

# A Path Toward Cost-effective SCA Compliance Testing

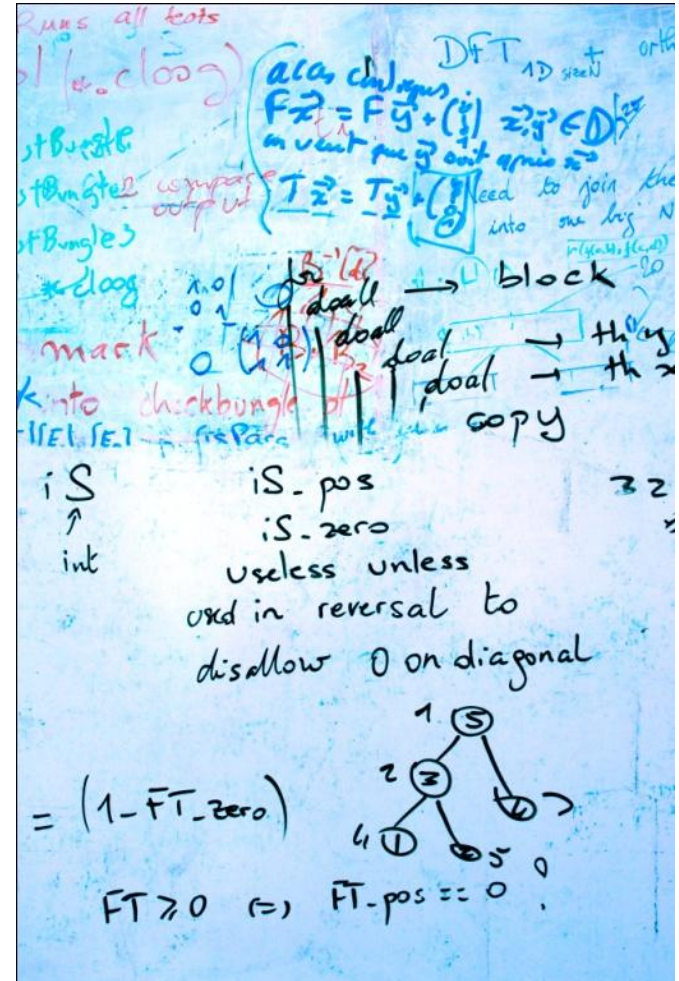
## R-Check™ SCA

**James Ezick,  
Jonathan Springer,  
Vassily Litvinov,  
David Wohlford**

**Reservoir Labs, Inc.  
New York, NY**

**SDR '10  
Wireless Innovation Conference &  
Product Exposition**

**30 November 2010**



# The Challenge

## Software Communications Architecture (SCA)

- Governs US Military's Joint Tactical Radio System (JTRS)
- References IEEE POSIX® and OMG CORBA®

## JTRS Test & Evaluation Lab (JTEL)

- JTRS program test authority for SCA compliance

## SCA Compliance Testing

- Multiple Operating Environments and CORBA ORBs
- Requirements cut across source code, IDL, XML files
- COTS products tend to be a poor fit or too expensive
- Vendors: test cycle can exceed upgrade-release cycle

## Challenge

- Deliver a *focused test capability* at *reasonable cost*

# R-Check™ SCA

Static Analysis Tool

Syntax **Structure** and **Context** Aware

**POSIX®** and **CORBA®** support, **C** and **C++**

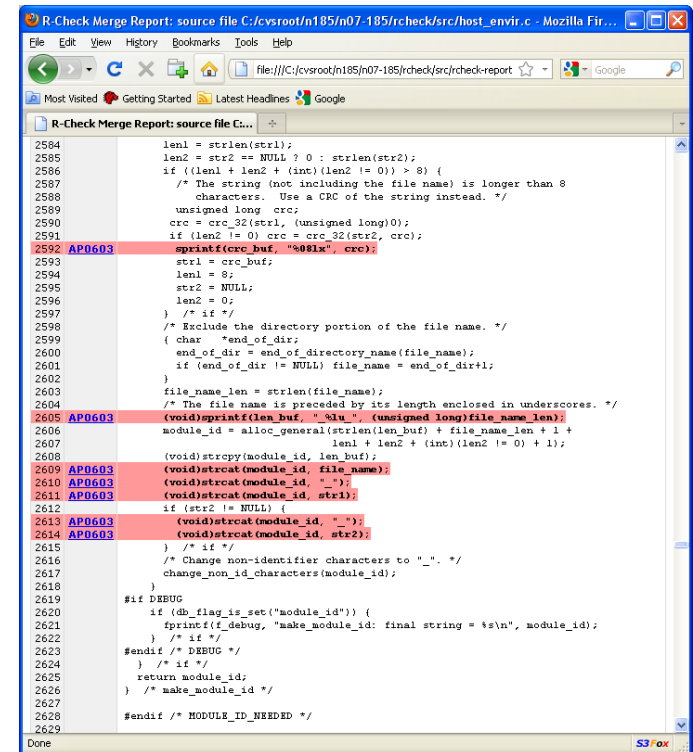
**Comprehensive analysis** utilizes  
build flags and included header files

**Partial Code Mode (PCM)** makes smart  
assumptions from incomplete information

**Compiler-grade speed and scalability:**  
scales to *millions of lines* of code,  
most files *analyzed in seconds*

Natural integration with development workflow

Robust report generation



```
2584 len1 = strlen(str1);
2585 len2 = str2 == NULL ? 0 : strlen(str2);
2586 if ((len1 + len2 + (int)(len2 != 0)) > 8) {
2587     /* The string (not including the file name) is longer than 8
2588     characters. Use a CRC of the string instead. */
2589     unsigned long crc;
2590     crc = crc_32(str1, (unsigned long)0);
2591     if (len2 != 0) crc = crc_32(str2, crc);
2592     sprintf(crc_buf, "%08lx", crc);
2593     str1 = crc_buf;
2594     len1 = 8;
2595     str2 = NULL;
2596     len2 = 0;
2597 } /* if */
2598 /* Exclude the directory portion of the file name. */
2599 { char *end_of_dir;
2600     end_of_dir = end_of_directory_name(file_name);
2601     if (end_of_dir != NULL) file_name = end_of_dir+1;
2602 }
2603 file_name_len = strlen(file_name);
2604 /* The file name is preceded by its length enclosed in underscores. */
2605     (void)sprintf(len_buf, "%5lu ", (unsigned long)file_name_len);
2606     module_id = alloc_general(strlen(len_buf) + file_name_len + 1 +
2607                             len1 + len2 + (int)(len2 != 0) + 1);
2608     (void)strcpy(module_id, len_buf);
2609     (void)strcat(module_id, file_name);
2610     (void)strcat(module_id, "_");
2611     (void)strcat(module_id, str1);
2612     if (str2 != NULL) {
2613         (void)strcat(module_id, "_");
2614         (void)strcat(module_id, str2);
2615     } /* if */
2616     /* Change non-identifier characters to "_". */
2617     change_non_id_characters(module_id);
2618 }
2619 #if DEBUG
2620 if (db_flag_is_set("module_id")) {
2621     printf(f_debug, "make_module_id: final string = %s\n", module_id);
2622 } /* if */
2623 #endif /* DEBUG */
2624 } /* if */
2625 return module_id;
2626 } /* make_module_id */
2627
2628 #endif /* MODULE_ID_NEEDED */
2629
```

*R-Check HTML report for the JTEL AP0603  
core POSIX requirement (SCA 2.2.2. App B)*

# Replace this ...

## Manual Test Steps

- Notes:
1. Test Result will include Pass, Fail, Untested, or N/A.
  2. The Test Recording Log is intended to record data for each step that requires recording.

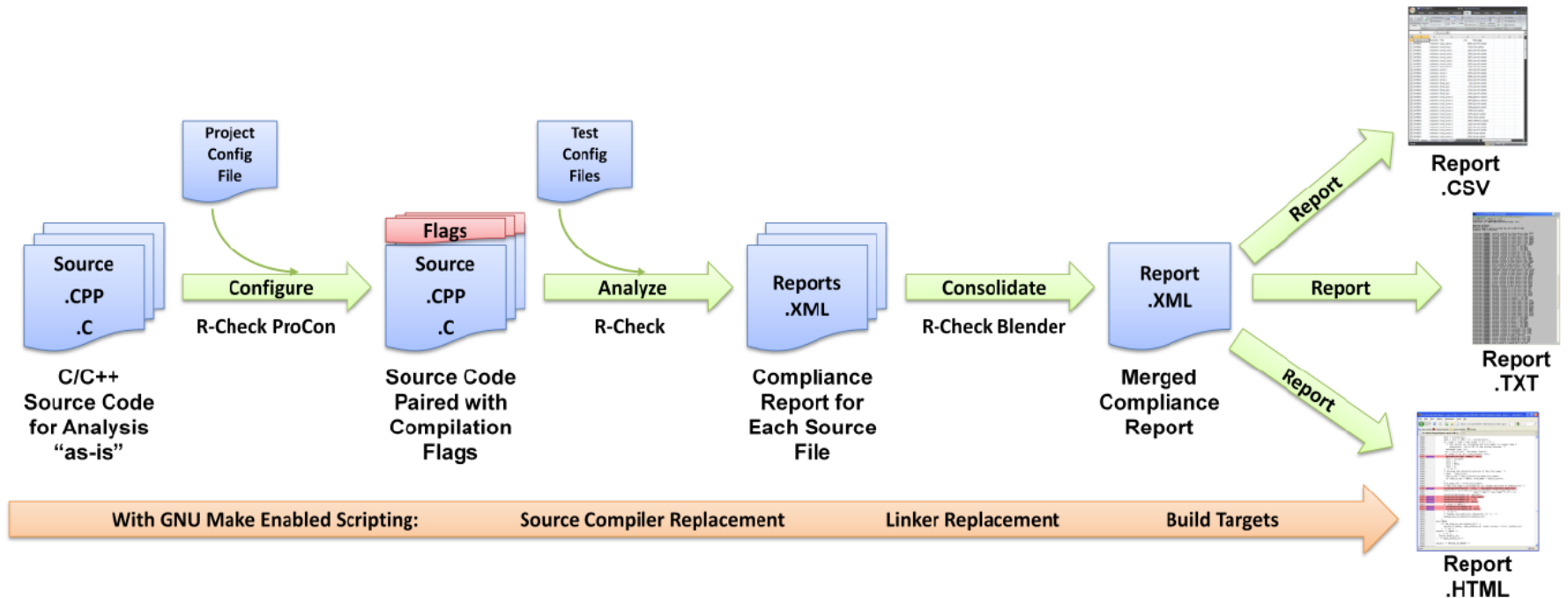
APP_TC_008				
Steps	Expected Results	Actual Results	Comments	Test Result
For each component, perform the following:				
<b>A. Identify the source code files that implement POSIX OS service functions. (AP0603)</b>				
1. Perform a search for all OS service functions in the application source code provided by the developer. Record the source file name(s).	N/A: The source code files are not available. (AP0603)			
For each entry in the APP_TC_008 Table 1 perform steps 2 & 3 :				
<b>B. Search the source code for occurrences of that entry. (AP0603)</b>				
2. Using the select entry from Table 1, perform a search for that entry.	Pass: There are no occurrences of the table entry. Move on to the next table entry. (AP0603)  Otherwise go to Step 3			
<b>C. Research occurrences within the source code to verify that they are not implementations of a POSIX interface. (AP0603)</b>				
3. Research occurrences within the source code to verify that they are not implementations of a POSIX interface.	Pass: All occurrences of the table entry found in the search do not represent implementations of a POSIX interface. (AP0603)  Fail: One or more occurrences of the table entry found in the search represent implementations of a POSIX interface. (AP0603)			
End of test				

... with this ...

```
rcheck.jtel --check=ap0603 --sca_version=2.2.2
```

... and get a ***complete, reproducible*** result with as few ***false positives*** as possible

# R-Check SCA Workflow



# Controlling Cost with Off the Shelf Components and Open Standards

## GNU Make Driver

R-Check functions as drop-in replacement for a compiler

Automatic dependency tracking and parallel source code evaluation

Operations are targets

Wrapper interface: ~2 days effort

## ProCon – EDG

EDG – Industrial-grade C/C++ parser with preprocessor support

ProCon – Open format to support binding compiler options to projects, file types, and modules

Simple to implement waivers

## Blender – XML

Open schema for reports

Linker replacement merges reports into a single summary

Also supports slices and comparisons

## Reports

Support: HTML, CSV, Text

Use 3<sup>rd</sup>-party tools to provide a robust, familiar user interface

In HTML, support hyperlinks to errors, references to SCA text

# Partial Code Mode (PCM)

R-Check performs a complete analysis –  
*given all of the information available to a compiler*

Under current practices, JTEL may not have access to  
all information necessary to rebuild

- Receives "in-development" code
- Does not receive build script or log
- Operating system headers (Integrity™, VxWorks™, ...) unavailable

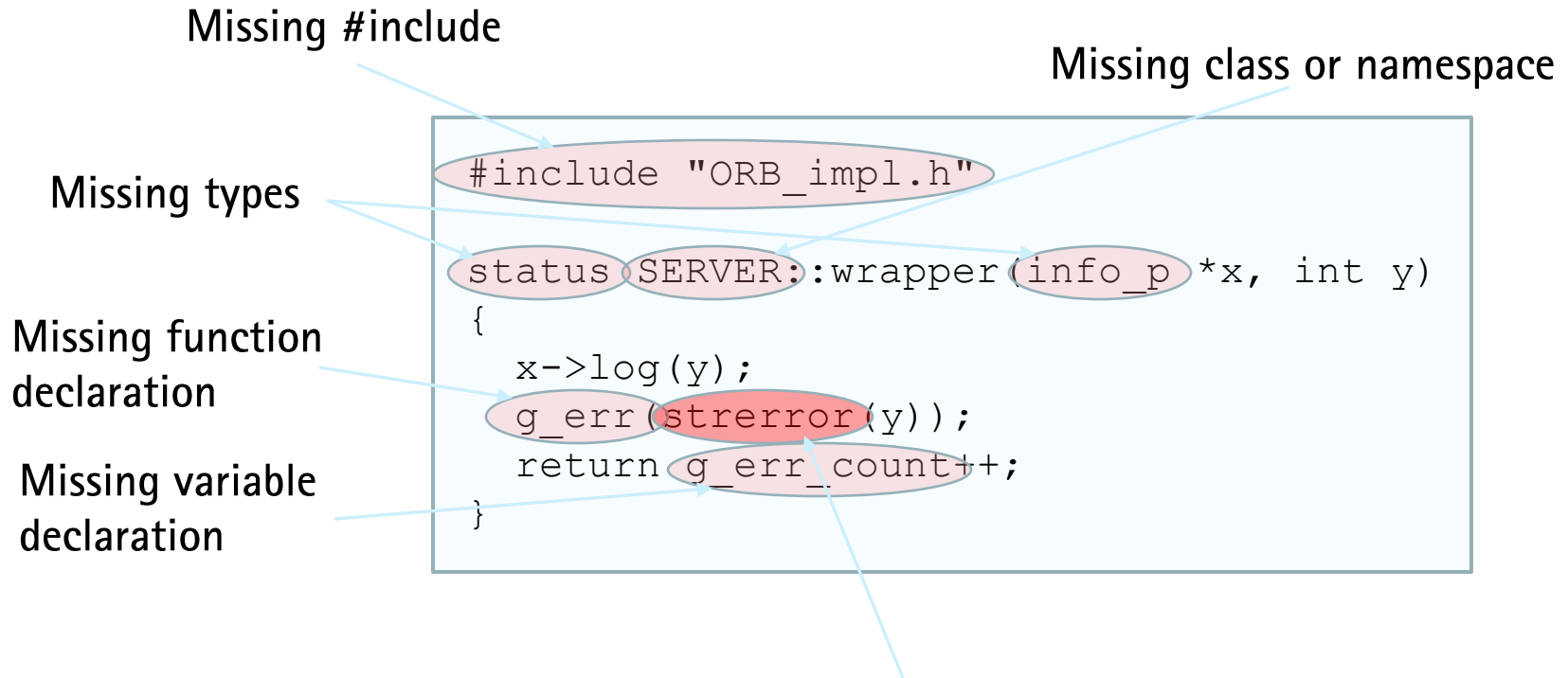
Problem exacerbated by CORBA development

- Missing proprietary CORBA ORB header files
- Missing proprietary IDL compiler necessary to build headers

Problem is less acute for vendors, but it may still be desirable  
to test on boxes without a full development installation



# Anatomy of Partial Code



**But we still want to find the violation!**

May be sandwiched between and/or nested within missing code references

# Partial Code Challenges

```
#include <tao/corba.h>
#include <tao/orbsvcs/orbsvcs/CosNamingC.h>
...
func(...)
{
    CORBA::ORB *op;
    ...
    (200+ lines omitted)
    ...
    op->shutdown (true);
}
```

- Seen by itself, **op->shutdown()** might be a non-CORBA call
- ORB headers are not available – the **#includes** cannot be inspected
- Need to infer existence of class **ORB** in namespace **CORBA**
- R-Check constructs classes as it sees references, so knows:
  - **op->shutdown()** call is a violation of Minimum CORBA requirement

# R-Check Partial Code Mode

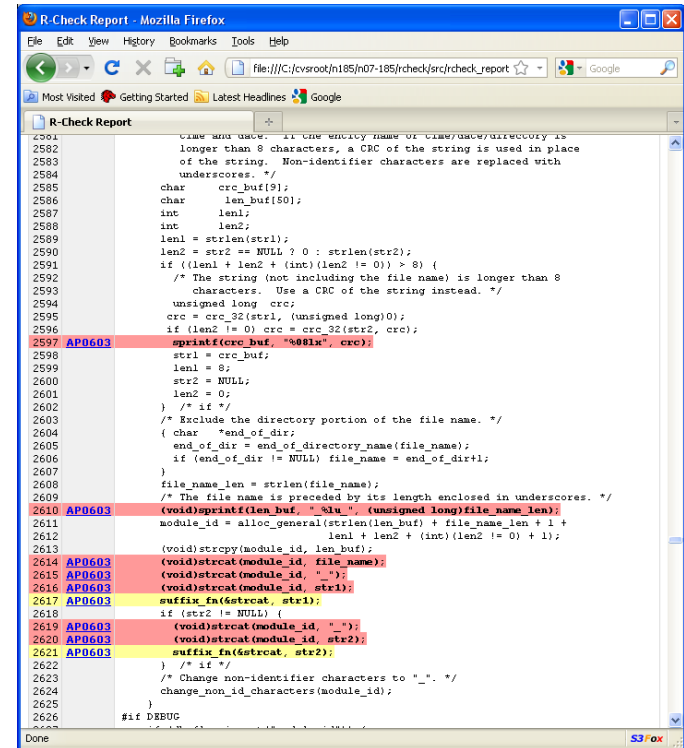
## Conservative analysis by making smart assumptions

- Add placeholders for missing types in declarations
- Reconstruct missing classes and namespaces
  - Used in declarations (types) and expressions (static members)
- Tolerate references to undeclared variables
- Tolerate calls to undeclared functions
- Can't overcome everything – need some information (e.g., #define macros)
- *Don't let missing information impede checking the code that is actually present*

Combines **domain knowledge** with solid **type-system theory** to provide a consistent, predictable result

## Most significant development challenge

- Required both theoretical and practical compiler expertise
- Approximately 50% of the development effort



```
2581 time and date. If the entity name of time/date/directory is
2582 longer than 8 characters, a CRC of the string is used in place
2583 of the string. Non-identifier characters are replaced with
2584 underscores. */
2585 char   crc_buf[9];
2586 char   len_buf[50];
2587 int    len1;
2588 int    len2;
2589 len1 = strlen(str1);
2590 len2 = str2 == NULL ? 0 : strlen(str2);
2591 if ((len1 + len2 + (int)(len2 != 0)) > 8) {
2592     /* The string (not including the file name) is longer than 8
2593     characters. Use a CRC of the string instead. */
2594     unsigned long  crc;
2595     crc = crc_32(str1, (unsigned long)0);
2596     if (len2 != 0) crc = crc_32(str2, crc);
2597 AP0603 sprintf(crc_buf, "%08lx", crc);
2598     str1 = crc_buf;
2599     len1 = 8;
2600     str2 = NULL;
2601     len2 = 0;
2602     /* if */
2603     /* Exclude the directory portion of the file name. */
2604     { char   *end_of_dir;
2605       end_of_dir = end_of_directory_name(file_name);
2606       if (end_of_dir != NULL) file_name = end_of_dir+1;
2607     }
2608     file_name_len = strlen(file_name);
2609     /* The file name is preceded by its length enclosed in underscores. */
2610 AP0603 (void)sprintf(len_buf, "%3u ", (unsigned long)file_name_len);
2611     module_id = alloc_general(strlen(len_buf) + file_name_len + 1 +
2612                               len1 + len2 + (int)(len2 != 0) + 1);
2613     (void)strcpy(module_id, len_buf);
2614 AP0603 (void)strcat(module_id, file_name);
2615 AP0603 (void)strcat(module_id, "_");
2616 AP0603 (void)strcat(module_id, str1);
2617 AP0603 suffix_fn(&strcat, str1);
2618     if (str2 != NULL) {
2619 AP0603 (void)strcat(module_id, "_");
2620 AP0603 (void)strcat(module_id, str2);
2621 AP0603 suffix_fn(&strcat, str2);
2622     } /* if */
2623     /* Change non-identifier characters to "_". */
2624     change_non_id_characters(module_id);
2625 }
2626 #if DEBUG
```

*R-Check HTML report for the JTEL AP0603 core POSIX requirement (SCA 2.2.2. App B)*

*Lines 2617 and 2621 are shaded as sources of a potential violation because the address of an NRQ POSIX call is taken and passed as a function pointer argument*

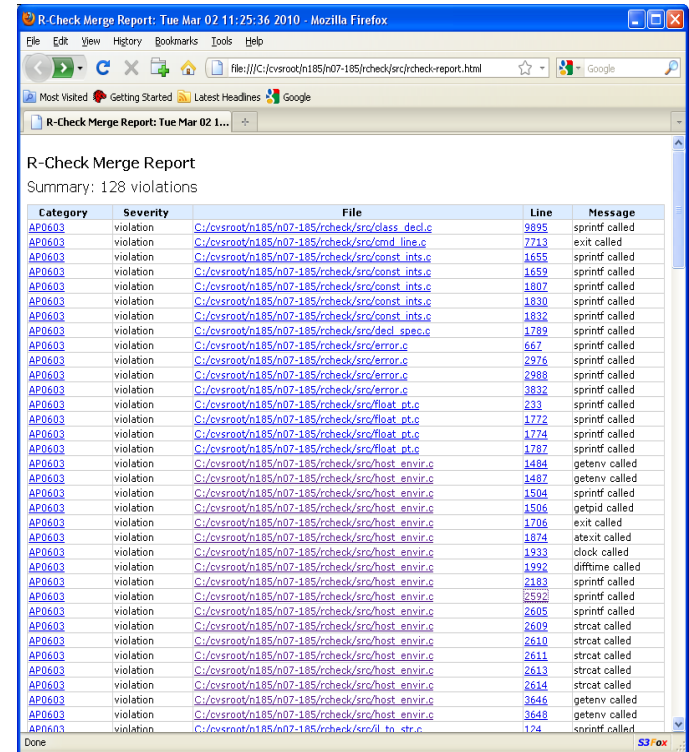
## Test Coverage & Performance

# Version 1.0 Test Coverage

- Core POSIX
- Minimum CORBA
- Read/Write Behavior

Tested against more than a dozen waveforms in the NED IR

- 100% successfully processed using PCM in the JTEL work environment
- Replaces 2+ days of JTEL search & inspect process per waveform
- Running times track with usual compilation times
  - Calit2 FM3TR – 2-3 minutes
  - Largest NED IR waveforms – 1000's files, >10MLOC, ~90 minutes



*R-Check HTML report for the JTEL AP0603  
core POSIX requirement (SCA 2.2.2. App B)*

*Hyperlinks jump to referenced code and a description of the analyzed requirement*

## Lessons Learned & Future Work

A specialized test capability for minimal cost is feasible

- Leverage off the shelf components and open standards
- Reusable platform

The less you develop, the easier it is to drop into a range of development environments

- The command line is your friend, GUIs are hard
- Testing folds into the normal edit-recompile-test cycle

Version 2.0 – Coverage for IDL, XML Configuration Files

- Several SCA requirements require consistency across source code, IDL, and XML files – existing tools are not set up to provide this functionality
- R-Check framework extends to provide this capability

# About Reservoir Labs

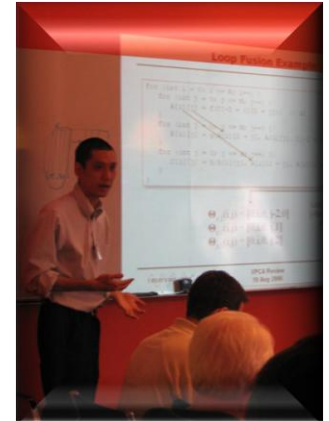
*Privately owned, Reservoir Labs has been providing leading-edge consulting and contract R&D to the computer industry, business, end-users, and the US Government since 1990*

## Expertise

- Applied *compiler research* for emerging high-performance and embedded architectures
- Advanced *polyhedral mapping* techniques
- Reasoning, *constraint solving*, and mathematics

## Technologies

- *R-Check* Static Analysis Platform
- *R-Stream* Mapping Compiler
- *Alef* Distributed Constraint Solver



*Reservoir offices in New York, NY  
and Portland, OR*

# Acknowledgements

Development funded by SPAWAR  
under Navy Phase II SBIR Contract  
"Static Analysis Tool for Interface  
Compliance Verification and Program  
Comprehension"



Thanks!

John Thom, TPOC (SPAWAR)

JTEL Test Execution Team

Jim Kulp, Mercury Federal Systems

