Software Security Practices
Lecture 5

Binuri Raigamkorale
binuriyr@dcs.ruh.ac.lk

Today’s Outline

5.1 Terminology

5.2 Introduction to Software Security.

5.3 History of Security Approaches in Software Developments

5.4 Secure Architecture Process

5.5 Application and Data security

5.6 Best practices in Devops

5. Conclusion.

Learning Outcomes

- Learn how to protect from threats that are every developer should be aware of
- Learn the Importance of Security in software.
- Discuss the steps in the secure architecture process to ensure the resulting architecture and system are as secure as possible.

objectives

- Extensive, practical knowledge about Software Security, from the basics to the advanced
- Software Security terminology (DDoS, MFA, SQL Injection, and lots more)
- The hacker's mindset
- The Software Architect's role in Software Security
- The main security threats every developer and architect should be aware of
- Proven methods of dealing with security threats
- The complete Secure Architecture Process
- Building Blocks of Secure Architecture
- Applying this knowledge on a case study

At the end of this course

- You will know what you need to protect against
- You will how to integrate security into your systems
- You will know how to manage the organization’s overall security
- You will be able to help the organization o define security policy

5.1 Software security terminology

- Threat

is a **malicious act that seeks to damage data, steal data, or disrupt digital** life in general.

ex: SQL injections

DDOS Attacks

- Attack

The actual execution of threat by an attacker(s).

Examples:

A malicious user enters SQL-Injection-bound input

A group of attackers orchestrate DDOS attack

- DDOS(Distributed Denial of Service) Attack

A special kind of attack used to overload sites and take them down

Source:
https://www.digitalattackmap.com/#anim=1&color=0&country=US&list=0&time=17890&view=map

5.1 Software security terminology..

- Vulnerability

  A problem in the system that can be used by attacker to execute an attack on the system, and make it compromised.

  Ex: misconfigured firewall exposes internal systems to the public web

- Authentication

  Establishing the identity of a user (human or not) based on reliable mechanism.

  Example:

  - Username / Password
  - SMS
  - Biometric Identification

- Authorization

  Establishing what a given user is allowed to do in the system.

  Example:

  - User X is allowed to create new service request
  - User X is NOT allowed to delete an existing service request

5.1 Software security terminology cont..

- Security breach

A security breach is any incident that results in unauthorized access of data, applications, services, networks and/or devices by bypassing their underlying security mechanisms.

A security breach occurs when an individual or an application illegitimately enters a private, confidential or unauthorized.

- Data inconsistency

  When the same data exists in different formats in multiple tables

5.2 Introduction to Software Security

5.21 What is Software Security?

"Someone logs into the system with fake identity."

"The system is attacked using DDOS"

5.22 Using software security we protect against:

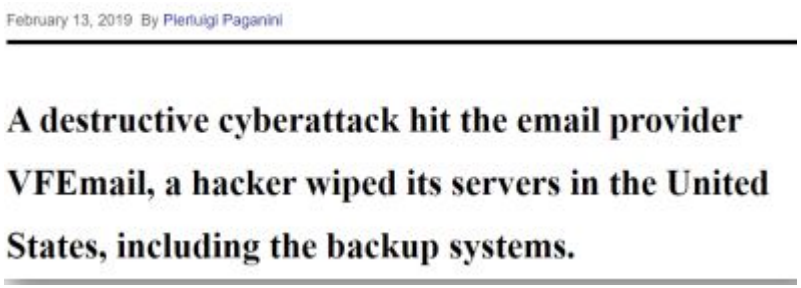
Data loss

- Sensitive data is lost due to security breach.
- Data disappear due to a vulnerability in the system, someone hacked the system and made an data loss.

i.e. when an attacker gains access to the main DBs and do delete data.

eg: https://securityaffairs.co/wordpress/81030/hacking/vfemail-destructive-cyberattack.html

hacker deleted all data from VFEMail servers, including backups.



February 13, 2019 By Pierluigi Paganini

A destructive cyberattack hit the email provider VFEmail, a hacker wiped its servers in the United States, including the backup systems.

5.2 Introduction to Software Security ..

Disruption of Service

- the system activity is disrupted due to attacker's actions

e.g.: Attackers orchestrate distributed denial-of-service attack, taking the service down

ex: https://internetofbusiness.com/ddos-attack-twitter/

5.2 Introduction to Software Security ..

- **Data Leak**

Sensitive data is stolen and made available to non-authorized recipients

e.g.: Attackers gain access to the DB and steal credit card information

source:
https://www.forbes.com/sites/thomasbrewster/2018/11/30/marriott-admits-hackers-stole-data-on-500-million-guests/#786737086492

- **Data Inconsistency**

Data is manipulated by non-authorized attackers and become inconsistent.

e.g.: Attackers impersonate as someone else and perform unauthorized actions.

5.2 Introduction to Software Security ..

Who is responsible for the security?

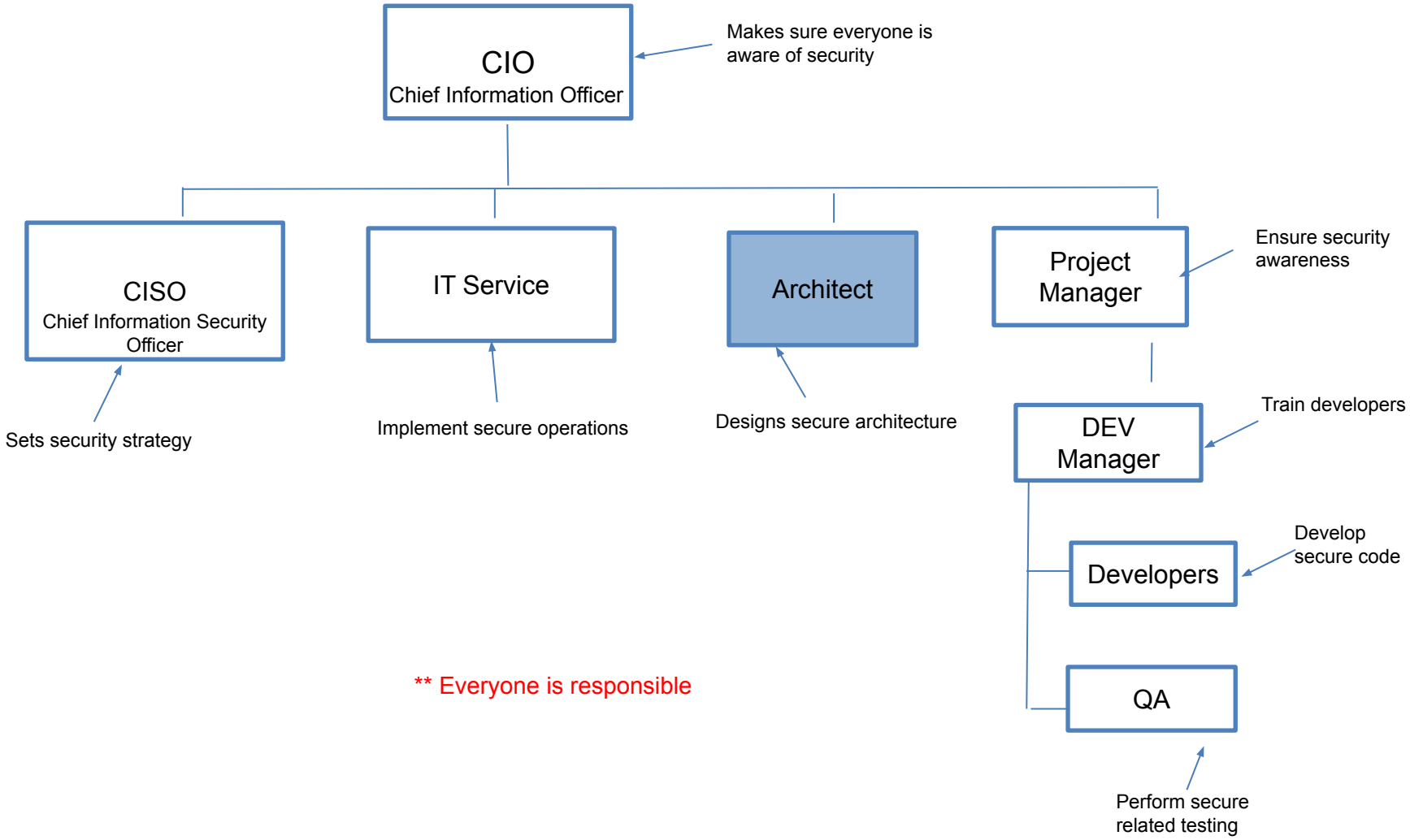


Figure 5.2 Typical organizational hierarchy of security

5.4 Secure architecture process

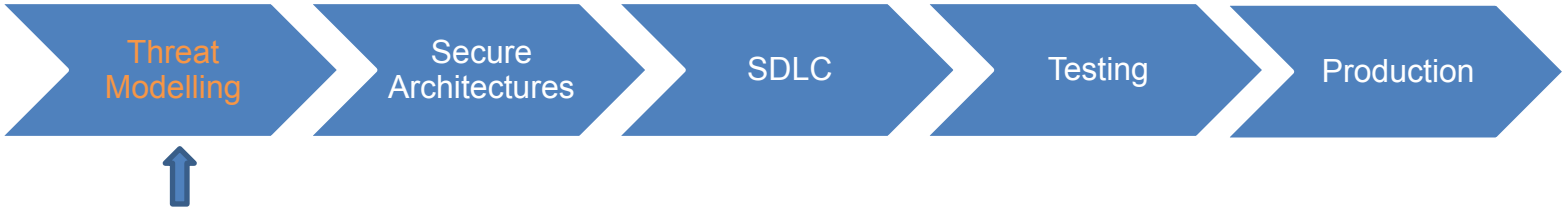
5.41 Introduction to secure architecture

- A well defined process for ensuring the system is as secure as possible.
- Goes through all the system's phases
- Should be led by the project manager/ dev manager
- Architect should be involved in all stages.

5 stages

5.4 Secure Architecture Process

5.41 "Threat Modelling" phase



Overview:

- A process for identifying potential threats for the system
- Prioritizes mitigations measures
- Has a great effect on the work plan
- Involves almost everyone in the team
- Might utilize formal methods and tools

Goal:	Identify potential threats for the system and discuss ways to mitigate them	
Participants:	Project Manager	CISO
	Architect	IT (OP)
	Dev Manager	Developers (OP)
	System Analyst	QA (OP)

(OP) = Optional

5.4 Secure Architecture Process

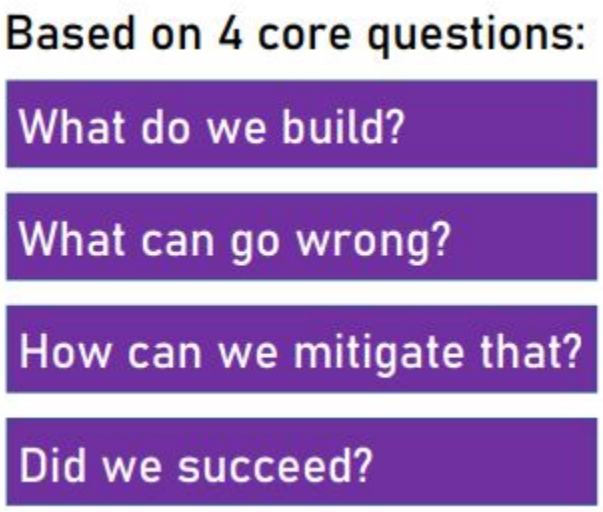
5.41 Threat Modelling phase

What is threat modelling?

- The process of identifying potential threats for the system
- Done once, but might be repeated later
- Should be very methodical
- Everyone's input is welcome.

How to conduct threat modelling?

Addresses the following 4 questions



5.4 Secure Architecture Process

5.41 Threat Modelling phase

What do we build?

- Describe functional and non-functional requirements
- If there are any known technical or architectural details – include them

Ex:

"We're designing an HR system to manage the employees' data, including salary, vacations, etc."

What can go wrong?

- Describe what are the main threats the application might face, based on:
  - Sensitivity of the information the system stores
  - Its location

Ex:

"Since we store sensitive data (salary) in the system, we want to make sure it won't leak"

5.4 Secure Architecture Process

5.41 Threat Modelling phase

How can we mitigate that?

- Discuss mitigations to the potential threats
- Research various mitigations methods if needed (best encryption algorithms)
- Make sure to include the mitigations in the work plan
- If the dev team does not know how to implement it – design a training plan

Ex:

"All the sensitive data is going to be encrypted. In addition, database access will be given on a least privilege basis only"

Did we succeed?

- Design tests to validate the solution designed
- Usually will be carried out in the testing phase, but sometimes also during development

Ex:

"We're going to ask a security expert to extract and decrypt the encrypted data and see if he succeeds."

5.4 Secure Architecture Process

5.41 Threat Modelling phase

Threat modeling example:

5.4 Secure Architecture Process

5.41 Threat Modelling phase

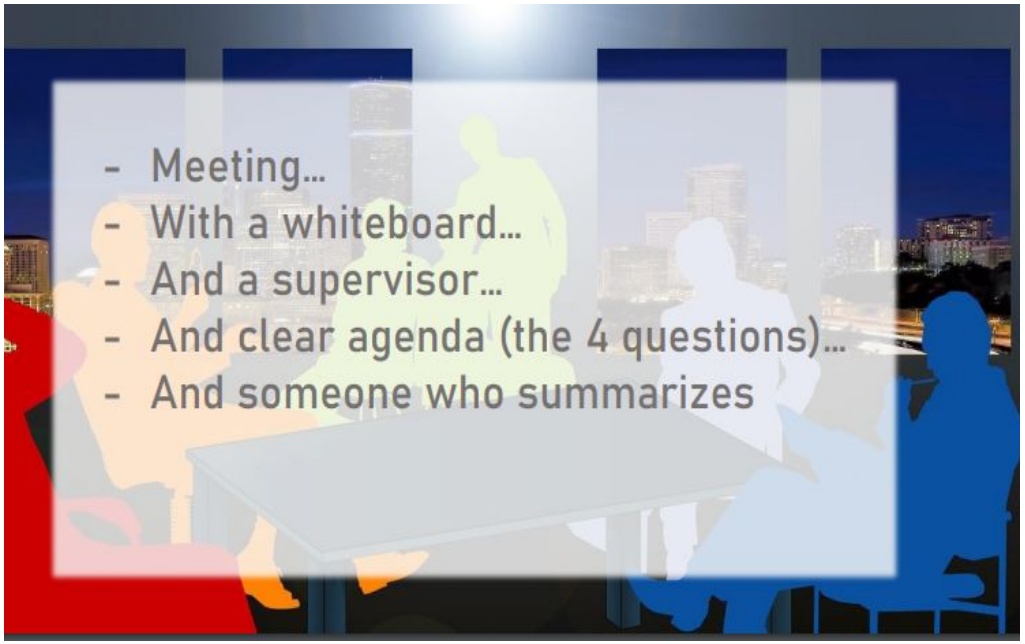
When ?

At the beginning of the project — (When there are functional and non-functional requirements)

After major changes — (That might present new security risks)

Following security incident — (To find out what went wrong)

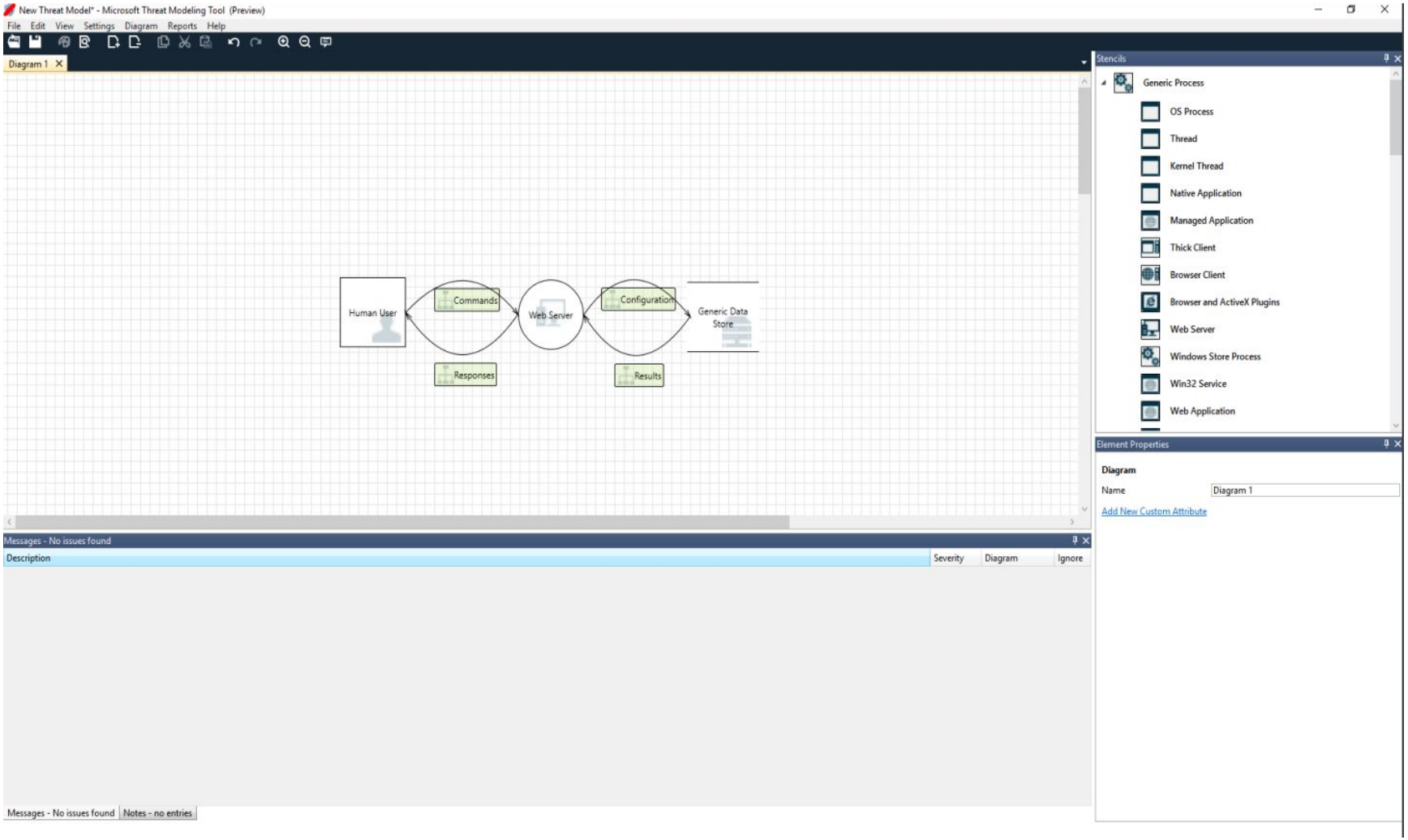
How ?



5.4 Secure Architecture Process

5.41 Threat Modelling phase

Tools ? Microsoft TMT , Threat dragon



5.4 Secure Architecture Process

5.42 "Secure Architecture" phase

Overview:

- Based on the Security Perimeters paradigm
- Integrates security defenses/protections into the core architecture
- Touches all aspects of the system(Dbs , networks).
- Secure Architecture Documentary

Goal:	Design Secure Architecture based on the threats defined in the Threat Modeling	
Participants:	Architect	CISO (OP)
	Dev Manager (OP)	IT (OP)
	System Analyst (OP)	Developers (OP)
		QA (OP)

(OP) = Optional

5.4 Secure Architectures Process

5.42 "Secure Architecture" phase

Security Perimeters

Physical :

No need to worry, it manage by cloud services providers

Network :

Controls the access to the organization's network.

Access Control

Usually performed by authentication engine

i.e. Active Directory

Various types of authentication

user/password , biometric, etc.

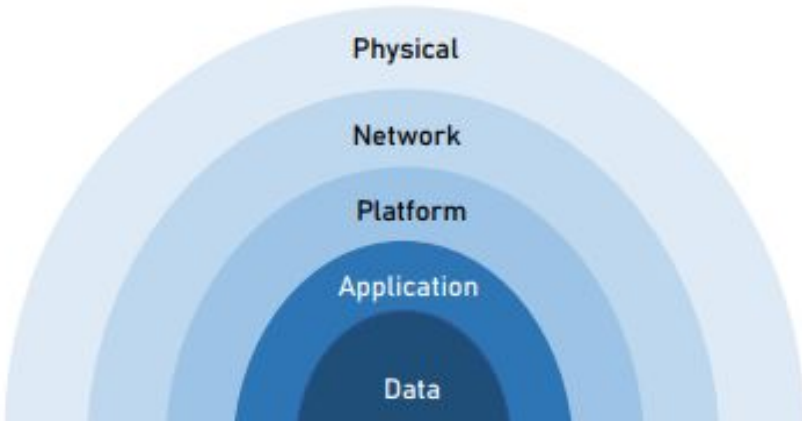


Figure : Security Perimeters

Reliability

Firewall that filters specific contents (ex: CISCO)

Makes sure the network stays up and running and reliable even under heavy attacks (DDoS…)

5.4 Secure Architectures Process

5.42 "Secure Architecture" phase

Security Perimeters

Platform Security

Secure the computers ,VMs , etc.

- by using modern OS versions
- patch management i.e. Security patches
- up to date antivirus
- DLP(data loss prevention) softwares to track in out data

Application Security & Data Security

- Authentication
- Authorization
- Secure Communication
- Secure Code
- Secure Data
- Logging and Monitoring

5.4 Secure Architectures Process

Application Security & Data Security

(this is the where the main works happens by the development team.)

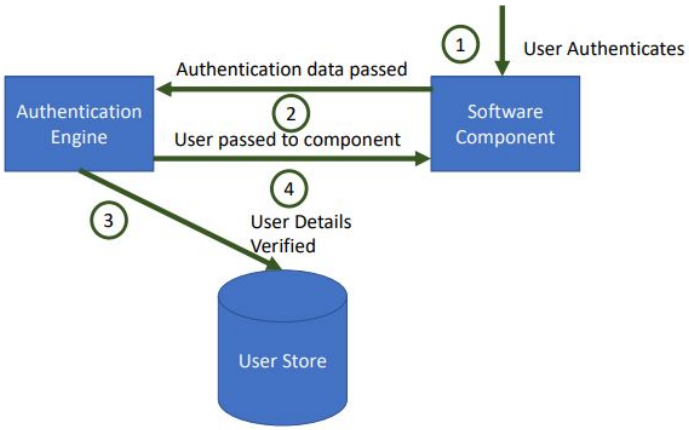
Authentication

Verifying the identity of a thing

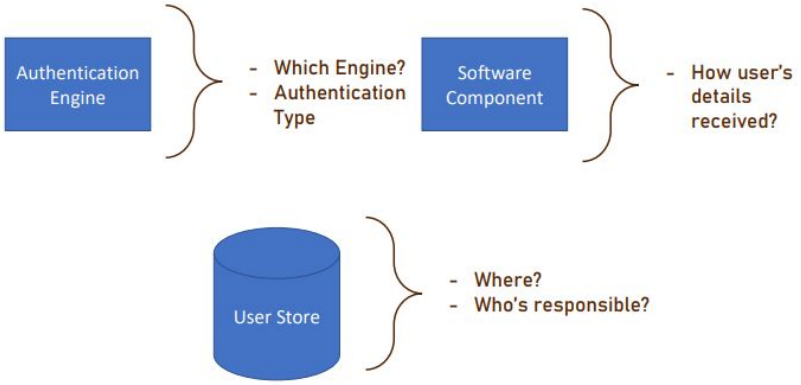
Where thing can be: Human, Computer, Software, IOT device, …

When knowing who uses the system, we can:

- Find out what he's allowed to do
- Allow / Deny certain actions
- Log all activities for future research
- Prevent data leak, data inconsistency



Authentication Design



Authentication consideration

5.4 Secure Architectures Process

Authentication

3rd party authentication engine

Cloud Based

On-Premise

Authentication divided into three factors

Something you know	(username / password, security question)
Something you have	(phone, smart card)
Something you are	(fingerprint, iris scan, other biometric data)

For extremely sensitive systems (bank, health, etc)

(username / password + code in Text)

(username / password + fingerprint)

(phone + fingerprint)

5.4 Secure Architectures Process

Authorization

Assigning least privileges to a thing

Where “thing” can be: Human, Computer, Software, etc.

Limits access to users so that they won’t do unintended actions

- Even if authentication was hacked – authorization will limit its impact
- Protects mainly against data leak, data inconsistency

Two types of authorizations

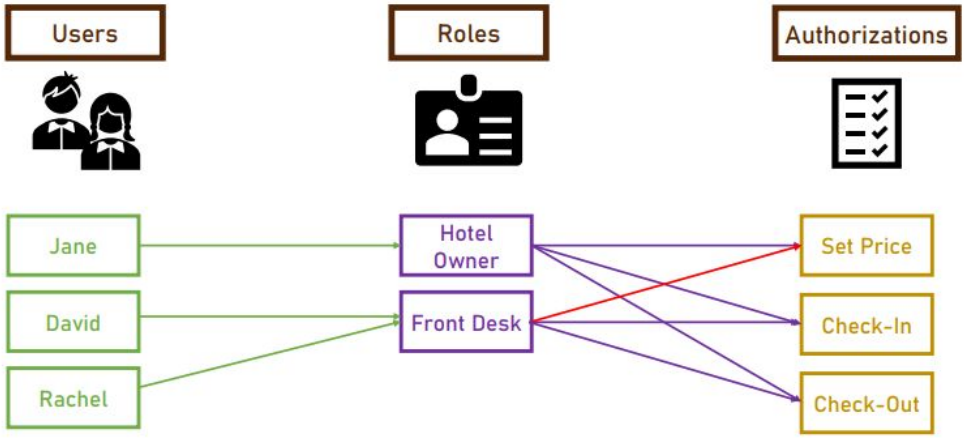
Action Authorization	Data Authorization
What actions are allowed or denied	Which data is accessible
Example: - Allowed to add service request - Allowed to update item inventory - Denied from updating exam score	Example: - Cannot see data from other teachers - Cannot see data of other counties - Can see data of his own patients only

5.4 Secure Architectures Process

Authorization

Role Based Access Control

With RBAC:



where are roles managed?

- In the Authentication Engine

(ie. User Groups in Active Directory)

Passed to the component as part of the user's token

RBAC implementation in .Net and Nodejs

Action Authorization

Usually using built-in support in the development platform

```
[Authorize(Roles = "Manager, Administrator")]
public class DocumentsController : Controller
{
    public ActionResult ViewDocument()
    {
        //Your code here
    }
    [Authorize(Roles = "Administrator")]
    public ActionResult DeleteAllDocuments()
    {
        //Your code here
    }
}
```

.NET Core

5.4 Secure Architectures Process

Data Authorization

Row level security (RLS)

Decide on the roles management

- Part of the Authentication Engine / part of the component
- If part of the component – design the roles' table, data access

implementation in .Net

```
Public MedData GetMedicalData(doctorId doc)  {

       using (var db = new MedContext())  {

              var medData=db.MedicalData
                  .Where(d=>d.ownerDoctor==doc)
                  .Order(d=>d.creationDate))
                  .ToList();

       }
}
```

5.4 Secure Architectures Process

Secure Communication

Making sure data in transit is secure to ensure Privacy & Data Consistency

Ensures sensitive data is not leaked to unauthorized recipients

- Prevents Man-In-The-Middle attacks
- Protects mainly against data leak, data inconsistency

Implemented using Secure Protocol

- SSL – secure Socket Layer
- TLS – transport level security

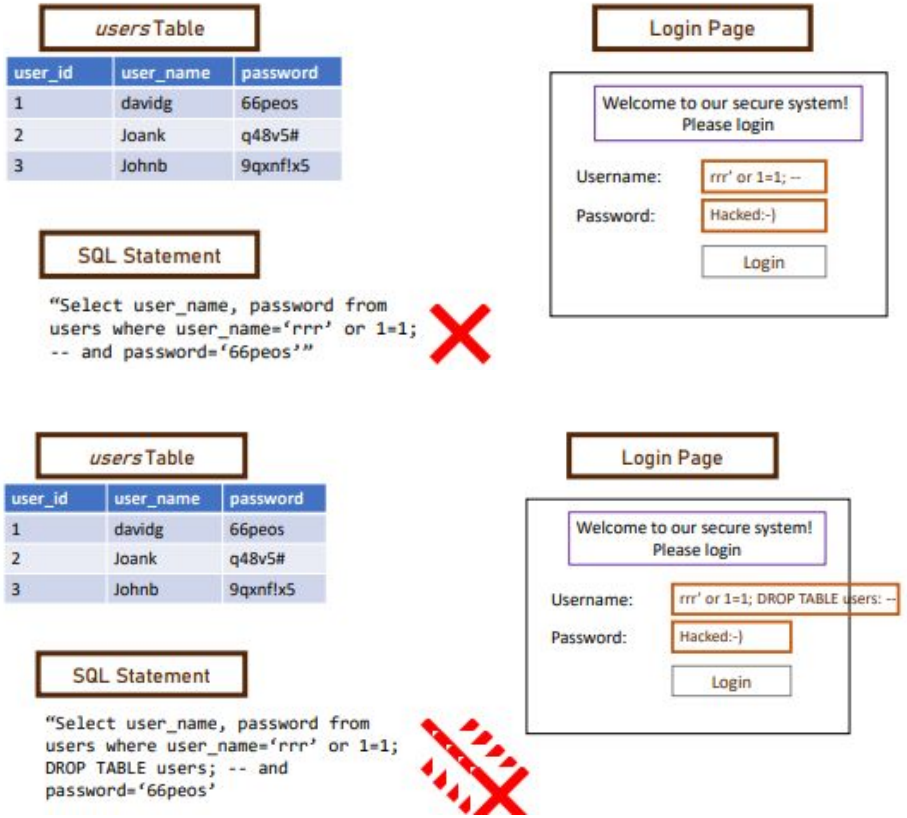
- ✔ Uses symmetric cryptography to encrypt the data
- ✔ Unique keys are generated for each connection
- ✔ Uses shared secret between both the parties
- ✔ Parties are authenticated using public key cryptography
- ✔ Messages transmitted includes message integrity check

5.4 Secure Architectures Process

Secure Code

- Often a result of mistake or lack of awareness
- unsecure code will disruption of service, data leak, data inconsistency, data loss.
- There are identified, widespread code vulnerabilities that must be handled in all systems
- The basic premise: Code is the last line of defense before the data, and must be well protected.

SQL injection



Use parameterized query

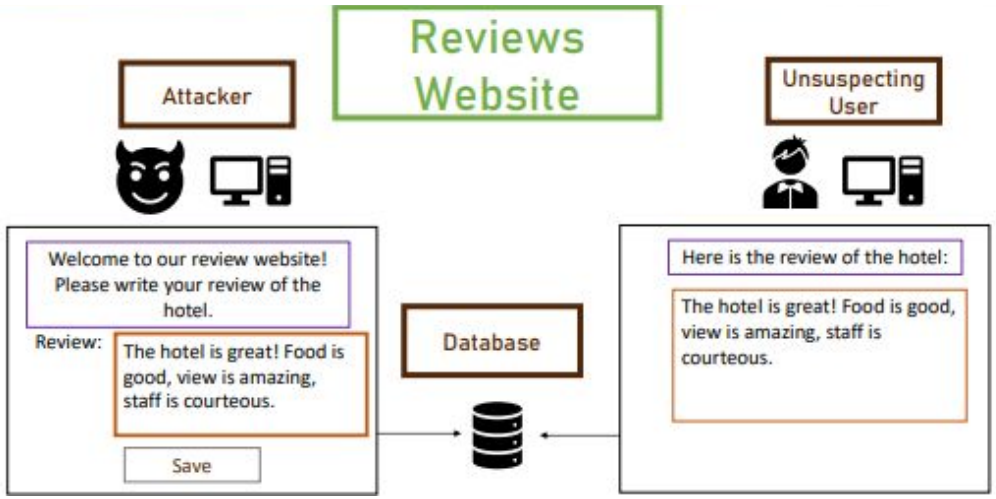
```
String sql="Select user_name, password from users where user_name=@userID
and password=@pwd"

SqlCommand cmd=new SqlCommand(sql, con);
cmd.Parameters.AddWithValue("@userId", txtUser);
cmd.Parameters.AddWithValue("@pwd",txtPassword);
```

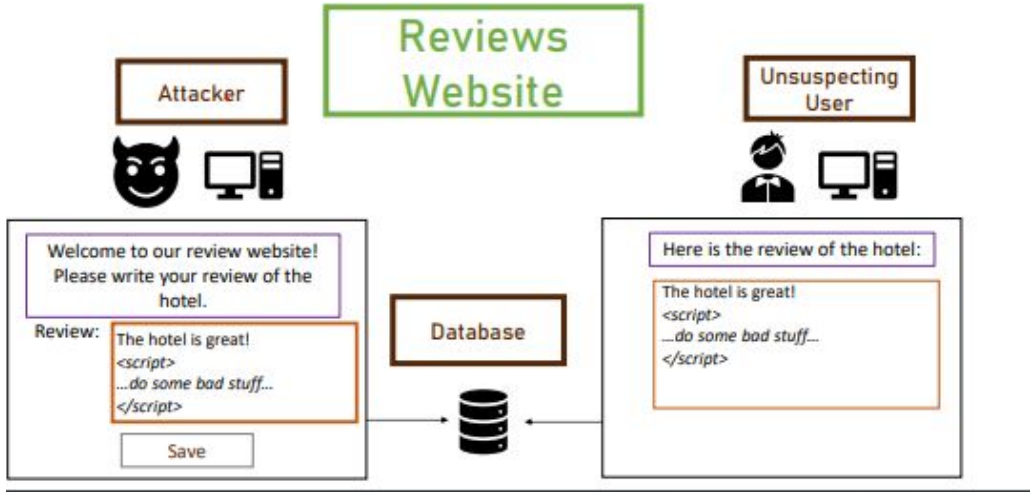
ALWAYS validate input

5.4 Secure Architectures Process

Secure Code

Cross site Scripting : Injects malicious scripts to an unsuspecting user's browser.





how to secure from it:

- ALWAYS validate input
- HTML-Encode the input

`<script>` → `&lt;script&gt;`

5.4 Secure Architectures Process

Secure Data

- Making sure the data is protected as much as possible. Usually data are in databases
- Two approaches of doing it.

Hard to access

- Have full-blown authorization in place
- Employ the least-privilege principle

Hard to read

Encrypt sensitive data

- key management: DB encryption (column, master key)
- key stores : securely store keys, certificates and more (windows certificate store , java keystore, azure key vault)
-

Using the database's encryption capabilities

The preferred way to go

Robust

Secure

5.4 Secure Architectures Process

Logging and Monitoring

- Making sure we know what's going on with our system
- Get notifications when something suspicious happens
- Collect data for future analysis of the system's behavior

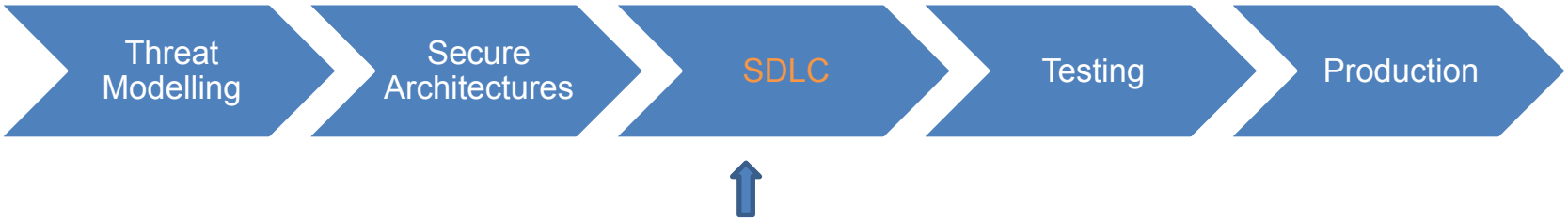
Log everything that might be security related (Last log-in time of users, Users' activity ,Validation problems)

Log metrics to provide aggregated data (No. of requests, No. of logins, No. of errors)

Define alerts for suspicious security-related activities (More than X validation problems in minute, More than 1,000 requests in 10 seconds, Log-in from an unknown location.

5.4 Secure Architecture Process

5.43 "SDLC" phase



Overview:

worked for Software Secure Development Life Cycle

https://www.microsoft.com/en-us/securityengineering/sdl/practices

Goal: Integrates security and privacy considerations into the development lifecycle

Participants: Architect
Dev Manager
Developers
QA

5.4 Secure Architecture Process

5.44 "Testing" Stage

Overview

- Security testing is an integral part of the development process
- Usually used to make sure the system works as expected
- Implement security-oriented testing should be used to make sure the system is secure.

Goal: To make sure the system is really secure

Participants: Architect
QA
Dev Manager
Developers

5.4 Secure Architecture Process

5.44 "Testing" Stage

Penetration testing: : A special type of test which simulates an attack on the system.

Purpose: to find weaknesses in the system that allow attackers to gain unauthorized access

Protect against: Data leak, data loss, data inconsistency, Disruption of Service

Types of penetration Testing: Black box testing, White Box testing, grey box testing

Black Box Testing

- The attacker (tester) has no prior knowledge on the system
- He's handed the URL or endpoint and that's it
- Might miss some important vulnerabilities
- Takes long time
- Best simulates real-world attacks

White Box Testing

- The attacker (test) is given full details and access
- Can see the source code, network, database, etc.
- Should scan everything for vulnerabilities
- Not simulates real world attacks

Grey Box Testing

- The attacker (tester) has some knowledge on the system
- Mainly around network and credentials
- Used to simulate an attack where the hacker already penetrated the network perimeter

5.4 Secure Architecture Process
5.44 "Testing" Stage

Which one to choose?

Black Box	- When main threats are from the outside (ie. Public website) - No time constraints
Grey Box	- When main threat is from the outside - Want to focus on the app's penentration testing
White Box	- When main threat is from the inside (ie. Internal org system) - Don't want to miss any potential vulnerability

Who conducts penetration testing?

- White Hat Hackers 
- For black box tests – preferably outside expert

Automated penetration testing

nessus Professional Qualys. RAPID7 Cymulate

5.4 Secure Architecture Process

5.45 "Production" Stage

- Continuous monitoring - System security is an on-going process that never ends
- System that was secure one year ago might no longer be so
- Need to maintain its security
- Using mainly:

  Security Review – continuously doing to protect from new virus, threats

  Penetration Testing - Make sure new changes didn't affect the system

  - Make sure the system is still secure

Goal: To make sure the system is still secure even in production

Participants: Everyone

5.5 DevOps Best Practices in Software Security

- Treat Software Security as a Priority from the start
- Conduct Security Awareness Training
- Use Code Reviews to Identify Potential Security Threats
- Use Static Code Analysis Tools
- Use Popular and Well-Maintained Libraries and Frameworks
- Secure coding guidelines and standards
  - Password Hashing
  - Encryption
  - sufficient Logging & Monitoring
  - Application Whitelisting/Least privilege(access to only the minimum resources needed to run securely)
  - proper Error Handling (This can allow hackers to execute their code or gain access through back-end servers by exploiting error messages)
  - Avoid SQL injections
- Penetration Testing
- Avoid as much as possible Legacy Infrastructures which creates security complications
- Incorporate secure software development practices into your DevOps practices
- Treat security vulnerabilities as software defects

5.6 conclusion

Secure software development is a journey that never ends.
Therefore, you should always look for new ways to improve and make your code more secure as technology evolves and hackers find new types of attacks to exploit against Software vulnerabilities.