SD3 Secure Web Application Development

Web Application Security Fundamentals

Web Application Security Fundamentals

- In this lecture ..
  - The Security Problem
  - Firewalls
  - What is meant by Security?
  - The Foundations of Security
  - Threats Vulnerabilities and Attacks Defined
  - How do you build a Secure Web Application
  - Secure your Network, Host & Application
  - Security Principles

The Security Problem

- When you hear talk about Web application security, there is a tendency to immediately think about attackers defacing Web sites, stealing credit card numbers, and bombarding Web sites with denial of service attacks.
- You might also think about viruses, Trojan horses, and worms.
- These are the types of problems that receive the most press because they represent some of the most significant threats faced by today's Web applications.

The Security Problem

- These are only some of the problems.
- Other significant problems are frequently overlooked.
- Internal threats posed by rogue administrators, disgruntled employees, and the casual user who mistakenly stumbles across sensitive data pose significant risk.
- The biggest problem of all may be ignorance.
- The solution to Web application security is more than technology.
- It is an ongoing process involving people and practices.

We Are Secure – We Have SSL/Firewall…

- This is a common misconception; it depends on the threat. For example, a firewall may not detect malicious input sent to your Web application. Also, consider the scenario where a rogue administrator has direct access to your application.
- Secure Sockets Layer (SSL) can be an integral part of your security, but not a complete solution by themselves.
  - SSL is great at encrypting traffic over the network.
  - However, it does not validate your application's input or protect you from a poorly configured server.

We Are Secure – We Have SSL/Firewall…

- They same can be said for firewalls. Do firewalls have their place? Of course they do.
  - Firewalls are great at blocking ports.
  - Some firewall applications examine communications and can provide very advanced protection.
  - A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules.
  - A firewall typically establishes a barrier between a trusted internal network and untrusted external network, such as the Internet.

We Are Secure – We Have SSL/Firewall…

- Firewalls are often categorized as either network firewalls or host-based firewalls.
- Network firewalls filter traffic between two or more networks and run on network hardware.
- Host-based firewalls run on host computers and control network traffic in and out of those machines.
- Firewalls are an integral part of your security, but they are not a complete solution by themselves.

What Is Meant By Security?

- Security is fundamentally about protecting assets. Assets may be tangible items, such as a Web page or a customer database or they may be less tangible, such as a company's reputation.
- Security is a path, not a destination.
- As you analyze your infrastructure and applications, you identify potential threats and understand that each threat presents a degree of risk.
- Security is about risk management and implementing effective countermeasures.

Foundations of Security

1. **Authentication** addresses the question: who are you?
   - It is the process of uniquely identifying the clients of your applications and services.
   - These might be end users, other services, processes, or computers.
   - In security parlance, authenticated clients are referred to as *principals*.

Foundations of Security

2. **Authorization** addresses the question: what can you do?
   - It is the process that governs the resources and operations that the authenticated client is permitted to access.
   - Resources include files, databases, tables, rows, and so on, together with system-level resources such as registry keys and configuration data.
   - Operations include performing transactions such as purchasing a product, transferring money from one account to another, or increasing a customer's credit rating.

Foundations of Security

3. Effective **auditing** and logging is the key to non-repudiation.
   - Non-repudiation guarantees that a user cannot deny performing an operation or initiating a transaction.
   - For example, in an e-commerce system, non-repudiation mechanisms are required to make sure that a consumer cannot deny ordering 100 copies of a particular book

Foundations of Security

4. **Confidentiality** also referred to as *privacy*, is the process of making sure that data remains private and confidential, and that it cannot be viewed by unauthorized users or eavesdroppers who monitor the flow of traffic across a network.
   - Encryption is frequently used to enforce confidentiality. Access control lists (ACLs) are another means of enforcing confidentiality.

Foundations of Security

5. **Integrity** is the guarantee that data is protected from accidental or deliberate (malicious) modification.
   - Like privacy, integrity is a key concern, particularly for data passed across networks.
   - Integrity for data in transit is typically provided by using hashing techniques and message authentication codes.

Foundations of Security

6. From a security perspective, **availability** means that systems remain available for legitimate users.
   - The goal for many attackers with denial of service attacks is to crash an application or to make sure that it is sufficiently overwhelmed so that other users cannot access the application.

Threats, Vulnerabilities and Attacks Defined

- A threat is any potential occurrence, malicious or otherwise, that could harm an asset. In other words, a threat is any bad thing that can happen to your assets.
- A vulnerability is a weakness that makes a threat possible. This may be because of poor design, configuration mistakes, or inappropriate and insecure coding techniques. Weak input validation is an example of an application layer vulnerability, which can result in input attacks.
- An attack is an action that exploits a vulnerability or enacts a threat. Examples of attacks include sending malicious input to an application or flooding a network in an attempt to deny service.

How Do You Build a Secure Web Application?

- It is not possible to design and build a secure Web application until you know your threats.
- An increasingly important discipline and one that is recommended to form part of your application's design phase is threat modelling.
- The purpose of threat modelling is to analyze your application's architecture and design and identify potentially vulnerable areas that may allow a user, perhaps mistakenly, or an attacker with malicious intent, to compromise your system's security.

How Do You Build a Secure Web Application?

- After you know your threats, design with security in mind by applying timeworn and proven security principles.
- As developers, you must follow secure coding techniques to develop secure, robust, and hack-resilient solutions.
- The design and development of application layer software must be supported by a secure network, host, and application configuration on the servers where the application software is to be deployed.

Secure Your Network, Host and Application

- **Securing Your Network:** *A secure Web application relies upon a secure network infrastructure. The network infrastructure consists of routers, firewalls, and switches.*

Component	Description
Router	Channel packets to ports and protocols that your application needs. Common TCP/IP vulnerabilities are blocked at this ring.
Firewall	The firewall blocks those protocols and ports that the application does not use. Additionally, firewalls enforce secure network traffic by providing application-specific filtering to block malicious communications.
Switch	Switches are used to separate network segments. They are frequently overlooked or over-trusted.

Secure Your Network, Host and Application

- **Securing Your Host:**

Component	Description
Patches and Updates	Many top security risks exist because of vulnerabilities that are widely published and well known. When new vulnerabilities are discovered, exploit code is frequently posted on Internet bulletin boards within hours of the first successful attack. Patching and updating your server's software is the first step toward securing the server.
Services	The service set is determined by the server role and the applications it hosts. By disabling unnecessary and unused services, you quickly and easily reduce the attack surface area.
Protocols	To reduce the attack surface area and the avenues open to attackers, disable any unnecessary or unused network protocols.

Secure Your Network, Host and Application

- **Securing Your Host:**

Component	Description
Accounts	The number of accounts accessible from a server should be restricted to the necessary set of service and user accounts. Additionally, you should enforce appropriate account policies, such as mandating strong passwords.
Files & Directories	Files and directories should be secured with relevant permissions.
Ports	Services running on a server listen on specific ports to serve incoming requests. Open ports on a server must be known and audited regularly to make sure that an insecure service is not listening and available for communication. In the worst-case scenario, a listening port is detected that was not opened by an administrator.

Secure Your Network, Host and Application

- **Securing Your Host:**

Component	Description
Auditing & Logging	Auditing is a vital aid in identifying intruders or attacks in progress. Logging proves particularly useful as forensic information when determining how an intrusion or attack was performed.
Registry	Many security related settings are maintained in the registry. Secure the registry itself by applying restricted Windows ACLs and blocking remote registry administration.

Secure Your Network, Host and Application

- **Securing Your Application:**

Component	Description
Input Validation	How do you know that the input that your application receives is valid and safe? Input validation refers to how your application filters, scrubs, or rejects input before processing.
Authentication	Authentication is the process where an entity proves the identity of another entity, typically through credentials, such as a user name & password.
Authorization	"What can you do?" Authorization is how your application provides access controls for resources and operations.

Secure Your Network, Host and Application

- **Securing Your Application:**

Component	Description
Configuration Management	Which database does your app connect to? How is your app administered? How are these settings secured?
Sensitive Data	Sensitive data refers to how your application handles any data that must be protected either in memory, over the wire, or in persistent stores.
Session Management	A session refers to a series of related interactions between a user and your Web application. Session management refers to how your application handles and protects these interactions.

Secure Your Network, Host and Application

- **Securing Your Application:**

Component	Description
Cryptography	How are you keeping secrets, secret (confidentiality)? How are you tamper-proofing your data or libraries?
Exception Management	When a method call in your application fails, what does your application do? How much do you reveal? Do you return friendly error information to end users? Do you pass valuable exception information back to the caller? Does your application fail gracefully?
Auditing and Logging	Who did what and when? Auditing and logging refer to how your application records security-related events.

Security Principles

Principle	Concepts
Compartmentalize	Reduce the surface area of attack. Ask yourself how you will contain a problem. If an attacker takes over your application, what resources can he or she access? Can an attacker access network resources? How are you restricting potential damage? Firewalls, least privileged accounts, and least privileged code are examples of compartmentalizing.
Use least privilege	By running processes using accounts with minimal privileges and access rights, you significantly reduce the capabilities of an attacker if the attacker manages to compromise security and run code.
Apply defence in depth	Use multiple gatekeepers to keep attackers at bay. Defence in depth means you do not rely on a single layer of security, or you consider that one of your layers may be bypassed or compromised.

Security Principles

Principle	Concepts
Do not trust user input	Your application's user input is the attacker's primary weapon when targeting your application. Assume all input is malicious until proven otherwise, and apply a defence in depth strategy to input validation, taking particular precautions to make sure that input is validated whenever a trust boundary in your application is crossed.
Check at the gate	Authenticate and authorize callers early at the first gate.
Fail securely	If an application fails, do not leave sensitive data accessible. Return friendly errors to end users that do not expose internal system details. Do not include details that may help an attacker exploit vulnerabilities in your application.

Security Principles

Principle	Concepts
Secure the weakest link	If an application fails, do not leave sensitive data accessible. Return friendly errors to end users that do not expose internal system details. Do not include details that may help an attacker exploit vulnerabilities in your application.
Create secure defaults	Is the default account set up with least privilege? Is the default account disabled by default and then explicitly enabled when required? Does the configuration use a password in plaintext? When an error occurs, does sensitive information leak back to the client to be used potentially against the system?
Reduce your attack surface	If you do not use it, remove it or disable it. Reduce the surface area of attack by disabling or removing unused services, protocols, and functionality. Does your server need all those services and ports? Does your application need all those features?