

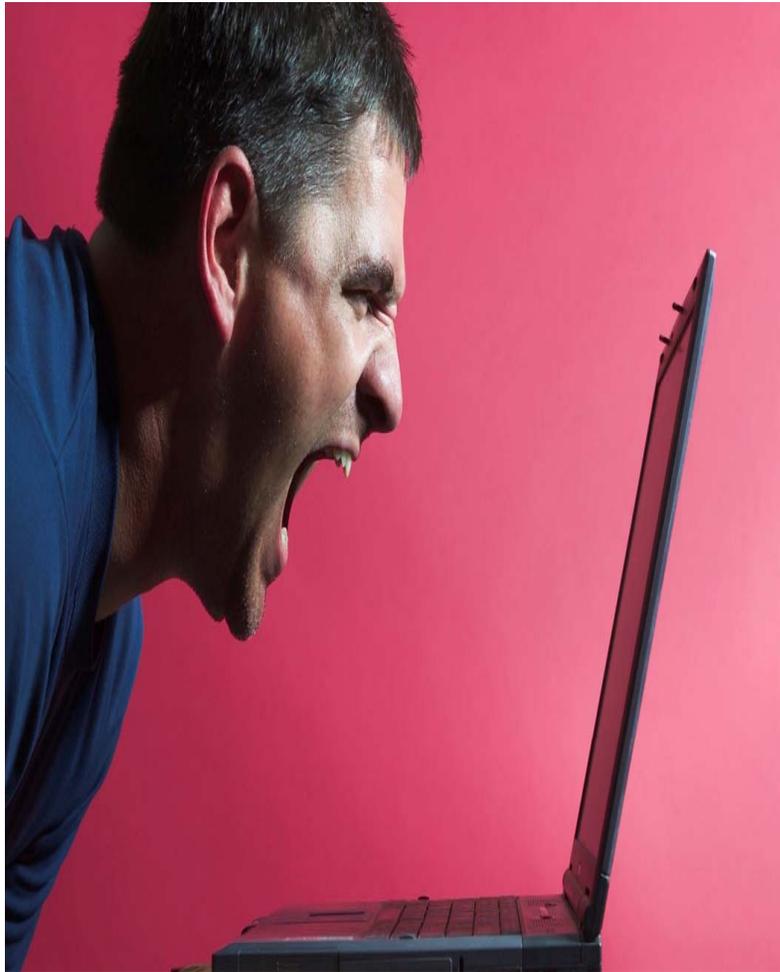
RCI-COTS-P08-008 (v1.1, 10/14/08)



Software Assurance in COTS and Open Source Packages

**Donald Reifer, CTO
Reifer Consultants, Inc.
Torrance, CA 90510
and
Eric Bryant, Chief Scientist
Arxan Defense Systems Inc.
Huntsville, AL 35805**

Agenda



- Identify the COTS assurance problem
- Discuss a solution
- Demonstrate the technology we've developed
- Entertain you and answer questions

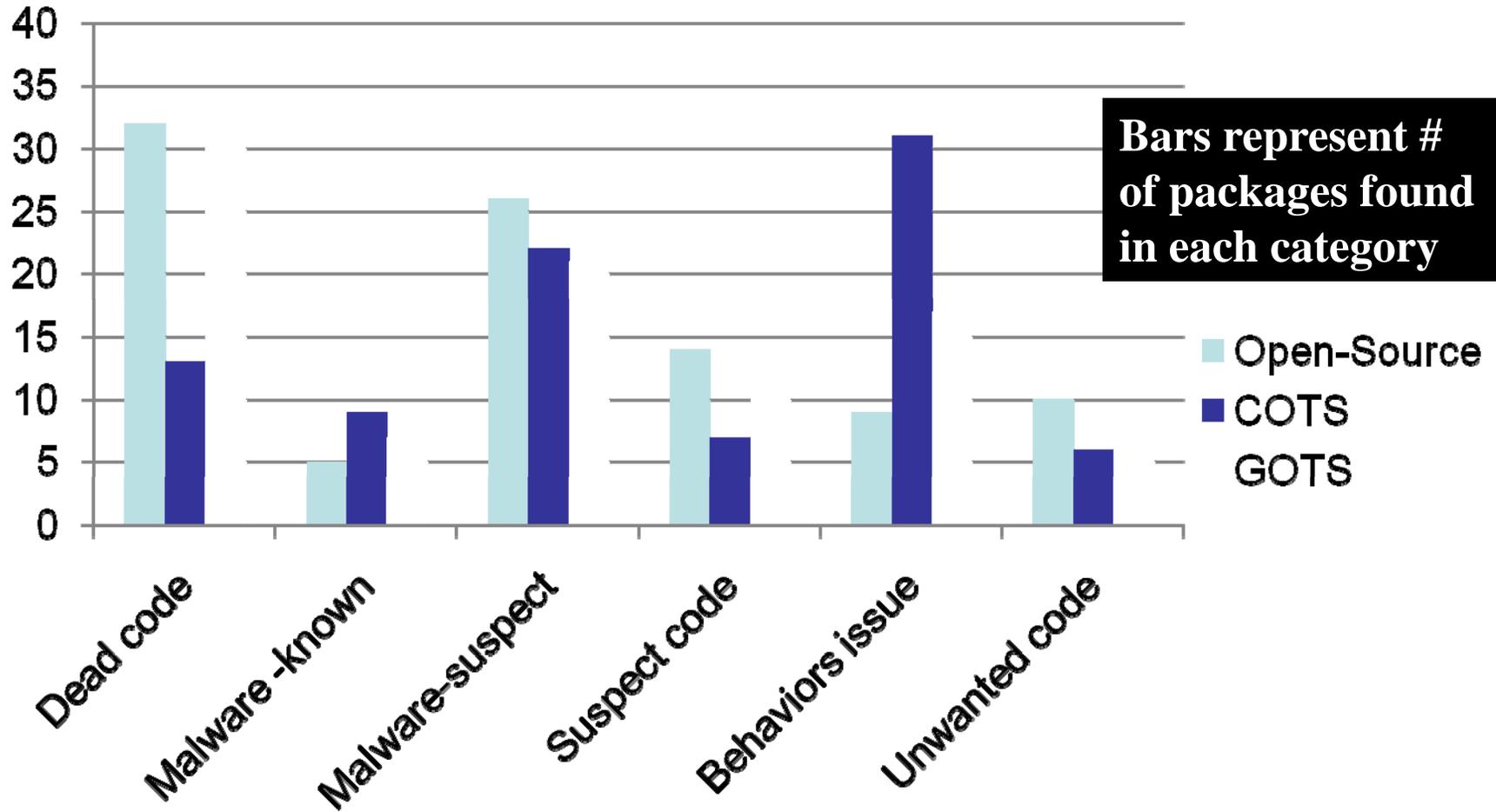
The Challenge

- Potential for malicious code in applications software is large as more and more packages are used
 - Example 1 – test instrumentation in COTS software that provides remote diagnostics/diagnosis opens a backdoor
 - Example 2 – poor coding practices in vendor or Open Source software creates exploitable vulnerabilities
- Best practices fail to provide confidence that software is trustworthy and of suitable quality
 - Elimination of “malware” in applications software is costly especially when packages are large and users do not have access to the source code and documentation
- The tools and techniques that exist primarily provide source code analysis capabilities
 - Software ecosystem frameworks and tool systems

How Big A Problem Is It?

- Selected one hundred packages at random from fifty public Open-Source and COTS libraries
 - Full range of applications and sites like SourceForge
- College students analyzed packages, spending at most four hours each, using a variety of tools to:
 - Determine if the packages are up-to-date (vendor identified vulnerabilities and patches)
 - Assess if packages are free of known viruses, worms, Trojans and Spyware
 - Assess if packages have weaknesses in the code and backdoors using reverse engineering techniques
 - Assess if packages have potential dead code, malware and unwanted behaviors (undesired functionality) by reversing the code

COTS Study Findings



Software Quality and Integrity Checker is being build to address these issues

Responding to the Problem

- MDA Phase II SBIR effort under topic entitled “Secure Computing Infrastructure Technologies”
 - Effort aimed at mitigating the “*Malicious Code*” risk
 - R&D developing work processes and tools to check quality and integrity of both applications **source code** and **executables (binaries)**
 - Addresses full range of MDA applications software (COTS, Open-Source and contractor code)
 - R&D contract awarded in May 2007
 - Prototype method and tool is available **now**
 - Production quality tool will be available in **near-term**

Addressing Executables as well as Source Code

- Unlike other applications software initiatives, we do not require access to developer's source code to assess its quality and integrity
 - It is naïve to think commercial vendors will supply it
 - However, if they do, we provide users with the tools they need to assess source code
- We focus on analyzing software executables
 - It is typically the only thing available from COTS suppliers
 - Even when it is not, executables should be still be assessed because such analysis facilitates the discovery of malware that cannot be detected otherwise

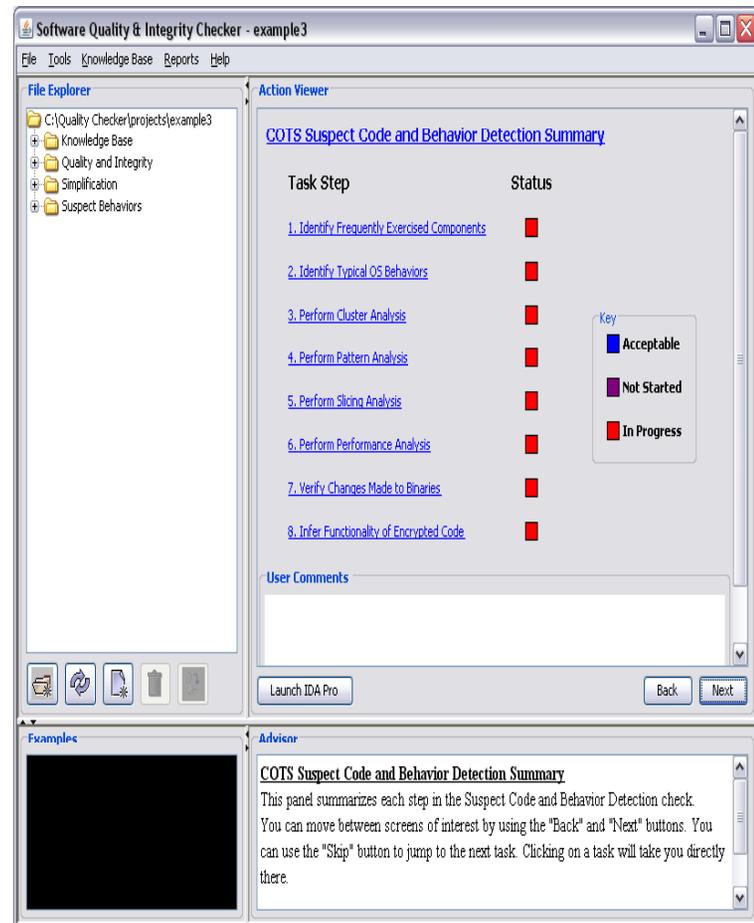
Technical Approach

- Methodology-based approach
 - Perform basic checks
 - Identify weaknesses and vulnerabilities
 - Perform “no harm” analysis
 - Remediate dead code, malware and suspicious behaviors
- Knowledge-based philosophy
 - Tools provide guidance and hints
- Filters to reduce false positives
- Extensible tool architecture



Let's See How It Works

- The package runs on Windows XP or VISTA
- It requires that you have a license for IDA Pro
- It permits you to bind your own tools into the toolset
- We will demo it to our customer on 5 Nov 2008
- We plan to have it ready for operational use in 2009



Summary and Conclusions

**ARE THERE
ANY
QUESTIONS?**

**WE ARE ON
TRACK AND
MAKING
PROGRESS**

