IN2120 Information Security Autumn 2019

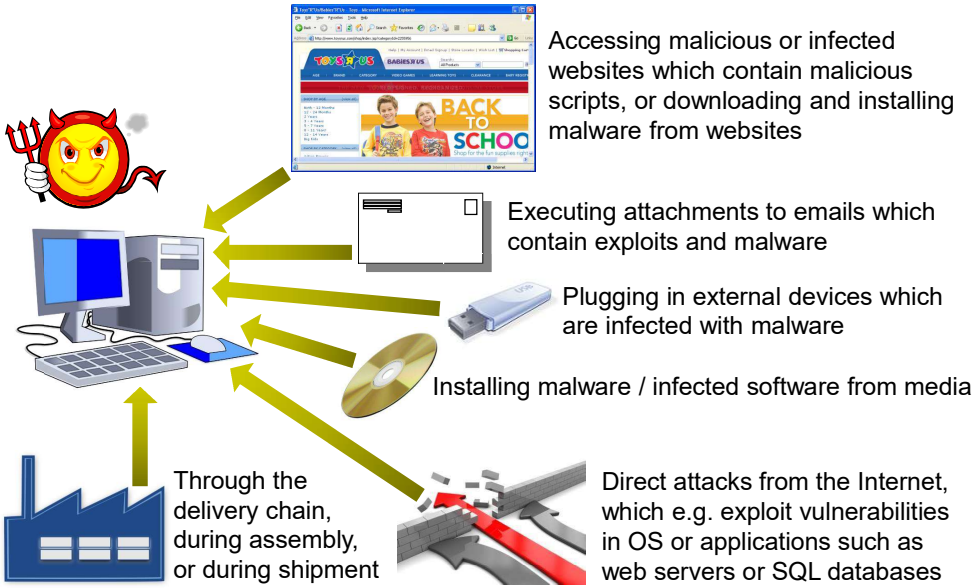
L12: Application Security and Secure System Development

Audun Jøsang
University of Oslo

Outline

1. Application Security
   - Malicious Software
   - Attacks on web applications
   - Secure System Development

How do computers get compromised ?



Accessing malicious or infected websites which contain malicious scripts, or downloading and installing malware from websites

Executing attachments to emails which contain exploits and malware

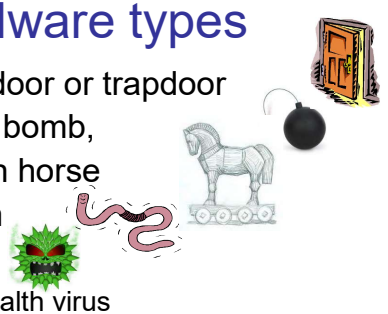
Plugging in external devices which are infected with malware

Installing malware / infected software from media

Through the delivery chain, during assembly, or during shipment

Direct attacks from the Internet, which e.g. exploit vulnerabilities in OS or applications such as web servers or SQL databases

Malware types



- Backdoor or trapdoor
- Logic bomb,
- Trojan horse
- Worm
- Virus
  - Stealth virus
    - Uses techniques to hide itself, e.g. encryption
  - Polymorphic virus
    - Different for every system
  - Metamorphic virus
    - Different after every activation on same system
- Exploit

  - A method to infect systems by using malicious program or input data (e.g. document) that triggers and exploits a software bug in the systems

Exploits

- A piece of software, data, or a sequence of commands that exploits a software/hardware vulnerability
- Can be carried in common data formats such as pdf documents, office documents or media files.

- Often contains carefully designed corrupt datatypes
- Causes unintended or unanticipated behavior to occur on computer software or hardware
- The functionality of exploits is typically to:
  - Download a malware/backdoor which allows the attacker to control the platform
  - Directly take control of a computer system, allowing privilege escalation, or a denial-of-service or other sabotage.

Backdoor or Trapdoor

Installed by exploit:

- Provides remote control capabilities by attackers
- Can reside on system for long periods before being used
- Can be removed after use

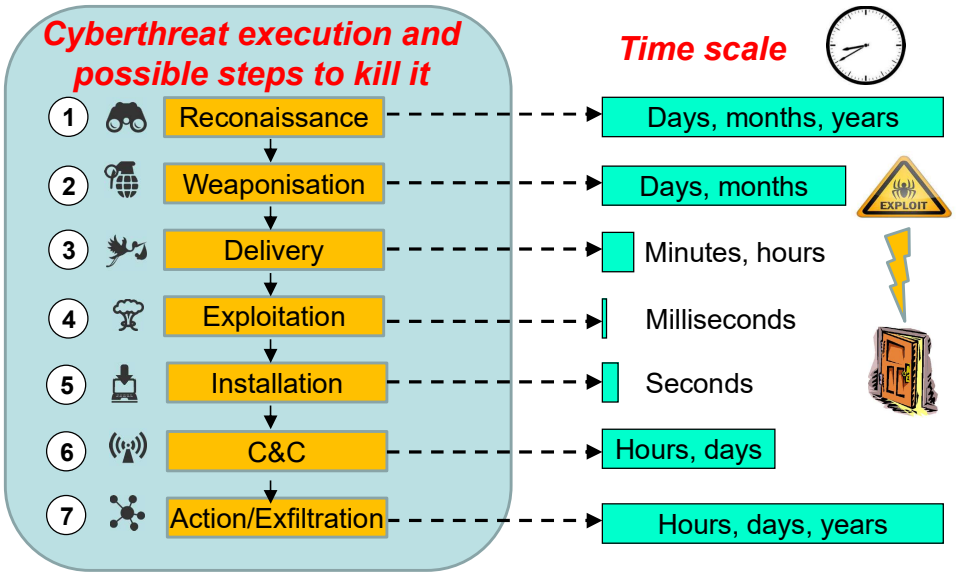
Installed by user:

- User can be tricked to install malicious program (see Trojan horse)

Installed during design:

- is a hidden/secret entry point into a program,
- allows those who know access bypassing usual security procedures
- is commonly used by developers for testing
- is a threat when left in production software allowing, exploit by attackers
- is very hard to block in O/S
- can be prevented with secure development lifecycle

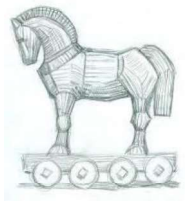
The Cyber Kill Chain (Hutchins et al. 2011)



Logic Bomb

- one of oldest types of malicious software
- code embedded in legitimate program
- activated when specified conditions met
  - eg presence/absence of some file
  - particular date/time
  - particular user
- causes damage when triggered
  - modify/delete files/disks, halt machine, etc

Trojan Horse



- program with hidden side-effects
  - e.g. a back door
- program is usually superficially attractive
  - eg game, s/w upgrade etc
- performs additional tasks when executed
  - allows attacker to indirectly gain access they do not have directly
- often used to propagate a virus/worm or to install a backdoor
- … or simply to destroy data

Malicious Mobile Code

- Program/script/macro that runs unchanged
  - on homogeneous platforms (e.g. Windows)
    - will only affect specific platforms
  - on heterogeneous platforms
    - will affect any platform that supports script/macro language
    - e.g. Office macros
- Transmitted from remote system to local system & then executed on local system
  - To inject Trojan horse, spyware, virus, worm etc. which can
    - directly perform specific attacks, such as unauthorized data access, root compromise, sabotage
    - indirectly infect other systems and thereby spread

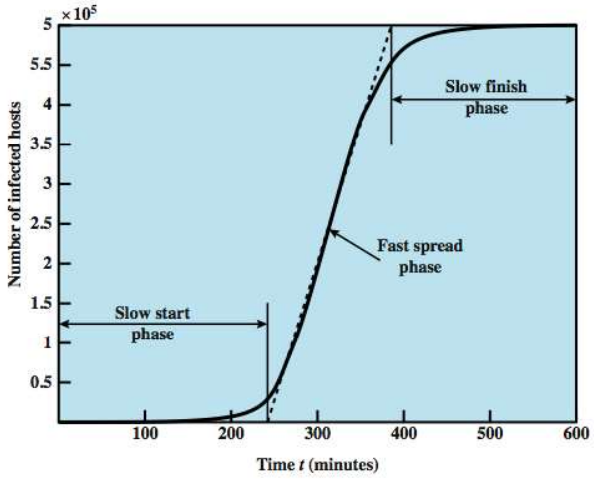
Viruses

- piece of software that infects programs
- specific to operating system and hardware
  - taking advantage of their details and weaknesses
- a typical virus goes through phases of:
  - dormant
  - propagation
  - triggering
  - execution

Worms

- Replicating programs that propagate over net
  - Access remote systems via network protocols to open ports
  - Attack vulnerable processes in remote systems
  - Can also use email, remote exec, remote login
- Can have characteristics like a virus:
  - Dormant, triggering, execution, propagation & replication
  - Propagation phase: searches for other systems to infect
  - May disguise itself as a system process when executing
- Morris Worm, the first and the best know worm, 1988
  - released by Robert Morris Jr., paralyzed the Internet (of 1988)
  - exploited vulnerabilities in UNIX systems
- WannaCry Worm, epidemic infection in May 2017
  - exploits known, but unpatched, vulnerability in Windows XP

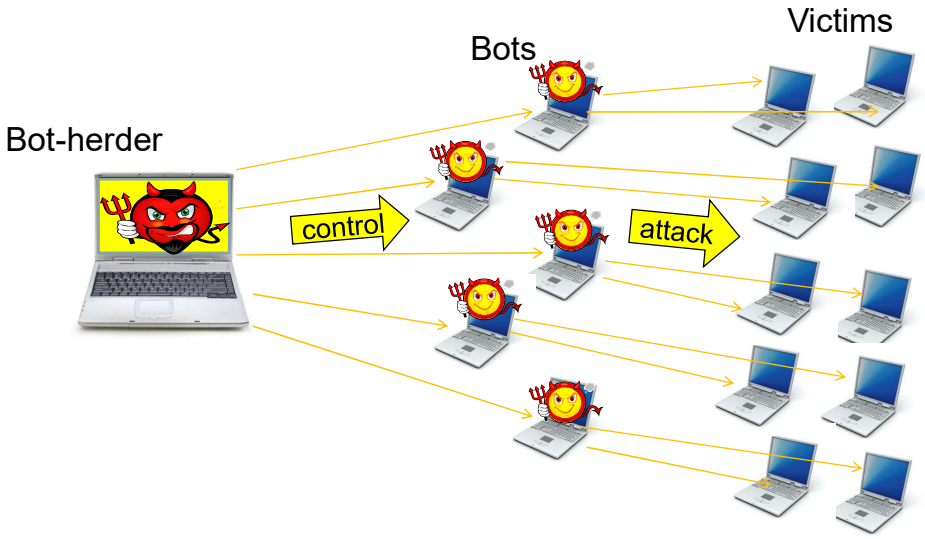
Worm Propagation Speed



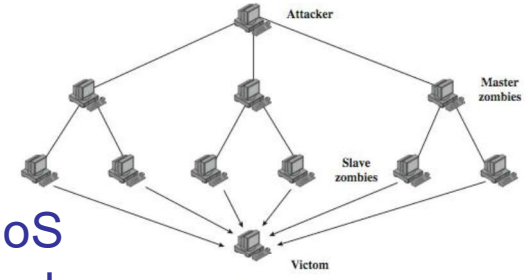
What is a botnet ?

- **A botnet** is a collection of computers infected with malicious software agents (robots) that can be controlled remotely by an attacker.
- Owners of bot computers are typically unaware of infection.
- Botnet controller is called a "bot herder" or "bot master"
- Botnets execute malicious functions in a coordinated way:
  - Send spam email
  - Collect identity information
  - Denial of service attacks
  - Create more bots
  - Bitcoin mining
- A botnet is typically named after the malware used to infect
- Multiple botnets can use the same malware, but can still be operated by different criminal groups

Botnet Architecture

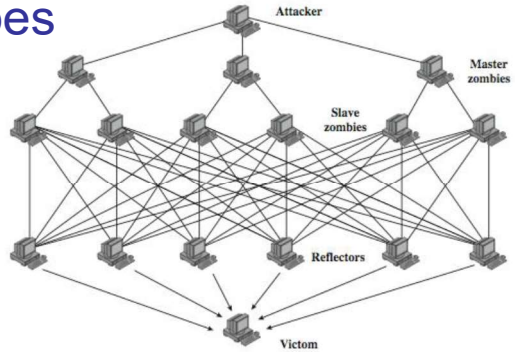


DDoS Flood Types



(a) Direct DDoS Attack

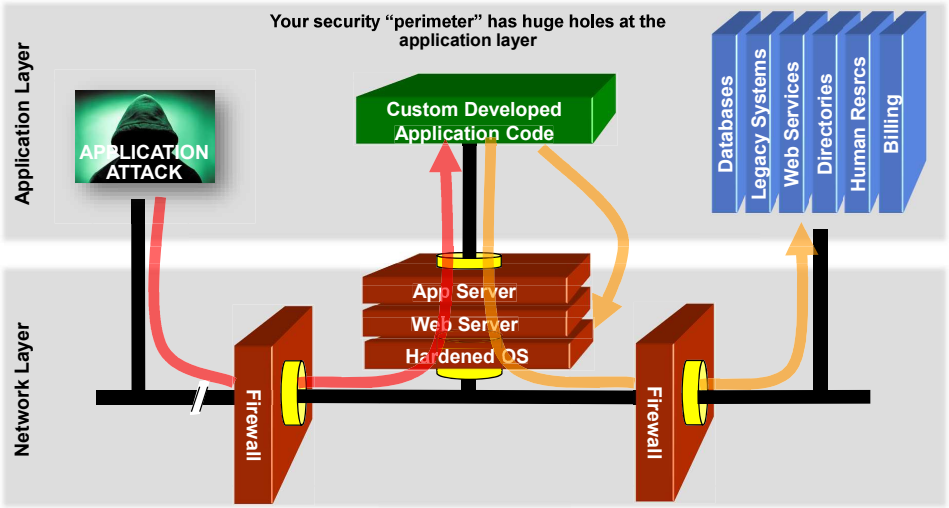
- Direct attack
  - Bots send traffic with own or spoofed sender address to victim



(b) Reflector DDoS Attack

- Reflected attack
  - Bots send traffic to innocent hosts with victim address as sender address. Innocent hosts become part of attack by replying to victim.

The web application security challenge



Network security (firewall, SSL, IDS, hardening) does not stop application attacks

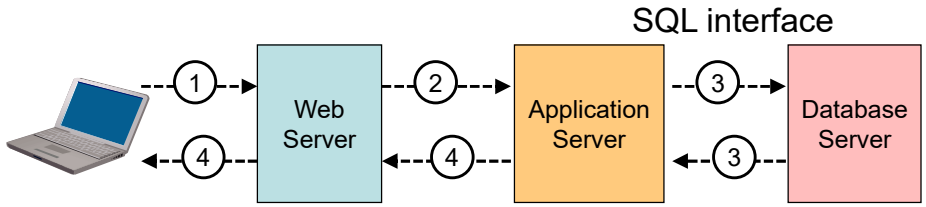
What is SQL?

- Structured Query Language: interface to relational database systems.
- Allows for insert, update, delete, and retrieval of data in a database.
- ANSI, ISO Standard, used extensively in web applications.
- Example:

```
select ProductName from products where
ProductID = 40;
```


SQL at back-end of websites

1. Take input from a web-form via HTTP methods such as POST or GET, and pass it to a server-side application.
2. Application process opens connection to SQL database.
3. Query database with SQL and retrieve reply.
4. Process SQL reply and send results back to user.



What is SQL Injection?

- Database system misinterpretation of input data
  - Attacker disguises SQL commands as data-input
  - Disguised SQL commands = 'injected' SQL commands
- With SQL injection, an attacker can get complete control of database
  - no matter how well the system is patched,
  - no matter how well the firewall is configured,
- Vulnerability exists when web application fails to sanitize data input before sending to it database
- Flaw is in web application, not in SQL database.

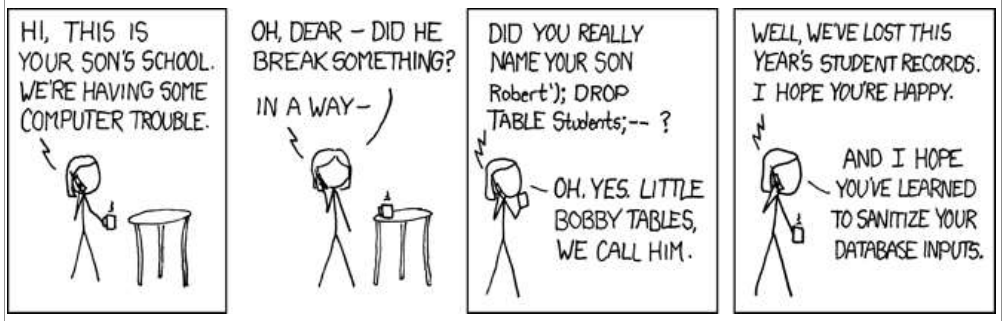
What is SQL Injection?

- For example, if input field ask for a product number, but the malicius user inputs "**40 or 1 = 1**"
- The result SQL command becomes:

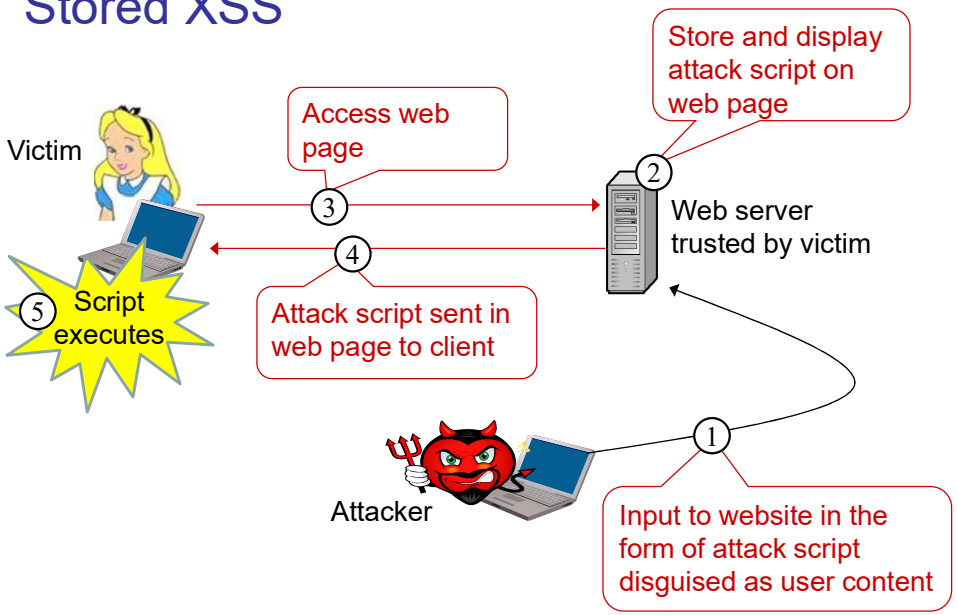
```
select ProductName from products where
  ProductID = 40 or 1 = 1
```

- 1=1 is always TRUE so the "where" clause will always be satisfied, even if ProductID ≠ 40.
- All product records will be returned.
- Data leak.

XKCD – Little Bobby tables



Stored XSS



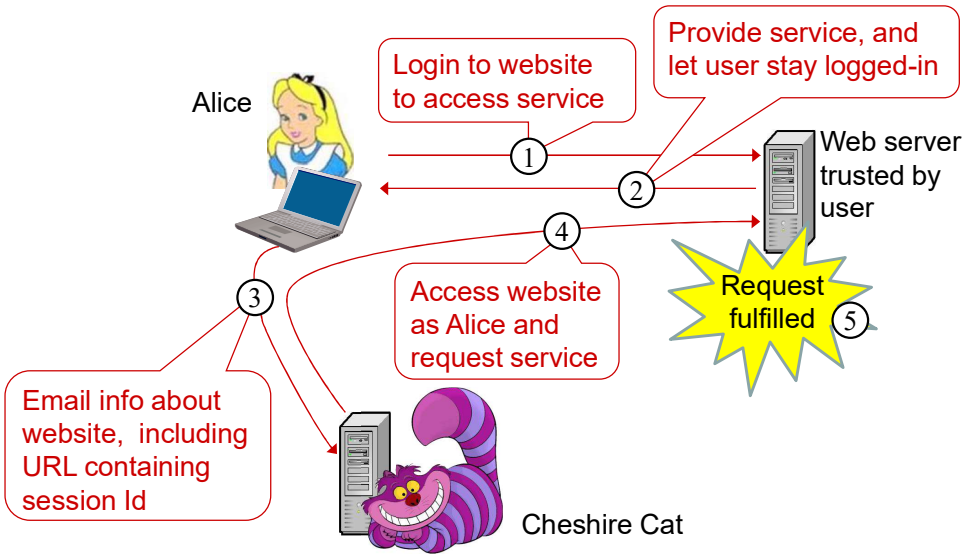
Stored XSS

- Data provided by users to a web application is stored persistently on server (in database, file system, …) and later displayed to users in a web page.
- Typical example: online message boards.
- Attacker uploads data containing malicious script to server.
- Every time the vulnerable web page is visited, the malicious script gets executed in client browser.
- Attacker needs to inject script just once.

Preventing SQL Injection and XSS

- Validate all user entered parameters
  - CHECK data types and lengths
  - DISALLOW unwanted data (HTML tags, JavaScript, SQL commands)
  - ESCAPE questionable characters (ticks, --,semi-colon, brackets, etc.)
- Hide information about Error handling
  - Error messages divulge information that can be used by hacker
  - Error messages must not reveal potentially sensitive information

Broken Authentication and Session Mgmt



Broken Authentication and Session Mgmnt Problem and Fix

- User authentication does not necessarily provide continuous authentication assurance
  - User authentication is only at one point in time
- Insecure implementation of session control with a static session Id which is passed in the URL
  - Unfortunately this can be misused
- Recommendations for session Id must be followed
  - E.g friom OWASP
- Examples of controls for session Id:
  - Link session Id to e.g. IP address, TLS session Id
- .

OWASP

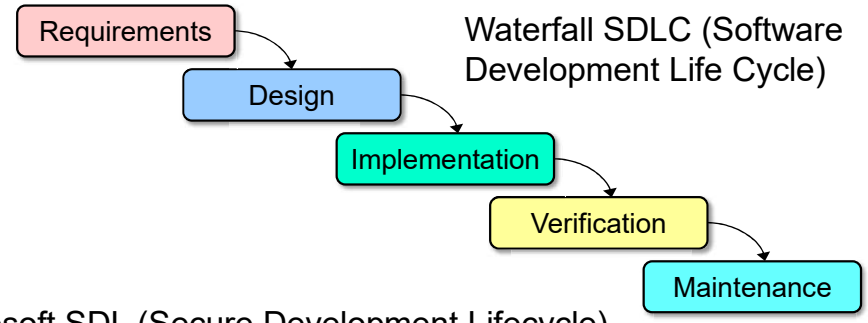
The Open Web Application Security Project

- Non-profit organisation
  - Local chapters in most countries, also in Norway
- OWASP promotes security awareness and security solutions for Web application development.
- OWASP Top-10 security risks identify the most critical security risks of providing online services
  - The Top 10 list also recommends relevant security solutions.
- OWASP ASVS (Application Security Verification Standard) specifies requirements for application-level security.
- Provides and maintains many free tools for scanning and security vulnerability fixing

Top-10 Web Application Risks

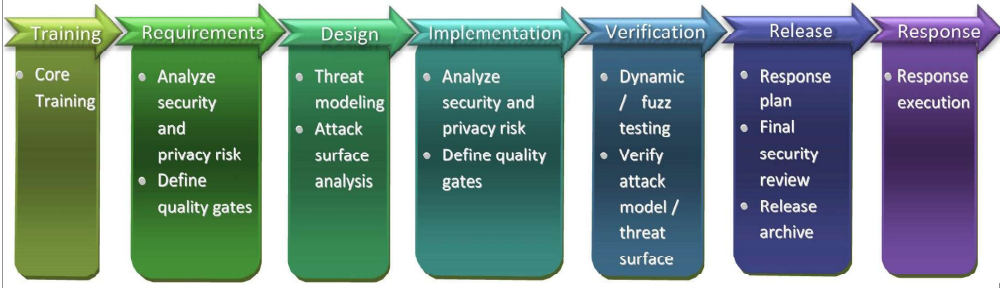
1. Injection
2. Broken Authentication and Session Management
3. Cross-Site Scripting (XSS)
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. Cross-Site Request Forgery (CSRF)
9. Using Components with Known Vulnerabilities
10. Unvalidated Redirects and Forwards

Waterfall and Secure Waterfall

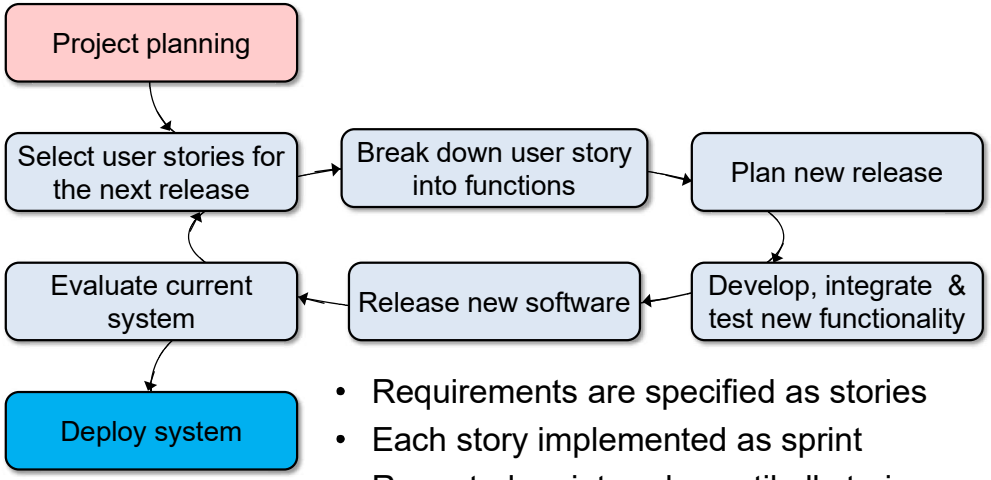


Waterfall SDLC (Software Development Life Cycle)

Microsoft SDL (Secure Development Lifecycle)



Agile Software Development (e.g. Scrum)



- Requirements are specified as stories
- Each story implemented as sprint
- Repeated sprint cycles until all stories are implemented

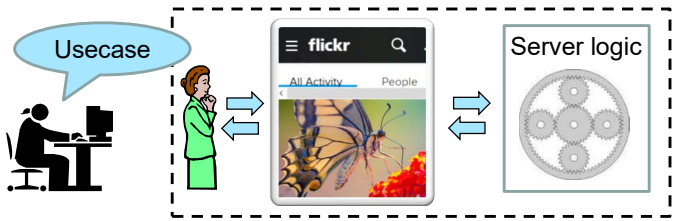
User Stories and Usecases

User Story – Seen from the user perspective:

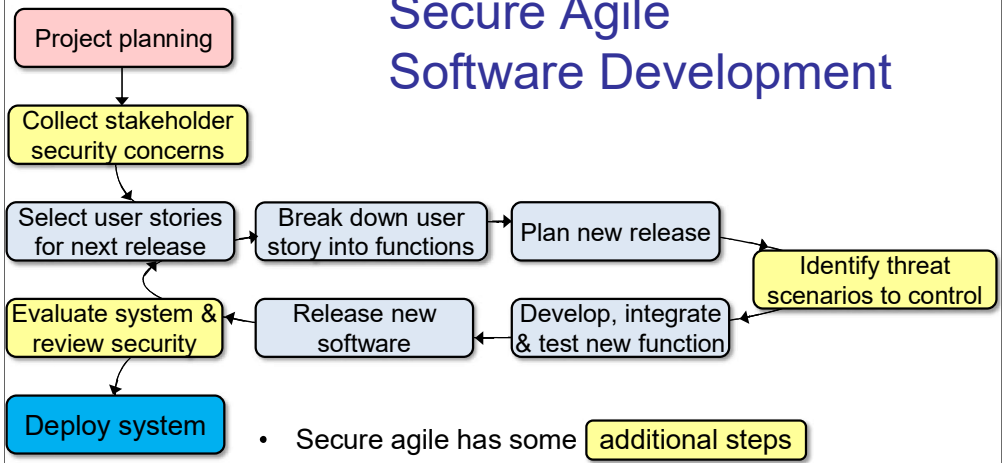
As an [actor] I want [action] so that [achievement].
For example: *As a Flickr member, I want to set different privacy levels on my photos, so I can control who sees which of my photos.*

Usecase – Seen from the design perspective:

Description of a set of interactions between a system and one or more actors (where an 'actor' can be a user or another system).

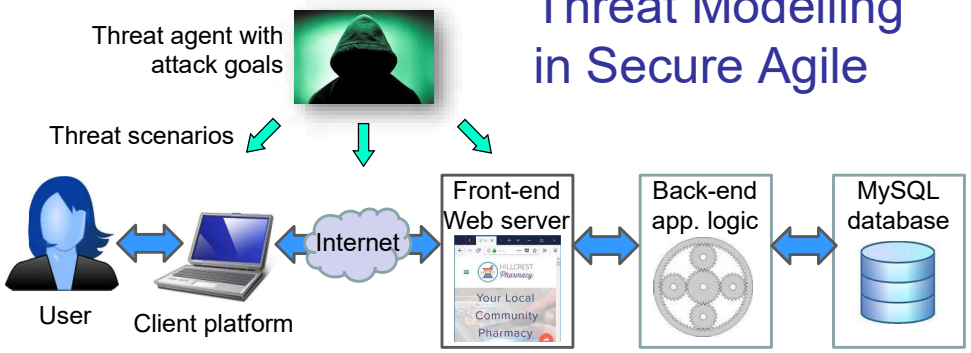


Secure Agile Software Development



- Secure agile has some additional steps
  - During project startup
  - During each sprint cycle
  - During final test and validation
- Secure agile necessarily makes it a little less agile

Threat Modelling in Secure Agile



- Threat modelling is the process of identifying, analysing and describing relevant threats (scenarios).
- Do threat modelling and (light weight) risk assessment in each sprint.
- Think: How could this new function be misused or attacked?
  Which assets could be harmed? What consequences?
- Stop or mitigate the threat (remove vulnerabilities) during the sprint.

STRIDE Threat Modelling

S **Spoofing**
Can an attacker gain access using a false identity?

T **Tampering**
Can an attacker modify data as it flows through the application?

R **Repudiation**
If an attacker denies doing something, can we prove he did it?

I **Information disclosure**
Can an attacker gain access to private or potentially injurious data?

D **Denial of service**
Can an attacker crash or reduce the availability of the system?

E **Elevation of privilege**
Can an attacker assume the identity of a privileged user?

Attacker Story and Misuse Case (Attacker Goal and Threat Scenario)

Attacker Story – The goal of the attacker:
As an [attacker] I want [action] so that [achievement].
So, for example: *As an attacker, I want to hack into Flickr accounts to steal photos and personal info.*

Misuse Case (Threat Scenario)

Seen from the threat scenario perspective:

Description of a set of steps and interactions to be executed by attacker to achieve goal.