

Network and Web Security

Pentesting

Dr Sergio Maffeis

Department of Computing

Course web page: <https://331.cybersec.fun>

Pentesting

- **Penetration testing:** pay someone to break into your system/organisation and report weaknesses
 - Network security, physical access, social engineering...
- Important to scope the pentesting exercise and avoid operational damage
 - Restrictions on targets to attack, tools and techniques to use, admissible side effects
- Access to information
 - Black box: no information
 - Gray box: selected information to help focus the exercise
 - White box: access to source code, system architecture, protocols, valid accounts
- Hard to ensure that pentester “tried hard enough”
 - Sometimes pentesting teams are played against each other
 - Pentester certifications (such as CISSP, <https://www.isc2.org/cissp/>) commend higher fees
 - Penetration Testing Execution Standard (PTES): fundamental principles and technical guidelines for penetration testing: <http://www.pentest-standard.org/index.php>

- Key steps
 1. Pre-engagement interactions
 - Sign contract, define scope, agree on rules of engagement
 2. **Intelligence gathering**
 3. **Threat modelling**
 4. **Vulnerability analysis**
 5. **Exploitation**
 6. **Post-exploitation**
 7. Reporting
 - Provide actionable intelligence to the customer: executive summary, technical report, etc
- We use penetration testing as a way to
 - Find weaknesses in our own web applications
 - Gain insight into attacker's approach
 - To inform better defenses



Attackers compared

	Black hat	Gray hat	White hat
Definition	<i>Penetrates a system without permission</i>	<i>Penetrates a system without permission but with good intentions</i>	<i>A pentester doing her job</i>
Is it legal?	No	Still no!	Yes
Avoid detection	Yes	Yes	Depends on job spec
Constraints	None	Avoid damaging target	Restriction on what, when, how to attack
Tools	Paid-for exploit kits 0-days Secret tools	Limited resources Public domain tools	Professional tools Insider information
Disclosure	None	Responsible, full	Confidential report

2. Intelligence gathering

- **Passive** information gathering
 - Aim to build a DFD, a network map, an architectural diagram, an organization chart, a sociogram of the target
 - Avoid any interaction with the target to avoid suspicion
 - To be sure, you can block your access to target using a proxy or firewall
 - Look for publicly available information about the target
 - Search online presence of company and/or employees (social networks, blogs), for comments, emails, company roles/data
 - Locate the target Web presence (services, webpages)
 - Use search engines cache, archive.org, alexa.org
 - Look at the source code of webpages for comments, hidden form fields, links
 - Find out what protocols are used: HTTP(S), FTP, SMTP
 - Look for any uptime statistics sites
 - Query the domain registrar, look for reverse DNS information
 - Look for comments in open source code used by the target
 - Are there open bugs?
 - Hardcoded credentials in older versions of the code
 - Caution: even publicly accessible data may be protected by law

Google Hacking

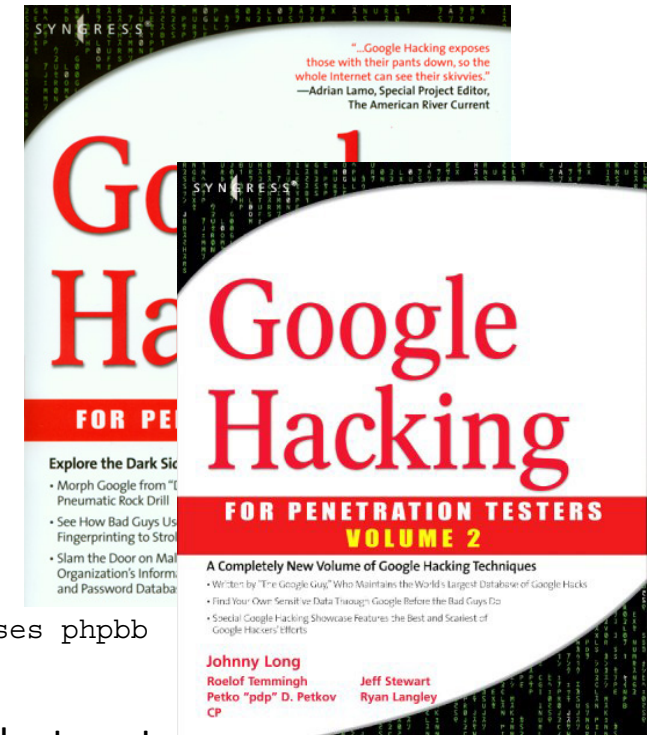
- 1,084 pages on passive information gathering (and more) with Google
- Some advanced operators

- Search for files with specific extensions
`ext:pdf`
- Search within a given website only
`site:example.com`
- Search inside url or title or body of a page
`"index.html" inurl: -html`

- Identifying potential targets using operators
 - Locate (unintentional) exposed directory listings
`intitle:index.of "parent directory"`
 - Locate sites running software known to be vulnerable
`allintext:"Powered by phpbb"`
`inurl:index.asp` Search for all index.asp pages that uses phpbb

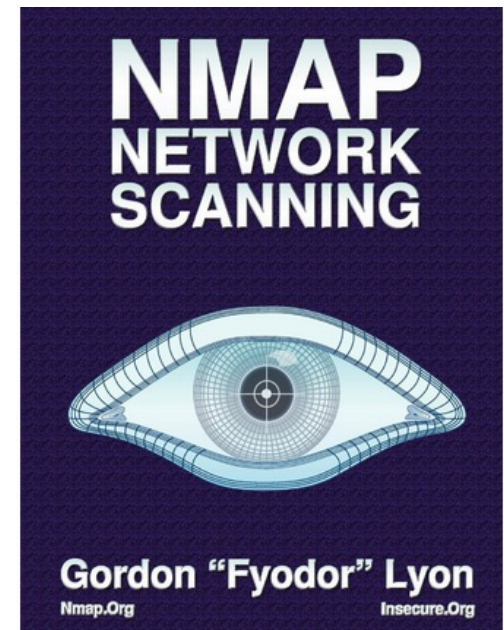
- Cache search
 - The code of cached webpages can be seen without accessing the target
 - Use proxy to prevent loading uncached elements (images)
- Google slows down some “interesting” queries using CAPTCHAs

Looking for
index.html in the
content without
having 'html' in
the URL itself



2. Intelligence gathering

- **Active** information gathering
 - Other useful information can be obtained by contacting target services
 - These activities may be detected as suspicious
 - Better to do it from IP addresses that you will not use for exploitation
 - What **version** is the **DNS** server running?
 - `dig` query to target DNS server
 - Verify potential email addresses, user names
 - Send an email, see if it bounces
 - Determine network perimeter
 - Are you accessing the target directly, or via a firewall?
 - Use `traceroute`, reverse DNS to identify intermediate hosts
 - Probe the network
 - Host discovery: identify what subnet addresses are active
 - Port scanning: identify what ports accept communications
 - OS fingerprinting: identify target OS
 - Identify services
 - *Banner grabbing*: some services send identifying information by default
 - By reverse-engineering the protocol
 - Sometimes it's enough to send a random string and observe the error msg
 - We'll see some of these in the network lab next week



4. Vulnerability analysis

- Unpatched systems
 - Search in the CVE database for vulnerabilities affecting any identified component
 - Scan for known vulnerabilities using tools
 - Nessus, Nikto, W3AF, Metasploit scanner, Nmap scripts
 - Beware that vulnerability scanners typically send a lot of probes, so are easy to detect
 - Same as above for exposures (misconfigurations)
- Patched systems
 - If source code is available, perform code review
 - By hand: we'll do that for JavaScript and PHP
 - Using static analysis tools
 - This activity can be very time consuming
 - Try to trigger unknown vulnerabilities
 - Using educated guesses: we'll do that for SQLi, XSS
 - Fuzzing input parameters: automated tools help with this task
- Obtain credentials
 - Investigate password policies, identify default passwords
 - Look for password hashes published by hackers (offline dictionary attack)

5. Exploitation

- Gain access to the target system by exploiting the identified vulnerabilities
- Use valid credentials, or try brute-force (online dictionary attack)
- Run publicly available exploits
 - From databases: <https://www.exploit-db.com/>
 - Using automated tools, like Metasploit
 - Use exploits tailored to verified vulnerabilities, don't just throw anything the tool has at the target (easy to detect)
- Build your own exploits
 - We'll do that for PHP, JavaScript, SQLi, XSS ...



6. Post-exploitation

- Well done, you're in! What now?
- If the exploited account is not admin, try **privilege escalation**
 - **pass-the-hash** technique in Windows
 - send the admin's hashed password to an internal authentication server
 - the server verifies the hash and grants you admin access
 - **Confused deputy**: exploit application that runs as *root* or *system*
- Steal data
 - Explore local and network disks for interesting files
 - Install keylogger
 - Capture screenshots, access webcam
- Send data back to hacker
 - Password hashes for off-line attacks
 - Other valuable data
 - Use information-hiding techniques to avoid detection
- Pivot: use compromised host to exploit other targets on LAN
- Maintain access
 - Open backdoors, reverse shells
 - Create new user accounts to login later
- Cover your tracks
 - Manipulate logs
 - Install rootkits to hide backdoors
- We'll practice some of these in the lab

