CSE 598i
Verification Methods for Security

Trent Jaeger
Systems and Internet Infrastructure Security (SIIS) Lab
Computer Science and Engineering Department
Pennsylvania State University

August 22, 2011
About Me

- Trent Jaeger (PhD, University of Michigan)
- Associate Professor, CSE -- after 9 years at IBM Research
- Research: Operating System Security
- Example Projects
  - L4 Microkernel -- minimal, high performance OS
  - Linux -- Open source, UNIX variant
  - Xen hypervisor -- Open source, virtual machine platform
- Office Hours: by appointment
- Office: 346A IST Bldg
- Email: tjaeger@cse.psu.edu
Motivation

Security mechanisms and policies have been implemented at several system layers (app, OS, VM, network)

Are we now secure?
Current Security Problems

Most current security problems are based on the failure of people to deploy hosts securely

- Botnets
- Rootkits
- Web attacks: XSS, SQL Inject, …
- Worms (Conficker, Stuxnet)
- Password Guessing
- Buffer Overflows
- Arbitrary App Flaws
Security State

SANS Top Security Risks
http://www.sans.org/top-cyber-security-risks/

• Client-side software is unpatched (apps patched slower)
• Web servers are vulnerable (XSS are 80%)
• Application vulnerabilities exceed OS vulnerabilities
• Attacks on Mac systems (QuickTime)
• US is the major attack target (30:1)
• Still buffer (and heap) overflows

We will study the structure of attacks on hosts and a general procedure for their prevention
Executive Summary

Vulnerability Exploitation Trends

Origin and Destination Analysis for 4 Key Attacks

Application vs. Operating System Patching

Tutorial: HTTP Client-Side Exploitation Example

Zero-Day Vulnerability Trends

Best Practices in Mitigation and Control

HTTP Server Threats

Real-Life HTTP Client-Side Exploitation Example

This section illustrates an example of a real life attack conducted against an organization that resulted in loss of critical data for the organization.

In this attack, Acme Widgets Corporation suffered a major breach from attackers who were able to compromise their entire internal network infrastructure using two of the most powerful and common attack vectors today: Exploitation of client-side software and pass-the-hash attacks against Windows machines.

Step 0: Attacker Places Content on Trusted Site

In Step 0, the attacker begins by placing content on a trusted third-party website, such as a social networking, blogging, photo sharing, or video sharing website, or any other web server that hosts content posted by public users. The attacker's content includes exploitation code for unpatched client-side software.

Step 1: Client-Side Exploitation

In Step 1, a user on the internal Acme Widgets enterprise network surfs the Internet from a Windows machine that is running an unpatched client-side program, such as a media player (e.g., Real Player, Windows Media Player, iTunes, etc.), document display program (e.g., Acrobat Reader), or a component of an office suite (e.g., Microsoft Word, Excel, Powerpoint, etc.).

Upon receiving the attacker's content from the site, the victim user's browser invokes the vulnerable client-side program passing it the attacker's exploit code. This exploit code allows the attacker to install or execute programs of the attacker's choosing on the victim machine, using the privileges of the user who ran the browser.

The attack is partially mitigated because this victim user does not have administrator credentials on this system. Still, the attacker can run programs with those limited user privileges.

Language Translations

Portuguese (PDF)

Check Them Out!

Upcoming Computer Security Training Events

Free Resources for Securing Your Network

SANS Network Security 2011 Security Awareness Training

Top Cyber Security Risks

SANS Reading Room

Career Roadmap

Storm Center

WhatWorks™ Newsletters

"A significant amount of information but with the information so well documented in the books, this will become a valuable reference tool for me in the future"

- Rick Funaro, KAPL, Inc
This section illustrates an example of a real life attack conducted against an organization that resulted in loss of critical data for the organization. In this attack, Acme Widgets Corporation suffered a major breach from attackers who were able to compromise their entire internal network infrastructure using two of the most powerful and common attack vectors today: Exploitation of client-side software and pass-the-hash attacks against Windows machines.

**Step 0: Attacker Places Content on Trusted Site**

In Step 0, the attacker begins by placing content on a trusted third-party website, such as a social networking, blogging, photo sharing, or video sharing website, or any other web server that hosts content posted by public users. The attacker's content includes exploitation code for unpatched client-side software.

**Step 1: Client-Side Exploitation**

In Step 1, a user on the internal Acme Widgets enterprise network surfs the Internet from a Windows machine that is running an unpatched client-side program, such as a media player (e.g., Real Player, Windows Media Player, iTunes, etc.), document display program (e.g., Acrobat Reader), or a component of an office suite (e.g., Microsoft Word, Excel, Powerpoint, etc.). Upon receiving the attacker's content from the site, the victim user's browser invokes the vulnerable client-side program passing it the attacker's exploit code. This exploit code allows the attacker to install or execute programs of the attacker's choosing on the victim machine, using the privileges of the user who ran the browser. The attack is partially mitigated because this victim user does not have administrator credentials on this system. Still, the attacker can run programs with those limited user privileges.
SANS Example

Activate reverse shell backdoor using HTTPS to control victim machine
Step 2: Establish Reverse Shell Backdoor Using HTTPS

In Step 2, the attacker’s exploit code installs a reverse shell backdoor program on the victim machine. This program gives the attacker command shell access of the victim machine, communicating between this system and the attacker using outbound HTTPS access from victim to attacker. The backdoor traffic therefore appears to be regular encrypted outbound web traffic as far as the enterprise firewall and network is concerned.

Steps 3 & 4: Dump Hashes and Use Pass-the-Hash Attack to Pivot

In Step 3, the attacker uses shell access of the initial victim system to load a local privilege escalation exploit program onto the victim machine. This program allows the attacker to jump from the limited privilege user account to full system privileges on this machine. Although vendors frequently release patches to stop local privilege escalation attacks, many organizations do not deploy such patches quickly, because such enterprises tend to focus exclusively on patching remotely exploitable flaws. The attacker now dumps the password hashes for all accounts on this local machine, including a local administrator account on the system.

In Step 4, instead of cracking the local administrator password, the attacker uses a Windows pass-the-hash program to authenticate to another Windows machine on the enterprise internal network, a fully patched client system on which this same victim user has full administrative privileges. Using NTLMv1 or NTLMv2, Windows machines authenticate network access for the Server Message Block (SMB) protocol based on user hashes and not the passwords themselves, allowing the attacker to get access to the file system or run programs on the fully patched system with local administrator privileges. Using these privileges, the attacker now dumps the password hashes for all local accounts on this fully patched Windows machine.

Step 5: Pass the Hash to Compromise Domain Controller

In Step 5, the attacker uses a password hash from a local account on the fully patched Windows client to access the domain controller system, again using a pass-the-hash attack to gain shell access on the domain controller. Because the password for the local administrator account is identical to the password for a domain administrator account, the password hashes for the two accounts are identical. Therefore, the attacker can access the domain controller with full domain administrator privileges, giving the attacker complete control over all other accounts and machines in that domain.

Steps 6 and 7: Exfiltration

In Step 6, with full domain administrator privileges, the attacker now compromises a server machine that stores secrets for the organization. In Step 7, the attacker exfiltrates this sensitive information, consisting of over 200 Megabytes of data. The attacker pushes this data out to the
Step 2: Establish Reverse Shell Backdoor Using HTTPS

In Step 2, the attacker's exploit code installs a reverse shell backdoor program on the victim machine. This program gives the attacker command shell access of the victim machine, communicating between this system and the attacker using outbound HTTPS access from victim to attacker. The backdoor traffic therefore appears to be regular encrypted outbound web traffic as far as the enterprise firewall and network is concerned.

Steps 3 & 4: Dump Hashes and Use Pass-the-Hash Attack to Pivot

In Step 3, the attacker uses shell access of the initial victim system to load a local privilege escalation exploit program onto the victim machine. This program allows the attacker to jump from the limited privilege user account to full system privileges on this machine. Although vendors frequently release patches to stop local privilege escalation attacks, many organizations do not deploy such patches quickly, because such enterprises tend to focus exclusively on patching remotely exploitable flaws. The attacker now dumps the password hashes for all accounts on this local machine, including a local administrator account on the system.

In Step 4, instead of cracking the local administrator password, the attacker uses a Windows pass-the-hash program to authenticate to another Windows machine on the enterprise internal network, a fully patched client system on which this same victim user has full administrative privileges. Using NTLMv1 or NTLMv2, Windows machines authenticate network access for the Server Message Block (SMB) protocol based on user hashes and not the passwords themselves, allowing the attacker to get access to the file system or run programs on the fully patched system with local administrator privileges. Using these privileges, the attacker now dumps the password hashes for all local accounts on this fully patched Windows machine.

Step 5: Pass the Hash to Compromise Domain Controller

In Step 5, the attacker uses a password hash from a local account on the fully patched Windows client to access the domain controller system, again using a pass-the-hash attack to gain shell access on the domain controller. Because the password for the local administrator account is identical to the password for a domain administrator account, the password hashes for the two accounts are identical. Therefore, the attacker can access the domain controller with full domain administrator privileges, giving the attacker complete control over all other accounts and machines in that domain.

Steps 6 and 7: Exfiltration

In Step 6, with full domain administrator privileges, the attacker now compromises a server machine that stores secrets for the organization. In Step 7, the attacker exfiltrates this sensitive information, consisting of over 200 Megabytes of data. The attacker pushes this data out to the
Causes?

Buffer Overflow

- (a) Situation when main program is running
- (b) After program A called
- (c) Buffer overflow shown in gray
Security Mythology

- Claim: *All these problems were solved in Multics*
- Is this claim true?
- Why not just use it?
- What is necessary?
- By whom?
- Can we make it happen?

- Claim: *We are still trying to solve the same security problems since Multics*
Answer?

- **Analysis Tools for systems and programs**
  - 1980s – 90s: formal verification methods
  - 2000s: Bug finding
  - 2010s: tools to find and fix security bugs?
  - Beyond??

- Problem: what bugs should be discovered?
- Problem: soundness and completeness
- Problem: how should analysis impact software development and system deployment?
Who Has a Role?

- Programmers (may be multiple groups)
- OS Distributors
- Administrators
- Users
- Service Providers
- Content Providers

**Challenge**: Must consider the balance between function and security
This course....

- Is a **software** course that teaches principles and techniques for verifying security properties

  - Lots of techniques have been developed, but we need to figure out how to use/extend them to improve systems security

  - **Topics**: What should “secure” mean in systems? How to find violations of security in programs and systems? How to fix such violations of security automatically? How to make such techniques tractable and practical?
Background

• Required:
  ‣ CSE 543

• Expected:
  ‣ Solid OS and PL background

• Additional:
  ‣ Willingness to read
    • We are going to read a lot of papers on security and analysis techniques
  ‣ Willingness to program
    • We are going to have some programming assignments
Course Materials

• Website
  ‣ [http://www.cse.psu.edu/~tjaeger/cse598-f11/](http://www.cse.psu.edu/~tjaeger/cse598-f11/)
  ‣ Course assignments, slides, etc. will be placed here
    • Check back often -- I may change some of the papers/assignments

• Readings
  ‣ Book: *Analysis Techniques for Security*
  ‣ Augmented with research papers
Course Calendar

- The course calendar has all the details
- Links to online papers for readings
- Links to projects
- Please check the calendar frequently
  - it’s the real-time state of the course

---

**Course Calendar**

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
<th>Assignments Due</th>
<th>Readings (read before class)</th>
<th>Slides</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/25/08</td>
<td>Introduction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/29/08</td>
<td>OS Security Enforcement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/1/08</td>
<td>Program Security Enforcement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/2/08</td>
<td>Enforcement in Practice</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/7/08</td>
<td>Security Goals</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/15/08</td>
<td>Security Challenge: Input</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/19/08</td>
<td>Security Challenge: Rootkit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/22/08</td>
<td>Security Challenge: Configuration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/26/08</td>
<td>Security Challenge: Command Injection</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/29/08</td>
<td>MAC OS Systems</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10/3/08</td>
<td>MAC OS Systems - SELinux</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10/9/08</td>
<td>OS and Program</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Below is the calendar for this semester course. It is the preliminary schedule, which may need to be altered as the semester progresses. It is the responsibility of the students to frequently check this web page for schedule, readings, and assignment changes. As the professor, I will attempt to announce any changes to the class, but this web page should be viewed as authoritative.

If you have any questions, please contact me (contact information is available at the course homepage).
Course Mailing List

• Via ANGEL
  ‣ Use with care

• I will send a test email
  ‣ Please reply if you do not receive by Fr
  ‣ May need to forward to your CSE account

• Can use to email me
  ‣ Please use “598” in the subject
Grading

- Exams (55%)
  - Midterm (25%)
    - Take home – do the readings
  - Final (25%)
    - In class
- Projects (35%)
  - 2 programming projects
  - Final project
- Presentations (15%)
Projects

• We are going to have three project deliverables
  ‣ Per person

• Topics
  ‣ Security analysis for programs
  ‣ Security analysis for systems (policies)

• Final Project
  ‣ Per my approval
Lateness Policy

- Assignments and project milestones are assessed a **20% per-day late penalty**, up to a maximum of 4 days. Unless the problem is apocalyptic, don’t give me excuses. Students with legitimate reasons who contact the professor before the deadline may apply for an extension.

- You decide what you turn in
Ethics Statement

- This course considers topics involving personal and public privacy and security. As part of this investigation we will cover technologies whose abuse may infringe on the rights of others. As an instructor, I rely on the ethical use of these technologies. Unethical use may include circumvention of existing security or privacy measurements for any purpose, or the dissemination, promotion, or exploitation of vulnerabilities of these services. Exceptions to these guidelines may occur in the process of reporting vulnerabilities through public and authoritative channels. Any activity outside the letter or spirit of these guidelines will be reported to the proper authorities and may result in dismissal from the class.

- When in doubt, please contact the instructor for advice. **Do not** undertake any action which could be perceived as technology misuse anywhere and/or under any circumstances unless you have received explicit permission from Professor Jaeger.
Road Map

• Introduction
  ‣ Current Attacks
  ‣ System Security Basics

• Static Analysis Techniques
  ‣ Foundations
  ‣ Detecting Bugs in Programs and Systems Policies
  ‣ Constraint Solving and Compiler Infrastructure

• More Advanced Problems
  ‣ Namespaces
  ‣ Attack Graphs

• More Advanced Analysis Topics
  ‣ Summary Functions, Runtime Analysis, Put It Together
Review

- Are we speaking the same language?

- General Terms
  - Principals/Subjects and Adversaries/Attackers
  - Trust Model
  - Threat Model
  - Security Model

- We will develop (semi-)formal models for each