April 3rd, 2003

Martin Nyström
Security Architect, Corporate Information Security
mnystrom@cisco.com

Who am I?
- Security Architect in Cisco’s internal InfoSec department
- Responsibility for consulting with application teams to secure their architecture
- Part of duty rotation responsible for monitoring infrastructure vulnerabilities
- Web services security architect
- 12 years developing application architectures
- Java programmer
- Grad student at NC State University
- Graduated Iowa State University – December 1990

Why worry?
- U.S. Army systems hacked using WebDAV vulnerability in IIS
  - It was a disturbingly successful attack, experts say, because the intruder found and exploited a flaw that took security researchers completely by surprise.
- Student charged with hacking at U. Texas
  - "It's a massive undertaking as to what [the hacker] did," he said, noting that identity theft is a growing problem nationwide. "All I need to steal your identity is your name and your Social Security number."
- Millions of credit card numbers compromised at DataProcessors International
- Utah ISP is victim of retaliation following hackers' attack on Al Jazeera
  - Impersonating an Al Jazeera employee, tricked the Web addressing company Network Solutions into making technical changes that effectively turned over temporary control of the network’s Arabic and English Web sites...

The goal of an attack
- Steal data
- Blackmail
- Beachhead for other attacks
- Bragging rights
- Vandalism
- Demonstrate vulnerability/satisfy curiosity
- Damage company reputation
- Others?

Commonly attacked services
- SMTP servers (port 25)
  - sendmail: "The address parser performs insufficient bounds checking in certain conditions due to a char to int conversion, making it possible for an attacker to take control of the application."
- RPC servers (port 111 & others)
- NetBIOS shares (port 139)
  - Opasoft worm
- FTP servers (ports 20, 21)
  - wuftp vulnerabilities
- SSH servers (port 22)
  - OpenSSH vulnerability
- Web servers (ports 80, 443)
  - Apache chunked encoding vulnerability
Web server attack

- Scan to find open ports
- Find out what’s running on open ports (banner grabbing)
- Profile the server
  - Windows (look for Kerberos, NetBIOS, AD)
  - Unix
  - Use TCP fingerprinting
- Probe for weaknesses on interesting ports
  - Default configuration files and settings (e.g., popular IIS ones)
  - Buffer overflows
  - Insecure applications
- Launch attack
  - Use exploit code from Internet…
  - …or build your own

Scanning…
What O/S is this system?

Example Web Application

Top 10 Web Application Security Vulnerabilities

1. Unvalidated parameters
2. Broken access control
3. Broken account/session management
4. Cross-site scripting flaws
5. Buffer overflows
6. Command injection flaws
7. Error handling problems
8. Insecure use of cryptography
9. Remote administration flaws
10. Web and app server mis-configuration
#1 Unvalidated parameters example

![Image of unvalidated parameters example]

#2: Broken Access Control

- Usually inconsistently defined/applied
- Examples
  - Insecure session IDs or keys
  - Forced browsing past access control checks
  - Path traversal
  - File permissions – may allow access to config/password files
  - Client-side caching

#3: Broken Account and Session Management

- Weak authentication
  - Password-only
  - Easily guessable usernames (admin, etc.)
  - Unencrypted secrets are sniffable
- How to break in
  - Guess password
  - Reset password
  - Have app email you new password
  - Sniff password
- Backend authentication
  - How database passwords are stored
  - Trust relationships between hosts (IP address can be spoofed, etc.)

#4: Cross-Site Scripting (XSS)

- Attacker uses a trust application/company to send malicious code to end-user
- Attacker can “hide” the malicious code
  - Unicode encoding
- 2 types of attacks
  - Stored
  - Reflected
- Wide-spread problem!
- Countermeasure: input validation
  - Positive
  - Negative: "< > ( ) # &"
  - Don’t forget these: "&lt &gt &#40 &#41 &#35 &#38"

#5: Buffer Overflows

- Mostly affects web/app servers
- Can affect apps/libraries too
- Goal: crash the target app and get a shell
- Example
  - [http://msfsecurity.pr.erau.edu/bom](http://msfsecurity.pr.erau.edu/bom)
- Buffer overflow example
  - [Code snippet]
- Countermeasures
  - Keep up with bug reports
  - Code reviews
  - Use Java

#6: Command Injection

- Allows attacker to relay malicious code in form variables or URL
  - System commands
  - SQL
  - Interpreted code (Perl, Python, etc.)
- Many apps use calls to external programs
  - sendmail
- Examples
  - Path traversal: ". . ."
  - Add more commands: ";, rm → r"
- SQL injection:
- Countermeasures
  - Taint all input
  - Avoid system calls (use libraries instead)
  - Run application with limited privileges
#7: Error Handling

- Examples: stack traces, DB dumps
- Helps attacker know how to target the app
- Inconsistencies can be revealing too
  - “File not found” vs. “Access denied”
- Fail-open errors
- Need to give enough info to user w/o giving too much info to attacker
- Countermeasures
  - Code review
  - Modify default error pages (404, 401, etc.)

## Daily News Example

```
Warning: Too many connections
in /web/include/classes/DBConnect_GF1.inc on line 14

Warning: mysql_query(): supplied argument is not a valid MySQL-Link
resource in /web/include/classes/DBConnect_GF1.inc on line 35
Too many connections

Warning: mysql_query(): supplied argument is not a valid MySQL-Link
resource in /web/include/classes/DBConnect_GF1.inc on line 35
Too many connections

Warning: mysql_query(): supplied argument is not a valid MySQL
result resource in /web/include/classes/StoryFetch.inc on line 23
Fatal error in class StoryFetch[1]
```

#8: Poor Cryptography

- Insecure storage of credit cards, passwords, etc.
- Poor choice of algorithm (or invent your own)
- Poor randomness
  - Session IDs
  - Tokens
  - Cookies
- Improper storage in memory
- Countermeasures
  - Store only what you must
  - Store a hash instead of the full value (SHA-1)
  - Use only vetted, public cryptography

#9: Remote Administration Flaws

- Problems
  - Weak authentication (username="admin")
  - Weak encryption
- Countermeasures
  - Don’t place admin interface on same server
  - Use strong authentication: certificates, tokens, strong passwords, etc.
  - Encrypt entire session (VPN or SSL)
  - Control who has accounts
  - IP restrictions

#10: Web/App Server Misconfiguration

- Tension between “work out of the box” and “use only what you need”
- Developers ≠ web masters
- Examples
  - Unpatched security flaws (BID example)
  - Misconfigurations that allow directory traversal
  - Administrative services accessible
  - Default accounts/passwords
- Countermeasures
  - Create and use hardening guides
  - Turn off all unused services
  - Set up and audit roles, permissions, and accounts
  - Set up logging and alerts