
On the Quality of Exploit Code
An Evaluation of Publicly Available Exploit Code,
Hackers & Threats II, February 17, 2:00 PM, San Francisco, CA
Ivan Arce, Core Security Technologies
RSA Conference 2005

OUTLINE

- Prologue: Context and definitions
- Why exploit code?
- Quality metrics
- Examples
- Epilogue: Future work

PROLOGUE

VULNERABILITIES & EXPLOITS

Lets start by defining a common language

- **Vulnerability** *(noun)*
  - "A flaw in a system that, if leveraged by an attacker, can potentially impact the security of said system"
  - Also: security **bug**, security flaw, security **hole**

- **Exploit** *(verb)*
  - "To use or manipulate to one's advantage" (Webster)
  - "A security hole or an **instance of taking advantage of a security hole**"

EXPLOIT CODE

Exploit code is not just “proof of concept”

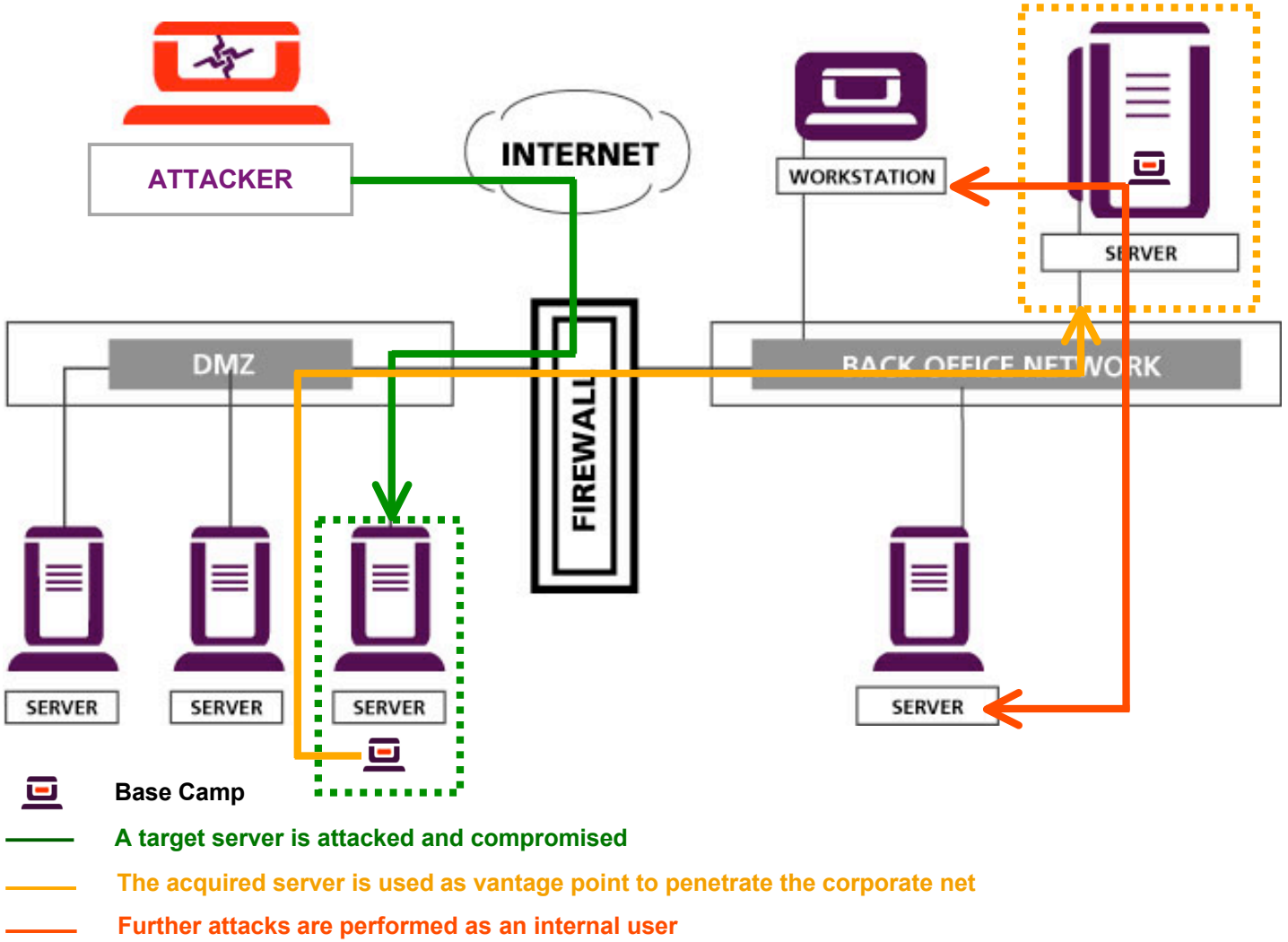
- **Proof of Concept exploit - PoC** *(noun)*
  - A software program or tool that exploits a vulnerability with the sole purpose of proving its existence.

- **Exploit Code** *(noun)*
  - A software program or tool developed to exploit a vulnerability in order to accomplish a specific goal.
  - Possible goals: denial of service, arbitrary execution of code, etc

WHY TALK ABOUT EXPLOIT CODE?

ANATOMY OF A REAL WORLD ATTACK

The classic attack uses exploit code...



EXPLOIT CODE FUNCTIONALITY

Exploit code becomes more sophisticated

- Add a simple “listen shell”

  echo "ingreslock stream tcp nowait root /bin/sh sh -i" >>/tmp/bob ; /usr/sbin/inetd -s /tmp/bob &"

- Add an account to the compromised system:

  echo "sys3:x:0:103::/:/bin/sh" >> /etc/passwd;

  echo "sys3:1WXmkX74Ws8fX/MFI3.j5HKahNqIQ0:12311:0:99999:7:::" >> /etc/shadow

- Execute a “bind-shell”
- Execute a “reverse shell”
- Deploy and execute a multi-purpose agent

  Command shell, FTP, TFTP, IRC, “zombies”, snifers, rootkits...

- Deploy and execute agent that re-uses existing connection.
- Deploy and execute agent that has low-level interaction with the OS
  - Syscall Proxing
- And more shellcode advances...
  - Loader payloads, InlineEgg, ShellForge, polymorphism, reusable components, encodings, etc.

A RECENT TREND IN THE INDUSTRY

Exploit code becomes a “valuable asset”

- Detailed information about vulnerabilities has value
- Exploit code is being bought and sold
- Included in commercial software offerings
- Exploit code development training
- Several books on exploiting software and exploit code development
  - “Exploiting Software”, Hoglund & McGraw
  - “The Shellcoder´s Handbook”, Koziol et. al.
  - “Hacking: The Art of Exploitation”, Jon Erickson

WHAT CAN I DO WITH MY EXPLOITS?

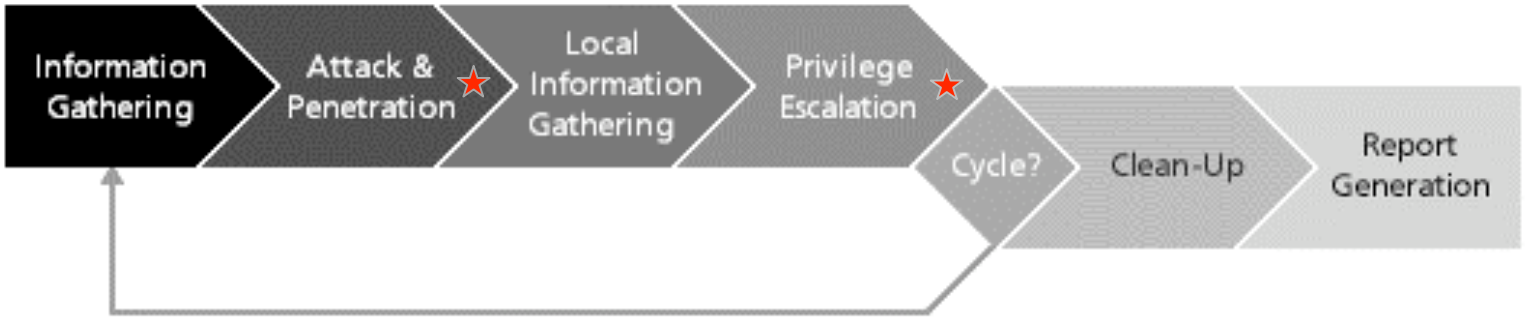
Some legitimate uses for exploit code

- Penetration Testing
- Test and fine-tune firewall configurations
- Test and fine-tune IDS configurations
- Test incident response capabilities
- Vulnerability Management

EXPLOIT CODE & PENETRATION TESTING

The penetration testing process

- Penetration Testing

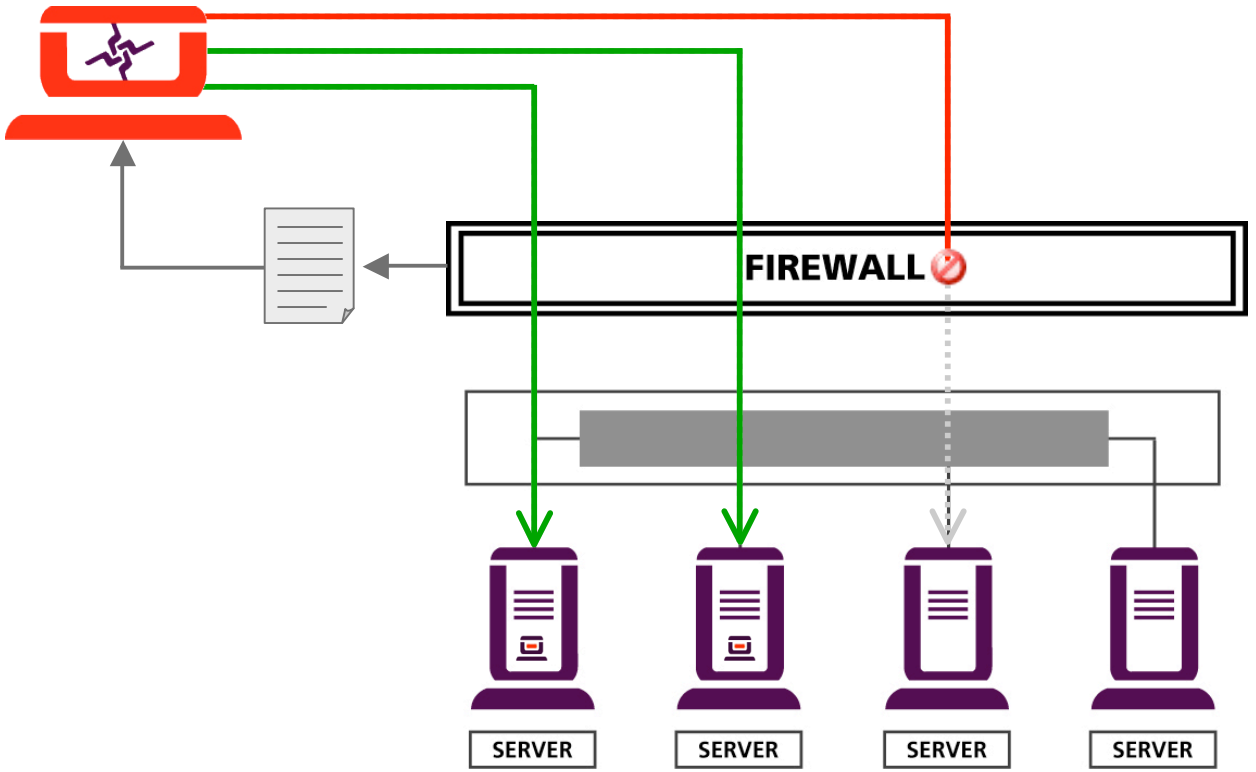


★ Using Exploits

EXPLOIT CODE & FIREWALLS

Using exploits to test and configure firewalls

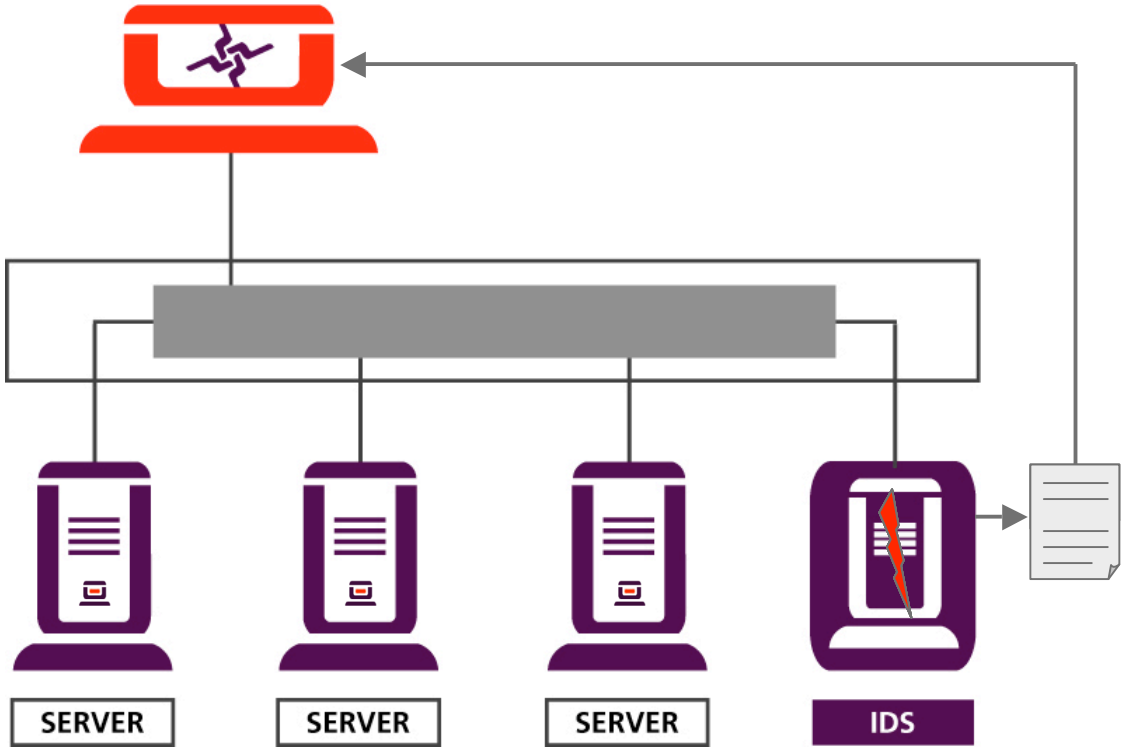
- Firewall configuration and testing



EXPLOIT CODE & IDS

Using exploits to test and configure Intrusion Detection Systems

- IDS configuration and testing



THE VULNERABILITY MANAGEMENT PROCESS

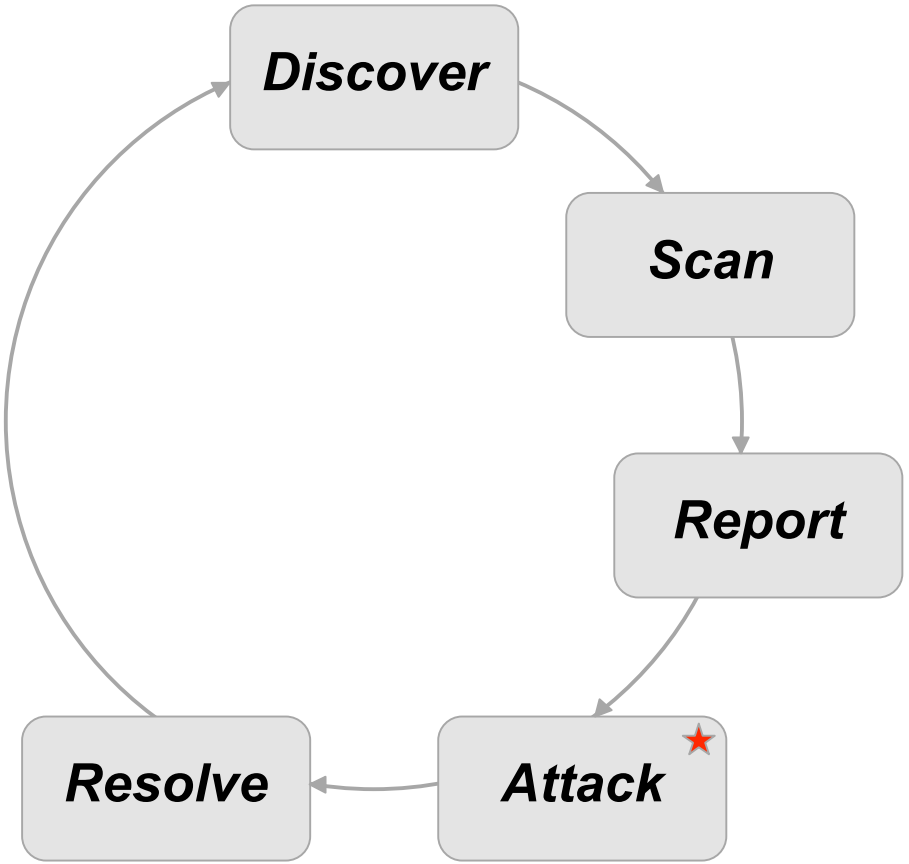
Vulnerability management: Scan & Patch strategy

- Vulnerability Management

IMPROVED VULNERABILITY MGMNT PROCESS

Use exploit code to minimize errors and prioritize better

- Vulnerability Management + Exploit Code

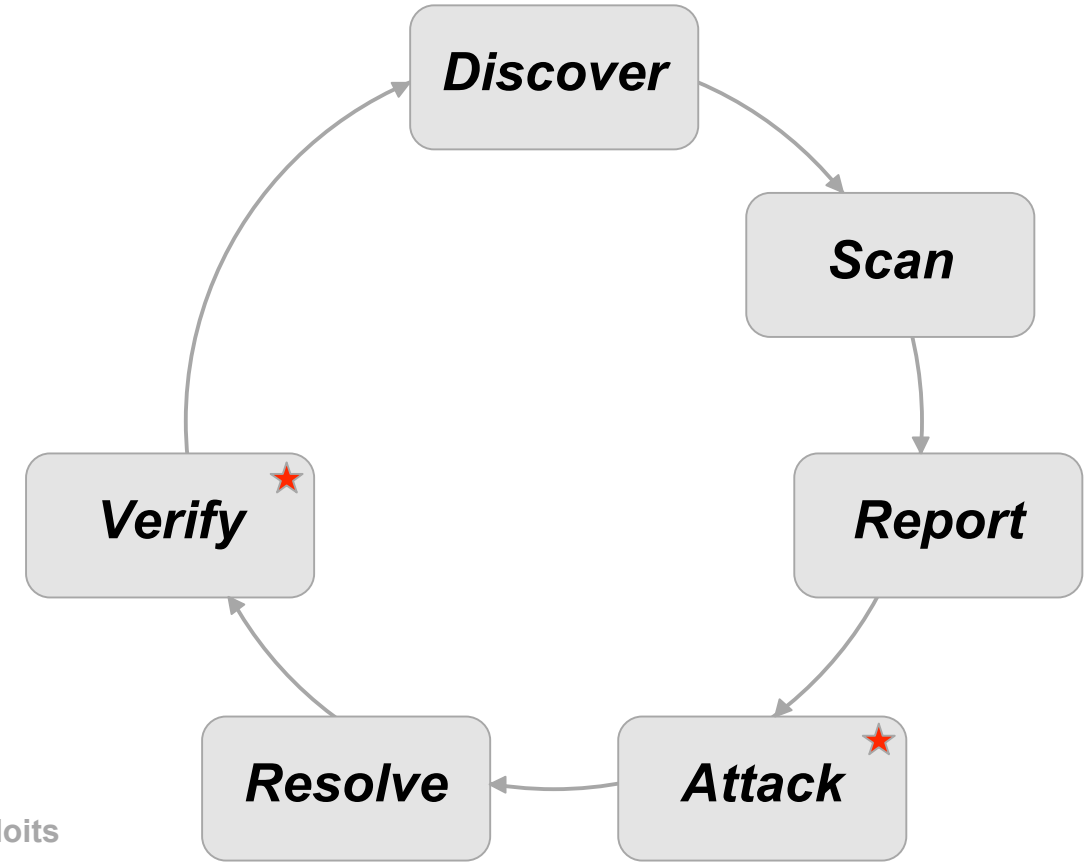


★ Using Exploits

AN ADDITIONAL IMPROVEMENT

Use exploit code to verify correct mitigation

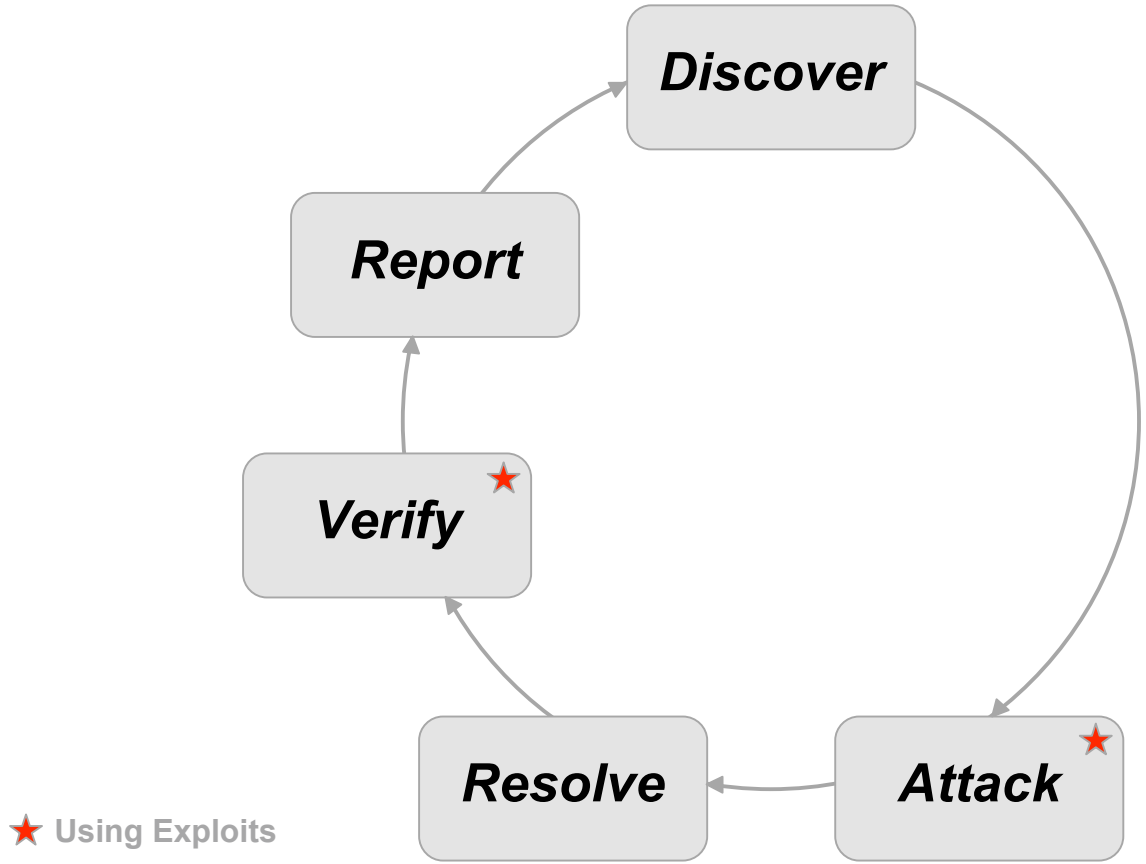
- Vulnerability Management + Exploit Code + Verification



VULNERABILITY MGMNT & PEN TESTING COMBO

Combine vulnerability management and penetration testing

- Vulnerability Management + Rapid Penetration Testing



QUALITY METRICS

QUALITY METRICS FOR EXPLOIT CODE

The legitimate uses of exploit code calls for quality metrics

- There are several legitimate uses for exploit code
- We need to understand the strengths and limitations of the tools we use
- A taxonomy helps organize our understanding of our tools
- Metrics provide a more objective way of measuring exploit code quality
- Caveat: Taxonomies and metrics are arbitrary

QUALITY METRICS FOR EXPLOIT CODE

What can we use the metrics for?

- Measure the quality of our exploits
- Comparative analysis
- Guidance for R&D
- Deploy timely and cost-effective mitigation measures
- Improve the security processes that use exploit code as components

EXPLOIT CODE INTERNALS

A few more definitions are needed...

- Remote exploit
  - A program or tool that does not require legitimate access to the vulnerable system in order to exploit the security flaw
- Exploit payload
  - The portions of the exploit code that implements the desired functionality after successful exploitation of a vulnerable system
  - Example payloads:
    - “add inetd service”
    - “add account”
    - “bind shell”
    - “reverse shell”

EXPLOIT CODE INTERNALS

A few more definitions are needed...

- Exploit attack vector
  - The means used by the exploit code to trigger the vulnerability on the target system

MS04-011 “Microsoft SSL PCT vulnerability” (CAN-2003-0719)
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0719
http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx
http://www.securityfocus.com/archive/1/361836

One vulnerability with seven attack vectors:

- MS IIS/Exchange ports

  https:443, smtp:25, imap:993, pop3:995, nntp:563

- MS Active directory ports

  ldaps:636, globalcatLDAPssl: 3269

EXPLOIT CODE INTERNALS

A few more definitions are needed...

- Exploit technique
  - The method used by the exploit code to alter the execution flow of a vulnerable system and force it to execute the exploit's payload.

Some exploit techniques

- Overwriting the stack memory
  - Read/write operations
  - Write/execute operations
  - Write operations
- Overwriting the heap memory
  - Read/write operations
  - Write/exec operations
  - Mirrored write operations
- Overwriting process flow control structures
  - Pointer overwrite (GOT, PLT, class pointers, destructors, atexit())
  - Program data overwrite (authorization keys, flags, credentials, FDs)

GENERIC QUALITY METRICS

These metrics can be used to assess the quality of exploit code

- Attack vectors
  - One
  - More than one
  - All
- Exploit logic
  - Brute-forcing vs. hard-coded addresses
  - Automatic OS fingerprinting vs. manual OS targeting
  - Network usage (number of connections, total bandwidth, etc.)
  - Total running time
  - Debugging capabilities, documentation, fixes
- Exploit technique and reliability
  - Some techniques are inherently more reliable than other
  - Lab testing under ideal conditions
    - 80% - 100%
    - 50% - 79%
    - 20% - 49%
    - Less than 20%

GENERIC QUALITY METRICS

Metrics related to network topology characteristics

- Network topology constrains
  - Link layer constrains (dialup, PPP, wireless, etc)
  - LAN vs. WAN
  - Attacker behind NAT device
  - Target behind NAT device
  - Target behind FW blocking incoming connections
  - Target behind FW blocking in/out connections
  - Target behind Proxy/Application gateway FW
  - IP Fragmentation/TCP Reassembly/Application-level Fragmentation
  - Network footprint
  - Latency
  - Constrained bandwidth

GENERIC QUALITY METRICS

Metrics related to the runtime environment of the vulnerable system/application

- Runtime environment
  - System load
  - Multi-threading
  - Fork & Exec
  - Multiplexing/Asynchronous service
  - File system access
  - Memory and file descriptors
  - Environment variables and command line arguments
  - Compile options, debugging, optimizations, logging
  - Service startup (manual, boot time, inetd, etc.)

GENERIC QUALITY METRICS

Metrics related to security hardened systems and services

- Security hardening measures
  - Vulnerable service runs as unprivileged process
  - Privilege separation/downgrade
  - Sand-boxing (chroot, jail, systrace, capabilities)
  - Non executable stack
  - Non executable heap
  - StackGuard, StackShield, ProPolice, Microsoft VS /GS flag
  - PaX, GrSecurity, W ^ X, DEP
- Portability and OS dependence
  - Exploit uses external helper libraries or programs?
  - Exploit run on specific OS?
  - Exploits requires local privileges?

GENERIC QUALITY METRICS

Metrics related to system stability

- System stability
  - After successful exploitation
    - Unstable service
    - Interrupted service
    - System reboot or halt
  - After unsuccessful exploitation
    - Unstable service
    - Interrupted service (One shot exploit)
    - System reboot or halt (One shot exploit & DoS)
- System pollution and clean-up
  - Modifies configuration
  - Modifies file system
  - Leaves audit trace
  - Stealthness

WINDOWS EXPLOITS: OS COVERAGE

OS coverage for exploits that target MS Windows

- Architecture
  - x86 - (32bit/64bit)
  - Multiprocessor kernels
- Operating System
  - WinNT, Win2k, WinXP, Win2003
- Operating System editions
  - WinNT 4.0: Workstation, Server, Enterprise, Terminal Server
  - Win2k: Professional, Server, Advanced Server
  - WinXP: Home, Professional
  - Win2003: Standard, Enterprise, Web
- Service Packs
  - WinNT 4.0: SP0-SP6,SP6a
  - Win2k: SP0-SP4
  - WinXP: SP0-SP2
  - Win2003: SP0-SP1
- Languages
  - English, Spanish, French , Portuguese, German, Japanese, Chinese

LINUX EXPLOITS: OS COVERAGE

OS coverage for exploits that target Linux

- Architecture
  - x86 - Intel IA32 (32bit), x86 - Intel IA64 (64bit), ARM, SPARC
- Linux Distribution
  - RedHat, Suse, Debian, Mandrake (Conectiva, Fedora, TurboLinux, Immunix, OpenWall, Gentoo, …)
- Linux Distribution versions
  - RedHat: 6.2, 7, 7.11, 7.2, 7.3, 8, 9
  - Suse: 7, 7.1, 7.2, 7.3, 8., 8.1, 9, 9.1
  - Debian: 2.0, 2.1, 2.2, 3
  - Mandrake: 7.1, 7.2, 8, 8.1, 8.2, 9, 10
- Kernel versions
  - Linux kernel 2.2.0 - 2.2.26
  - Linux kernel 2.4.0 – 2.4.26
  - Linux kernel 2.6.0 - 2.6.x
- User Space and Applications
  - Glibc and Gcc versions, default application versions, default compile options

SOLARIS EXPLOITS: OS COVERAGE

OS coverage for exploits that target Solaris

- Architecture
  - Intel x86, sun4m, sun4u
- Solaris versions
  - 2.5.1, 2.6, 7, 8, 9, 10
- Patch clusters and individual patches
- Software Packages and compiled applications
- Security settings
  - no_exec_user_stack = 1

EXAMPLES

MS RPC DCOM VULNERABILITY

The MS RPC DCOM vulnerability exploited by the Blaster worm

- Vulnerability: CAN-2003-0528, BID 8205

  Microsoft Security Bulletin MS03-026

  http://www.microsoft.com/technet/security/bulletin/MS03-026.mspx

- Vulnerable Systems

  winNT 4, winNT4 Terminal Services, win2k, winXP,win 2003

- Attack vectors

  Ports 135/tcp, 135/udp, 139/tcp, 445/tcp, 593/tcp, 80/tcp, >1024/tcp

  Plus 135/udp broadcast

- Publicly available exploit code
  - winrpcdcom.c (FlashSky, xfocus.org)
  - dcom.c (HD Moore, modified from xfocus.org)
  - msrpc_dcom_ms03_026.pm (HD Moore, included in metasploit 2.0)
  - Rpcexec.c (ins1der, trixterjack at yahoo.com)
  - dcom48.c (OC192 www.k-otik.com)

MS RPC DCOM VULNERABILITY

The MS RPC DCOM exploit used by the Blaster worm

- Exploit: dcom.c (written by FlashSky, Benjurry, re-written by H.D. Moore)
  http://downloads.securityfocus.com/vulnerabilities/exploits/30.07.03.dcom.c
- Attack vectors: 1/8
- Exploit logic
  Hard coded addresses, no brute forcing, manual OS selection
  1 exploitation attempt per connection
- Exploit payload: Bindshell (port 4444/tcp)
- Exploit technique: Stack smashing,return address overwrite, stack exec.
- Exploit targets: 11/385 (30/385 maximum)
  - Windows 2000 SP0-SP4 (English): 5
  - Windows 2000 SP2-SP4 (German): 4
  - Windows XP SP0-SP1 (English): 2
  - Windows XP SP1 (German): 1
- Other:
  Leaves service unstable (one shot), halt/reboot of system, ingress filtering blocks shell access,
  OS independent (FlashSky's original was OS dependant), requires additional FDs
  The worm required file system and registry access, deployed multi-purpose agent

MS LSASS VULNERABILITY

The MS LSASS.EXE vulnerability exploited by the Sasser worm

- Vulnerability: CAN-2003-0533, BID 101108

  Microsoft Security Bulletin MS04-011

  http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx

  http://www.eeye.com/html/Research/Advisories/AD20040413C.html

- Vulnerable Systems

  win2k, winXP,win 2003

- Attack vectors

  Ports 139/tcp, 445/tcp

- Publicly available exploit code
  - HOD-ms04011-lsasrv-expl.c (houseofdabus)
  - ms04011lsass.c (www.k-otik.com)

MS LSASS VULNERABILITY

The MS LSASS.EXE exploit used by the Sasser worm

- Exploit: HOD-ms04011-lsasrv-expl.c (houseofdabus)
  http://downloads.securityfocus.com/vulnerabilities/exploits/HOD-ms04011-lsasrv-expl.c
- Attack vectors: 1/2
- Exploit logic
  Hard coded addresses, no brute forcing, manual OS selection (-t option just prints output)
  1 exploitation attempt per connection
- Exploit payload: Bindshell on user specified port (reverse-shell optional)
- Exploit technique: Stack smashing,return address overwrite.
- Exploit targets: 56/161 (119/161 maximum)
  - Windows 2000 SP2,SP4: 2 (1 “universal” address + Adv. Svr.)
  - Windows XP SP0,SP1: 1 (1 “universal” address)
- Other:
  Leaves service unstable (one shot), halt/reboot of system, ingress filtering blocks shell access,
  OS dependant (previous exploit used modified system DLL), requires additional FDs, requires access to target’s IPC$ endpoint
  The worm required file system and registry access, deployed multi-purpose agent, bindshell on port 9996/tcp, FTP server on port 5554/tcp

EPILOGUE

EPILOGUE

Conclusion and future work

- Conclusion
  - There are several legitimate uses for exploit code
  - We need to understand the tools we use
  - We propose a set of metrics to measure quality of exploit code
- Future work
  - Refine the proposed metrics
  - Test them against publicly available exploits
  - Comparative analysis
  - Extend into a model with more quantifiable parameters and possibly a suitable “QoE”metric

THANK YOU!

CONTACT INFORMATION

Iván Arce | ivan.arce@coresecurity.com | www.coresecurity.com

Headquarters · Boston, MA
46 Farnsworth St
Boston, MA 02210 | USA
Ph: (617) 399-6980 | Fax: (617) 399-6987
info@coresecurity.com

Research and Development Center
Argentina (Latin America)
Florida 141 | 2º cuerpo | 7º piso
(C1005AAC) Buenos Aires | Argentina
Tel/Fax: (54 11) 5032-CORE (2673)
info.argentina@coresecurity.com

www.coresecurity.com