Tutorial 9

Aims

- To explore different attack and protection methods for the web

Questions

1. How can a sandbox and the same origin policy improve web-page security? Which method is more efficient and why?
   Sandbox:
   - The browser, and any application running in it, have limited access to most local resources
   - Sites are put in different security contexts, and each context has its own protection
     - Protection can be tailored to sites' trustworthiness
     - Web sites cannot access/ abuse other sites' privileges
   - Used by e.g. Google Chrome

   Same origin
   - Documents and scripts originating from the same site can access each other's methods and properties, but cannot do the same with documents from different origins
     - Less protection than what is offered by sandbox
     - Still protects from abusing other sites' privileges
     - Easy to implement
     - Easier to circumvent than Sandbox

2. Consider an application that uses predictable cookies to maintain system state. Explain an easy attack on the system, and how the attack can be prevented.
   Session hijacking:
   - Guess the session ID of an existing connection, the intruder can take over the connection and
     - Fool the server by masquerading as the client, while sending a disconnect message to the client
     - Fool the client by masquerading as the server, while sending a disconnect message to the server
   - Guess a session ID, and convince a user to use it with the server. Same problems as above.

   Can be prevented by
   - Using a secondary authentication at least for the client, possibly for the server too
   - Not using predicable session IDs

3. Why are browsers filtering code types, such as JavaScript or Java and not actual content? What method is there to filter actual content? Could these methods be improved?
   - Filtering content requires processing/executing the script/code, but it is complex and time consuming. Filtering by code type is easy and fast but less effective; in the vast

majority of cases the malicious effect is not recognisable by looking at the source code.

- Text parsing can be used to filter actual content, but its usefulness is limited to simple cases, e.g. blocking indecent language or parental filter. Image recognition is too hard at the moment.
- For improvement, learning methods, i.e. when the filter is trained on real examples to recognize content to be blocked represent a promising approach, but still not reliable enough.

4. Compare cross-site scripting and cross-site request forgery. What is the main difference between the two?

XSS	**CSRF**
Exploits the user's trust in the server	Exploits the server's trust in the user
Exploit introduced via the server	Exploit introduced via the client
Requires no user action (apart from visiting the malicious page)	Requires preparation, user has to download the malware (from email, other web page, etc)
Intruder has to manipulate the server only	Intruderhas to manipulate the user and then the target server
Mostly(but not exclusively) exploits the user machine	No limit to exploitation (user machine, other servers, etc)

5. How can web crawlers affect a web-server's performance? How can web pages manipulate web crawlers?
   Web crawlers can overload the server by visiting many web pages on the site, and downloading too many web pages.
   The Robots Exclusion Protocol (/robots.txt file, <robots> meta tag) is to help
   Web servers. The servers can tell crawlers to avoid certain pages or actions e.g. not to download.
   The Robots Exclusion Protocol is observed by honourable crawlers, but rogue crawlers (e.g. email address harvesters) usually ignore it.