Vulnerability Assessment for Middleware

Elisa Heymann

Computer Architecture and
Operating Systems Department
Universitat Autònoma de Barcelona

elisa@cs.wisc.edu

NFS Cybersecurity Summit for
Cyberinfrastructure and Large Facilities
September 30, 2013

Who we are

Bart Miller
Jim Kupsch
Vamshi Basupalli
Salini Kowsalya

Elisa Heymann
Eduardo Cesar
Manuel Brugnoli
Max Frydman

http://www.cs.wisc.edu/mist/

What do we do

- Assess Middleware: Make cloud/grid software more secure
- Train: We teach tutorials for users, developers, sys admins, and managers
- Research: Make in-depth assessments more automated and improve quality of automated code analysis

http://www.cs.wisc.edu/mist/papers/VAshort.pdf

Our History

2001: "Playing Inside the Black Box" paper, first demonstration of hijacking processes in the Cloud.

2004: First formal funding from US NSF.

2004: First assessment activity, based on Condor, and started development of our methodology (FPVA).

2006: Start of joint effort between UW and UAB.

2006: Taught first tutorial at San Diego Supercomputer Center.

2007: First NATO funding, jointly to UAB, UW, and Ben Gurion University.

2008: First authoritative study of automated code analysis tools.

2009: Published detailed report on our FPVA methodology.

2009: U.S. Dept. of Homeland Security funding support.

2012: National Software Assurance Marketplace (SWAMP) research center.

Our experience

Condor, University of Wisconsin
Batch queuing workload management system
15 vulnerabilities 600 KLOC of C and C++

SRB, SDSC
Storage Resource Broker - data grid
5 vulnerabilities 280 KLOC of C

MyProxy, NCSA
Credential Management System
5 vulnerabilities 25 KLOC of C

glExec, Nikhef
Identity mapping service
5 vulnerabilities 48 KLOC of C

Gratia Condor Probe, FNAL and Open Science Grid
Feeds Condor Usage into Gratia Accounting System
3 vulnerabilities 1.7 KLOC of Perl and Bash

Condor Quill, University of Wisconsin
DBMS Storage of Condor Operational and Historical Data
6 vulnerabilities 7.9 KLOC of C and C++

Our experience

Wireshark, wireshark.org
Network Protocol Analyzer
2 vulnerabilities 2400 KLOC of C

Condor Privilege Separation, Univ. of Wisconsin
Restricted Identity Switching Module
2 vulnerabilities 21 KLOC of C and C++

VOMS Admin, INFN
Web management interface to VOMS data
4 vulnerabilities 35 KLOC of Java and PHP

CrossBroker, Universitat Autònoma de Barcelona
Resource Mgr for Parallel & Interactive Applications
4 vulnerabilities 97 KLOC of C++

ARGUS 1.2, HIP, INFN, NIKHEF, SWITCH
gLite Authorization Service
0 vulnerabilities 42 KLOC of Java and C

Our experience

VOMS Core INFN
Virtual Organization Management System
1 vulnerability 161 KLOC of Bourne Shell, C++ and C

iRODS, DICE
Data-management System
9 vulnerabilities (and counting) 285 KLOC of C and C++

Google Chrome, Google
Web browser
1 vulnerability 2396 KLOC of C and C++

WMS, INFN
Workload Management System
in progress 728 KLOC of Bourne Shell, C++, C, Python, Java, and Perl

CREAM, INFN
Computing Resource Execution And Management
5 vulnerabilities 216 KLOC of Bourne Shell, Java, and C++

What is Software Security?

- Software security means protecting software against malicious attacks and other risks.
- Security is necessary to provide availability, confidentiality, and integrity.

What is a Vulnerability?

"A vulnerability is a defect or weakness in system security procedures, design, implementation, or internal controls that can be exercised and result in a security breach or violation of security policy."

- Gary McGraw, *Software Security*

What is an Exploit?

"The process of attacking a vulnerability in a program is called exploiting."

The Art of Software Security Assessment

- Exploit: The attack can come from a program or script.

What is a Threat?

"A potential cause of an incident, that may result in harm of systems and organization."

ISO 27005

"Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, or individuals through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service. Also, the potential for a threat-source to successfully exploit a particular information system vulnerability."

NIST

What is a Threat?

- Threat may come from many sources:
  - External attackers.
  - Legitimate users.
  - Service providers.
  - Technical failure.

Cost of Insufficient Security

- Attacks are expensive and affect assets:
  - Management.
  - Organization.
  - Process.
  - Information and data.
  - Software and applications.
  - Infrastructure.

Cost of Insufficient Security

- Attacks are expensive and affect assets:
  - Financial capital.
  - Reputation.
  - Intellectual property.
  - Network resources.
  - Digital identities.
  - Services.

Things That We All Know

› All software has vulnerabilities.
› Critical infrastructure software is complex and large.
› Vulnerabilities can be exploited by both authorized users and by outsiders.

Key Issues for Security

- Need independent assessment
  - Software engineers have long known that testing groups must be independent of development groups
- Need an assessment process that is NOT based on known vulnerabilities
  - Such approaches will not find new types and variations of attacks

Key Issues for Security

- Automated Analysis Tools have Serious Limitations:
  - While they help find some local errors, they
    - MISS significant vulnerabilities (false negatives)
    - Produce voluminous reports (false positives)
- Programmers must be security-aware
  - Designing for security and the use of secure practices and standards does not guarantee security.

Addressing these Issues

- We must evaluate the security of our code
  - The vulnerabilities are there and we want to find them first.
- Assessment isn’t cheap
  - Automated tools create an illusion of security.
- You can’t take shortcuts
  - Even if the development team is good at testing, they can’t do an effective assessment of their own code.

Addressing these Issues

- Try First Principles Vulnerability Assessment
  - A strategy that focuses on critical resources.
  - A strategy that is not based on known vulnerabilities.
- We need to integrate assessment and remediation into the software development process.
  - We have to be prepared to respond to the vulnerabilities we find.

First Principles Vulnerability Assessment Understanding the System

Step 1: Architectural Analysis

- Functionality and structure of the system, major components (modules, threads, processes), communication channels.
- Interactions among components and with users.

First Principles Vulnerability Assessment Understanding the System

Step 2: Resource Identification

- Key resources accessed by each component.
- Operations allowed on those resources.

Step 3: Trust & Privilege Analysis

- How components are protected and who can access them.
- Privilege level at which each component runs.
- Trust delegation.

First Principles Vulnerability Assessment
Search for Vulnerabilities

Step 4: Component Evaluation

- Examine critical components in depth.
- Guide search using:
  - Diagrams from steps 1-3.
  - Knowledge of vulnerabilities.
- Helped by Automated scanning tools (!)

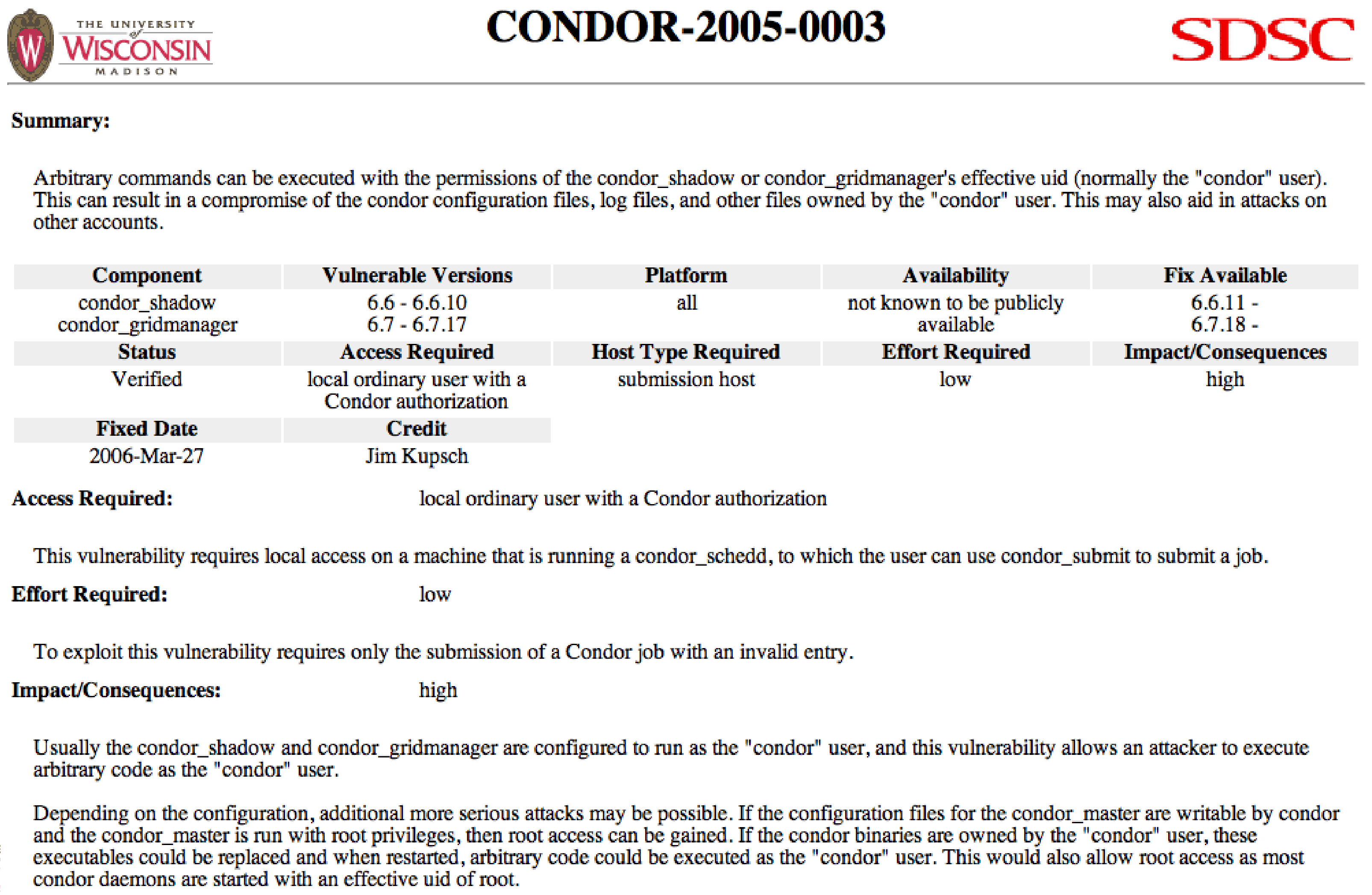
First Principles Vulnerability Assessment
Taking Actions

Step 5: Dissemination of Results

- Report vulnerabilities.
- Interaction with developers.
- Disclosure of vulnerabilities.

First Principles Vulnerability Assessment Taking Actions

Step 5: Dissemination of Results



THE UNIVERSITY of WISCONSIN MADISON

CONDOR-2005-0003

SDSC

Summary:

Arbitrary commands can be executed with the permissions of the condor_shadow or condor_gridmanager's effective uid (normally the "condor" user). This can result in a compromise of the condor configuration files, log files, and other files owned by the "condor" user. This may also aid in attacks on other accounts.

Component	Vulnerable Versions	Platform	Availability	Fix Available
condor_shadow condor_gridmanager	6.6 - 6.6.10 6.7 - 6.7.17	all	not known to be publicly available	6.6.11 - 6.7.18 -
Status	**Access Required**	**Host Type Required**	**Effort Required**	**Impact/Consequences**
Verified	local ordinary user with a Condor authorization	submission host	low	high
Fixed Date	**Credit**			
2006-Mar-27	Jim Kupsch			

Access Required: local ordinary user with a Condor authorization

This vulnerability requires local access on a machine that is running a condor_schedd, to which the user can use condor_submit to submit a job.

Effort Required: low

To exploit this vulnerability requires only the submission of a Condor job with an invalid entry.

Impact/Consequences: high

Usually the condor_shadow and condor_gridmanager are configured to run as the "condor" user, and this vulnerability allows an attacker to execute arbitrary code as the "condor" user.

Depending on the configuration, additional more serious attacks may be possible. If the configuration files for the condor_master are writable by condor and the condor_master is run with root privileges, then root access can be gained. If the condor binaries are owned by the "condor" user, these executables could be replaced and when restarted, arbitrary code could be executed as the "condor" user. This would also allow root access as most condor daemons are started with an effective uid of root.

Vulnerability Assessment of DIRAC

- We need resources.
  - We provide training.
- Slow process.
- Critical activity.

Questions?

http://www.cs.wisc.edu/mist