Web vulnerability scanning and exploitation tools
Scaling vulnerability scanning

- Companies with 1000+ web applications running
  - Move to μ-services architectures making things worse
- Huge shortage of skilled security engineers to perform red-team (adversarial) analysis
- Hackers employing automation to speed compromise
  - Equifax (admin/admin) or Mirai default usernames and passwords discovery
  - Shodan scans and reveals the same
- Must increasingly employ automation in security (i.e. use software to improve security)
Word of caution

- Must not rely solely on what tools find
- Tools can not automatically solve all of your labs
- Tools are very loud
  - Can crash stuff
  - Can do things like print 9000 pages on a printer
- Penetration testing requires creative humans of diverse disciplines and modes of thinking
  - Example: social engineering methods
Kinds of tools

- **Command-line web vulnerability scanning and auditing**
  - `nmap` (via NSE scripts)
  - `nessus` (OpenVAS)
  - `nikto`
  - `w3af`
  - `WPS`can (WordPress)

- **Proxy-based web vulnerability scanners**
  - `zap`

- **Command-line exploitation tools**
  - `metasploit` (general)
  - `sqlmap` (database)

- **Command-line password brute-forcing**
  - `hydra`
nmap

- **Open-source network scanner**
  - For target discovery typically
  - Scan huge networks of literally hundreds of thousands of machines

- **Portable, flexible, extensible**
  - Plug-in scripts to allow for web scanning

- **Uses raw IP packets in novel ways**
  - To determine what hosts are available on the network,
  - What services those hosts are offering
  - What operating systems and versions are running
  - What type of packet filters/firewalls are in use
  - Many of other characteristics.
nessus (OpenVAS)

• **Free, open-source vulnerability scanner**
  • Free version of nessus at [https://tenable.com/products/nessus-home](https://tenable.com/products/nessus-home)
  • Does both operating system and web vulnerabilities
  • Vulnerability checks are modularized via plug-ins
    • 20,000+ plug-ins in Nessus vulnerability database
  • Customizable – user can write new plug-ins
    • In C
    • In Nessus Attack-Scripting Language (NASL)
wikto

- **URL:** http://cirt.net/nikto2
- **Vulnerability scanner for web servers**
  - Similar to Nessus - runs off plug-ins
- **Tests for:**
  - Web server version
  - Known dangerous files/CGI scripts
  - Version-specific problems
Web Application Attack Audit Framework

- Python-based tool for securing web applications
  - Portable across Windows, OS X, Linux, OpenBSD, etc.
- Phases supported:
  - Discovery: *Finding new URLs, forms, and other “injection points”*.
  - Audit: *Probe injection points by sending* crafted data into all of them to find vulnerabilities.
  - Attack: Exploit vulnerabilities found
- Integrations with Metasploit and sqlmap
w3af

**audit**
xrft
htaccessMethods
sql
sslCertificate
fileUpload
mxeInjection
generic
localFileInclude
unSSL
xpath
osCommanding
remoteFileInclude
dav
ssi
eval
buffOverflow
xss
xst
blindSql
formatString
preg_replace
globalRedirect
LDAPi
phishingVector
frontpage
responseSplitting

grep
dotNetEventValidation
pathDisclosure
codeDisclosure
blankBody
metaTags
motw
privateIP
directoryIndexing
svnUsers
ssn
fileUpload
strangeHTTPCode
hashFind
getMails
httpAuthDetect
wsdlGreper
newline
passwordProfiling
domXss
ajax
findComments
httpInBody
strangeHeaders
lang
errorPages

**collectCookies**

**Exploit**
sqimap
osCommandingShell
xssBeef
localFileReader
rfiProxy
remoteFileIncludeShell
davShell
eval
fileUploadShell
sql_webshell

Also…………

discovery, output, mangle,
bruteforce, evasion
WPScan

- Black box WordPress vulnerability scanner
  - https://wpscan.org/
  - WordPress and its plug-ins are extremely popular targes
  - Checks for CVEs specific to WordPress
zap

- **OWASP Zed Attack Proxy**
  - Open-source web proxy for capturing and modifying traffic from a browser
  - Provides automation for finding security vulnerabilities in web applications
  - Similar to Burp Suite

- **Setup**
  - Automatically listens on port 8080
  - Point web browser HTTP proxy settings to port 8080
  - Requests sent by browser captured in Zap for subsequent replay
zap
Metasploit

- Defacto tool for penetration testing
- Framework for exploiting vulnerabilities
- Attack scripts written in Ruby
- Contains a rich set of modules organized in systematic manner
- 1000 + exploits, 200 + Payloads, 500+ Auxiliary Modules
Metasploit CLI

Validate lots of vulnerabilities to demonstrate exposure with Metasploit Pro -- Learn more on http://rapid7.com/metasploit

metasploit v4.14.22-dev
[ 1658 exploits - 947 auxiliary - 293 post ]
[ 486 payloads - 40 encoders - 9 nops ]
[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
Exploits

- Actual code which works on the target vulnerability system.
- Modular organization based on OS and service classification
  /usr/share/metasploit-framework/modules/exploits
- Ranked to determine reliability of exploit for success
  - Manual, Low, Average, Normal, Good, Great, Excellent
**Encoders**

- How to encode payload and morph it to bypass anti-virus and detection

/usr/share/metasploit-framework/modules/encoders
Payloads

- **What to run on target after initial exploit**
  
  `/usr/share/metasploit-framework/modules/payloads`
  
  - Web shell, stager to download additional code
  - Meterpreter
    - Common payload for Windows
    - Provide an enhanced, extensible shell for adversary
    - Delivers common post-exploitation functionality via an injected DLL onto victim machine
Example use

Attacker

Exploit + 1st Stage Meterpreter Payload

Payload Connects Back to Metasploit

2nd Stage DLL Injection Payload Sent

Metasploit Sends Meterpreter Server DLL

Client And Server Communicates

Victim
Post-exploitation

- Perform additional operations after gaining access
  /usr/share/metasploit-framework/modules/post
- Gather information about exploited system
- Enhance environment
  - Privilege escalation
  - Credential stealing (password manager hacking)
  - Key-logging
  - Activity viewing
  - Web camera
  - Desktop capture (screen_spy)
- Operating system specific
Auxiliary

- Additional functionality for...
  - Scanning
  - Fuzzing/brute-forcing
  - Crawling
  - Sniffing
  - Password guessing

/usr/share/metasploit-framework/modules/auxiliary
Plug-ins

- For popular third-party apps
  - nessus
  - nexpose
  - OpenVAS

/usr/share/metasploit-framework/modules/plug-ins
Demo video
sqlmap

- Automate detection and exploitation of SQL injections
  - Form submission via GET
    sqlmap -u <URL> -p <injection parameter>

    $ sqlmap -u 'http://foo.com/view.php?id=1141' -p id

  - Form submission via POST
    sqlmap -u <URL> --data=<POST_DATA> -p <injection parameter>

  - Will automatically try Blind SQL injection on all fields to dump entire database
Hydra

- Parallelized network authentication cracker
- Supports Cisco auth, HTTP, IMAP, RDP, SMB, SSH, LDAP, MySQL, VNC
- Uses dictionaries of dumped usernames and passwords
- Does brute-force attacks

```
# hydra

Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only


Options:
- -1 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 be attacked in parallel, one entry per line
- -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 server (use either 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 asf cisco cisco-enable cvs firebird ftp ftps http[s]-{head|get} http[s]-{get|post}-form http-proxy http-proxy-urlenum icq imap[s] irc ldap2[s] ldap3[-{cram|digest|md5}[s] mssql mysql ncp nntp oracle-listener oracle-sid pcanywhere pcmfs pop3[s] postgres rdp rexec rlogin ssh s7-300 sip smb smtp[s] smtp-enum smtp snmp socks5 ssh sshkey svn teamspeak telnet[s] vmauthd vnc xmpp
```
Hydra

- Can also supply a list of usernames and passwords to it

  `hydra -L users.txt -P pass.txt ssh://foo.com`

- HTTP basic-auth example

```bash
# hydra -L users.txt -P pass.txt http-get://localhost/
Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2015-02-10 15:11:57
(DATA) 1 task, 1 server, 1 login try (1:1/p:1), ~1 try per task
(DATA) attacking service http-get on port 80
[80][www] host: 1.2.3.4 login: user password: tester
1 of 1 target successfully completed, 1 valid password found
```
Services

- Third party sites for vulnerability scans
  - Free
    - [https://www.scanmyserver.com/](https://www.scanmyserver.com/)
    - [https://www.qualys.com/forms/freescan/](https://www.qualys.com/forms/freescan/)
    - [https://app.webinspector.com/](https://app.webinspector.com/)
  - Pay
    - Tenable (Nessus Pro)
    - Netsparker
    - Acunetix
    - Rapid7 (Nexpose, Metasploit Pro)
- SSL
  - [https://www.ssllabs.com/ssltest/](https://www.ssllabs.com/ssltest/)
Web application firewalls
Web application firewalls

- **Function**
  - Proxy incoming connection
  - Pull in request
  - Examine request for common exploitation payloads and block automatically
  - Forward request to destination if OK
  - Often part of Layer-7 load balancing (i.e. application layer)
Examples

- **Open-source**
  - modsecurity
    - [https://modsecurity.org/](https://modsecurity.org/)
    - Prevent XSS, SQL injection, other common attacks
    - Toss requests based on OWASP’s modsecurity core rule set
    - For efficiency, throw out rules your site does not need
  - NAXSI
    - [https://github.com/nbs-system/naxsi](https://github.com/nbs-system/naxsi)
    - Prevents XSS and SQL Injection
  - Shadow Daemon
    - [https://shadowd.zecure.org](https://shadowd.zecure.org)
    - Prevents SQL/XML/Code/Command injection, XSS, local/remote file inclusion

- **Commercial**
  - CloudFlare, Barracuda, AWS
Labs

- Handout walkthrough
GCP labs

- Set up kali, wfp1, and wfp2 VMs
- Set up a VM to run a docker image of vulnerable Apache Struts server (cve-2017-5638)
- Lab #1: Use metasploit on kali VM to...
  - Compromise Apache Struts server
  - Perform a directory scan of wfp1 VM
  - Brute-force the HTTP authentication on wfp2 VM’s Authentication #1 example
- Lab #2: Use sqlmap on kali VM to
  - Solve wfp1’s SQL injection #1 example
  - Solve wfp1’s SQL injection #2 example
  - Solve natas15’s Blind SQL injection level (please do in pairs)
- Lab #3: Use hydra to
  - Brute-force the HTTP authentication on wfp2 VM’s Authentication #1 example
linuxlab labs (for CS 510 students)

- Download a kali VM image via BitTorrent
- Bring kali VM up in VirtualBox
- **Lab #1: Use WPScan on kali VM to**
  - Find all of the known vulnerabilities in a given WordPress installation
- **Lab #2: Use zap and firefox on kali VM to**
  - Solve wfp1’s SQL injection #1 example
  - Solve one of the other SQL injection levels in wfp1 or wfp2
  - Solve a level in Google’s XSS firing range
  - Solve wfp1’s XSS #1 example
  - Launch a command injection on WebScantest’s test page
- **Lab #3: Use w3af to**
  - Identify vulnerabilities on wfp1 in two OWASP categories
  - Identify one XSS vulnerability on Google’s XSS firing range
- **Optional:** [https://flaws.cloud](https://flaws.cloud)
linuxlab labs (CS 510)

- Extra credit labs flaws.cloud
Questions

- [https://sayat.me/wu4f](https://sayat.me/wu4f)
Extra
Homework: nmap

```python
import socket

target = input('Enter the IP address to scan: ')
portrange = input('Enter the port range to scan (es 5-200): ')

lowport = int(portrange.split('-')[0])
highport = int(portrange.split('-')[1])

print('Scanning host ', target, ' from port', lowport, ' to port', highport)

for port in range(lowport, highport):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    status = s.connect_ex((target, port))
    if(status == 0):
        print('** Port', port, ' - OPEN **')
    else:
        print('Port', port, ' - CLOSED')
s.close()
```
Lab: nikto

- **Install nikto on linuxlab**
  - `wget https://github.com/sullo/nikto/archive/master.zip`
  - `unzip master.zip`
  - `cd nikto-master/program`
  - `./nikto.pl`
    - Point it at several URLs in WFP1 and WFP2
Lab: nikto

- Run nikto on each of the instances deployed via its Internal IP address
  - nikto –h http://w.x.y.z
- Answer the following questions
  - Briefly compare the outputs generated by each of the deployed web servers.
    - What software versions differ?
    - Are there any vulnerabilities?
  - Provide one screenshot of each tool’s output
Do not use

- Run `w3af_console` on a Web for Pentester 1 instance the instructor gives you
- Use tool to identify an XSS vulnerability and a command injection automatically

```bash
w3af>>> plugins audit xss
w3af>>> target set target http://10.138.0.2/xss/example1.php?name=hacker
The configuration has been saved.
w3af>>> start
A Cross Site Scripting vulnerability was found at: "http://10.138.0.2/xss/example1.php", using HTTP method GET. The sent data was: "name=" The modified parameter was "name". This vulnerability was found in the request with id 37.
Scan finished in 8 seconds.
Stopping the core...
w3af>>> []
```

```bash
w3af>>> plugins audit os_commanding
w3af>>> target set target http://10.138.0.2/commandexec/example1.php?ip=127.0.0.1
The configuration has been saved.
w3af>>> start
OS Commanding was found at: "http://10.138.0.2/commandexec/example1.php", using HTTP method GET. The sent data was: "ip=%3B%2Fbin%2Fcat%20%2Fetc%2Fpasswd" The modified parameter was "ip". This vulnerability was found in the request with id 45.
Scan finished in 23 seconds.
Stopping the core...
w3af>>> []
```
Add to Recon
PTES

- Penetration testing execution standard
- Many tools across many protocols
Finding targets

- **DNS**
  - robtex, netcraft
    - Third-party services for finding subdomains
  - censys
    - Third-party service for finding subdomains via brute-forcing cloud IP addresses to get TLS certs
  - sublist3r
    - Tool for Google/Bing/Baidu searching for subdomains
  - knockpy
    - Tool for brute-forcing subdomains via dictionary
Finding targets

- **Vulnerable users**
  - E-mail addresses (simplyemail)
    - HR and account/order management, accounts payable addresses
    - Example

How a Single Email Stole $1.9 Million from Southern Oregon University
Finding targets

- **Vulnerable users**
  - Social media profiles and job postings for security engineers in company
    - Reveals the technology (anti-virus) being run in enterprise
    - LinkedIn, Monster, Twitter, Google+, FB
  - Information on people in company
    - pipl.com
    - Great for monitoring if someone is stealing your ID?

- **Calling in to gather intelligence on technology**

- **Tailgating and implanting physical devices**
  - Smokers and a Raspberry Pi with kali that phones home (Kim)
Finding targets

- **API keys**
  - Searching “aws key” in github
  - Truffle Hog, Git-Secrets, GitAllSecrets
  - Google dorking
    - `filezilla inurl:recentServers.xml` to find creds that are remembered
    - `filetype:pdf "Assessment Report" nessus` to find vulnerability reports
    - `inurl:login` to find all login pages
    - Strings within https://github.com/JohnTroony/Google-dorks/blob/master/google-dorks.txt
Finding targets

- **All-purpose tools (discover)**
  - Aggregates information found with
    - dnsrecon (includes squatting reports)
    - goofile, goog-mail, goohost
    - theharvester
    - urlcrazy, urlvoid
    - whois
    - dnssy
    - ewhois
    - myipneighbors
    - recon-ng (includes known breached usernames/passwords)
      - cnn.com
Finding targets

- **All-purpose tools (discover)**
  - Example

mark.reed@cnn.com => Breach found! Seen in the River City Media Spam List breach that occurred on 2017-01-01.
[*] [contact] <blank> <blank> (mark.reed@cnn.com) - <blank>
[*] [credential] mark.reed@cnn.com: <blank>
[*] test@cnn.com => Breach found! Seen in the Adobe breach that occurred on 2013-10-04.
[*] test@cnn.com => Breach found! Seen in the iMesh breach that occurred on 2013-09-22.
[*] test@cnn.com => Breach found! Seen in the LinkedIn breach that occurred on 2012-05-05.
[*] test@cnn.com => Breach found! Seen in the MySpace breach that occurred on 2008-07-01.
[*] test@cnn.com => Breach found! Seen in the River City Media Spam List breach that occurred on 2017-01-01.
[*] test@cnn.com => Breach found! Seen in the vBulletin breach that occurred on 2015-11-03.
[*] [contact] <blank> <blank> (test@cnn.com) - <blank>
[*] [credential] test@cnn.com: <blank>