."*

In the context of the Orange Book the terms **Subject** and **Object** as used above are defined as follows:

*"**Subject** - An **active** entity, generally in the form of a person, process, or device that causes information to flow among objects or changes the system state. Technically, a process/domain pair."*

*"**Object** - A **passive** entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are: records, blocks, pages, segments, files, directories, directory trees, and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes, etc."*

A web-search for this principle will provide a source of contemporary discussions of this topic as it applies to many current applications. These are useful sources and may help to answer the three questions raised above (See 7.2.1)

The least privilege principle requires that each **Subject** in a system be granted access to an **Object** under conditions of the most restrictive set of privileges needed for the performance of authorized tasks. Thus granting read and write privileges to a subject than only needs to read the contents of the object would be a violation of the principal. As we can see the term "subject" is broadly interpreted to mean any user, executable process or application, or device (internal or external), while the "object" in today's parlance can be considered to apply to other active processes, devices, processing resources (e.g., memory), services, files and other types of data objects such as keys, configuration data etc., as well as hardware objects such as processors and interfaces. Practical solutions to this principle also require the separation and isolation of subjects and objects. Furthermore, the implications of varying levels of access imply that the objects themselves must be isolated from objects at a different level.

An example can perhaps illustrate the notion. For example, there is no need for a web browser to access a password file; even if the file is encrypted, unless the web browser is validating a user's entry. This concept also extends to the level of privilege at which a process is running in systems which employ two or more levels of privilege. There are at least two levels of privilege in systems which employ an operating system (OS) that has only a defined subset of the OS capable of access at the root level. This subset is often referred to as the "kernel". Some systems, which have to deal with multiple levels of security, may have several privileged states, each linked to a security level.

It is readily understandable how the application of this principle is intended to limit the damage that can result from accident, error, or malicious/ unauthorized use. From our web browser example; if the browser were to retains an open file pointer to a password file; and if it were to crash (an all too common event), the password file may become corrupt or unavailable to other processes. Similarly faults in non-security applications and services can cascade to critical functions if proper isolation mechanisms are not in place. For example; the use of protected or partitioned memory and restriction of access are mechanisms that can prevent a fault or malicious actions from spreading (A well defined memory container can prevent a hostile buffer overflow from running over into adjacent memory containing critical operating system functionality).

Implementing the concept of least privilege can be a challenge; particularly the tasks of identifying the minimal set of access and privileges needed for any given process can be tedious and some might say impossible to fully define. But depending upon the resultant security architecture it may not be necessary to fully define each and every instance for each and every process or subject. For example, in today's SDRDs user applications downloaded from internet sites may contain viruses and Trojans. If the SDRD can allocate a distinct and isolated processing environment for each application then any failure or corruption is confined to that environment.

It is also perhaps evident how the security architecture elements from our earlier examples (see inset below) derive from this principle since they all support this principle in some way. Even Defense in Depth is a consequence of design considerations intended to limit the extent of potential damage. (e.g., granting the browser access to only a copy of the password file would ensure that the original is uncorrupted).

- **Isolation/Separation**
- **Restricted Access and Access Control**
- **Information Flow Control**
- **Active/Passive Surveillance**
- **Defense in Depth**
- **Audit Logs**

Relevant applications of this principle will be addressed later in this chapter.

7.2.3 The Reference Monitor Concept and COMPUSEC Guards

The Reference Monitor concept relates to our earlier examples concerning access control, Information flow control and, to a lesser extent, active/passive surveillance.

The concept of a "Reference Monitor" is another that derives from early computer security design investigations. The Orange Book defines the Reference Monitor (RM) concept as:

> *"An access control concept that refers to an abstract machine that mediates all accesses to objects by subjects."*

The terms subject and object are as previously defined in Orange Book context.

This concept was created as result of a US Air Force initiated study conducted in 1972 by James P. Anderson & Co. *[Anderson, J. P. Computer Security Technology Planning Study, ESD-TR-73- 51, vol. I, ESD/AFSC, Hanscom AFB, Bedford, Mass., October 1972 (NTIS AD-758 206).]*

In that report, the concept of "a **reference monitor**" was introduced. The reference monitor concept was found to be an essential element of any system that would provide multilevel secure computing facilities and controls and is the basis for much of the criteria and principles present in the Orange Book.

The Anderson report goes on to define the *reference validation mechanism* as "*an implementation of the reference monitor concept . . . that validates each reference to data or programs by any user (program) against a list of authorized types of reference for that user*."

There are three essential properties that a Reference Monitor must possess in order for it to be effective: It then listed the three design requirements that must be met by a reference validation mechanism:

1. "It must be tamper proof."
2. "It must always be invoked." (i.e. it should not be possible to bypass the RM)
3. "It must be small enough to be subject to analysis and tests, the completeness of which can be assured."

A secure operating system kernel is perhaps the most obvious example of a reference monitor.

As we have seen Reference monitors are software (logical) based processes. For the purposes of this document we shall refer to the term Computer Security (COMPUSEC) Guards, or more simply as just "Guards", when the reference monitor function mechanism includes a non-abstract (e.g., hardware based) element as an essential component of its implementation. Thus electronic locks controlled by ergonometric sensors such as retinal scanners can be considered as a "guard". A memory management unit (MMU) is another example of a guard mechanism. In both cases, there are software components managing the hardware but the hardware is an essential component to this "non-abstract machine" reference monitor.

7.2.4 Trusted Computing Base

The Orange book defines a Trusted Computing Base (TCB) as

> *"The totality of protection mechanisms within a computer system-including hardware, firmware, and software-the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a trusted computing base to correctly enforce a security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (e.g., a user's clearance) related to the security policy."*

Once again one can see the original context as having emphasis on "user access". Nonetheless to the security architect it should be apparent how this concept can be extended to the Security Architecture components within an SDRD. Furthermore, depending upon the primary functionality of the SDRD and the system architecture needed to support that functionality, it may be that a given SDRD might have more than one TCB within its design, particularly when there are multiple processing environments using several processors.

For example, a base station radio might have a CPU and associated software to accommodate reconfigurability as well as remote control and management of the SDRD. A second CPU environment might be managing baseband interfaces used by backhaul links processing user traffic, while yet a third processing environment employing GPPs, DSP s and/or FPGAs is employed to process the air interface modulation, demodulation and lower levels of the air interface stack. Within such a distributed control environment a distributed TCB may be a preferable implementation.

The Orange Book notion of a TCB should not be confused to be the same term as that defined by the Trusted Computing Group although there may be many similarities. In the Orange book context the term security kernel has special significance since it is directly tied to the Reference Monitor concept:

> ***"Security Kernel*** *- The hardware, firmware, and software elements of a Trusted Computing Base that implement the reference monitor concept. It must mediate all accesses, be protected from modification, and be verifiable as correct."*

The notion is again derived from the Andersen Report. As explained in the Orange Book "The *Anderson Report described the security kernel as "that combination of hardware and software which implements the reference monitor concept." In this vein, it will be noted that the security kernel must support the three reference monitor requirement*s".

7.2.5 Inter-Process Communication Channel Security Concepts

Any SDRD will possess a variety of internal communication channels by which processes in different processing environments may communicate and over which user information and internal data may flow. These are the known and defined information flow paths within the SDRD. These may provide critical flows of information within a platform in need of internal protection mechanism, or they may be communication between processes and external entities. As such some of these perform SDRD platform configuration and control functions, while others are simple physical internal transport mechanisms for information. The information may represent user communications, download paths for data, user applications, and even platform software updates. The activation of certain channels may require the modification of the behavior of the platform to ensure the channel's integrity and confidentiality. Mitigation of today's SDRD threats requires that information flows within the SDRD be managed and controlled to constrain or prevent the spread of malicious code insertion on internal communication channels. There are several concepts used to describe and characteristics these channels from a security perspective.

7.2.5.1 A Trusted path

The notion of a trusted path has its origins once again in the Orange Book. In that context the trusted path was defined as "*A mechanism by which a **person at a terminal** can communicate directly with the Trusted Computing Base. This mechanism can only be activated **by the person** or the Trusted Computing Base and cannot be imitated by untrusted software.*" This notion continues to the present time with the use of the "Ctrl-Alt-Del" three key method to initiate computer log-in as a mechanism to activate the trusted path. Unfortunately, without a secure OS and associated secure operating environment to go along with it, it is remains a purely notional concept. However, with SDRD's that are designed to protect highly sensitive communications, the concept of a trusted path must be extended beyond the bounds of human involvement and the limitations of human interaction with a system. Thus for purposes of this document we shall define a Trusted Path as follows

> TRUSTED PATH: A mechanism by which a **person using a user interface or an internal process within the SDRD** can communicate directly with the Trusted Computing Base. This mechanism can only be activated **by the person** or the Trusted Computing Base and cannot be imitated or initiated by untrusted software.

Note that this definition does not permit the "process" to activate the trusted path, only the user or a process within the trusted computing base. This in essence places any process which is not a part of the TCB at a different privilege level than users. This is not a result of placing more trust in the users; it is a matter of necessity to support the notion of user initiated log-in. Given today's technology available for use in an SDRD, even the user log-in initiation could be subject to additional restrictions.

The intent of the Trusted Path is to create a hardware path used to transport information flows from one point to another that is protected from access by any other process except the requesting user/process and the TCB. Implicit to this notion is that the design includes intrusion detection techniques and monitoring as part of anti-tamper mechanisms. A separate port on a device used to load keys directly into an integrated cryptographic unit is an example of a trusted path as should be the path by which users enter PINS or passwords as part of their log-in process. Another example of use for a trusted path might be to transfer keys from internal secure storage to the cryptographic unit or key stream from a cryptographic unit to a TRANSEC process.

A trusted path is usually always physically present but may be enabled and disabled by a trusted control mechanism (e.g., guard) accessible only by the TCB or by user action such as connecting the key loading device or the CTRL-ALT-DEL key combination.

7.2.5.2 Trusted Channel

A Trusted channel is one in which information flows from one trusted entity to another trusted entity and is secured in some manner. It may involve the mutual authentication of the trusted entities at each end of the channel and provides for the transfer of information with high integrity and confidentiality. The confidentiality need not be cryptographically based but could be.

For example consider the circumstances of our earlier example where an SDRD has multiple TCBs, one in each of several processing environments. Communication paths between the TCBs could be via some form of a physically common shared system bus that is used for many purposes. Exchanges between the TCBs could occur by the creation of a trusted channel, which in this instance probably would involve encryption for security.

7.2.5.3 A Protected channel

A Protected Channel is similar to the trusted channel (and path) but it need not involve communications either with or between TCBs. One example, involves the use of an operating system that has a separation kernel (the ability to securely set up and maintain logically separate processing spaces). This type of OS can usually support one or more implementation options for passing information between separate partitions, neither of which necessarily contains any trusted components, and the OS kernel operation (which becomes part of the TCB) ensures that only the designated processes have access to the information.

Another example might be transferring data via an internal bus between processes in physically different processing environments. In this instance, the two processes could mutually authenticate each other, and transfer the information in encrypted form using mutually generated key material. The important aspect in the identification of this type of channel is that it is secure and that it need not involve any part of the TCB. What is not important in the definitional sense is how the security protection is obtained.

7.2.5.4 Unprotected Channel

This is the simplest channel to define in that it is neither a trusted path nor a trusted channel, nor is it a Protected Channel. This channel may be logical or it may be physical.

7.2.5.5 Illicit Communications - Covert Channels

We now delve briefly into the topic of illicit communications. This topic is considered relevant only to those SDRDs which are used to provide security in Governmental and Military communication environments or which may be used in high value financial matters.

Once again we shall refer to the Orange Book for our initial definition and explanation:

> *A **Covert Channel** is any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy. There are two types of covert channels: storage channels and timing channels. **Covert storage channels** include all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another. **Covert timing channels** include all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by the second process would provide information.* (Ed. Note: For example, Morse Code can transfer information with combinations of simple dots and dashes.)

As is evident, covert channels are used as a means of bypassing other security measures in order to compromise information that otherwise would be protected. In doing so the channel is used to transmit protected information from a protected zone to one in which there is no protection. It should also be evident that considerable resources would be involved in being able to introduce this class of exploit into the design of an SDRD and then be able to capitalize on its presence. Hence the rather narrow application focus identified above.

Developing a security architecture and design when covert channels are of concern requires specialized expertise and is beyond the scope of this document.

7.2.6 Defense in Depth

Another important security architecture concept is that of **Defense in Depth**. When warranted by the threat environment and subsequent risk assessment, the use of Defense in Depth works when the breaching of one security measure does not provide a means to facilitate subverting another. It is analogous to using electrified fences within a pair of barbed wire topped fences surrounding a building with guard dog patrols on the exterior and guards and electronic locks on the interior.

Certainly Defense in Depth may have SDRD cost implications both from a non-recurring and recurring perspective and should be applied only to the extent it is deemed to be warranted. However, one area which should be considered in all SDRD architectures are measures used to guard against buffer overflow attacks because they are so prevalent and can be very effective.

When it is employed there are some considerations that are relevant. First is to recognize from the cascading principle, that several low hurdles do not make a high hurdle. Thus implementing a cascade consisting of several weak mechanisms does not provide the safety of a single stronger mechanism. Another is to ensure that components return to default (secure) settings upon failure conditions, and wherever possible should be designed to "fail secure" rather than "fail insecure" (see fail-safe for the equivalent in safety engineering). Ideally, a secure system should require a deliberate, conscious, knowledgeable and free decision on the part of legitimate authorities in order to make it insecure.

7.2.7 Security Architecture Assurance Levels

In this document **assurance** is defined as grounds for confidence that a SDR Platform meets the Radio Platform Security Policy (RPSP) as expressed by the Radio Platform Security Design Requirement. References to “design robustness” or similar may be found throughout this document. The term “design robustness” or similar, relates in part to the concept of “defense in depth” and in part to the assurance level. A design is considered robust if significant effort and/or multiple failures are needed in order to effect a compromise.

For computer systems in general there are several methods for measuring the level of assurance, and in the public arena there is at least one that can support independent product certification. Unfortunately there are currently no recognized evaluation or certification standards that apply to the design of an SDRD. Even if certification is not a goal, these existing standards and criteria can provide valuable insight into the processes and needs for building a secure SDRD.

The term assurance levels have been derived from concepts and evaluation for security certifications initially based on the Trusted Computer System Evaluation Criteria (TSSEC) Rainbow series which we have already mentioned. That series provided for systems to be evaluated to one of four “divisions” of certification. Each division was assigned a letter (“D” lowest to “A” highest).

Division D represented "Minimal Protection "meaning that it failed to meet the criteria of any of the higher divisions. Division C had two classes (C1 and C2) while Division B had three (B1, B2, B3) and Division A , one (A1). Interested readers can learn more on this from the Orange book the last version of which was designated "DoD-5200.28-STD". These criteria define six fundamental security requirements and the certification levels specify the extent to which the evaluated system could be trusted to conform to this set of requirements.

When the TCSEC was replaced by the Common Criteria Evaluation and Validation Scheme (CCEVS), a new set of terms were created known as Evaluated Assurance Levels (EALs). The CCEVS has seven levels of evaluation driven by requirements based on a Protection Profile (PP) for a given Information Technology (IT) product. The CCEVS website has a listing of US Government approved protection profiles as well as several profiles that are under development. The Common Criteria Portal website (see link on next page) has a list that reflects additional accepted profiles. There are currently no profiles which are applicable to commercial SDRDs. Each protection profile is targeted for a specific EAL and a specific IT technology. Evaluation Assurance Levels (EAL) levels 1-4 tend to be applied to already developed products with levels 5-7 usually reserved for products developed with certification in mind at the beginning their life cycle development. This latter is also a consequence of the effort required to produce and test a design to one of these levels.

The Common Criteria has become an internationally adopted International Standard (ISO/IEC 15408) by over two dozen nations and there are approved laboratories providing independent assessment and evaluation assistance (Certificate issuers) in 14 of the 26 countries. This collaboration is known as Common Criteria Recognition Arrangement (CCRA). The agreement includes recognition/acceptance of certificates issued by other member nations up through EAL-4. Above level 4, recognition may be granted but it is up to individual nations to determine whether or not the certification meets their own national standards. Additional details are available on the CCEVS website (link below) Note that many U.S. Governmental agencies have regulations that require them to purchase IT systems that conform to a specific protection profile and EAL level as well as other criteria. It is believed that member nations of the CCRA also have their respective national criteria. Also in preparation is a document defining a set of common security criteria. It is not known when this latter document will be published. Further information about the Common Criteria can be found on the NIAP CCEVS or the CCRA websites. This link (http://www.niap-ccevs.org/cc-scheme/) is to the US website and on the International CCRA (http://www.commoncriteriaportal.org/) additional information is available including links to various member nation websites (http://www.commoncriteriaportal.org/members.html). The Forum strongly recommends that readers visit these websites and review some of the protection profiles to perhaps better appreciate the security principles espoused in this document.

Thus while this criteria may not be directly applicable to an SDRD, it does establish a basis for SDRD security architects to consider the steps necessary to assess the assurance level of their design.

7.2.8 Anti-Tamper

Physical access to an SDRD allows anyone access to the underlying hardware. Depending upon the technology used within the SDRD it may be possible to alter the contents of non-volatile storage devices and significantly affect the behavior of the SDRD or gain access to data stored within. We have earlier addressed the tamper threats during the manufacturing process and how these might be countered by process controls.

After the device has been manufactured and placed into service, practically speaking there is little one can do to prevent anybody from attempting to tamper with a device for any purpose and it is unlikely that any preventative measures can reasonably be taken. However, anti-tamper concerns are not limited to prevention. Often it is sufficient to have the knowledge that an attempt was made that could have resulted in a successful tamper event. Armed with this knowledge the SDRD can be examined and when necessary replaced with a new unit thereby averting the threat. This approach is warranted when there is economic or other (e.g., safety, ant-terrorism) concerns that provide the justification. This justification should be considered as part of the security risk assessment.

7.2.9 Accountability -Auditing

Auditing is the last architectural concept which we shall address in this Section. No matter the SDRD application, maintaining a proper audit trail is essential to security. Without an audit trail or log recording security related events it may not be possible to determine if a security compromise has occurred or has been attempted, much less who, if anybody, might have been involved. Perhaps the breach was not deliberate and was the result of human error granting privileges to an individual who should not have had them. While, the audit log doesn't prevent the event from occurring it does allow some level of recovery when it is discovered and would certainly enable preventative measures to be implemented to minimize the likelihood of a similar mistake or to prevent a future attack of the same type from succeeding.

Earlier in Section 5.1.8 specific recommendations concerning the security log were presented. Several of those are worth emphasizing since the ways in which they are implemented are security architecture related.

1) The security audit log function should have the capability to specify and record any specifiable subset of the total set of events. (i.e. An authorized individual needs the ability to select specific events to be audited from the total set of defined events)
2) The log should be maintained in a way that prevents its destruction or alteration. This includes attempts to exceed log storage capacity to prevent an event from being logged or causing a recorded event from being overwritten.
3) Access to the log should be restricted to authorized individuals whose access privileges do not violate the "separation of duties" principal of access control.

7.3 Architectural Design Considerations for Security

The Security Architecture is an integral and inseparable part of the overall architecture of an SDRD. Thus it is not possible to define either independently although there are somewhat different considerations involved depending upon whether the focus is on satisfying security requirements or those requirements defining the basic SDRD architecture. The latter of course is directly related to the primary operational functions of the SDRD, while the former addresses the security specific concerns necessary to support the primary functionality within the defined threat environment as determined by the Vulnerability and Threat analysis and quantified by the Risk Assessment. It is the primary functional architecture that forms the basis upon which the security architecture is structured so long as the primary architecture does not create a situation that violates the RPSP. Such a circumstance can be avoided by considering the relevant security requirements in each stage of development of the primary functional architecture.

Generally speaking, the development of an SDRD architecture to satisfy a particular set of requirements begins with the development of set of functional block diagrams for the SDRD. This set of diagrams lays out functions first at a high level and depicts the interfaces that each functional block in the diagram must support. Then each block within that diagram is further decomposed into the various sub functions comprising the larger function. The 2nd level diagram depicts the interfaces among the various sub functions comprising the function as well as illustrating which of the sub functions support the interfaces to other functions defined in the top level diagram. Depending upon the complexity of the system and the choice of functions, as many as three or four tiers of functional diagrams may be created. Armed with the functional design, the architectural choices and selections as to how the functions will be implemented is then undertaken. This includes determinations as to what aspects will be implemented by hardware and what will be software based and these decisions are moderated by others. For example, an early set of choices involves defining the computational environment of the SDRD. This includes defining how many processors and of what types (GPP, DSP, FPGA, special purpose, etc.) will be used to satisfy the functional needs of the SDRD, and the choice of operating systems for each environment. It is absolutely essential that these functional architecture and detailed implementation designs include not only the security functions defined by the RPSDRs, but security design principals and considerations must be applied throughout these processes.

In the sections that follow, additional design considerations especially pertinent to the development of the Security architecture and design are presented. In several instances we will be examining the same design aspect from different security principle perspectives. This approach provides the architect with a better understanding of the security issues involved and leads to the best solution for any given application.

7.3.1 Isolation and Separation Considerations

Isolation/separation is arguably the most fundamental security consideration and principle applied to any security architecture development. As we have illustrated, isolation and separation are key aspects of any security architecture whether the architecture applies to a prison or an SDRD. Isolation and separation mechanisms are essential to being able to enforce the Least Privilege Principle under any and all conditions. The application of this class of mechanism can simplify the following security aspects:

- Preventing malicious processes from unauthorized access to data and other system resources.
- Preventing malicious processes from violating the integrity of (i.e., altering) protected information and processes.
- Limiting the scope of any potential damage when a process has already been corrupted.

This design area revolves around the need to isolate and separate information as a means to control and restrict access. Isolation and separation mechanisms establish the barriers which:

- Provide confidentiality, by denying/not granting access to a logical or physical path which could be used to read information belonging to another process/subject;
- Maintain integrity by denying/not granting access to a logical or physical path which could be used to modify either information belonging to another process/subject or the object/subject of interest.
- Provide service/process availability, by preventing actions that would deny/prevent a legitimate process being able to access needed services or processes.

The use of isolation and separation mechanisms as a means to enforce confidentiality, integrity and availability for security critical functions and processes allows placement of a higher degree of confidence in their operation. As a side effect, isolation and separation requires inputs of information to be highly constrained, thus minimizing the interfaces to critical resources; reducing potential malicious action, increasing uptime or availability. This is known as reducing the attack surface.

Isolation and separation can be implemented through multiple means. To understand how information can be isolated and separated, a better understanding of how it needs to exist in a system is essential and how it needs to move within the SDRD operating environment. The previous discussion concerning the functional block breakdown is useful for this purpose. It is also important consider that the SDRD operating state can affect how and where information and processes reside. For example, in the power off state, some code will reside in the ROM/PROM, but the bulk will be stored in some form on Non-volatile memory (NVM). At boot time the code and related data will have to move from the NVM to the processing space where it will be instantiated. This movement may require the use of basic security services for integrity and authentication. In this type of process the sequence of boot operations will define the dynamics of this environment and will require additional security analysis to understand the threats and vulnerabilities that may exist during such a process. This analysis may show either a simpler or a more complex threat environment than the steady state environment of an operating system. Thus the separation and isolation mechanisms may have different operational constraints under these conditions.

It is also important to consider where the highest degree of vulnerability may exist. Typically this would exist in processing environments where standard protocols may be employed and the protocols are known to be vulnerable to one or more exploits. This is typically either an air or terrestrial (e.g., backhaul) interface, and of course user applications are another known area of potential vulnerability. Thus the security architect would first consider mechanisms that would allow these processes to be constrained to their own separate processing spaces if this is a feasible option. An alternative might involve implementing specific security features that would guard against specific exploits such as those involving buffer-overflow.

Another form for consideration is separation in the time domain. As the name suggests, the separation relies on different information flows being able to share common resources, be it a processor or on a bus, a storage or interface device but not at the same time. Time separation may also only allow certain functions to occur within specified time frames or intervals. For example a firewall may allow specific users access to the internet only at certain times of day. Other examples of time isolation will be addressed in sections which follow.

As we shall see there are multiple ways available for isolating and separating information from either a software and/or hardware perspective.

7.3.1.1 Operating System Selection

Operating Systems (OS) can be a primary enforcer of security policy while also providing the required process separation and functional isolation for the platform. There are many security experts who would opine that without a secure OS there is no such thing as software security and these same experts will also state that the OS alone cannot be responsible for maintaining system security; the hardware operating environment must have complementary features to support the OS. In particular, the OS needs to have features and mechanisms which ensure its ability to protect against tampering, bypass and spoofing attacks by malicious processes.

In 1998 six individuals employed at that time by the National Security Agency, wrote a paper: "*The Inevitability of Failure: The Flawed Assumptions of Security in Modern Computing Environments*" The paper appears in the Proceedings of the 21st National Information Systems Security Conference pages 303 - 314, October 1998 and is available on the web. (http://csrc.nist.gov/nissc/1998/proceedings/paperF1.pdf). The paper, which is highly recommended, defines a set of necessary operating system features which are too often lacking in operating systems and addresses how these feature support the security needs of the computational environment. Most, if not all of the concepts and principles addressed in this document are discussed in that paper.

A new class of highly robust Real Time Operating Systems (RTOS) which provide a separation kernel are emerging. One such OS has been approved and there are others in various stages of Common Criteria Approval. These new RTOS are using the U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness (currently at Version 1.03) as the basis for their Common Criteria certification. (Details may be found at the CCEVS website: http://www.niap-ccevs.org/pp/pp_skpp_hr_v1.03/). These operating systems are being evaluated at EAL-6 and higher. The current versions are limited in application to single CPU processing cores and unfortunately cannot be applied to multi-processor core environments. For most SDRD designs this limitation should not be an impediment.

Such a method can found in operating systems supporting Multiple Independent Levels of Security (MILS) where various levels of classified processes can operate within the same operating system with assurance information will not leak. This form of isolation, while adding some overhead, helps to solve problems where security critical processes need to be separated from a user and untrusted applications. A user’s space and processing resources can be contained within a memory partition, preventing malicious code or actions from accessing outside (from the perspective on the container) memory which may contain critical processes. To be able to properly support memory based separation between a user space and all other spaces including critical security processes, this type of operating system should be selected for any GPP based platform with even moderate security requirements. The type of memory separation allows for the concurrent running of critical software alongside untrusted, downloaded from the internet, malware ridden screen savers; there is no need to worry about cross contamination. The trade off is the operating selection supporting these features is limited; competent developers may be harder to find and the development of application may take longer. Also, because the operating system adds another level of abstraction with the memory partitions, it requires at least one more level of memory abstraction. To be able to properly support memory based separation between a user space and all other spaces, including critical security processes, a partitioning operating system should be selected for any GPP based platform with even moderate security requirements.

There is also another class of RTOS providing separation kernels that are compliant with the high reliability and safety requirements for avionics applications. While perhaps not as rigorous as the latest they offer important security benefits in lower risk threat environments. It is incumbent on software security experts to carefully evaluate available candidate operating systems including those that can be used in a multi-processing core environment. The Vulnerability and Threat analysis, as quantified by the Risk Assessment will provide guidance as to what security constraints are applicable for RTOS selection. It is of course also pertinent to note that selection of the operating system will perhaps have the greatest impact on the overall security of the system and will drive many if not most of the other architectural choices, thus OS selection should be done early in the process.

There are other considerations in RTOS selection as well. Any planned use of an RTOS that supports separation/partitioning of processes, must also consider the impact of that selection on processing time. There will of course be a timing impact and performance overhead associated with switching between different partitions as the OS cleans up working storage and shifts memory boundaries etc, and this must be factored into the evaluation. Too many partitions may not be a good thing because of this. Another consideration concerns how to accommodate the flow of information/data between partitions. In the context of the Software Communication Architecture (SCA) that has been adopted and endorsed by the Forum, this communication is provided by the Middleware, which for the SCA is CORBA. CORBA effectively hides the existence of partitioning since processes need not care where a resource is located or how it is accessed. For the High Assurance RTOS applications in an SCA context, a corresponding high assurance version of CORBA (or equivalent) is needed. While none as yet have been approved, there are several vendors who are actively pursuing Common Criteria Certification although there is as yet no approved protection profile.

While the OS may support memory partitioning in terms of process separation, the OS need not be the only factor to be considered.

7.3.1.2 Memory Partitioning/separation

Some degree of memory partitioning occurs in any system. This basic partitioning is done on the basis of memory types; RAM, ROM, (E)PROM, and various forms of non-volatile storage (NVS). Typically non-volatile storage will be used to store data and program code during power off conditions, while PROM retains the boot and recovery code. Program code is typically not executed from the Non-volatile storage since it is too slow to meet performance requirements, thus the program code is moved to volatile (RAM) during the start-up/instantiation process. Thus the non-volatile storage forms the basis for the file system and as such as serves as a "data" repository. The volatile storage serves as both executable code space and as a short term data repository.

Since NVS should never need to be used when code is being executed a basic form of memory partitioning for security purposes would be to ensure that any information in NVS can never be used as executable code. This prevents malicious processes from storing "data" and through some other exploit transferring execution control to that memory space. This can be done by simply disabling the "read" control for NVS during an instruction fetch cycle. This same technique can be done with volatile storage also, but it requires that physically separate devices be used to store data and code. These techniques can virtually stop any exploit that employs techniques which cause data to be executed as code or which try to overwrite executable code space. In the case of the latter, the "write enable" control could only be enabled during the time frame that code is being transferred into the working RAM during the instantiation process by either trusted boot code or another trusted loader

When there are multiple levels of sensitive data, there are a variety of techniques beyond having the partitioning kernel serve as the reference monitor function. A trusted centralized security module/processor, allows that process to serve as a "guard"/reference monitor exercising control over memory access (read/write/execute) might be a more viable alternative in some applications.

7.3.1.3 Additional Memory Separation Considerations

Volatile Memory holds active information as it is needed by processors and I/O devices. Containment of information while it exists in volatile memory is one of the principle ways isolation/separation is enforced within a secure platform. The objective is to allow trusted and untrusted processes to be able to operate at the same time within a single processor radio without interference.

Certainly the most common way to enact memory separation is though an operating system. The OS implements various levels of isolation ranging from individual process separation as is commonly found in Linux, Windows and Apple operating system to partitioning micro kernels utilizing memory containers to separate groups of processes. For very sensitive information, physically separate memory with limited access may be required.

However, some memory in the system may need to be truly shared between trusted and untrusted processes. The allocation of residual memory can violate isolation and separation requirements unless steps are taken either by the OS or another RM like process.

When a process request access for more memory, an operating system usually delivers a pointer to the next available segment that fits the requested size. This new memory section may contain information from the process that last occupied the space. To solve this problem, the memory manager needs to be trusted to securely erase the memory before passing it on to the requesting program; this results in a higher level of security but at the cost of a higher latency (time) for providing memory requests.

7.3.1.4 Processors and Separation Issues

Processors take information from memory or I/O to enact some sort of transformation. Some processors directly interface to I/O devices, accepting streams of information for transformation and delivery. Programmers of devices such as FPGAs need to be cognizant of how the information flows through the processing elements and uses that knowledge to prevent cross over between processing flows within the processor. General Purpose Processing (GPP) devices interact with memory to process information; these devices have various high speeds, on chip caches and registers (memory) that may allow for information to leak outside of acceptable ranges. Depending on the needed level of security, some processes, when finished with a time slice or being preempted, will flush information from cache and registers. Usually it is the responsibility of the operating system to carry out this function. Certain GPPs implement hardware features, such as Memory Management Units (MMUs), to assist and help the OS in enforce memory isolation. As a practical matter Hardware MMUs are a requirement for memory partitioning kernels.

The recent introduction of multi-core processing units adds complications to the subject of isolation at the processor and memory levels. Often the different cores share resources such as cache memory and busses. It is usually the OS’s responsibility to assure the shared resources are not a source of violation of isolation and separation policy, but this type of functionality must be inherent to the OS.

Needless to say, selection of the proper processing environment, including selection of RTOS, processor technology etc. is a critical factor in the development of the Security Architecture.

7.3.2 Information Flow Control

Another important area for consideration in the architecture is determination of how the various types of information/data are passed between hardware and/or software components in the system. Earlier in the discussion of the functional decomposition of the system requirements into functional block diagrams it was stated that the diagrams needed to depict the various interfaces that need to be supported by a given functional block. What was not said at the time is that these interfaces also need to depict the functional purpose of the interface and for each different interface function that has to be supported a separate functional interface should be depicted. This is important because at some level in the functional decomposition the interfaces will diverge and terminate on specific (and different) functional components. This approach allows the designer/architect to understand the information flow needs of a system from the highest level to the lowest and to develop an approach which satisfies not only the information flow needs, but also addresses potential vulnerabilities and allows for their mitigation if not their elimination. The threat mitigation/elimination approaches will most certainly rely on the principals we have already introduced such as separation/isolation, LPP enforcement, and perhaps others.

In some instances the need to pass all types of information over a given path is unavoidable. For example, the air interface, in addition to passing user communications, may also be used for downloads of software updates to any of the four types of software identified in Section 3.1.5, policy updates, receipt of new Key Material, platform configuration data updates and any other conceivable type of information. Thus while sharing this interface and processing path is unavoidable to a certain point, the design should include provisions to separate and isolate these different information types from one another as soon as practicable.

In general, information in a system flows within defined paths, holding objects/information in transit. Examples of this can be found in network data transferring into a system and through an IP stack, in the sequence of hardware and software data flows in a boot sequence as objects are read in from storage and loaded into memory, and in the hardware and software path that user information passes for encryption/decryption or during an authentication event. An information flow can be considered as an abstract object and the nature of its movement, interaction and means of protection require consideration. Understanding the nature of this abstract object and the information behavior internally will help when deciding proper protection mechanisms.

For example let’s assume that all of the downloaded information is encrypted and must be decrypted before further processing is possible. In this example, the decryption and/or other applicable security services (e.g., integrity verification, authentication etc.), could deliver each of the different decrypted information types to interfaces designated for further processing of that type of information (this assumes of course that there is some way of identifying to which type the encrypted data belongs). Some of this downloaded information may require authentication and integrity checking (e.g., downloaded software, policies) after being decrypted. Then, in the case of downloaded software, after integrity and authentication and other security related functions the code package may be passed to the file management system for placement and storage. User communications would be passed to the appropriate user communications interface and a user downloaded application would be routed to the application’s execution environment as well as to the file system for storage. In these instances separation/isolation mechanisms should be considered to prevent the user communications path from providing access to the file management system or to the user application processing space and consequently eliminating these as potential sources of vulnerability. This illustrates a general security design principle to avoid sharing information flow paths among different types of information unless there is another mechanism which can provide the necessary separation. We shall address this latter topic in more depth shortly, but before that it is necessary to address several other security aspects related to information flow.

7.3.2.1 Direction of Information Flow

The direction of the flow of information is another factor for consideration in the development of the security architecture. When attempting to preserve confidentiality, it may be allowable to let information flow from a lower sensitivity security container to a higher one, but not the other way around; this is the premise of the Bell-LaPadula model which was modeled to protect sensitive and classified US Government Information in information processing systems. To preserve confidentiality, the model states it is allowable to let information flow from a lower classification container to a higher classification level container but not the other way around. Thus there are constraints on the direction that information may flow. Although the specific model is limited to a very restricted set of SDRD applications, the principle is applicable to practically any SDRD although the sensitivity and data classifications are likely to be very different. However, the Bell-LaPadula model was limited to confidentiality and did not address integrity concerns. It also did not consider differences between strategic and tactical information classes, e.g., the value of tactical is perishable and degrades quickly over time.

In 1977, Kenneth Biba of the Mitre Corporation developed the Biba Integrity Model, developed to address a “flaw” in the Bell LaPadula model. By allowing information to flow from a lower (and perhaps untrusted) classification level to a higher one the potential for using this flow path direction to corrupt data at the higher level existed. The unconstrained flow of information in the same direction (less sensitive to sensitive) also can be used to support the activities of covert channels by allowing a malicious process in one level to communicate with another malicious process in another. The Biba model was developed to address this weakness among other potential weaknesses. The Biba models purpose is to:

- Maintain internal and external data consistency
- Prevent data modification by unauthorized parties
- Prevent unauthorized data modification by authorized parties

This last point is highly relevant because even though a party may have authorized access to the data, that party may not be authorized to change it. This too is the essence of the LPP.

For example, in a system utilizing memory partitions, all security services could be contained in their own protected partition. So while it may be ok to allow certain information to flow from the less secure container to the security space (e.g., encrypted traffic must be decrypted) the application of the LPP per the Biba Model would prevent unrestricted information from that flow and all flow with be constrained by an appropriate security policy. This same policy would define how the information is to be processed and how it is to be dispositioned after processing. This type of policy enforcement can prevent potential corruption or replacement of data such as keys, or the insertion of viruses and malware infections as well as a host of other similar exploits. Readers should also be aware that other integrity models exist and should be studied for consideration during the development of the security architecture for an SDRD. Each model tends to bring additional nuances and aspects to light.

While the Bell-LaPadula model may have valid applicability to the class of information processing systems for which it was intended, this model only has limited applicability to the SDRD operating environment and then only to protecting the same type of classified information. SDRD's must in general violate this model. For example users must be able to control the transmitter by activating it by a key line control. This control exists on a radio or perhaps a handset. While the information being conveyed from the handset has to conform to the Bell LaPadula model, the keyline cannot. It must pass from the "sensitive data" environment, past the security services and thence to the transmitter where it will be used to enable and disable radio output. Of course this is a control signal, and as we shall discuss in the next section, user information/data needs to be appropriately isolated and separated from SDRD control and configuration information, but even this is sometimes difficult. There are numerous other examples which illustrate the practical aspects of the limited applicability of the Bell-LaPadula model.

7.3.2.2 Data and Control Information Flow

In our example above, all of the downloaded information comprised of user data or configuration data constitutes data in one form or another. However even software during the download process and while stored in the file system is still nothing but a form of data. As yet we have not addressed the information flow class that is called "control". We must consider both internal and external control information.

7.3.2.2.1 Internal Control Information

Within an SDRD these various data types must be processed before being delivered to their point of use. An SDRD has another form of information flow that is primarily, but not exclusively internal. This is information that for purposes of this document we shall define as control. This control information is used to manage and set-up the specific configuration and operation of the SDRD and may use an internal message structure to pass commands and control parameters via system busses. In some instances the "control" may be discrete electrical control signals enabling or disabling specific functions, such as the use of a push-to-talk control on a portable radio as we mentioned above, but for systems which transmit data transmitting another form of control may be required. Perhaps another way of distinguishing between data and control is that "data" is the information being transformed or moved while it is control information that instructs how data will be transformed or moved.

An architectural choice will be to determine whether or not to separate or share the information paths for the two information types, and what protection mechanisms may be needed. Sharing introduces the risk than an attacker may be able to send information on a data channel then trick the system into treating it as control information, adversely changing platform's functionality. Good security design practices separate control information from data through the use of trusted paths (e.g., a dedicated control bus), protected channels or trusted channels depending on the robustness of the design and the level of assurance needed for the information being transported.

These mechanisms may be applied only to the “data” or only to “control”, or if different security parameters are being used, (e.g., different keys) then the same process may be applied to both with the “parameters” being used as the separation factor. In fact, the application of a common security measure to both types allows them to safely transmit an untrusted process space (like the internet or an internal IP stack).

7.3.2.2.2 External Control Information

Control information arriving at the platform from external sources is subject to the SSP as well as the RPSP and may place additional requirements on the platform’s security architecture. As we have discussed earlier during downloads, one or more digital signatures may have been placed on the downloaded information in accordance with the SSP and the RPSP.

In some cases it may be difficult to separate external control information from user data. Examples of areas might involve, routing information, Quality of Service, network security policy etc. In future cognitive radio networks individual radios may be subject to some external controls regarding cognitive behavior. In some of these instances the control will flow over the same communication network paths used for user communications while in others there may exist specialized interfaces used for control. (e.g., a base station might have a separate IP address for remote control and network management purposes.). Using different digital certificates (because control originates from one or more designated devices and data from others) means separate private/public key pairs and these can provide the needed degree of separation.

Also in these instances protocols used to convey information should have built in mechanisms that create a virtual separation of control from data. One example is Simple Network Management Protocol (SNMP). SNMP is used to transmit configuration information to platforms that have the capacity to affect the function and behavior of a platform. SNMP messages flow in the same space as other IP based traffic in an IP stack. As of SNMP version 3, the messages are isolated from the other information by authentication mechanisms and other security services. Once an SNMP message is authenticated and the SNMP agent determines the sender is authorized the instruction is executed. Even though the SNMP control information is transported over the same communication channel as user data, the authentication and authorization mechanisms assist in protecting and isolate the control from user data.

7.3.2.3 Buses and I/O Ports

Buses and I/O ports are used to move information from one physical entity to another. A common use of busses and I/O is between volatile memory and some peripheral device such as hard drives and flash memory, but the transfer may also be from one I/O device to another or from one processing space to another. Ethernet switching chips have also been incorporated into a design used to provide communications paths between separate physical devices/processing environments within an SDRD. This is definitely not a recommended approach because every packet appears at every switch port and relies on the IP stack to reject packets not destined for that port. Clearly a corrupted stack process has access to all of the information passed through the switch unless each packet is individually encrypted…a potentially horrendous overhead penalty. It is therefore important to recognize that regardless of the transport mechanism any given bus or I/O port, unless appropriate safeguards are employed, can introduce vulnerability by providing an access point to external attacker or an internal corrupted/malicious process.

As indicated earlier separation of control and data information flows can mitigate these risks to some extent, but other measures may be needed, depending of course on the radios operating environment. An example will serve to illustrate.

Consider an SDRD which uses a design that has a non-IP/MAC based master bus over which all information is passed between processes and devices in the system. Thus unencrypted user communications, sensitive key material, network control information, configuration data etc. all flow over this bus at one time or another. Unless some form of control is implemented to constrain access from any given process or device to authorized interval of time a corrupted/malicious process could collect this information and then send it off the platform as though it were part of the user communications. Relatively simple controls (managed by a Reference Monitor) could control read and write access to the bus and thus assure that at no time would any unauthorized access occur. The purpose of course, is to illustrate the importance of measures that need be taken to prevent unwanted intermingling/access.

Transport over buses can also employ trusted communication components such as tagging and routing data over protected paths or by separate physical paths. Multiplexing or other time sharing techniques are also feasible and of course distinctly separate physical paths can be used to protect sensitive flows of information.

7.3.2.4 Input Output Flows

For a system to be useful, information must at least flow in or out, usually both. The way information flows within external I/O elements and interacts with shared busses and memory should be carefully scrutinized as indicated above. Attempting to dissect how this information flows at a physical layer in shared hardware devices and understanding the mechanisms used for separation can be challenging. In many cases, such separation mechanisms were built with data efficiency in mind and not security.

DMA interfaces such as “Firewire” can allow external connections to have direct access to a platform’s active memory, bypassing operating systems and memory partitions[6]. Other considerations involve examining how the external device I/O data flows on internal system buses and the impact this will have on separating information flows such as control and data. Awareness and avoidance is the principle guideline in this case.

Though useful in meeting performance goals, allowing external ports or devices to connect to the platform through such I/O types may allow devices with access from untrusted environments access to trusted functions and memory, circumventing monitoring functions that might

[6] A. Boileau (aka Metlstorm) “*Hit By A Bus: Physical Access Attacks with Firewire*” Security-Assessment.com, Ruxcon 2006

otherwise provide the ability to detect/deny the access. Such devices can read and write to memory without processes in a GPP being aware of the event, circumventing most operating system protection mechanisms.

Direct Memory Access (DMA) allows for a very efficient means for transferring information to and from memory to I/O devices. While efficient, it may open avenues of attack and should be carefully considered before it is allowed.

7.3.3 Simplicity verses Complexity

There is a balance which must be played when considering the need to lock down a system versus allowing a system to be usable. A system with minimal security is easier to use from a user's perspective, at least until something goes wrong. Usage and placement of security mechanisms and services is a balance of complexity verses simplicity. A small, simple, security design and implementation is easier to understand and verify as compared to one that is more feature rich and complex. A more complex design may allow for greater flexibility but may increase the potential for misconfiguration. A simpler security design may not allow users to interact with the platform in a desired manner. For assurance purposes, the implementation needs a high degree of examination for verification; this complexity adds development time and cost. The vulnerability analysis and risk assessment are the primary considerations for determining the proper mix and balance between these factors for any given SDRD.

7.3.4 Hardware versus Software Implementation Choices

The security architecture not only defines the software processing environment but must also determine the hardware environment both of which in combination enforce the radio platform security policy (RPSP) and providing the required security design, services and mechanisms.

Hardware architecture and design choices can help or harm the overall security of a system. Hardware choices will help to protect the assets of the system; this can range from tamper resistant storage, how data is routed in a radio and can simplify software security designs by providing and enforcing separation, ensuring availability and a stronger basis for trust. Hardware separation, using physically separate paths, separate processing elements or specialized security hardware can simplify a platform's software security design. It can also add a level of reliability and robustness to the design as hardware security features tends to be more difficult to bypass; hardware separation allows for a higher level of assurance but usually at a higher cost. Software can also be used to provide separation with a greater level of complexity to the software design, but may be more flexible. It also can be less costly as compared to a hardware solution as the number of units produced increases. Just about any security objective provided by software can also be provided by hardware. Providing clear requirements based on a security architecture will guide hardware designers in making proper choices.

Dedicated processors can be used to provide functionality for highly critical functions. A common example in tactical radios for the past 20 years or so is the usage of a separate cryptographic processor. This processor is inserted permanently in the user information flow path so that user data cannot be inadvertently bypassed. In this way, no information can flow in or out of the system without passing through the cryptographic processor unless intentionally bypassed through user action. Early versions of this technology were ASICs customized to support specific applications and cryptographic methods. Within the last decade, software programmable devices allowing the development and insertion of new cryptographic algorithms in fielded devices have emerged as the predominate technology. There are advantages and disadvantages to this approach.

By using a programmable cryptographic processor, it is possible to include code to support any or all of the other security services that may be needed in an SDRD besides cryptography. The functionally constrained interfaces of this class of devices can provide high assurance as well as design robustness for these services, minimizing if not eliminating the possibilities of corrupting or compromising these services. The downside is of course additional hardware and software and the associated recurring and non-recurring development costs. ASIC designs are still highly relevant, and in some instances ASICs are needed to achieve the required efficiency or speed of operations. ASICs do offer the advantage of not being vulnerable to attacks but, once cast into silicon they are costly to change. Of course there are hybrid approaches using a combination of ASICs, processors and even FPGAs which allow meeting performance needs and still retaining the necessary flexibility for change. In fact, recent developments in the FPGA field have yielded a patented approach which can achieve high assurance certifications for process separation.

It thus devolves to the architect to balance security risks, cost, performance, reliability, vulnerability and availability factors to make the best selection for the given application.

7.3.5 Object Labeling

Inherent to the original concept of the Least Privilege Principle is the ability to be able to identify the subjects and objects in a system and define their security attributes. As stated earlier LPP is a form of access control that is applied to internal processes (subjects) and determines which objects (data, or other processes) to which they may access. It also applies to user access control and was a way of determining not only the level of classification of the data objects to which the subject was granted access, but depending upon the attribute set, it could also be applied to "need to know" aspects within a classification level. In today's application, LPP goes much beyond its original intended application; the actual attributes that are to be used and their associated values would be defined by the RPSP. These security attributes would then enable the reference monitor function to enforce the security policy applicable to these attributes. However, a means of associating these attributes with the corresponding subjects and objects is needed.

A standard method involves labeling, either implicitly or explicitly each object/subject. The label might contain the attributes or it may serve as a pointer to a list of attributes. In any case there would be a one to one correspondence between the label and the attribute set. In addition to the label/attribute set would be the definition of the rules concerning how the subject's attributes related to the request objects attributes. In some instances there may need to be a perfect correspondence (no differences allowed) while for others the object's attributes may have to have either perfect correspondence or a complete subset (intersections not allowed). The specific rules should be defined as part of the RPSP.

As the reader may imagine, there are many forms and techniques by which labeling can be accomplished but they generally fall into one of two classes; explicit or implicit labels. As we shall see the method of object labeling differs based on the methods used for access control to enforce separation.

7.3.5.1 Explicit Labels

Explicit labels are associated with defined sets of attributes for both subject and object and the labels and their associated attributes are interpreted and enforced by a reference monitor function according to the rules stipulated in the RPSP.

Explicit labels allow for individual object access by subjects as long as there is a corresponding permission granted by the defined attributes. The association can be based on an object's ownership, security classification, sensitivity level, functional characteristics or any other parameters needed to support the security policies of the architecture. Labels and their attributes assigned to internal software processes are usually permanent and determined during the design. These are typically the objects to which they need to access. In this instance the “object” may be whatever data exists in a particular data structure rather than having to have specific rights granted for each data element.

However there are classes of labels which may need to be assigned to specific subjects as part of an operational process. For example the assignment of individuals to specific roles, where each role has a defined set of privileges is one example of the need for such a capability, while user input of private data into a cellular handset might be another. Others may involve network functional associations. Added thought must be given in deciding how and by what means and by whom it is possible to assign labels to the various forms of subject and objects, particularly when the subjects and objects involved do not have a permanently assigned label.

While file systems may provide mechanisms for adding labels with the attributes associated with the individual files, because of the function of the labels from an LPP view, special security considerations are needed in the architectural design of this capability. Explicit labels associated with a specific subject or object must have two critical security properties enforced.

1) The integrity of the specific association between the label and the subject/object to which it belongs must always be maintained.
2) The integrity of the label and its parameters must always be maintained.

Stated another way, it must not be possible to change the association of a particular label with its object nor shall it be possible to alter the attributes except by an authorized process behaving in accordance with the RPSP. If a malicious process could associate the label with a different subject, then that subject could be given access to an object which it should not have, or alternatively, it may be denied access to objects to which it should have access. Similarly if the contents of a label can be changed without authorization, then similar violations of security policy can occur. Thus the labels and their associated parameters need to be bound to the subjects and objects to which they are assigned in a way that protects the integrity of the labels, their parameters and the binding. This is a non-trivial architectural design matter, particularly in devices such as an SDRD which has real time performance parameters with which it must comply.

In information management systems it is feasible to use techniques such as digital signatures to bind attributes and labels to subjects and objects. This allows a reference monitor function to validate the signature and the integrity of the binding and the attributes. The overhead associated with this technique is probably inconsequential for this type of application. However, such an approach could seriously impact the performance of a real time system so other methods need to be considered for any process that has real-time implications. For other processes that do not have this limitation then explicit labels are a viable choice. The security architect then must consider the following in developing his approach:

1) How does the Reference monitor enforce the LPP with the approach under consideration?
2) When is the enforcement applied? On startup? During installation? Upon each access?
3) What is the need for a label? Why is this item being labeled; what purpose does it serve?
4) What is the label going to represent (level, groups, classification, roles)?
5) Who is going to decide the object's label representation level? Who within the system will have the privileges to set initial label values?
6) How is the label going to be associated with the object (database lookup, tied to the file, etc)?
7) What services will provide the label binding to the object and when?
8) How is this binding association going to be protected?
9) How is modification of the label going to be controlled?
10) How is the label going to be accessed for verification and matching?

Other considerations may also apply for a given SDRD application. For example, an object in one state might have an explicit label, while in a different state its labeling may be purely implicit. We shall discuss examples of this in the next section.

7.3.5.2 Implicit labels

Subjects and objects are implicitly labeled when they exist in a condition where all subjects and objects in the same condition have identical attributes and the RPSP can be enforced without the need for explicit labeling. Many objects within a system are labeled by default and may not need specialized labeling. An implicit label for a subject or object is the inherited label of the container holding said object; an object inherits the label and attributes of its container.

The simplest conceptual example involves the use of a dedicated processor to provide security functions and the code package for the processor has been subjected to rigorous analysis and testing. It therefore is a trusted process and its own reference monitor. Another example could result from the use of a trusted OS that has a separation kernel. In such an instance it would be possible to place different types of subjects in various partitions (the containers), and various objects in other partitions (other containers), such that subjects with the same attributes are located in the same partition and likewise objects with the same parameters in the same partition. In this example the subjects and objects inherit the properties of the partitions into which they are placed. Of course one could also place both subjects and objects with corresponding attributes in the same partition. There are valid architectural and security considerations for both approaches.

Regardless, the act of placing any subject or object into a partition for such purposes must fall within the responsibility of a reference monitor function. Furthermore this also serves as an example of converting an explicitly labeled entity into an implicitly labeled entity. Let's clarify this point. During instantiation of a set of software code, the RTOS will have to associate the various processes and data with a specific partition using whatever parameters that are associated with this process. The association of the parameters to the process constitutes the explicit labeling of that process but that explicit labeling does not persist with the object in the instantiated form in its designated partition, but the label is implicitly associated with the object because the partition inherits the properties of the label, and passes it down to the subjects and objects contained there-in.

Keeping subjects and their allowed objects in a common partition is very efficient from a processing perspective since each and every access does not have to be validated, nor does binding and associated integrity checking apply each time. There is always the overhead of course of the RTOS switching from one partition to another in order to give each its allocated processing time. This is an overhead itself and provides a practical limit on the number of partitions any given processor design can support.

We have addressed how an object can have its label removed when it moves into a partition, but we must also consider if the reverse operation is permissible, as it may be for some objects (e.g., a security audit log is extracted from the SDRD for use and analysis.). (Note: it seems unlikely that "subjects" would be 'moved' for any purpose.) Thus if an object is moved outside of the container, it may need an explicit label attached unless the attributes of the object to which it is being moved are the same. Again any such movement is the responsibility of a reference monitor type function.

Appendix A Document Definitions

Administrator/Owner (role) -The Administrator/Owner role corresponds to the individual or an entity that not only has owner rights to the device but also possesses the unilateral control over which of the set of authorized or permitted communication services are enabled by the device.

Application, Radio Platform (RPA) - A **Radio Platform Application** is the software that controls the behavior of the radio to make it behave as a radio. It is sometime known as or called a waveform.

Application, Service Provider (SPA) - A **Service Provider Application** is software used to support services provided by a service provider for the user of the radio. This might include special messaging services, video services, etc.

Application, User (UA) – A **User Application** is a software application that Users download that reside and run on an SDRD using the SDRD computational resources.

Assets - Assets are hardware, firmware, software, information or intangibles such as reputation that is resident on or associated with an SDRD that has value to the owner of the asset. Assets can also be services or information that may be provided, leased or sold to the user of an SDRD.

Assurance - The grounds for confidence that a SDRD security features and architecture accurately mediate and enforce the Radio Platform Security Policy as expressed by the Radio Platform Security Design Requirement

Assurance level - Assurance level is a measure of assurance based on the quality and robustness requirements needed to assure that sufficient protection mechanisms are properly implemented to address the commensurate threats as identified by the risk assessment. (See for example definitions for Common Criteria levels)

Back Door: A vulnerability intentionally left in system software which may be used for later exploitation.

Computer Security (COMPUSEC) Guard - A COMPUSEC Guard or more simply as just “Guard”. The term “Guard” applies when the **Reference Monitor** function mechanism includes a non-abstract (e.g., hardware based) element as an essential component of its implementation

Covert Channel - A Covert Channel is any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy. There are two types of covert channels: storage channels and timing channels. Orange Book

Covert storage channels include all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another. Orange Book

Covert timing channels include all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by the second process would provide information. Orange Book

Defense in depth – Defense in depth is a design principal the employs more than one (usually multiple) security measure that must be overcome to compromise the integrity of the system and the information it holds.

Design robustness - Design robustness or similar, relates in part to the concept of "defense in depth" and in part to the assurance level of a design. A design is considered robust if significant effort and/or multiple failures are needed in order to effect a compromise.

Discretionary access control (**DAC**) - In the Orange book, Discretionary Access Control is defined as "a means of restricting access to objects based on the identity of subjects and/or groups to which they belong.

Download Authorization Authority (DAA) [Role] -In the context of this document, the DAA is an entity with the authority to approve the download of software/firmware of a designated type. This role is defined solely within the context of enforcing security policy on the SDRD.

Evaluated Assurance Level - In the Common criteria one of seven possible levels for which a given product is certified to comply with an identified and approved protection profile.

Explicit Radio Platform Security Policy (ERPSP) - An Explicit Radio Platform Security Policy **is a** subset of the Radio Platform Security Policy that can be expressed in digital form and that can be interpreted and acted upon by an SDRD policy enforcement mechanism.

Least Privilege Principle – The Orange book definition of this term states that this principle requires that each subject in a system be granted the most restrictive set of privileges (or lowest clearance) needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use. The terms "Subject:" and "Object" also have specific Orange Book meanings.

Malicious Code: Software which if able to be installed and instantiated on a system performs functions that are in violation of the security policy.

Mandatory Access Control (MAC) - The Orange Book defines mandatory access control as "*a means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (i.e., clearance) of subjects to access information of such sensitivity*". However, in non-DoD computer applications the term has broadened to encompass "*a type of access control by which the operating system constrains the ability of a subject or initiator to access or generally perform some sort of operation on an object or target.*

Manufacturer - A manufacturer is the entity that produces, assembles and integrates the hardware and software comprising the device and assumes a liability for the performance of the device.

Object – As defined in the Orange Book, an Object is a **passive** entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are: records, blocks, pages, segments, files, directories, directory trees, and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes, etc."

Organizational Security Policy (OSP) - The Organizational Security Policy is the broadest and most general of the security policies. The OSP is a formal statement of the rules by which people who are given access to an organization's technology and information assets must abide. It is a document intended to guide humans rather than equipment. The OSP is enforced by individuals in the organization. It is not directly part of an SDR but it does include a definition of the assets of the system and the individual components that must be protected and the assurance level of protection mechanisms to be applied during the design and development phase of the system and it component parts.

Physical Access Control - In physical access control access to an object is restricted by means of physical barriers and requires that the person desiring access have the authorized capability of passing through or around the physical barriers.

Platform Configuration and Operating Data - Information which is used to configure the hardware and software the device needs proper operation. Configuration information includes information on allowed operations and operating parameters, and information related to platform security such as digital certificates used by the platform and it users. For public safety and military organizations it could include operating parameters such as definition of the specific frequency channels to be used. This information or data can control the behavior of the operating platform applications as well as specific behaviors of the SDRD

Policy: (a) A set of rules governing radio system behavior. Policies may originate from, for example regulators, manufacturers, developers, network and system operators, and system users. (b) A machine interpretable instantiation of policy as defined in (a)

Policy-Based Radio: Radio in which certain aspects of the behavior of communications systems are governed by machine-interpretable policies that are created and are modifiable by authorized entities.

Policy, Organizational Security (OSP) – The **Organizational Security Policy** is the broadest and most general of the security policies. The OSP is a formal statement of the rules by which people who are given access to an organization's technology and information assets must abide. It is a document intended to guide humans rather than equipment. The OSP is enforced by individuals in the organization. It is not directly part of an SDR but it does include a definition of the assets of the system and the individual components that must be protected and the assurance level of protection mechanisms to be applied during the design and development phase of the system and it component parts.

Policy, System Security (SSP) – The **System Security Policy** is a set of rules, requirements and practices that specify or regulate how a **system** (e.g., the networked hardware components, the SDRDs, as well as software and physical plant elements of the system) provides security services to protect resources. The SSP is therefore a component of the System Security Architecture and Design that implement the relevant aspects of the Organization Security policy. It supplies the technical goals and objectives against which the System Security Architecture and Design are evaluated. The SSP is one element of the decomposition of the OSP.

Policy, **Radio Platform Security (RPSP) -** The **Radio Platform Security Policy** is the portion of the SSP relevant to the SDRD. The RPSP is a set of rules, whose enforcement is either implicit in the design and/or explicit via machine interpretable expressions. In either case, these rules 1) Define and constrain the application of security services, and 2) Govern or restrain a system's possible actions as defined by the SSP.

Policy Distributor (Role) – The policy distributor role parallels the Software Distributor role, because it characterizes those entities which are designated as being authorized to distribute policies of a designated type to an SDRD and to components of the network in which the SDRD operates. This role is defined solely within the context of enforcing security policy on the SDRD.

Policy Issuer (Role) - This is a broad class of roles each defined by the type and the nature of the policy being issued. Examples of such policies are regulatory policies, network security policies, network management policies, as well as individual SDRD security policies. From an SDRD security policy enforcement perspective, a Policy Issuer is an entity who is authorized to issue a corresponding type of policy. There many variations possible and are of course SDRD system and network design dependent. This role is defined solely within the context of enforcing security policy on the SDRD.

A **Protected Channel** is similar to a trusted channel or trusted path but it need not involve communications either with or between Trusted Computing Bases.

Radio Platform Applications (RPA) - A **Radio Platform Application** is the software that controls the behavior of the radio to make it behave as a radio. It is sometime known as or called a waveform.

Radio platform software/firmware consists of any and all software classes that utilize the computational resources available on an SDRD.

Radio Platform Operating Environment (RPOE) - In the context of the Wireless Innovation Forum endorsed SCA [1] this software consists of the Core Framework, the operating system software, devices, drivers, middleware, and services such as a downloader and installer. It includes any other software fundamental to the operation of the radio platform with the exception of the RPA, SPA and UA classes of applications. This might include a voice Codec, and other software components (e.g., interleavers, Viterbi encoders/decoders etc.) which can be used by the RPA or any other software. Of particular import to this document is the fundamental point that the RPOE includes all of the radio security services. For a policy based radio this class would also include policy enforcement mechanisms and/or services.

Radio Platform Security Architecture (RPSA) – the Security Architecture of an SDRD can be viewed as the framework upon which all of the security functions, services and mechanisms of a system or platform are implemented. It defines and constrains the design of the system/platform from a security perspective and must allow for the proper selection and implementation of security services and mechanisms in conjunction with all of the other platform functional requirements. As such it determines the means by which security policy is enforced throughout all aspects of the SDRD design to ensure that it will enforce the RPSP.

Radio Platform Security Requirements (RPSR) - The Radio Platform Security Requirements capture the relevant aspects of the Radio Platform Security Policy as a set of requirements that guide the development, deployment, provisioning, and operation of the SDRD to assure that the SDRD RPSP will be enforced by the SDRD. These are further broken down into Radio Platform Security Design Requirements and the Radio Platform Security Operational Requirements

Radio Platform Security Design Requirements (RPSDR) are those requirements that impose constraints on the functions and the overall architecture as well as the Radio Platform Security Architecture of the SDRD.

Radio Platform Security Operational Requirements (RPSOR) are those requirements that impose constraints on the production, distribution and operation of an SDRD within the intended network environment.

Radio Platform Security Policy (RPSP) – The Radio Platform Security Policy is the portion of the SSP relevant to the SDRD. The RPSP is a set of rules, whose enforcement is either implicit in the design and/or explicit via machine interpretable expressions. In either case, these rules: 1) Define and constrain the application of security services, and 2) Govern or restrain a system's possible actions as defined by the SSP.

Reference Monitor - The Orange Book defines the **Reference Monitor (RM**) concept as “An access control concept that refers to an abstract machine that mediates all accesses to objects by subjects.” The terms “Subject:” and “Object” also have specific Orange Book meanings.

Regulator (role) -The regulator is the legal authority that assigns spectrum rights to communication service providers and establishes limits for safe operation of radio equipment for a given regulatory domain.

Risk assessment – Risk assessment is a probabilistic rating assessment of the likelihood that a hostile entity will devote the required effort (and cost) to exploit a potential or real vulnerability, combined with the probabilistic likelihood of the success of exploiting that vulnerability given the planed protective mechanisms.

Role - A role is an entity who is either a specific stakeholder or someone representing a stakeholder, and who is involved in some aspect of the use, operation, management, control, deployment, maintenance, and/or security of the device and the network in which it operates. For any given system and SDRD a role has a specific defined set of capabilities or functions associated with it.

Role Based Access Control (**RBAC) -** Role Based Access Control is sometimes referred to as role-based security. RBAC is such that the role of a user defines a set of allowed operations allocated to the role. Thus by being assigned a specific role, the user is granted permission to perform the functions associated with the role and no others. A user may have more than one role subject to the separation of duties rule.

Root Certificate – In cryptography and computer security, a root certificate is either an unsigned public key certificate or a self-signed certificate that identifies the Root Certificate Authority (CA). A root certificate is part of a public key infrastructure scheme. The most common commercial variety is based on the ITU-T X.509 standard, which normally includes a digital signature from a certificate authority (CA).

Root Certificate Authority - Digital certificates are verified using a chain of trust. The trust anchor for the digital certificate is the **Root Certificate Authority** (CA). The PKE certificate is a self-signed by the Certificate Authority. It is the most fundamental certificate in a hierarchical chain of certification authorities for a defined PKI.

Secure Boot – Secure Boot is a process which brings the radio platform from a shut down state to a defined secure state. This state is one that 1) enforces the platform security policy and 2) insures that platform operations transforms only to operational states that enforce the portions of the security policy that a apply to that state. The process of transiting from a shutdown state to a secure startup state is called a secure boot. In the context of this document this would typically complete when the Radio Platform Operating Environment (RPOE) has achieved a fully operational state and that all services provided by the RPOE are available for use.

Secure Download: A method for downloading software, firmware or other information , that provides means for protecting the information being downloaded from modification during distribution and for detection of any changes to the information that may occur and authenticating the source(s) of the information that has been downloaded as well as the entity which is providing the download service.

Security Mechanism: A function of a specific security service class**.**

Secure Memory: Memory that has hardware/firmware/software support such that execution access, as well as read and write access to the memory is restricted to specific, identified , authorized and/or trusted operations and processes.

Security policy: A security policy provides 1) constraints that govern and restrain a system's possible actions to those within the acceptable trust boundary and 2) identification of the security services that are applied to ensure the constraints are enforced.

Security Critical Process - A security critical process is one which, if compromised, could prevent the enforcement of the radio platform security policy.

Security Kernel - The hardware, firmware, and software elements of a Trusted Computing Base that implement the reference monitor concept. It must mediate all accesses, be protected from modification, and be verifiable as correct." [Orange Book]

Security Policy Enforcement Engine (SPEE) - When an SDRD supports the use of an ERPSP it will need one or more functional elements that, for purposes of this document will be identified as a **Security Policy Enforcement Engine (SPEE).** The SPEE is the mechanism that interprets the downloaded policy and implements the enforcement of the rules expressed by policy.

Service Provider Application (SPA) - **A Service Provider Application** is software used to support services provided by a service provider for the user of the radio. This might include special messaging services, video services, etc.

Software Distributor (SD) (role) - A software distributor is an specifically identified entity who has the role and is authorized to provide a distribution of software/firmware via download or direct physical contact with an SDRD interface (e.g. USB etc.).

Software/Content Provider (SCP) - The software/content provider is an entity who contributes a portion or all of the software that resides on an SDRD. In most cases it is the entity that wrote the code and/or integrated open source code with code written by it.

Stakeholder - An entity that has an asset associated with the SDRD and consequently a stake in the resulting behavior of the SDRD is termed a stakeholder.

Subject – As defined in the Orange Book, a Subject is an active entity, generally in the form of a person, process, or device that causes information to flow among objects or changes the system state. Technically, a process/domain pair.

System Security Policy (SSP) - The System Security Policy is a set of rules, requirements and practices that specify or regulate how a **system** (e.g., the networked hardware components, the SDRDs, as well as software and physical plant elements of the system) provides security services to protect resources. The SSP is therefore a component of the System Security Architecture and Design that implement the relevant aspects of the Organization Security policy. It supplies the technical goals and objectives against which the System Security Architecture and Design are evaluated. The SSP is one element of the decomposition of the OSP

Threat: Any circumstance, event or entity with the potential to cause harm or disruption to a system with the intent to exploit a vulnerability for the purpose of violating the security policy of the target.

Trusted: Always behaving in an expected manner for an intended purpose.

Trusted Computing Base (TCB) - The Orange book defines a Trusted Computing Base as "The totality of protection mechanisms within a computer system-including hardware, firmware, and software-the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a trusted computing base to correctly enforce a security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (e.g., a user's clearance) related to the security policy."

Trusted channel – A trusted channel is one in which information flows from one trusted entity to another trusted entity and is secured in some manner. The securing of the channel need not involve encryption but may be enforced by trusted separation mechanisms.

Trusted Path - A mechanism by which a **person using a user interface or an internal process within the SDRD** can communicate directly with the Trusted Computing Base. This mechanism can only be activated **by the person** or the Trusted Computing Base and cannot be imitated or initiated by untrusted software.

User (role) - The user role corresponds to the individual or entity that uses the communications device to access communication based services and has access to the communication services of the device as a minimum.

User Application (UA) – A **User Application** is a software application that Users download that reside and run on an SDRD using the SDRD computational resources.

User Data - The user's personal information and other stored in the memory of the SDRD.. Examples of such data include: credit card numbers, pins, user log-in names, home addresses, account numbers, address books, date books and other personal information.

Vulnerability – A Vulnerability is a weaknesses (which may have been intentionally included/ inserted) that exists in processes, procedures, protocols, hardware and/or software design or implementation that provide the means or opportunity for exploitation.

Waveform - A waveform, in the context of the WINNF endorsed SCA is the entire set of software that provides the functions associated with a specific air interface.

Appendix B – NSA/NCSC Rainbow Series

The information in this appendix was extracted from the following website:

http://www.fas.org/irp/nsa/rainbow.htm

Note that the Federation of American Scientists is not an U.S. Government sponsored Organization

The Rainbow Series is six-foot tall stack of books on evaluating "Trusted Computer Systems" according to the National Security Agency. The term "Rainbow Series" comes from the fact that each book cover is a different color. The main book (upon which all other expound) is referred to as the Orange Book. Its title is Trusted Computer System Evaluation Criteria [DoD 5200.28-STD] and is available on the web from a number of locations including : http://www.fas.org/irp/nsa/rainbow/std001.htm

Portions of the Rainbow Series (e.g., the Orange** book and the Red** Book) have been superseded by the Common Criteria Evaluation and Validation Scheme (CCEVS). For background and further information, see the CCEVS web site (http://www.niap-ccevs.org/cc-scheme/).

*[**ed Note – There are multiple volumes which use the colors Orange and Red. These other volumes are viewed as extensions of the source book hence the same color. In the context of this reference the Red book is –NSC TG-005 "Trusted Network Interpretation"]*

Listing of the NSA/NCSC Rainbow Series

NCSC-TG-001 [Tan Book] - A Guide to Understanding Audit in Trusted Systems [Version 2 6/01/88]

NCSC-TG-002 [Bright Blue Book] -Trusted Product Evaluation - A Guide for Vendors [Version 1 3/1/88]

NCSC-TG-003 [Orange Book] - A Guide to Understanding Discretionary Access Control in Trusted Systems [Version 1, 9/30/87]

NCSC-TG-004 [Aqua Book] -Glossary of Computer Security Terms [Version 1, 10/21/88]

NCSC-TG-005 [Red Book] -Trusted Network Interpretation [Version 1 7/31/87]

NCSC-TG-006 [Orange Book] -A Guide to Understanding Configuration management in Trusted Systems [Version 1, 3/28/88]

NCSC-TG-007 [Burgundy Book]-A Guide to Understanding Design Documentation in Trusted Systems

NCSC-TG-008 [Lavender Book] -A Guide to Understanding Trusted Distribution in Trusted Systems [Version 1 12/15/88]

NCSC-TG-009 [Venice Blue Book] -Computer Security Subsystem Interpretation of the Trusted Computer System Evaluation Criteria

NCSC-TG-010 [Teal Book] -A Guide to Understanding Security Modeling in Trusted Systems

NCSC-TG-011 [Red Book] -Trusted Network Interpretation Environments Guideline - Guidance for Applying the Trusted Network Interpretation

NCSC-TG-013 [Pink Book] - Rating Maintenance Phase Program Document [Version 2 - 01 Mar 1995]

NCSC-TG-014 [Purple Book] -Guidelines for Formal Verification Systems [4/1/89]

NCSC-TG-015 [Brown Book] - A Guide to Understanding Trusted Facility Management [6/89]

NCSC-TG-016 [Yellow-Green Book]- Writing Trusted Facility Manuals

NCSC-TG-017 [Light Blue Book] - A Guide to Understanding Identification and Authentication in Trusted Systems

NCSC-TG-018 [Light Blue Book] - A Guide to Understanding Object Reuse in Trusted Systems

NCSC-TG-019 [Blue Book] - Trusted Product Evaluation Questionnaire [Version-2 - 2 May 1992]

NCSC-TG-020A [Grey/Silver Book] -Trusted UNIX Working Group (TRUSIX) Rationale for Selecting Access Control List Features for the UNIX System

NCSC-TG-021 [Lavender/Purple Book] - Trusted Database Management System Interpretation

NCSC-TG-022 [Yellow Book] - A Guide to Understanding Trusted Recovery

NCSC-TG-023 - [NOT USED]

NCSC-TG-024 - [NOT USED]

NCSC-TG-025 [Forrest Green Book] - A Guide to Understanding Data Remanence in Automated Information Systems (Ver.2 09/91)

NCSC-TG-026 [Hot Peach Book] - A Guide to Writing the Security Features User's Guide for Trusted Systems

NCSC-TG-027 [Turquoise Book] - A Guide to Understanding Information System Security Officer Responsibilities for Automated Information Systems

NCSC-TG-028 [Violet Book]- Assessing Controlled Access Protection

NCSC-TG-029 [Blue Book] -Introduction to Certification and Accreditation (09/94)

NCSC-TG-030 [Light Pink Book] -A Guide to Understanding Covert Channel Analysis of Trusted Systems (11/93)

The NIST RAINBOW SERIES LISTING

A somewhat different listing of the series is available from the NIST website:

http://csrc.nist.gov/publications/secpubs/rainbow/

The NIST website provides links to text versions of the documents

```
std001.txt [277123 bytes] 1993-09-30 "Department Of Defense Trusted Computer System Evaluation Criteria" ("Orange Book"), 12/85 (DoD 5200.28-std) -Rainbow Series

std002.txt [60908 bytes] 1993-11-10 "Password Management Guideline" 4/12/85 (CSC-STD-002-85) - Rainbow Series

std003.txt [20194 bytes] 1993-10-04 "Computer Security Requirements" 6/25/85 (CSC-std-003-85) – Rainbow Series

std004.txt [76867 bytes] 1993-10-04 "Technical Rationale Behind CSC-std-003-85:Computer Security Requirements", 6/25/85 (CSC-std-004-85) - Rainbow Series

tg001.txt [56311 bytes] 1993-11-09 'Audit in Trusted Systems" Version 2 6/01/88 (NCSC-tg-001) - Rainbow Series

tg002.txt [52053 bytes] 1993-11-10 "Trusted Product Security Evaluation Program" 3/1/88 (NCSC-tg-002) - Rainbow Series

tg003.txt [86796 bytes] 1993-11-09 "Discretionary Access Control in Trusted Systems" Version 1, 9/30/87 (NCSC-tg-003) - Rainbow Series

tg004.txt [65376 bytes] 1993-10-04 "Glossary of Computer Security Terms" Version 1, 10/21/88 - Rainbow Series

tg005.txt [819493 bytes] 1993-11-09 "Trusted Network Interpretation" Version 1 7/31/87 (NCSC-tg-005) - Rainbow Series
```

tg006.txt [137992 bytes] 1993-11-09 "Configuration Management in Trusted Systems" Version 1, 3/28/88 (NCSC-tg-006) - Rainbow Series

tg008.txt [55484 bytes] 1993-11-09 "A Guide to Understanding Trusted Distribution in Trusted Systems" Version 1 12/15/88 (NCSC-tg-008) - Rainbow Series

tg014.txt [57408 bytes] 1993-11-10 "Guidelines for Formal Verification Systems" 4/1/89 (NCSC-tg-014) - Rainbow Series

tg015.txt [97701 bytes] 1993-11-10 "Guide to Understanding Trusted Facility Management" 6/89 (NCSC-tg-015) - Rainbow Series

tg019.txt [60038 bytes] 1993-11-09 "Trusted Product Evaluation Questionnaire" Version 1, 10/16/89 (NCSC-tg-019) - Rainbow Series

.

Appendix C – On-Line Reference Resources

The following list of Websites provides potential resources of security information. Some are run by governmental agencies while others are commercial enterprises. Information provided on non-Governmental websites must be construed in the context of the organization or enterprise which is providing it.

Government Websites

U.S. Government

http://csrc.nist.gov/

http://csrc.nist.gov/publications/PubsFIPS.html

http://www.niap-ccevs.org/announcements/

http://www.nsa.gov/ia/index.shtml

http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml#guides

http://www.cnss.gov/policies.html

http://niatec.info

International

The Common Criteria Portal: http://www.commoncriteriaportal.org/

Commercial Websites

The SANS Institute http://www.sans.org/

The Trusted Computing Group http://www.trustedcomputinggroup.org/

The International PGP Home Page www.pgpi.org

Certicom http://www.certicom.com

PKI Certificate Suppliers

RSA Security Inc. www.RSA.com

Verisign Inc. www.Verisign.com

Entrust www.entrust.net

Other Non-Commercial

Federation of American Scientists (FAS) – www.fas.org