1

a)

i) - Collect information to be used in pen-testing (vulnerable software versions, open ports)

* Recognize you for authentication (warning when logging in from unknown place)
* Access control for particular fingerprints
* Serve personalised ads
* Remember personal website settings
* Identify who you are

Web fingerprinting is one of the techniques used in web tracking.

Web tracking is one of the goals of web fingerprinting.

ii) Web server fingerprinting:

* User agent
* Accept
* Accept-Language
* Accept-Encoding
* Content language
* Referer

Strengths: Always available to collect, since sent every request. Cannot be detected by the target. Doesn’t affect the target.

Weaknesses: Easier to spoof, limited number of properties.

JavaScript fingerprinting:

* DOM properties
* Navigator properties
* Plugins
* Screen size
* WebGL info
* Input devices
* Sensors
* Canvas rendering

Strengths: More unique/precise features that are more difficult to spoof correctly & consistently. Can detect anti-fingerprinting.

Weaknesses: Can be detected/prevented by target and may disrupt target (such as rendering fonts/objects/etc)

b) Practical part, can only assume.

Fingerprint: Some (maybe hashed) combination of browser properties.

Possibly stored as a cookie.

Spoofing your fingerprint: Modify properties (e.g. userAgent), by going into the settings, or intercepting the request with burp/wireshark

ii) Search own root directory?

Implement some fingerprinting.

2)

a)

i) Security HTTP headers:

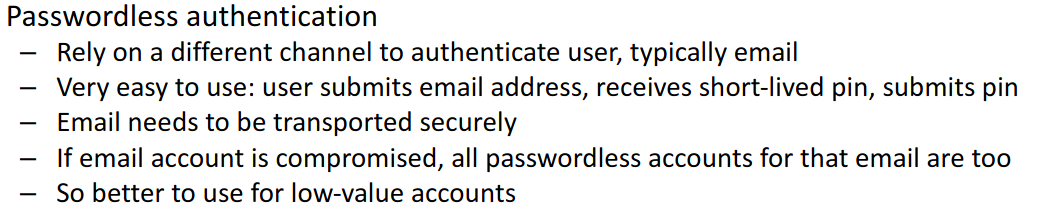
* Content-Security-Policy (CSP) : Specifies what the user is allowed to load
* Strict-Transport-Security (HSTS) : Enforces HTTPS, prevents SSL warning override.
* X-Frame-Options: Control which resources are allowed to frame the site
* Cross-Origin resource sharing (CORS): Control which domains we are allowed to retrieve resources from.
* X-XSS-Protection: If a URL is reflected by a script in the body, santise it or block the page entirely (mode=block).
* Public-Key-Pins (HPKP): Associate a certificate to a particular domain by storing a hash of the domain certificate’s public key for max-age.

ii) Offline dictionary attack: Attempting to crack a hashed (maybe salted) password, by comparing it against the hash of thousands/millions of common passwords or words (i.e. a dictionary). Possible if unsalted-hashed passwords have been leaked/extracted or the salts are known.

Online dictionary attack: Attempting to login to an account by trying many common passwords or words. Possible if the site doesn’t enforce login limits (delays, bans, captchas)

Attack against passwordless authentication: Session Fixation and Session Hijacking. Perform MITM attack? Possible if infected victim browser/PC or on unsecured connection.

Answer 2:



* Intercept the email sent from the web server to the victim containing the login link, and use that link to login instead
* The link/code must be used quickly as it is often valid for only a short window of time
* Steal someone’s thumb

b) i) Hypothetically:

* Add 2FA?
* Force stronger password
* Prevent repeated login attempts
* Use HTTPS
* Protect against SQL injection (filter input)
* Use captchas (prevent DoS)

3)

a)

i) Passive:

* Look up the company, social networks, etc for data
* Study the source code (without visiting the site)
* Find DNS/Domain info
* Study open source code used by target

Active:

* Use dig to query the DNS server
* Send emails to company emails
* Traceroute
* Identify subnet addresses
* Scan open ports
* Identify target OS
* Study information sent by server
* Send random info and study services

ii) Database based attacks:

* Perform SQL injection to access potentially sensitive info (passwords, emails, addresses, payment information) Is part of the “I” from STRIDE (Information Disclosure)
* Acquire admin logins using SQL injection to then install backdoor, explore local files, etc. Part of the “E” from STRIDE (Elevation of Privilege)
* Perform SQL injection to alter/delete data at will (Tampering)
* Launch expensive SQL queries to put the DB under heavy load (DoS)
* Repudiation – (if logs on SQL) delete logs showing your presence on site

b)

i) Hypothetically:

* Look at the source code
* Nmap the IP for ports
* Look up for info on the other sites

ii) Hypothetically:1

* Attempt simple SQL injection: OR 1=1
* Use # to remove any code after what you enter
* Union select user, password from TABLE\_NAME
* Select \* from information\_schema.tables
* Perform boolean queries (maybe this takes too long to be used in the exam)

4)

a)

i) DNS hijacking: Take over the victim’s DNS server address or a legitimate DNS server to redirect requests to malicious IPs

What needs to be done: Compromise victim machine to modify DNS server address, through malware on the machine or router, or compromise a DNS server itself, by gaining admin access through phishing, social engineering, etc.

The goal: Redirect legitimate requests to malicious sites, such as fake phishing sites, or to access data that is sent afterwards, such as emails, passwords, etc.

Countermeasure: DNSSEC: public/private keys used to verify authenticity of a DNS response.

ii) SSL stripping: MITM attack where attacker intercepts unsecure (HTTP) connection request from victim, and establishes HTTPS connection with the target site, then sends back a HTTP response to the victim.

What needs to be done: Have the victim send a HTTP request through a network controlled by the attacker (TOTALLY LEGIT FREE WIFI)

The goal: Intercept data sent by victim in plain-text (passwords, credit card details)

Countermeasure: Enforce HSTS, victim must load site through HTTPS

b)

i) Use nmap 10.39.26.128-159

ii) Common PHP vulnerabilities:

* Taking user input and executing it as a shell command (use stuff like escapeshellarg())
* Taking user input and executing it as a DB query (Use prepared statements)
* Taking user input and including it in HTML (Use whitelist)

iii) Intercept requests using wireshark.

Not sure what “exploit BobTheHacker” here means? Take over the DNS server somehow?