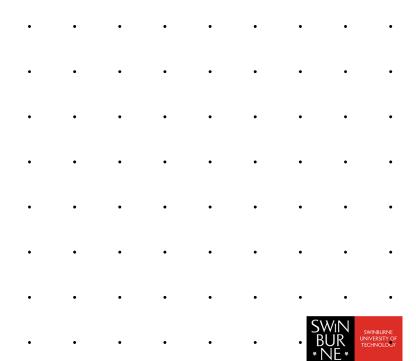


# Session Hijack Attack



## HTTP sessions

Stateful information needs to be maintained a server when it deals with a client.

- Load page by making an HTTP Request (POST or GET)
- 2. Server sets cookie and puts session ID into it. Browser sends it back with each request.
- 3. Upload info by POST or GET
- 4. Server compares session ID with it's list of sessions and "remembers" data for that client.

All vulnerable to sniffing.



# HTTP session hijack 1

- □Attacker uses *packet sniffing* to read session ID.
  - Attacker can take over a http session by writing the sniffed session ID into the attacker's cookie.

#### □ Defense:

- Cookie expiration date
- Https
- Cookie secure bit (cookie sent by https)



## HTTP session hijack 2

- □ Attacker uses *XSS* to read cookies of authenticated visitors to a site.
  - Attacker can take over a session by writing the received session ID into the attacker's cookie.

#### ☐ Defense:

- Server side filter/sanitise input/output
- Client side turn off javascript, turn on Application Boundary Enforcer (ABE)
  privacy plugins (noscript)
- Https no protection



#### Sandbox

- ☐ A virtual container which restricts the rights of a program.
  - e.g. Program X not allowed to write to disk
  - Sandbox contains a virtual disk (which program X can write to)
  - Sandbox and virtual disk are deleted when Program X terminates.
  - Built into some browsers (Chrome), plugins (Adobe), javascript, Java applets.
  - 3<sup>rd</sup> party sandboxes (Sandboxie)
  - Virtual machines are the ultimate sandbox
- ☐ But weakened by usability features (unity, vmtools)



### Sandbox

■Sandbox escape exploits exist for many sandboxes.

## ■Sandbox escape:

- Sandboxed (untrusted) application runs code in sandbox (e.g. browser runs js)
- 2. Untrusted (sandboxed) script runs trusted application from OS
- 3. Trusted application runs untrusted application from OS



## Sandbox example

■ Excel + Flash + download. 10/09/2017

https://threatpost.com/patched-flash-player-sandbox-escape-leaked-windows-credentials/127378/

- Normally Flash applications on web sites run in the default 'remote sandbox'

   can't run code outside sandbox.
- 2. User clicks on a link which downloads an Excel file.
- 3. Excel file contains and launches a Flash app.
- 4. Flash app runs with 'local with networking' privileges.
- 5. Flash app downloads malware and launches it outside sandbox.



## Sandbox example

☐ iOS sandbox escape and Priv-esc 1/08/2017

#### https://www.exploit-db.com/exploits/42407/

- 1. Set flags in a user-defined message (serialized object) to use shared memory
- 2. Return value contains a pointer to the shared memory
- 3. Call 2 function on the serialized object in rapid succession. The first allocates memory, the second sets the data type which will use the memory.
- 4. A race condition allows the type to be changed after the memory is allocated, allowing a buffer overflow to occur.
- 5. Send over-sized string to memory, including shell code (the exploit) runs as root!

