Research on the Design and Development of Server Security and Ethical Hacking

Konstantinos Mantzavinatos
7110132100210

MSc Thesis in
Control and Computing

Committee

Prof. Dionysios Reisis (Supervisor)

Assoc. Prof. Anna Tzanakaki

Prof. Hector E. Nistazakis

Depts of Physics & Informatics and Telecommunications

National and Kapodistrian University of Athens

Athens, Greece

January 12, 2024

Abstract

This paper delves into a research on the design and development of server security and ethical hacking. It is structured by four pivotal chapters.The initial chapter, focuses on defense, and introduces the fundamental concepts of server security, including the CIA triad, threats, attacks and vulnerabilities, setting the foundation in understanding the complexity in securing servers. The next chapter moves a more practical guide in setting and hardening a public web server in an enterprise setting, explaining essential security measures and best practices for robust defense mechanisms.

The next half ,focus on the attack, and delves into the specifics of common web server vulnerabilities such as SQL Injections and Cross-site Scripting, providing and insight into their way of working and the ways of mitigation.The final chapter introduces the role and methodologies of an Ethical Hacker, provides a discussion on vulnerability measurement standards and essential tools used in ethical hacking. This paper aims to provide an overall understanding of the challenges of server security and the ethical hacking practices employed in order to address them.

Chapter 1: Principle of Web Server Security

1.1 Introduction

Throughout history, integrity has not always been a staple of human conduct. In our physical environment, we've gathered vast experience in guarding against ill-intentioned individuals, this was achieved with the support of a framework of complex legal systems that describe lawful and unlawful behaviors. Furthermore, a variety of technological methods are employed to safeguard our belongings and secure information.

The digital realm, or cyberspace, however, still poses significant security challenges. It's a term increasingly used to describe the digital universe formed by interconnected computer networks, impacting many aspects of our lives. Cybersecurity, in this realm, is particularly demanding. As noted by Bruce Schneier (Schneier, 2012), *"complexity is the enemy of security."* The exponential increase in the number of Internet-connected devices and their manufacturers worsen the scale and diversity of cyberspace, elevating the risk of system failures.

Cybersecurity also grapples with considerable imbalances. Attackers have a plethora of tactics at their disposal, whereas defenders must be vigilant and ready for any possible threat at all times. Consequently, not all successful cyberattacks are due to oversight. In some instances, security measures are present but improperly implemented, potentially due to conflicts with user requirements. The growing interest in responsive security acknowledges that not all attacks can be prevented.

This chapter introduces the fundamental aspects of cybersecurity. It commences with an exploration of common cyber threats, followed by an examination of standard attack and defense strategies . Then, it delves into security basics across various fields: data security through cryptography , malware , software security , and network security . The chapter underscores the importance of ongoing testing.

1.2 Threats

Understanding the stakes is crucial before delving into cyberspace attacks and defenses. We will discuss the primary security objectives that offer a comprehensive view of security aspects.

Before "cybersecurity" became a common term, the focus was on "computer security," aimed at safeguarding assets like hardware (e.g., computers, smartphones), software, and data from potential threats leading to damage or loss.

Computer security encompasses information security and systems security, both foundational to cybersecurity. Information security protects data and its derived information, while systems

security focuses on ensuring that computer systems function as intended, preventing unauthorized interference. (Christen et al., 2020, #) [3]

The discussion on threats begins with information security, emphasizing three core objectives: confidentiality, integrity, and availability, collectively known as the 'CIA triad' (origin of this acronym is unclear). Security measures target one or more of these goals:

- **Confidentiality**: Prevent unauthorized access to information.
- **Integrity**: Halt or detect unauthorized data alterations.
- **Availability**: Avoid unauthorized deletion or disruption of services.

These objectives pertain to both stored data (on computers or paper) and transmitted data (over networks). The term 'unauthorized' implies established norms regarding who can interact with the data.

In some cases, only one individual is authorized, like in the case of a smartphone or computer with encrypted storage (full-disk encryption). In other instances, such as online banking, both confidentiality and integrity are crucial for secure message exchanges between two authorized parties.

The three fundamental goals of confidentiality, integrity, and availability focus on content protection. However, the identity of involved parties is also a concern. For instance, ensuring the authenticity of an email sender is crucial, as is non-repudiation to prevent denial of certain actions like message transmission. Authenticity and non-repudiation are essential for holding individuals accountable. (*The CIA Triad*, n.d.) [4]

1.2.1 Systems Security

Systems security revolves around designing systems to safeguard stored data, pursuing the same objectives as information security.

Achieving these goals can be approached in various ways. For instance, confidentiality might be ensured through data encryption or a combination of authentication (like password entry) and access controls (dictating user file access). Crafting systems that effectively integrate security measures is a complex task.

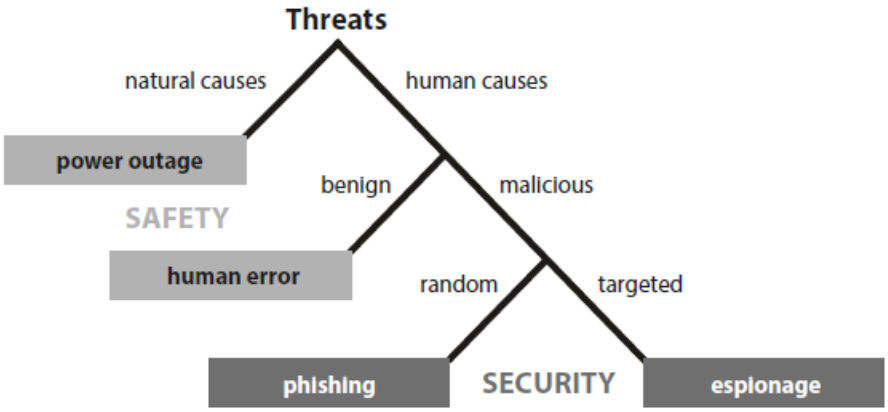
Systems security also extends beyond safeguarding information. Some systems, while not containing sensitive data, are critical for reliable functionality. For example, bugs in an

operating system's authentication component could allow attackers to disrupt or gain unauthorized control. Integrity and availability are primary concerns in systems security, with confidentiality sometimes aimed at protecting intellectual property. However, relying on the obscurity of a system's workings for security is generally frowned upon.

Cyber-physical systems, like traffic lights or industrial control systems, are of particular interest in systems security. Some of these are critical infrastructures, where failures could significantly impact society. Policy makers worry about future conflicts targeting these infrastructures to create chaos without physical force. Notable examples include the Stuxnet malware attack on an Iranian uranium enrichment facility **(Ralph Langner, 2013)[18]** and the 2015 attack on a Ukrainian power plant **(Zetter, 2016) [28]**.

1.2.2 Security Versus Safety

Safety and security, while often intertwined, address different threats. Safety concerns include natural disasters and benign human errors, such as accidental data deletion. Safety is crucial in cyber-physical systems, where system failure could result in human harm. Cars and airplanes, for example, incorporate numerous safety-critical systems.



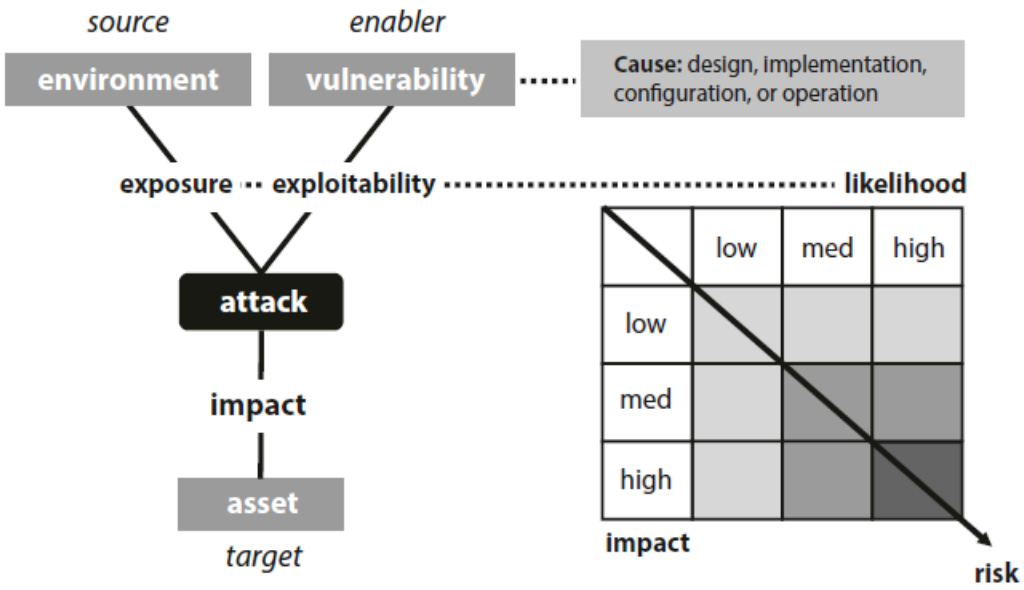
Security, however, primarily focuses on protecting against malicious human acts, known as attacks. These include both random and targeted attacks, with the latter posing greater defensive challenges due to their strategic nature.

1.2.3 Security as Risk Management

Developing software and hardware is fraught with complexity and susceptibility to errors. On average, thousands of lines of code may contain multiple bugs, and even meticulous code

reviews only partially mitigate this. The 'Common Weakness Enumeration' (https://cwe.mitre.org) lists frequent mistake types, with buffer overflows and SQL injections being common examples.

A risk shows the possibility of an asset or data being compromised or destroyed by an attacker



A specific weakness manifestation in a product is called a vulnerability. **Vulnerability** is a weakness that can be exploited by cybercriminals to gain unauthorized access to a computer system. **(Sen, 2023) [22]**

Vulnerabilities can be registered through the '**Common Vulnerabilities and Exposures**' (https://cve.mitre.org) database archives, which contain over 115,000 entries as of mid-2019.

There they receive a **Common Vulnerability Scoring System (CVSS)** score to determine the severity. This score is used as a standard from many companies to determine the accurate scores for their system' vulnerabilities. This scoring system helps decide the priority in dealing with every given threat. Those scores are calculated using values such as the attack vector (network, local,adjacent, physical), the complexity of an attack, the privileges required, whether or not the attack needs user interaction to function , as well as the impact that a successful attack could deliver to the organization's confidentiality, integrity and availability if data. This score goes from 0 to 10 depending on the metrics.

A ***Threat*** is a process that amplifies the potential of an adverse event, such as an attacker (or threat actor) exploiting a vulnerability. Some vulnerabilities raise more threat concerns over others due to the increased probability of the vulnerability being exploited. The higher the

reward for an attacker can gain from an exploitation the more likely the attack to be carried over.

An ***exploit*** in any code can be used to take advantage of a weakness, many exploits can be be found through open source platforms such as exploitdb (https://www.exploit-db.com) and we can also find a lot of exploits hosted in GitHub and GitLab

The notion of ***risk*** encapsulates the uncertainty associated with these vulnerabilities. Risk severity is a function of the potential attack impact on an asset (usually monetary) and the attack's likelihood. Predicting impact and likelihood is challenging, as depicted in the next Figure illustrating the link between risk and vulnerabilities. (*CyberSecurity Definitions*, n.d.) [7]

(https://www.bmc.com/blogs/security-vulnerability-vs-threat-vs-risk-whats-difference/)

Risk management can involve avoiding, mitigating, transferring, or accepting risks. Unfortunately, system designers often transfer risks to users, creating negative externalities due to asymmetric power dynamics. This can reduce incentives for system operators to prioritize security.

1.3 Approaches for Attack and Defense

Successful attacks require a method, opportunity, and motive. Understanding various attacker types and techniques is insightful.

1.3.1 Attackers and Their Motives

Cyber attackers' motives often mirror those in the physical world. Corporate spies, cybercriminals seeking financial gain, and nation states pursuing political or power objectives are common perpetrators. Nation states can execute sophisticated 'advanced persistent threat' (APT) attacks, requiring substantial resources.

Other attackers include hobbyists ('script kiddies'), hacktivists advocating causes like free speech, and rogue hackers attacking out of curiosity or personal gain. '**Black hats**' denote

malicious attackers, while **'white-hat hackers**' or '**Ethical Hackers**' focus on enhancing security by reporting vulnerabilities. We will discuss more about it later in chapter 4.

A comprehensive security view must also consider insiders, such as disloyal employees or vendors. Insider attacks can be more potent due to their access and knowledge. Supply-chain attacks, infecting vendor systems to target high-profile entities, are an example of this (Korolov 2018).

1.3.2 Defenses

Effective defense strategies blend proactive and reactive measures. Six approaches are commonly employed:

- **Preventive controls**: Block or prevent attacks through exposure control (like firewalls) or exploitability reduction (fixing vulnerabilities).
- **Deterrence**: Increase adversary effort to make targets less appealing. Two-factor authentication is an example.
- **Deflection**: Redirect adversary efforts to other targets, achievable with honeypots.
- **Detection**: Employ real-time alerts (like Intrusion Detection Systems) or log data for post-incident analysis.
- **Mitigation**: Limit attack impacts, such as through network segmentation to isolate compromised systems.
- **Recovery**: Revert attack effects and resume normal operations quickly, using off-site backups and emergency protocols.

Organizations typically combine these techniques for comprehensive security. The goal is to prevent as many attacks as possible and manage the rest reactively.

Saltzer and Schroeder's Security Design Principles (1975, updated by Smith 2012) **[19]** provide a framework for building secure systems:

- Continuous improvement: Security is an ongoing process.
- Least privilege: Limit access rights to what's necessary.
- Defense in depth: Employ multiple security mechanisms.
- Open design: Avoid reliance on obscurity.
- Chain of control: Execute trustworthy software and restrict non-trustworthy components.
- Deny by default: Grant access only when explicitly specified.
- Transitive trust: Trust extends through a chain of trust relationships.
- Trust but verify: Confirm even trustworthy components' identities.
- Separation of duty: Distribute critical tasks among different entities.
- Principle of least astonishment: Ensure usability and predictability of security mechanisms.

1.3.3 Stages of an Intrusion

The **'Cyber Kill Chain**' framework by Lockheed Martin (*Cyber Kill Chain*®, 2011) [6] outlines an attack's typical stages:

1. Reconnaissance: Identifying and selecting targets.
2. Weaponization: Coupling malware with exploits.
3. Delivery: Delivering the weapon to the target.
4. Exploitation: Triggering the malicious code.
5. Installation: Establishing a presence on the target system.
6. Command and Control: Gaining control over the compromised system.
7. Actions: Achieving the attacker's goals, like data exfiltration.

While widely used, this framework has limitations, especially in modern networks employing principles like least privilege and defense in depth. Moreover, not all attacks require these steps, such as data theft via an SQL injection.

1.4 Threats and Solutions in Data Security

Data storage and transmission are central to computing. This section reviews attacks on data and introduces cryptographic countermeasures, focusing on 'data in transit.'

1.4.1 Unauthorized Disclosure of Information

Attacks on confidentiality typically involve gaining access to data, either at rest or in transit. Eavesdropping is a common method for intercepting data in transit, particularly in distributed systems using public networks. Passive attackers, like eavesdroppers, do not disrupt transmissions.

Encrypting data is the primary countermeasure against confidentiality breaches. Secure communication requires cryptographic key establishment. Symmetric cryptography uses the same key for encryption and decryption, exchanged via a secure channel. Asymmetric cryptography (public-key cryptography) uses a public-private key pair, shared openly and kept secret, respectively.

Traffic analysis, discerning communication patterns without accessing the content, is another threat. Techniques like multiple encryption layers and routing through additional nodes (as in the Tor network - http://torproject.org) can counter traffic analysis.

1.4.2 Unauthorized Modification and Fabrication

Active attackers may seek to alter or forge messages. Encryption alone doesn't prevent data modification, necessitating additional integrity protection. Message authentication codes (MACs) combined with cryptographic keys help detect tampering. However, MACs don't prevent replay attacks, where intercepted messages are resent . Countermeasures include using counter values in messages to detect replays.

Relay attacks, like those on 'smart' car entry systems, pose a different challenge. These attacks exploit the communication delay between the key and the car, tricking the system into unlocking. Manufacturers could potentially mitigate this by measuring message delays.

Using cryptography to secure data is only the first step in implementing basic safety features. Symmetric cryptography though has limitations: scalability issues due to key management, increased key compromise risk, and insufficient non-repudiation (proof of message origin). Asymmetric cryptography addresses these by using public-private key pairs, enhancing security in applications like email (S/MIME, OpenPGP) and the World Wide Web.

Case Study: Secure HTTP

Web browsers communicate with servers using HTTP (Hypertext Transfer Protocol), with many servers redirecting to HTTPS for secure communication. HTTPS employs the Transport Layer Security (TLS) protocol, providing eavesdropping and tampering protection. It also prevents server impersonation.

TLS uses asymmetric keys for key establishment, not data encryption, to ensure forward secrecy. The server signs messages during key establishment to prevent man-in-the-middle (MitM) attacks. Server certificates, digitally signed by Certification Authorities (CAs), verify server authenticity. Browsers come with pre-installed public keys of major CAs (root certificates) to validate these signatures.

Attacks against HTTPS include sslstrip, targeting router-controlled networks to prevent HTTPS redirections, and CA compromise, where rogue certificates are issued. Solutions like the HTTPS Everywhere browser extension and Strict Transport Security headers address sslstrip attacks. Certificate Transparency helps detect rogue certificates by requiring CAs to log issued certificates publicly.

1.4.3 Malware Threats and Solutions

Malware, or malicious software, compromises confidentiality, integrity, and availability. It can propagate autonomously (like worms) or require human assistance (via email attachments or drive-by downloads).

Self-propagating malware, such as the SQL Slammer worm, exploits vulnerabilities in systems. With increased firewall use, malware often relies on human assistance, employing social engineering for delivery. Techniques include email attachments, drive-by downloads (exploiting browser vulnerabilities), malvertising (malicious ads), and waterholing attacks (compromising frequented websites or software updates).Countermeasures against malware include regular software updates, user education, and robust security protocols.

When malware operates on a target system, it executes its payload, often disguised as benign or masking as an error, known as a Trojan horse. Initially, malware aimed to corrupt systems, either by deleting files or preventing the operating system from booting. Later, ransomware emerged, encrypting files and demanding ransom for decryption keys. Other payloads include keyloggers for stealing credentials and remote control tools for creating botnets, which can be used for spam emails or Distributed Denial of Service attacks.

Countermeasures

Key countermeasures against malware include prompt installation of security patches and user awareness training. These measures are designed to prevent exploitation of known vulnerabilities and accidental malware execution. Anti-virus (AV) solutions monitor systems for malware indicators, using either signature-based detection or heuristics for behavior analysis. However, AV solutions have limitations, including the inability to detect modified malware and false positives. Intriguingly, some AV tools can introduce vulnerabilities themselves, as seen in cases like Windows Defender. An alternative to AV solutions is using sandboxes for isolating and analyzing suspicious files. While effective, sandboxes are not foolproof, as malware can be designed to evade sandbox detection.

Active defense against malware, like attempting to disrupt its command and control infrastructure, remains a legal and ethical debate. Actions like deleting a ransomware author's mailbox can have unintended consequences, either hindering recovery efforts or, in cases where the malware is destructive, proving futile.

1.4.4 Threats and Solutions

Threats and Solutions in Software Security

Software security deals with vulnerabilities arising from programming errors. Buffer overflows and SQL injections are common examples. These vulnerabilities can be identified through code

audits, reverse engineering, and fuzzing. Fuzzing, in particular, has been effective in uncovering vulnerabilities that might be missed during code reviews.After discovering a vulnerability, reporting it can either be done publicly (full disclosure) or privately to the vendor (responsible disclosure), with each approach having its pros and cons. Vendors should have established practices for handling vulnerability reports, including acknowledging reports, confirming problems, and scheduling fixes. Transparency and quick response are crucial, and some vendors offer bug bounty programs as incentives.

Threats and Solutions in Network Security

Networked systems are exposed to various threats, including reconnaissance, where attackers gather information about a target using public services like WHOIS or DNS. Tools like Shodan and Censys can also provide attackers with information about vulnerable systems.

Firewalls are key for perimeter security, controlling access to network services. They filter packets based on headers, though this can be circumvented using techniques like port redirection or Deep Packet Inspection (DPI). DPI, however, raises privacy concerns and isn't foolproof against sophisticated tunneling methods.

Network Intrusion Detection Systems (NIDS), like Snort, use signature-based or anomaly-based methods to detect attacks, but they have limitations, especially with encrypted traffic. The effectiveness of NIDS is often hampered by the base rate fallacy, where the number of false positives can overshadow actual threats.

Continuous Testing

Regular testing is essential for maintaining security, with practices including running security scanners and conducting penetration tests. Security scanners like Nessus check for known vulnerabilities, while penetration tests simulate real attacks to assess systems' resilience. Tools like Metasploit facilitate penetration testing but raise ethical concerns regarding their dual-use nature and regulation.

Overall, combating cyber threats requires a multifaceted approach, encompassing proactive security measures, continuous monitoring, and effective response strategies. Education, awareness, and cooperation between vendors, users, and security professionals are critical in maintaining a robust cybersecurity posture.

Chapter 2:Server Creation using proper security

Now that we have some basic concept of cyber security we can continue discussing ways to properly set up a secure web server. We will discuss the most difficult case which is a public web server of an enterprise scale. For a local and private server a lot of those steps could be unnecessary to implement but good to know. Also when using cloud providers to host our server, most of those are being done automatically by the provider which, as we will see also in the chapter, is called mitigation of vulnerabilities and it is a solid way to overcoming them.

2.1 Security Standards and Compliance

To ensure server security, it's crucial to first identify and understand the threats that need to be addressed. Awareness of potential threats is key to understanding the importance behind various fundamental technical security practices. Threats to data and resources often originate from errors, whether they are bugs in operating system and server software, creating vulnerabilities, or mistakes by users and administrators. These threats might originate from deliberate actions, such as an attacker aiming to access server information, or from unintended mistakes, like an administrator forgetting to disable a former employee's user accounts. Threat sources can be internal, like a dissatisfied employee, or external, like a remote hacker. Organizations are advised to perform **risk assessments** to pinpoint specific threats to their servers and evaluate the efficacy of existing security controls, following guidelines in **NIST Special Publication (SP) 800-30**, Risk Assessment Guide for Information Technology Systems. This helps organizations better understand their security posture and decide on security measures for their servers.

The security practices in this document draw from widely recognized technical security principles and practices documented in various NIST SPs and other sources, including the Department of Defense (DoD) Information Assurance Technical Framework. NIST SP 800-27, NIST Special Publication 800-44 Version 2 **[25]** ,and Engineering Principles for Information Technology Security **[26]**. This chapter tries to provide a foundation for a more structured approach to IT security design and implementation.

Understanding the threats relevant to the server's operational environment is vital for determining the right security controls. These recommendations, as said before, are focusing on web servers that are in typical enterprise settings, facing usual threats and security needs. If a Server is in high-security or legacy environments then we might require more strict controls. Security categorization of information and systems, as described in FIPS Publication (PUB) 199, is essential in determining the level of protection needed, based on the potential impact of a security breach.

Server security involves several steps, starting with planning the OS installation and deployment, securing the underlying OS, and securing the server software. For content-hosting servers, securing the content is critical, though the specifics vary by server type and content nature. Network protection mechanisms and secure administration processes, including patch application, log monitoring, and periodic security testing, are also key components for the server's maintained security

When addressing server security issues, it is an excellent idea to keep in mind the following general information security principles

- **Simplicity:** Security mechanisms and systems should be as straightforward as possible, minimizing complexity to reduce security risks**.**
- **Fail-Safe Stance:** In case of failure, systems should default to a secure state where security controls remain active and enforced, prioritizing security over functionality.
- **Complete Mediation:** Access to all resources should be controlled and mediated, using systems like file permissions and firewalls to enforce access policies.
- **Open Design:** Security of a system should not rely on the obscurity of its design or implementation.
- **Separation of Privilege:** System functions should be segregated to provide granularity in access control. Roles, like system and database administrators, should be distinct wherever feasible.
- **Least Privilege:** Users, processes, and systems should have only the essential privileges needed for their function, limiting the impact in case of compromise.
- **Psychological Acceptability:** Security mechanisms should be understandable and acceptable to users. Training and sensible design can help users embrace security practices without seeking workarounds.
- **Least Common Mechanism:** Minimize the use of shared mechanisms to reduce the chances that a breach in one area could affect others.
- **Defense-in-Depth:** Employ multiple layers of security controls to protect systems, ensuring that a breach in one layer doesn't compromise the entire system.
- **Work Factor:** The effort required to breach system security should be disproportionately higher than the potential gain for an attacker.
- **Compromise Recording**: Maintain comprehensive logs and records to document any security breaches, aiding in post-incident analysis and helping to refine future security measures.

These principles form the foundation of a robust and adaptable security posture for servers, ensuring a balanced approach that takes into account the dynamic nature of cybersecurity threats.

2.2 Planning before Deployment

The key to successfully deploying a secure web server lies in meticulous planning before installation, configuration, and deployment. This planning ensures not only robust security but also adherence to all pertinent organizational policies. Many security and performance issues of web servers are often attributable to inadequate planning or ineffective management controls, highlighting the significance of these controls. In many organizations, IT support is fragmented, leading to inconsistencies that can create security gaps and other problems.

Security considerations should be integral from the early stages of system development to optimize security and reduce costs. Addressing security post-deployment is often more challenging and costly. Detailed, well-thought-out deployment plans are crucial for making informed decisions about usability, performance, and risk trade-offs and for maintaining secure configurations. Such plans also help in identifying vulnerabilities, which often appear as deviations from the plan. **(*Securing Network Servers*, 2000) [21]**

Key considerations in the planning stages include:

- Determining the Web server's purpose, including the types of information it will store, process, or transmit, and the security requirements for this information.
- Identifying network services and software to be installed, users or user categories, their privileges, and management protocols for the Web server.
- Deciding on authentication methods and data protection strategies.
- Selecting appropriate Web server applications, considering factors like cost, compatibility, employee expertise, manufacturer relationships, vulnerability history, and functionality.
- Collaborating with manufacturers during planning stages.

Choosing a Web server application often dictates the choice of the operating system (OS). The selected OS should provide capabilities like restricting administrative activities, controlling data access, disabling unnecessary network services, controlling executables like CGI scripts, logging activities for intrusion detection, and offering host-based firewall features.

2.1.1 Security Management Staff

The security of a web server is closely linked to the overall information system security posture of an organization. Therefore, various IT and system security staff should be involved in web server planning, implementation, and administration. These roles might vary across organizations but generally include:

- Senior IT Management/Chief Information Officer: Responsible for developing and maintaining security policies and ensuring compliance across the organization.
- Information Systems Security Program Managers: Oversee the implementation and adherence to security standards and policies.

- Information Systems Security Officers: Oversee information security within an organizational entity, ensuring compliance with organizational and departmental policies.

Management Practices
Effective management practices are essential for a secure Web server. These include:

- Establishing an organizational information system security policy.
- Implementing configuration/change control and management.
- Conducting risk assessment and management.
- Developing standardized configurations.
- Adopting secure programming practices.
- Providing security awareness and training.
- Planning for contingencies and disaster recovery.
- Undertaking certification and accreditation processes

System Security Plan
A system security plan is vital for safeguarding information system resources. It should include:

- System identification with key contact details, purpose, sensitivity level, and environment.
- Description of control measures in place for system protection.
- Categorization of controls into management, operational, and technical.

2.2.2 Alternative Web Server Platforms

Organizations may also consider alternative web server platforms like Trusted Operating Systems (**TOS**), Web Server Appliances, Pre-Hardened Operating Systems and Web Servers, and Virtualized Platforms. TOS offers high security but may require intricate knowledge and administration. Web Server Appliances are "plug-and-play" solutions optimized for web server support, offering security and performance benefits. Pre-Hardened Systems come with OS and Web server applications modified for high security, while Virtualized Platforms allow running multiple Web servers on a single host, offering cost efficiency and response flexibility to attacks.

Each alternative platform comes with its considerations, including compatibility with existing infrastructure, ease of administration, response time to vulnerabilities, and support lifespan.

Web server appliances are integrated hardware and software solutions designed to simplify the deployment and management of web servers. These appliances use a streamlined operating system optimized for web server functions, enhancing security by reducing unnecessary features and services. The web server software on these appliances is typically pre-hardened and configured for security, making them an ideal choice for small to medium-sized organizations that may lack the resources for a full-time web administrator. However, they may not be suitable for larger, more complex websites due to limitations in supporting diverse types of active content and difficulties in integrating appliances from different manufacturers. Additionally, they may restrict the separation of backend and frontend functions, limiting scalability and flexibility.

Pre-Hardened Systems: Pre-hardened operating systems and web server packages provide higher security out of the box. These packages, either as hardware systems or software distributions, combine an OS with a web server application, both modified for enhanced security. Common features include secure default configurations, hardened OS and web server software, comprehensive auditing capabilities, and simplified security administration interfaces. These systems are particularly suitable for organizations handling sensitive data or operating in high-risk environments. When considering these systems, organizations should assess the compatibility with their existing web applications, the ease of administration, and the manufacturer's responsiveness to vulnerabilities.

Virtualization technology allows multiple virtual machines, each with its own OS and applications, to run on a single physical host. This approach offers flexibility and cost-effectiveness, enabling efficient management of multiple web servers. There are different types of virtualization, including full virtualization, native virtualization, and paravirtualization, each with its unique advantages and performance considerations. However, virtualization adds complexity to the web server setup, requiring diligent security and configuration management for both the host and guest OSs. Regular security updates and backups are essential to maintain a secure environment. Additionally, each virtual machine should be treated as an independent risk to minimize potential network-wide impacts from security breaches.

2.3 Hardening OS

Hardening the underlying operating system (OS) is crucial in protecting the server from malicious attacks. This process entails configuring the OS away from default settings, which are often focused on ease of use and functionality, to a setup that emphasizes security. Each organization must tailor its server configurations to meet its specific security needs. The following steps provide a general guide for maintaining basic OS security:

1. **Planning Installation and Deployment**:
   - Determine the purpose and requirements of the web server.
   - Choose an appropriate OS based on security needs and compatibility with the web server application.
   - Plan the network placement and architecture of the web server.

2. **Patching and Updating the OS**:
   - Regularly update and patch the OS to fix vulnerabilities and enhance security features.
   - Establish a routine for checking and applying updates from reliable sources.

3. **Hardening and Configuring the OS**:
   - Remove or disable unnecessary services, features, and components to reduce the attack surface.
   - Configure user access controls and permissions to limit unnecessary access to the system.
   - Set up system auditing and logging to monitor and record system activities.
   - Ensure secure network configurations, including firewall settings and network service configurations.
   - Refer to security configuration guides and checklists for detailed recommendations specific to the OS being used.

4. I**nstalling Additional Security Controls**:
   - Implement security measures such as antivirus programs, intrusion detection systems, and firewalls.
   - Consider using automated tools for additional hardening and monitoring of the OS (refer to Appendix D for tool recommendations).

5. **Testing the OS Security**:
   - Conduct regular security assessments and vulnerability scans to identify and address potential weaknesses.
   - Review and analyze security logs to detect any unusual activities or potential security breaches.
   - Periodically reassess and update security configurations to adapt to new threats and organizational changes.

These steps should be part of an ongoing process to ensure that the web server remains secure against evolving threats. The security of the web server application and the network, covered in the next sections, also plays a crucial role in the overall security of the web server. We always have to remember that security is a continuous effort that requires regular updates, monitoring, and adjustments to stay effective.

2.2.1 Patch and Upgrade Operating System

Applying patches and upgrades to an operating system (OS) is a critical step in maintaining the Web server. Known vulnerabilities in an OS can be exploited by malicious attackers, making it essential to correct these vulnerabilities, especially when the server is exposed to untrusted users. The process of patching and upgrading an OS should be methodical and carefully documented. The key steps involved include:

1. Developing a Patch Management Process:
   - Establish a documented and consistent process for patch management. This includes procedures for identifying, testing, applying, and verifying patches.
   - Regularly review and update the patch management process to adapt to new challenges and changes in the IT environment.

2. Identifying Vulnerabilities and Applicable Patches:
   - Regularly check for updates and security advisories from the OS vendor and other trusted sources.
   - Utilize tools and services that notify administrators of new vulnerabilities and available patches.

3.Temporary Mitigation of Vulnerabilities:
   - Implement interim measures to mitigate vulnerabilities if patches are not immediately available or if they need time for testing.
   - These measures can include configuration changes, additional monitoring, or employing compensating controls.

4. Installing Permanent Fixes:
   - Apply patches, hotfixes, service packs, or updates that address identified vulnerabilities.
   - Ensure that patches are obtained from a reliable and authentic source to avoid introducing malware.

5. Securing Web Servers During Patching:
   - New Web servers should be protected during the patching process. This can be achieved by keeping the servers disconnected from public networks, or connecting them to a secure, isolated network until patching is complete.
   - Consider using out-of-band methods, like transferring patches via CDs, to avoid exposure to threats.

6.Testing Patches Before Deployment:
   - Test patches on a system with an identical configuration to the Web server before applying them to the production environment.

- This helps to ensure that the patch does not adversely affect the system or introduce new vulnerabilities.

7. Controlled Patch Installation:
   - While automatic patch downloading can be enabled, automatic installation should be avoided.
   - Administer patches manually after thorough testing to prevent unintended disruptions or compatibility issues.

Effective patch management is a crucial aspect of maintaining the security and integrity of a Web server. It requires a proactive approach, with regular monitoring for updates and swift action to address vulnerabilities. The process should be integrated into the overall security strategy of the organization to ensure continuous protection against emerging threats.

2.2.2 Minimizing attack surface

Minimizing the attack surface involves removing or disabling any unnecessary services and applications that are not essential for the Web server's function. Here are the recommended steps for securing a Web server by managing its services and applications:

1. **Use Dedicated Hosts for Web Servers:** Ideally, a Web server should operate on a host dedicated solely to this purpose. This minimizes the risk of other services on the host being exploited to compromise the Web server.

2. **Disable Unnecessary Services:** By default, operating systems may come with various services enabled that are not needed for a Web server's operations. These should be disabled to reduce potential vulnerabilities. Common examples of services to consider disabling include file and printer sharing, wireless networking services, remote control programs (especially those not using strong encryption), directory services, email services, and network management tools.

3. **Opt for Minimal Installation:** When installing the OS, choose the 'minimal installation' option if available. This approach installs the least number of components necessary, reducing the effort required to remove or disable unnecessary services.

4. **Remove Unwanted Applications**: Rather than just disabling unneeded applications, it's often safer to completely uninstall them. This ensures that these applications cannot be re-enabled inadvertently or through an attack.

5. **Secure Remote Access:** If remote access is necessary, ensure it uses secure methods like SSH or IPsec, particularly if the default remote access service doesn't provide strong encryption.

6. **Regularly Review and Update Configuration:** Regularly review the services and applications on the Web server to ensure that only necessary components are running. This should be part of routine maintenance and performed especially after any major changes to the server or its software.

7. **Monitor Logs:** By reducing the number of services, logs become more manageable and any unusual behavior is easier to detect.

8. E**valuate Additional Services Carefully**: If services beyond the basic Web server functionality are required (like database access, file transfer protocols, or remote administration services), assess the risks versus benefits for each service. Enable such services only if they are essential and if their security can be appropriately managed.

These steps are designed to streamline the Web server's operation, focusing it on its primary function and reducing the potential for security breaches through extraneous applications or services. We always have to remember that in server management, "**less is often more"** in terms of security.

2.4 Configuring operating system (OS)

This process involves setting up the OS to confirm the identity of users, ensuring only authorized individuals can access administrative features or other sensitive areas. Here are the key steps to configure user authentication on an OS for Web servers:

1. **Remove or Disable Unneeded Default Accounts and Groups:** Default OS configurations often include generic user accounts (like guest accounts) and groups that are unnecessary for a Web server's operations. These should be removed or disabled to reduce potential security risks. Change the names and passwords of necessary default accounts to comply with organizational security policies.

2. **Disable Non-Interactive Accounts**: For accounts that need to exist but do not require interactive logins (like service accounts), disable their login capabilities. For Unix systems, this can involve setting the login shell to a non-functional option like `/bin/false`.

3. **Establish User Groups**: Organize users into groups and assign rights at the group level, rather than to individual users. This simplifies the management of access rights and permissions.

4.**Create Necessary User Accounts**: Based on your deployment plan, create only those user accounts that are necessary. Avoid using shared accounts except in situations where no alternatives are available.

5.**Adhere to Organizational Password Policies**: Passwords should be set in accordance with the organization's policies regarding length, complexity, aging, and reuse. Consider implementing requirements like a minimum of eight characters, a mix of character types, and regular password changes.

6. **Configure the System Against Password Guessing**: To prevent unauthorized access, configure the OS to delay or lock accounts after several unsuccessful login attempts. This measure can also inadvertently lead to denial-of-service if abused, so balance between security and usability is key.

7. **Prevent Network Login to Sensitive Accounts**: For high-security accounts, like those of administrators, disable the ability to log in over the network if remote administration isn't required.

8. **Implement Strong Authentication Mechanisms**: For high-value or high-risk sites, consider stronger authentication methods like biometrics, smart cards, or one-time password systems. Ensure these mechanisms align with organizational policies.

9. **Protect Passwords in Transit**: Use encryption technologies like SSL/TLS, SSH, or VPNs to protect passwords from being intercepted during

2.5 Enhancing Security Measures for Web Servers

2.5.1 Initiating the Installation of Web Server Software

After successfully securing the operating system, the next step is to commence the installation of the selected web server software. It's crucial to meticulously review the documentation provided by the web server's developer to grasp the installation options thoroughly. Additionally, it's advisable to check the manufacturer's website or a reputable vulnerability database, such as the National Vulnerability Database (NVD) (*National Vulnerability Database*, n.d.) for any known vulnerabilities and corresponding patches that might need to be applied during the installation. Only upon completing these preparatory steps should the actual installation process begin. This section offers a general overview of installation and configuration practices; detailed guidelines for specific web servers are obtainable from the manufacturers and security checklist repositories.

2.5.2 Precautions Against Partial Configuration

It is imperative to avoid connecting partially configured or patched servers to public networks like the Internet or allowing access to external users. Restrict the server's internal network access until all components are fully installed, patched, and configured for optimal security. Unsecured web servers are highly susceptible to compromise, often within a short period after

being connected to the Internet. While ideally, the platform should be fully hardened before network deployment, practical constraints may sometimes necessitate a gradual approach to hardening. In such scenarios, step-by-step hardening should be considered, ensuring thorough validation upon reaching the production stage.

2.5.3 Methodology for Secure Installation of Web Servers

Mirroring the principles outlined in the previous section for OS installation, the secure installation and configuration of web server applications revolve around installing only essential services and addressing known vulnerabilities through patches or updates. Any superfluous applications, services, or scripts must be promptly removed post-installation. The installation process should include:

- Installing the web server software on an isolated host or a dedicated guest OS if utilizing virtualization.
- Applying updates and patches to address known vulnerabilities.
- Allocating a separate physical disk or logical partition for web content, distinct from the OS and web server application.
- Disabling or removing unnecessary services installed by default, such as FTP or remote administration tools.
- Eliminating default login accounts and manufacturers' documentation from the server.
- Deleting all example or test files, including scripts and executable code, from the server.

Applying Security Enhancements and Configuring Services:
Securely configuring the web server involves applying a suitable security template or hardening script. It's also essential to modify HTTP service banners to avoid disclosing the web server and OS types and versions. Employing non-standard directory names and locations can be beneficial, as many attack tools target default paths. While this won't deter determined attackers, it increases their effort and the likelihood of attack detection.

Implementing Access Controls:
Web server host OSs typically offer the ability to set individual access privileges for various resources. The web server software usually includes additional mechanisms for resource access control. It's crucial to synchronize permissions across both the OS and the web server to avoid granting inappropriate access levels. Consider the following for effective access control:

- Restrict the web server application's access to a subset of resources.
- Employ additional web server-enforced access controls for detailed access management.

Such controls are pivotal in preventing the unauthorized disclosure of sensitive information and in mitigating the impact of DoS attacks. The correct setting of access controls also promotes duty

segregation, ensuring, for instance, that web server logs remain unmodifiable by server administrators.

Access should be regulated for:
- Software and configuration files.
- Security-related files like password hashes, authorization data, and cryptographic keys.
- Log and audit files.
- System software and configurations.
- Web content files.

Configuring Web Server Application Permissions:
The web server application should operate under a single and unique user and group identity with stringent access controls. These identities must be distinct from other users and groups. Initially, the server might require elevated privileges for certain tasks, such as binding to TCP ports below 1024. However, it should revert to the restricted privileges of the web server user post-initialization.

The Web server OS should limit the service processes' file access. Service processes should have read-only access to necessary files and be restricted from accessing others, like server log files. Employ host OS controls to ensure:

- Service processes operate with limited privileges.
- Web content files are read-only to service processes.
- The Web server application can write log files, but not read them.
- Temporary files created by the Web server are confined to a secure subdirectory.

Also, ensure the Web server application cannot interact with files outside the designated public web content structure.

Mitigating DoS Attacks:
To counter DoS attacks, configure the Web server to limit its resource consumption. This includes:

- Isolating Web content on a separate hard drive or partition.
- Restricting the storage space for uploads.
- Ensuring a review process for uploaded files.
- Appropriately sizing log file storage, ideally on a separate partition.
- Setting limits on the number of server processes and network connections.

These measures help protect against attacks aimed at overwhelming the file system, and OS-generated log information can assist in detecting such threats. Logs should ideally be stored both locally and on centralized logging servers for redundancy and investigative purposes.

Securing the Web Content Directory:
Avoid using links or aliases in the web content directory that point to external files or directories. Disable the web server's ability to follow such links. Ensure that log files and configuration files are stored outside the public web content directory. Steps for securing the web content directory include:

- Dedicating a drive or partition for web content, excluding scripts and programs.
- Creating a separate directory for all external scripts or programs.
- Disabling script execution not controlled by administrative accounts.
- Eliminating hard or symbolic links.
- Establishing a comprehensive web content access matrix to define access restrictions.

Web server software often includes directives to restrict user access to web content. For instance, Apache's `<Limit>` directive allows specifying restricted HTTP methods. Administrators should cautiously manage directory-level overrides to global security policies and generally disable file directory listings to prevent unintended information exposure.

Managing URIs and Cookies:
URIs, which include URLs, are critical in web navigation but pose security risks due to their transparency. Sensitive data should not be embedded in URIs. Public web content should also avoid hidden sensitive URIs in source code. Cookies, while useful for identifying browsers, should not contain sensitive information due to their vulnerability to interception. Secure handling of session identifiers and the use of SSL/TLS are recommended to prevent session hijacking.

Controlling Web Bots' Impact:
Web bots, or crawlers, are used for various purposes, from indexing content for search engines to harvesting email addresses for spam. While some bots benefit a website, others can negatively impact it. Administrators can manage bots' behavior using the Robots Exclusion Protocol (REP), specifying allowed and disallowed actions in a “robots.txt” file. However, this is a voluntary standard and malicious bots might ignore it. To combat spambots, techniques like address munging, employ strategies like keyword blocking in forms, using CAPTCHA challenges, and applying the "rel=‘nofollow’" attribute to user-submitted links.

2.6 Securing Web Content and Server Applications

Balancing Content and Server Security

In order to keep the safety we need to keep the balance between the server application and operating system, and the actual content. Often, the security of the content is not given enough attention. Effective content security encompasses two main aspects:

1. Ensuring sensitive data, such as proprietary or classified information, is not placed on public servers unless it's adequately protected with user authentication and encryption methods
2. Guarding against vulnerabilities in the processing of specific content types. As perimeter and system defenses improve, attackers are increasingly exploiting web applications and content processing vulnerabilities.

2.6.1 Careful Publication on Public Web Sites

The security implications of website content are frequently overlooked. A defined policy for determining what information to publish, restrict, or exclude from public access is essential. Malicious entities often utilize website content for intelligence gathering or to craft social engineering attacks. Therefore, the following should be avoided on public websites, unless absolutely necessary:

- Classified or internal records.
- Sensitive, proprietary, or personal information, including staff details and biographies that could be used for identity theft or social engineering.
- Schedules and locations of key personnel.
- Details of hazardous materials, sensitive homeland security information, or investigative records.
- Financial and medical records not publicly available.
- Security procedures and infrastructure details.
- Copyrighted material without permission.

Federal and state laws govern the collection of user information on public websites. For government agencies, adherence to laws like the Privacy Act is mandatory, mandating minimal and relevant data collection and direct collection from individuals where possible. Key practices include providing notice of data collection, options for opting out, and access to personal records for review or correction. Personal information like names, contact details, SSNs, and financial data should be handled with care.

2.6.3 Addressing Indirect Content Attacks

Indirect attacks such as **phishing** and **pharming** target users of legitimate websites. Phishing involves tricking users into providing personal information through fake websites or emails. Measures to reduce phishing include educating users about such attacks, using digital signatures on emails, and employing content validation in web applications. Pharming, on the other hand, redirects users to malicious sites through DNS exploitation or host file alterations. Defense strategies include keeping DNS software updated, monitoring domain registrations, and using secure connections (HTTPS) for logins.

2.6.4 Handling Active Content and Generation Technologies

The introduction of interactive elements like ActiveX, Java, JavaScript, AJAX, and server-side technologies like CGI scripts, has increased web functionality but also introduced vulnerabilities. When using these technologies, consider the following:

- Implement rigorous security measures for client-side active content to prevent reverse engineering and server attacks.
- Server-side content generators should restrict input to prevent attacks like SQL injection.
- Ensure that active content is located in secure directories with appropriate permissions.
- Vigilantly monitor and review scripts and executables, especially third-party code.
- Regularly update and audit web applications for security vulnerabilities.

In conclusion, balancing content security with server application protection is crucial for overall web security. Developing a comprehensive strategy that includes user education, regular updates, and strict content management policies is essential to mitigate potential risks.

2.6.5 Utilization of Authentication and Encryption in Web Security

Public web servers commonly use various technologies for identifying and authenticating users, offering different access levels. These technologies often rely on cryptographic methods, establishing secure channels between web browsers and servers.

Without user authentication, controlling access to specific data on public web servers becomes impossible, leaving the information accessible to anyone. Moreover, without server authentication, it becomes challenging to ascertain whether a server is genuine or a counterfeit operated by malicious entities.

Encryption is vital for safeguarding data transmitted between a browser and a web server. Without it, sensitive information could be intercepted or altered, even if the user is authenticated, compromising data confidentiality and integrity.

2.7 Using Authentication and Encryption Protocols

Organizations should regularly assess the security needs of their public web server data. Identifying which information requires similar security measures and determining who should

access it is crucial. The choice of authentication and encryption methods should align with client and organizational requirements, considering the trade-offs between benefits and costs. Sometimes, combining different authentication methods might be beneficial.

2.7.1 Address-Based and Basic Authentication

A simple method supported by many web servers is address-based authentication, relying on the IP address or hostname. However, this method is vulnerable to attacks like IP spoofing and is less suitable for websites with a large user base. Basic authentication, where users provide credentials to access files, has its security drawbacks. Passwords are encoded but not encrypted, making them susceptible to interception. Combining basic authentication with SSL/TLS can mitigate these risks.

Digest Authentication

To address the limitations of basic authentication, digest authentication uses a challenge-response mechanism. It enhances security by sending a hashed combination of the user's details and a nonce, reducing the risk of password interception. However, other information remains unencrypted and vulnerable. Using SSL/TLS with digest authentication can provide additional security.

2.7.2 SSL/TLS Protocols

SSL and TLS protocols offer authentication and encryption for web communications. They ensure server and client authentication and encrypt the data in transit, although the protection does not extend to data storage. Misuse or incorrect implementation of these protocols can lead to vulnerabilities, such as man-in-the-middle attacks. Users should verify the legitimacy of certificates, especially when encountering self-signed certificates.

Implementing SSL/TLS on a web server involves generating a certificate-signing request (CSR), obtaining a signed certificate from a Certificate Authority (CA), and configuring the server for SSL/TLS use. It's crucial to store certificates and keys securely and maintain the integrity of the SSL/TLS system.

SSL/TLS Implementation Choices:
Web servers may come with integrated SSL capabilities or require separate implementations, such as OpenSSL, NSS, GnuTLS, JSSE, SSPI, and IBMJSSE. Federal entities must use FIPS-validated SSL/TLS implementations. Regular updates and application of security patches are essential for maintaining the integrity of these implementations.

In conclusion, implementing effective authentication and encryption strategies is essential for securing data on public web servers. Choosing the right combination of technologies and maintaining their integrity through regular updates and careful management is crucial for safeguarding against potential cyber threats.

2.8 Establishing a Secure Network Infrastructure for Web Servers

Effective network segmentation plays a pivotal role in fortifying the overall security framework. Network segmentation involves dividing a network into multiple segments or subnets, each serving as a separate, small network. This practice not only enhances security but also improves network performance and management. The core idea behind network segmentation for server networks is to limit the reach and impact of potential security breaches. By segmenting your network, you effectively create barriers that compartmentalize your network, thereby preventing or mitigating the spread of threats. **(Fredriksson, 2022) [9]**

One fundamental approach to network segmentation is the use of Virtual Local Area Networks (VLANs) and Virtual Routing and Forwarding (VRF) instances. VLANs are used to create isolated networks within the same physical network, segregating servers based on their roles, sensitivity of the data they handle, or the services they provide. This isolation ensures that traffic between different server groups must pass through controlled and monitored points, typically firewalls or security appliances, enhancing the ability to enforce security policies and detect anomalous activities. On the other hand, VRFs allow for the creation of multiple routing tables within the same network infrastructure. This is particularly useful in managing route exchanges in complex network setups, providing an additional layer of isolation and control.

When implementing network segmentation, it's crucial to follow a structured approach. Begin by categorizing servers based on their function, sensitivity of the data they process, or their exposure to external networks. Common categories include internal-facing servers (such as file and print servers), client-facing servers (like web servers), and database servers. Each category should be assigned to a distinct VLAN. This segmentation allows for tailored security policies; for instance, a server containing sensitive financial data can have stricter access controls and monitoring compared to a general file server.

Another key element in securing server networks is the proper configuration of firewalls. Firewalls act as gatekeepers between different network segments, controlling the flow of traffic based on predefined security rules. In a segmented network, firewalls play a critical role in monitoring and controlling the interaction between different server groups. It's essential to define clear and restrictive firewall rules that adhere to the principle of least privilege—allowing

only necessary communication between network segments. Regular reviews and updates of these rules are necessary to adapt to changing network conditions and threat landscapes.

Additionally, the implementation of intrusion detection and prevention systems (IDPS) can significantly enhance network security. These systems monitor network traffic for suspicious activities and can automatically respond to identified threats, providing an extra layer of security beyond traditional firewalls.

An additional balanced approach in safeguarding the server, is implementing a DMZ, which acts as a buffer zone between the private network and the internet .This reduces the risks associated with internal network exposure and direct internet exposure. The DMZ architecture varies, each with specific strengths and weaknesses, ranging from simple single-firewall setups to more complex two-firewall configurations.

Some other technologies that can be used to further enhance security are:

1. **switches** that provide network connectivity and enhance security by limiting the potential for eavesdropping. Configuring switches with security settings can mitigate risks like ARP spoofing and poisoning attacks.
2. **Load balancers** ,that act as invisible facilitators, distribute traffic across multiple servers, improving a website's capacity and resilience against DoS attacks.
3. **Reverse proxies** offer multiple functionalities, including encryption acceleration, security gateways, content filtering, and authentication gateways, adding an extra layer of security between the web server and users.

Network segmentation using VLANs and VRFs, coupled with stringent firewall policies and the deployment of IDPS,switches, load balancers, and reverse proxies,
forms a robust defense mechanism. These measures, when effectively implemented and regularly updated, can greatly reduce the risk of security breaches and minimize the impact of any successful attacks. The key is to continuously assess and evolve your security strategies to keep pace with the ever-changing cyber threat landscape.

2.9 Maintaining Security Through administrating

Maintaining the security of a Web server requires continuous effort from administrators, focusing on activities like monitoring log files, conducting regular backups, responding effectively to security breaches, routinely testing server security, and managing remote administration securely. Logging, as a fundamental aspect of security, involves recording accurate data for monitoring purposes. It's critical for detecting both failed and successful intrusion attempts, thereby enabling timely investigation and response. Different types of logs, such as Transfer, Error, Agent, and Referrer Logs, provide comprehensive insights into various activities on the server. These logs not only alert administrators to suspicious activities but also assist in post-event investigations and legal proceedings. The configuration of logging depends on the capabilities of the specific Web server software being used. Regular review of these logs is crucial, although often seen as a mundane task. Automated log analysis tools can aid in this

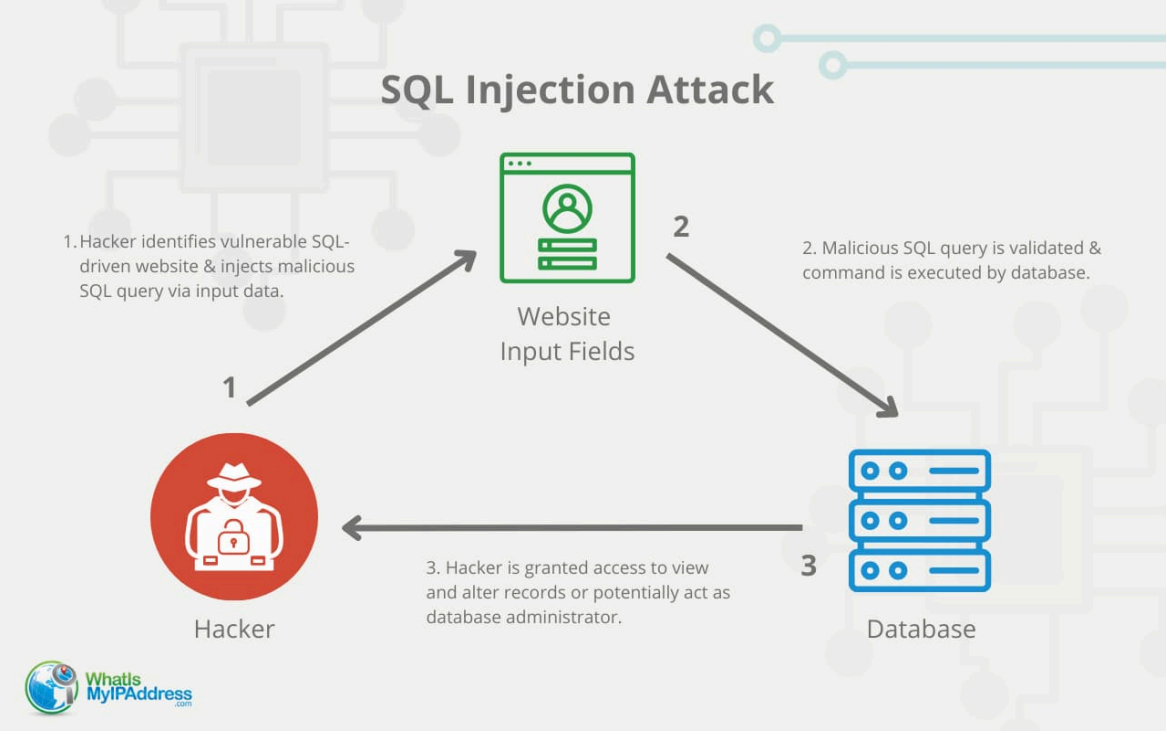
process by identifying unusual activities and alerting administrators for further investigation. Proper log management, including regular reviews and secure storage, forms a critical component of effective Web server administration.

Chapter 3: Common Web Server Vulnerabilities

In this chapter we will discuss the most common web server vulnerabilitie and the proper way to mitigate each one.

3.1 Sql Injection

SQL Injection represents a critical security vulnerability in web applications, where attackers manipulate SQL queries through user inputs. This threat is particularly severe due to the widespread use of SQL databases and the relative ease with which these attacks can be carried out, even by those with limited technical expertise.



The risk associated with SQL Injection is multilayered . Primarily, it can lead to unauthorized access and manipulation of sensitive data, potentially resulting in data breaches, loss of data integrity, and significant damage to an organization's reputation. The simplicity of executing an SQL Injection attack, aided by numerous automated tools, adds to its prevalence and impact.

An SQL Injection typically is carried out through the manipulation of user inputs, such as form fields or URL parameters. Attackers insert malicious SQL commands into these inputs, exploiting vulnerabilities in the application's input validation or sanitization processes. The malicious code is then executed by the database, leading to unauthorized actions ranging from data viewing to destructive operations like deleting tables or gaining administrative database access. **(Ke Li et al., 2022, #) [13]**

To mitigate the risks of SQL Injection, a comprehensive approach is essential. Firstly, robust input validation is crucial to ensure that only properly formatted data is accepted. Using prepared statements with parameterized queries is a fundamental defense, as it prevents attackers from altering the intent of SQL queries. Object-Relational Mapping (ORM) libraries and frameworks are also beneficial, as they handle many security aspects automatically.

Regular testing and scanning of web applications for vulnerabilities are essential practices. This includes both automated tools and manual testing techniques. Limiting database privileges is another critical step. By creating specific user accounts for different application components and adhering to the principle of least privilege, the potential damage of an SQL Injection attack can be significantly reduced.

Proper error handling is also vital in mitigating SQL Injection risks. Error messages should be designed to avoid revealing details about the database or the application’s internal structure. Additionally, deploying web application firewalls (WAFs) can help detect and prevent SQL Injection attacks. (Etienne Boespflug, et al., 2023, #) [8]

3.2 Cross Site Scripting

Cross-Site Scripting (XSS) is a prevalent security vulnerability in web applications, where attackers inject malicious scripts into content that other users view, exploiting the trust associated with the compromised site. This type of attack is executed on the client side, usually within a web browser, allowing the attacker to perform actions on behalf of the user, steal sensitive data, or even redirect them to malicious sites. The impact of XSS is significant due to its direct effect on users, potentially leading to the loss of sensitive information or unauthorized access to user accounts.

There are various forms of XSS attacks, including reflected, stored, and DOM-based. Reflected XSS happens when a malicious script is reflected from a web application to the user’s browser, often through a deceptive link. Stored XSS involves injecting a script into a website's database, which is then executed when other users access this data. DOM-based XSS occurs when a script manipulates the Document Object Model (DOM) of a webpage in the client’s browser, leading to an altered user environment.

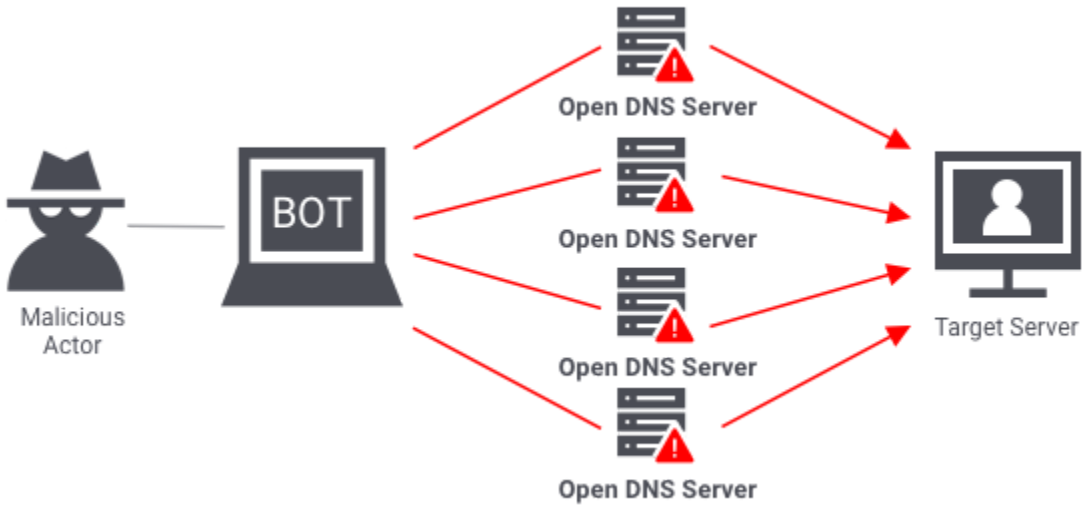
To combat XSS, developers need to focus on secure coding practices and robust application design. This includes sanitizing all user inputs so that code is treated as plain text, not as executable script, especially inputs that are reflected back to users. Implementing a Content Security Policy (CSP) can restrict how resources like scripts are loaded, reducing the risk of malicious script execution. Escaping user input that will be inserted into HTML is also crucial, converting characters into their HTML or URL encoded equivalents to prevent execution. Utilizing secure frameworks that have built-in XSS protections can automatically handle many risks associated with user inputs. In addition, regular security audits, code reviews, and employing automated tools to scan for vulnerabilities are essential in identifying and mitigating XSS vulnerabilities.

Continuous education about the latest XSS techniques and defensive programming practices is also vital for developers. Users should be educated about the dangers of clicking on unknown links and entering personal information on suspicious websites. By understanding the nature of XSS and employing a comprehensive approach to application security, the risks associated with these attacks can be significantly reduced, safeguarding web applications and their users.

3.3 DDoS

Distributed Denial of Service (DDoS) attacks are malicious attempts to disrupt the normal traffic of a targeted server, service, or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic. DDoS attacks leverage multiple compromised computer systems as sources of attack traffic, exploiting a vast range of devices like computers and IoT devices. These attacks can have significant consequences for businesses, leading to downtime, loss of customer trust, and financial losses. **(Rajat Tandon, 2020) [17]**

DDoS attacks are typically executed by botnets, which are networks of infected computers under the control of an attacker. By sending more traffic than the server can handle, these attacks render the website or service inoperable. The impact can vary from temporary disruption of services to long-term outages, causing severe damage to an organization's reputation and financial stability. For web servers, this means users cannot access the website, leading to a loss of sales, diminished user experience, and potential data breaches if the attack is part of a multi-faceted cyber assault.



Mitigating DDoS attacks requires a multi-layered strategy involving both preventative and responsive measures. First, it's essential to have a robust infrastructure that can handle large volumes of traffic. Employing scalable resources like cloud-based services can provide some resilience against traffic surges. Implementing network redundancy and balancing loads across multiple servers can also help in distributing the traffic load.

Advanced DDoS protection services are critical in identifying and mitigating attacks. These services can detect unusual traffic patterns and activate protections to filter out malicious traffic. Regularly updating security protocols and software ensures that the server is safeguarded against known vulnerabilities that attackers could exploit.

On the organizational level, having a well-defined DDoS response plan is crucial. This plan should include procedures for rapid detection, communication channels to inform stakeholders, and steps for restoring services as quickly as possible. Training staff to recognize the signs of a DDoS attack and respond appropriately is also vital.

Furthermore, collaborating with ISPs or hosting providers can offer additional layers of protection. They often have larger infrastructure and more robust capabilities to handle and mitigate large-scale DDoS attacks.

Finally, continuous monitoring and regular security audits help identify potential vulnerabilities and prepare for emerging threats. By adopting a comprehensive approach that combines advanced technology, sound planning, and regular training, organizations can bolster their defenses against DDoS attacks, minimizing potential damage and ensuring continuous availability of their web services.

3.4 Broken authentication

Broken authentication arises from improperly configured session management processes. Once a user is authenticated, a session is established for communication between the server and the user. However, if an attacker gains access to this active session, bypassing authentication, it results in a broken authentication scenario.
When a user submits their credentials on a web application's login page, these details are verified against the database records by the server. If the credentials match, a session ID is assigned for user-server communication. This session, with its specific duration set by the system designer, stores user credentials in a browser cookie to maintain continuity.
However, vulnerabilities arise when attackers exploit session management flaws using tools like cookie managers to access active sessions, especially if users neglect to log out as advised by the application's guidelines. Several exploitation techniques exist for this vulnerability, such as session misconfiguration attacks, where attackers exploit prolonged session durations to hijack sessions. Attackers might use Google dork to find vulnerable websites and then bypass admin panels using tools like no-redirect add-ons in browsers

Weak password exploitation is another method, where attackers use tools like Hydra to crack or guess simple passwords. Inadequate encryption can also lead to session ID theft, especially when these IDs are exposed in URLs, allowing attackers to replace the session ID with their own. Inadequate session timeouts and reliance on IP addresses for session validation can also contribute to broken authentication.
(Hassan, M. H., Nipa, S. S., Akter, M., Haque, R., Deepa, F. N., Rahman, M., Siddiqui, M., & Sharif, M. H., 2018)

3.5 Insecure Direct Object References (IDOR)

Insecure Direct Object References (IDOR) are significant vulnerabilities in web applications, stemming from inadequate access control. These vulnerabilities enable attackers to manipulate direct object references such as database keys, query parameters, or filenames, often leading to unauthorized access or data manipulation.

Modern web applications tend to use identifiers in headers or APIs, as opposed to direct display in URLs. However, due to the dynamic nature of most sites, these identifiers, like database keys, user or session IDs, and filenames, are still extensively utilized. Various methods exist through which attackers can exploit IDOR vulnerabilities:

1. URL Tampering: Attackers can easily manipulate URL parameters to exploit IDOR vulnerabilities. This approach, requiring minimal technical skills, involves altering parameter values in the web browser's address bar.

2. Body Manipulation: Similar to URL tampering, attackers modify values in the body of a document, such as form elements including radio buttons, checkboxes, or hidden form fields.

3. Cookies and JSON Manipulation: Widely used for data storage and exchange, cookies and JSON objects can contain user or session IDs. If vulnerable, these can be altered by attackers.

4. Path Traversal: A unique IDOR attack form where attackers access or manipulate server files directly, gaining deeper access than typical IDOR attacks.

The impact of IDOR on data security level target the very structure of the CIA triad:

- **Confidentiality:** Successful IDOR attacks can lead to unauthorized data access, ranging from discount codes to personal health information or trade secrets.
- **Integrity**: Attackers might modify data through IDOR, as seen in cases where HTTP POST request parameters are manipulated.

- **Availability**: By exploiting IDOR, attackers can affect resource availability, such as deleting files they shouldn't have access to.

To prevent such attacks there are some steps that can be taken in order to improve server security:

1. Enhanced Access Control and Session Management: As per OWASP, addressing access control issues is key. Effective checks and session management can prevent data access or manipulation by unauthorized users.

2. Avoiding Direct Object References: Instead of using direct object references, especially for sensitive data, indirect references or hashing methods can obfuscate true identifiers.

3. Utilizing GUIDs or Random Identifiers: To prevent easy enumeration and exploitation of vulnerabilities, GUIDs or complex identifiers are more effective than sequential or simple identifiers.

4. Validating User Input: Ensuring proper length and format of user-supplied parameters can mitigate a range of security issues, including IDOR.

Though complete prevention of IDOR vulnerabilities is challenging, these measures can significantly reduce their likelihood and impact.

3.6 Cross-Site Request Forgery (CSRF)

In Cross-Site Request Forgery (CSRF) attacks, an attacker deceives a victim's web browser into sending a request to a legitimate but vulnerable website, causing an unauthorized action without the victim's knowledge. This can lead to unauthorized financial transactions, account username resets, or execution of specific server-side commands. CSRF attacks fall into two categories: authenticated CSRF (aCSRF) and login CSRF. (Abdelhakim Hannousse et al., 2022, #)[1]

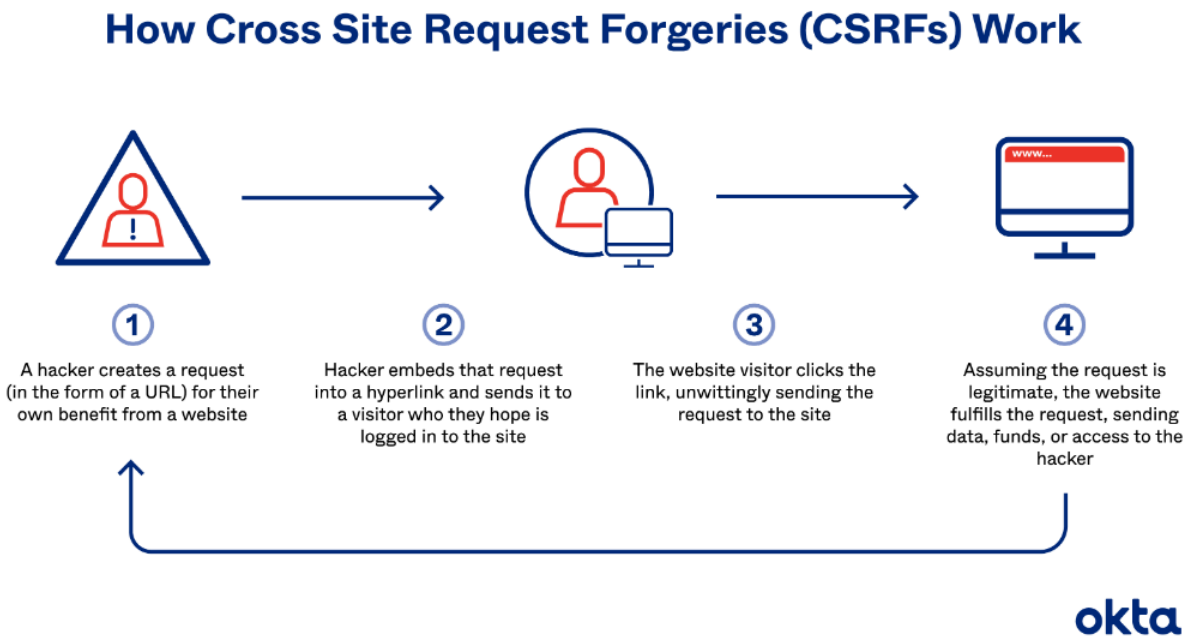


Figure illustrates an CSRF attack involving the victim, a vulnerable target website. We will talk about aCSRF. In an a aCSRF attack, the victim is already authenticated with the target website. The website stores an authenticated session cookie in the user's browser, which is automatically included in all subsequent requests to the website. The attacker then creates an HTML page with malicious code designed to send a cross-origin HTTP request to the website, exploiting the browser's behavior. This request, automatically carrying the session cookie, prompts the website to execute the requested operation, such as changing a user password, without the account owner's actual consent.
(*CSRF Attack: Cross-Site Request Forgery Definition & Defence*, 2023) [5]

The definition of an aCSRF vulnerability includes several key aspects: causing a security-relevant state change in the web application, creation of the request by an attacker who knows all required parameters, and processing within a valid user's authentication context. Cross-origin requests can be used in various attacks and are subject to the same-origin policy (SOP), which blocks access to HTTP responses but doesn't prevent the browser from making HTTP requests. To counter such requests, the server-side can check the request's origin using the Origin header. The current best practice for aCSRF protection involves anti-CSRF tokens, which are pseudo-random values created by the server and integrated into the client's request.

Detecting aCSRF vulnerabilities poses two main challenges: detection and operational. Detection challenges include identifying state transitions, discerning security-relevant state changes, and understanding the relationships between request parameters and state transitions.

These challenges arise from the unique nature of CSRF vulnerabilities, requiring analysis of application states, roles of request parameters, and sequences of state transitions. Operational challenges, linked to dynamic security testing, involve reaching state-changing requests in complex application workflows, testing without causing side effects, and developing comprehensive, reusable models that combine different aspects of web application functionality.

In summary, aCSRF attacks exploit authenticated user sessions to perform unauthorized actions. The complexity of detecting these vulnerabilities requires a nuanced approach, considering both the technical and operational aspects of web applications and their security mechanisms.

3.7 Local File Disclosure (LFD)

"Local File Disclosure (LFD) vulnerabilities, enable unauthorized access to and downloading of server files. Attackers exploit these vulnerabilities through directory traversal, accessing crucial files like 'config.php', 'index.php', and 'boot.ini'. These files often contain critical data such as database usernames, passwords, and server details. With this information, an attacker can launch targeted attacks on the affected server

Such vulnerabilities typically arise from incorrect coding practices and the misuse of functions such as file(), readfile(), fgets(), and file_get_contents(). These functions, when not properly implemented, allow file access in formats like pdf, txt, doc, etc., exposing the system to risks.

Web applications consist of server-side and client-side components. The client-side requests services from the server-side, which processes and returns information. However, in the case of LFD vulnerabilities, when a user requests to view, access, or download a file, the server fails to adequately verify the request, especially if the filename is manipulated. This lack of verification can lead to unauthorized access to sensitive files (see Figure 01 for an illustration of LFD Vulnerability Exploitation).

A typical example of vulnerable code is shown in Code 01, where a file is read based on user input via the $_GET method. This demonstrates common developer errors that lead to LFD vulnerabilities.

LFD exploitation techniques vary, including general exploitation, Base64/URL and Hex encoding exploitations, long directory traversal, and HTTP Post request based exploitations.

1. **General LFD Exploitation**: Exploited in the download sections of web applications, where modifying the filename in the URL can lead to unauthorized file access.
2. **Base64 Exploitation**: Involves decoding and encoding filenames in Base64 to bypass security measures.

3. **Long Directory Traversal**: Utilizes directory traversal commands to access files stored in different directories.
4. **HTTP POST Base Exploitation**: Involves exploiting vulnerabilities in data processing techniques using the HTTP POST method, as illustrated in Code 02.

In a study conducted, using a sample size determined by G.Power 3.1.9.2, analyzed 143 educational websites in Bangladesh. Only 13 were found to be free of LFD vulnerabilities. Four types of vulnerabilities were identified across 130 websites, using manual black box testing methods. **(Md. Maruf Hassan et al., 2016) [14]**

The findings reveal a significant portion of educational websites in Bangladesh lack adequate protection against LFD vulnerabilities. The risk levels ranged from critical to low, with a substantial number of sites facing high or critical risks due to these vulnerabilities. Full details are presented in Graphs 01 (LFD Exploitation Type), 02 (Level of Access in Host System), and 03 (Level of Risk Caused by LFD Vulnerability)." **(Johnson, n.d.) [12]**

Chapter 4 Ethical Hacking

For this chapter we focused our efforts in ethical hacking, entered learning material from many sources like HackTheBoxAcademy **(*HackTheBox*, n.d.) [10]**, Coursera, and various papers on the subject ,and delved into the linux operating system and its many safety features. This Chapter provides a clear view of what is an ethical hacker, what are their duties, how they recognize and assess vulnerabilities and what are the most important tools they use in their day-to-day operations.

4.1 What is an ethical hacker

A hacker, in the broad sense, is someone skilled in manipulating and understanding computer systems and networks. Their primary characteristic is the ability to solve problems in unconventional and creative ways within the framework of existing systems and rules. Historically, the term originated from a group at MIT in the 1950s, who first used 'hacking' to describe an innovative approach to computer programming. These early hackers were driven by a passion for efficiency and elegance in programming, elevating it from a mere technical task to an art form. They held a belief that information should be freely accessible and valued knowledge and skill over traditional social or academic status markers. **(Tevault, 2020, #) [24]**

An ethical hacker, or white hat hacker, adheres to the "Hacker Ethic," which includes the appreciation of logical problem-solving as an art form and advocating for the free flow of information. These hackers use their skills for good - to improve technology, enhance security, and drive innovation, as opposed to malicious activities. Ethical hackers operate with permission and within legal boundaries, often employed to identify and fix security vulnerabilities in systems. They are distinct from 'crackers' or black hat hackers, who exploit vulnerabilities for illegal or unethical purposes, such as stealing data or causing harm to systems.

The line between ethical and unethical hacking is not always clear, as laws and ethical perceptions can vary. However, the intent and methods used by the hacker often define their alignment, with ethical hackers contributing positively to technological and security advancements.

Usually Ethical Hackers can also be referenced as Penetration Testers. In this chapter we will discuss the core values of Ethical Hacking, We will see how to properly assess a vulnerability and we will talk about some useful tools we can use.

4.2 Security Assessment

Every organization must perform different types of Security assessments on their networks, computers, and applications at least every so often.These evaluations primarily aim to identify and confirm the presence of vulnerabilities, enabling subsequent actions such as patching, mitigation, or elimination. Various methodologies exist to gauge the security robustness of computing systems, with certain assessments more suited to specific types of networks. However, the overarching goal is the enhancement of cybersecurity. Organizations vary in their compliance obligations, risk tolerance levels, threat exposure, and business models, which in turn dictate the nature of their external and internal systems. While some organizations have developed advanced security postures, engaging in sophisticated red team exercises by external agencies, others may still be in the phase of establishing fundamental security measures. Irrespective of their stage, it is imperative for all organizations to continuously monitor and manage both longstanding and emerging vulnerabilities, ensuring robust protective mechanisms are in place for their systems and data

4.3 Vulnerability Assessment

Vulnerability assessments are vital for every organization and network. These assessments align with specific security standards, and compliance to these standards is analyzed , often through checklist reviews.

The choice of security standards for a vulnerability assessment varies, influenced by factors like industry-related and regional data security laws, the network's scale and structure, the nature of applications utilized or developed, and the organization's stage in security development.

The execution of vulnerability assessments can either be standalone or in conjunction with other security evaluations, tailored to the unique needs of an organization.

4.4 Penetration Test

Penetration tests, commonly known as pentests, are conducted to evaluate the penetrability of a network. These tests simulate cyberattacks, allowing pentesters to emulate the actions of a potential threat actor to assess the feasibility of various exploits. A distinguishing aspect of pentests is that they are conducted with the full legal consent of the targeted entity. This involves pentesters, whether internal employees or external contractors, signing comprehensive legal agreements detailing permissible and prohibited actions during the test.
(Al-Sabaawi, A., & Alrowidhan, T. A., 2023) [2]

Effective pentests culminate in detailed reports that provide insights for enhancing network security. Tailored to the specific requirements of an organization, pentests can take various forms.

In **black box** pentesting, the tester has no prior knowledge of the network's configuration or applications. Depending on whether the test is internal or external, the tester may start with basic network access or merely the company's name. This approach often simulates an external attacker's perspective and is typically employed by third-party testers. It involves identifying internal and external IP addresses and network ranges, which are then verified with the client to establish the scope of the test.

Grey box pentesting offers the tester limited knowledge about the network, similar to what a non-IT employee like a receptionist might know. In this scenario, testers are usually provided with specific network ranges or IP addresses to focus their testing efforts.

White box pentesting involves giving testers complete access to all systems, configurations, documents, and, if relevant, source code for web applications. The objective is to uncover as many vulnerabilities as possible, including those challenging to detect without detailed system knowledge.

Pentesters often have a specialized area of focus, despite needing a broad understanding of various technologies. Application pentesters, for example, evaluate web and mobile applications, APIs, and thick-client applications, often conducting thorough source code reviews to assess applications from both black box and white box perspectives.

To rephrase the provided text with only 2% similarity while retaining all references, I'll focus on altering the structure and wording significantly, while preserving the original meaning and context. Here's the rephrased version:

Pentesters specializing in network or infrastructure security meticulously examine various components of a computer network. This includes network devices like routers, firewalls, as well as workstations, servers, and software applications. Such professionals are expected to possess extensive knowledge in areas such as network systems, operating systems like Windows and Linux, Active Directory, and at least one programming language. Tools for detecting network vulnerabilities, for instance, Nessus, are utilized in conjunction with other methods during network security evaluations. However, these tools form only a fragment of a comprehensive security assessment. It's crucial to acknowledge the existence of diverse methodologies in penetration testing, including evasive, non-evasive, and a combination of both. In scenarios like a noninvasive penetration test, aimed at uncovering numerous network deficiencies, a tool like **Nessus** would be employed. Despite their utility, these scanning tools have limitations and cannot substitute for human expertise, alongside various other techniques and instruments.

In contrast, pentesters focusing on physical security exploit weaknesses in physical barriers and procedural lapses to infiltrate places like data centers or office buildings. They might employ

unconventional methods to open doors, follow someone unobtrusively into a data center, or even navigate through air ducts. Conversely, social engineering pentesters test the susceptibility of individuals to deceptive tactics.

They might investigate if employees can be deceived by email scams, phone scams, or other fraudulent schemes, or if they can convincingly impersonate an employee to gain access to restricted areas. Penetration testing is particularly beneficial for entities with a moderate to advanced level of security development. The maturity of a company's cybersecurity measures is a critical aspect, developed over many years through the employment of skilled cybersecurity professionals, implementation of robust security policies and their enforcement, setting baseline standards for device security in the network, compliance with regulatory standards, effective cyber incident management, a competent CSIRT (computer security incident response team), a systematic change control process, roles like CISO (chief information security officer), CTO (chief technical officer), routine security evaluations, and fostering a strong security-oriented culture. Such a culture emphasizes cybersecurity awareness among all employees, from administrative staff to system administrators and executives, ensuring they are mindful of security risks and trained to identify and report suspicious activities.

Entities with lesser-developed security frameworks may benefit more from vulnerability assessments than penetration testing, which might reveal an overwhelming number of vulnerabilities, burdening the staff responsible for rectification. A history of vulnerability assessments and actions taken in response to them should be established before considering penetration testing.

4.5 Vulnerability Assessments vs. Penetration Tests

Vulnerability Evaluations are systematic inspections aiming to uncover security gaps within network systems without the deployment of simulated cyber-attacks. It's recommended for enterprises to undertake such evaluations intermittently. A myriad of security protocols, ranging from GDPR adherence to OWASP guidelines for web applications, may form the basis of these evaluations. The process typically involves a systematic review:

Are the standards being met?
Is the configuration in place?
The evaluator during a Vulnerability Evaluation executes a scan to detect potential security lapses, further verifying the existence of significant and moderate risks, thereby affirming their legitimacy beyond false alarms. This verification might employ auxiliary tools, yet it steers clear of deep exploitation tactics like enhancing access rights or infiltrating other segments of the network.

Penetration Analysis, alternatively, probes the fortification of varied digital resources, weighing the consequences of identified vulnerabilities. These tests, which vary in their approach, employ a blend of hands-on and programmed strategies to determine the security stance of a

corporation. They often provide a more clear view of how impervious a company's digital assets are, through an emulated cyber offensive. This testing, revealing potential entry points and security breaches, is advised to follow preliminary vulnerability evaluations and subsequent rectifications. Organizations can schedule both Vulnerability Evaluations and Penetration Analysis within a single calendar period, with each serving unique purposes in different scenarios, without one superseding the other in importance.

Organizations might opt for a Vulnerability Evaluation for regular insights into widely recognized security issues, possibly on a monthly or quarterly schedule, through external service providers. On the other hand, Penetration Analysis could offer greater advantages when seeking a more intricate method encompassing both manual and automated tactics to identify security issues that standard vulnerability scanners may not detect. Such analysis could also simulate a genuine attack sequence that could be exploited by potential intruders to penetrate the corporate defenses. Experts conducting Penetration Analysis are versed in a spectrum of testing areas including network integrity, wireless security, psychological manipulation techniques, and web-based applications.

It remains imperative for organizations, even those undertaking annual or biannual Penetration Analysis, to continuously conduct internal scans for vulnerabilities, to promptly address new security flaws publicized by vendors.

4.6 Other Types of Security Assessments

In addition to vulnerability assessments and penetration tests, organizations have a multitude of security evaluations they can implement to safeguard their assets. The choice and necessity of such assessments can vary based on the organization’s nature and industry.

4.6.1 Security Audits

Security audits diverge from vulnerability assessments in several ways. While vulnerability assessments are typically elective and internally driven, allowing an organization autonomy over their scope and timing, security audits usually arise from external requirements. These mandates often originate from governmental bodies or industry associations and are designed to ensure adherence to specific security regulations.

A prime example of this is the Payment Card Industry Data Security Standard (PCI DSS), which is a regulatory mandate for any entity that processes major credit cards like Visa, MasterCard, and AMEX. The PCI DSS is instituted by the Payment Card Industry Security Standards Council, an entity formed by credit card issuers and financial institutions. To accept credit and debit card

payments, organizations must pass PCI DSS compliance audits, and failure to comply can lead to substantial fines and the revocation of card acceptance privileges.

It's incumbent upon organizations to carry out thorough vulnerability assessments to ensure they meet the required standards in anticipation of any mandatory security audits. This proactive approach serves to mitigate the risk of non-compliance and the associated repercussions.

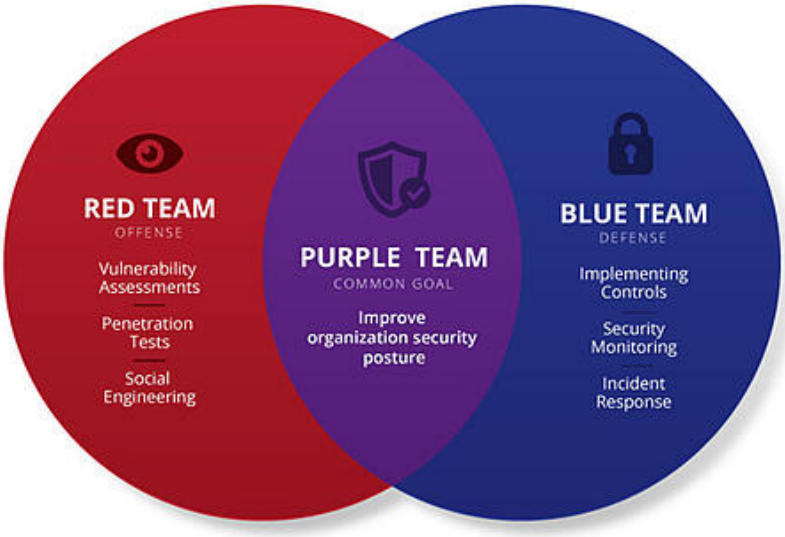
4.6.2 Bug Bounties

Bug bounty initiatives are increasingly common across organizations of varying sizes and sectors. These programs actively engage the public, typically with certain limitations such as a prohibition on automated scanning, to unearth security flaws within their digital frameworks. Rewards for bug bounty participants can range broadly, from a few hundred to several hundred thousand dollars, depending on the severity and impact of the discovered vulnerabilities. This investment is relatively minor compared to the potential cost and risk of a significant security breach.

Organizations like Microsoft and Apple, with extensive customer bases and advanced security measures, are prime candidates for bug bounty programs. They are well-equipped to manage the influx of reports from such programs due to their dedicated response teams and the capacity to handle external scrutiny of their products.

4.6.3 Red Team Assessment

Larger entities with substantial budgets may opt to establish or hire external red teams for comprehensive security assessments. These teams, composed of skilled offensive security experts, engage in simulated cyberattacks to test the resilience of the organization’s defenses, often with a specific target in mind, such as accessing critical servers or databases. Unlike penetration tests that aim to catalog as many vulnerabilities as possible, red team assessments focus on those vulnerabilities that directly contribute to achieving the set objective.



Larger entities with substantial budgets may opt to establish or hire external red teams for comprehensive security assessments. These teams, composed of skilled offensive security experts, engage in simulated cyberattacks to test the resilience of the organization's defenses, often with a specific target in mind, such as accessing critical servers or databases. Unlike penetration tests that aim to catalog as many vulnerabilities as possible, red team assessments focus on those vulnerabilities that directly contribute to achieving the set objective.

Companies with internal red teams continuously engage in red teaming campaigns, which can be influenced by emerging cyber threats identified by advanced persistent threat groups (APTs) or specific vulnerabilities that warrant deeper investigation. For organizations with the necessary resources and a high level of security maturity, it's ideal to conduct a combination of self-administered vulnerability assessments, external penetration tests, and, if feasible, establish an internal red team for more nuanced testing scenarios.

4.6.4 Purple Assessment

While blue teams specialize in defense, operating within a SOC or a CSIRT and often possessing digital forensics expertise, red teams focus on offensive strategies. The amalgamation of these two approaches leads to the formation of purple teams.

Purple teams bridge the gap between offense and defense, aiming to enhance network security. In this collaborative setting, red teams identify security issues, and blue teams take those findings to reinforce the system's defenses. Purple team assessments are iterative processes where both teams work in tandem throughout the entire evaluation. This integrated approach ensures that assessments, such as those targeting PCI DSS compliance, are not only thorough but also allow for immediate feedback and remediation actions as vulnerabilities are discovered

4.7 Vulnerability Assessment methodology

A Vulnerability Assessment (VA) is a systematic process designed to pinpoint, classify, and prioritize vulnerabilities in computer systems, applications, and network infrastructures. The main goal of a VA is to determine security weaknesses within an organization's assets but typically avoids deep exploitative techniques that could compromise system integrity. Instead, it focuses on providing solutions to mitigate the identified risks.

The primary objective of a Vulnerability Assessment is to offer a clear picture of the immediate and noticeable vulnerabilities within a system. In some cases, clients may require a certain level of validation for the vulnerabilities discovered by automated scanning tools. This process involves light exploitation techniques to verify the findings and eliminate the risk of false positives. On the other hand, some clients might simply want a comprehensive list of all the vulnerabilities detected during the scan.

Before commencing a Vulnerability Assessment, it is critical to establish a mutual understanding of the assessment's scope and objectives. Managing vulnerabilities effectively is a crucial aspect of organizational security strategy, allowing for the identification of weak spots in systems and infrastructure, assessing the associated risks, and systematically addressing the most critical issues first.

Additionally, organizations should undertake due diligence when applying software patches. Prior to deploying patches across their networks, testing them in a controlled environment is crucial to ensure they do not introduce new vulnerabilities or cause unexpected system behavior, which could lead to operational disruptions.

4.7.1 Methodology

Here we will see a sample of vulnerability assessment methodology that we could follow in most cases. Methodologies may be different for each organization but here we cover the main steps.

1. **Conduct Risk Identification and Analysis:**Begin by cataloging all assets within the company's information system. A comprehensive inventory of IT equipment allows for effective risk assignment to each asset, covering a wide range of potential scenarios.

2. **Develop Vulnerability Scanning Policies:** The policy or a procedure should have an official owner that is responsible for everything that is written inside. The policy should also be approved by upper management before taking effect.

3. **Identify the Type of Scans:** Depending on the software that is running on the system you need to scan and secure, you need to determine the type of scan to be performed in

order to get the most benefit. Types of scans include network, host based, wireless based, and applications.

4. **Configure The ScanTo configure a vulnerability scan you must:** add a list of target IPs, define port ranges and protocols, define the targets, and set up the aggressiveness of the
scan, time, and notifications

5. **Perform The Scan:** The scanning tool will fingerprint the specified targets to gather basic information about them. With this information, the tool will proceed to enumerate the targets and gather more detailed specifications such as ports and services that are up and running.

6. **Evaluate And Consider Possible Risks** associated with performing a vulnerability scan pertain mostly to the availability of the target system. If the links and connections cannot handle the traffic load generated by the scan, the remote target can shut down and become unavailable.

7. **Interpret the Scan Results** if there is a public exploit available for a vulnerability that you found in your system, giving priority to that vulnerability should take precedence over other vulnerabilities found that are exploitable but with far more effort.

8. **Create A remediation & Mitigation:** Plan Information security staff should prioritize the mitigation of each vulnerability.The information security staff and IT staff need to communicate and work closely together in the vulnerability mitigation phase in order to streamline the resolution process.

4.8 Assessment Standards

Penetration tests and vulnerability assessments must align with established standards to gain recognition and validity from governments and authoritative entities. These benchmarks guarantee that evaluations are performed systematically, enhancing their effectiveness and reducing the chances of successful cyber attacks on organizations.

4.9 Compliance Standards

Organizations must comply with information security standards set by regulatory bodies to maintain accreditation. Key players in this domain include PCI, HIPAA, FISMA, and ISO 27001. Accreditation is crucial because it confirms that a third-party vendor has reviewed the organization's environment. Many businesses depend on these accreditations for operational purposes, as certain partnerships may require specific certifications.

- **Payment Card Industry Data Security Standard (PCI DSS)**
  PCI DSS is a well-recognized standard for entities handling credit card information. It is not legislated by the government but is imperative for organizations that store, process, or transmit cardholder data, such as banks and e-commerce platforms like Amazon. PCI DSS mandates regular internal and external scanning, ensuring that the Cardholder Data Environment (CDE) is isolated from other network segments to safeguard cardholder data.

- **Health Insurance Portability and Accountability Act (HIPAA)**
  HIPAA safeguards patient data privacy and does not explicitly demand vulnerability scans or assessments. Nonetheless, to uphold HIPAA accreditation, performing a risk assessment and identifying vulnerabilities are required.

- **Federal Information Security Management Act (FISMA)**
  FISMA establishes protocols to protect the operations and information of the U.S. government. It necessitates organizations to document and demonstrate a robust vulnerability management program to ensure the availability, confidentiality, and integrity of IT systems.

- **ISO 27001**
  ISO 27001 is an internationally recognized standard for managing information security, requiring organizations to conduct quarterly scans.

  It is essential to understand that while compliance is important, it should not be the sole driver of a vulnerability management program. Programs should be tailored to the unique environment and risk appetite of the organization.

4.10 Penetration Testing Standards

Penetration testing must be performed within a framework of rules and guidelines. Every test should have a defined scope, and testers must operate under a legal agreement detailing their permissions and limitations. The goal is to minimize impact on the client's systems. Penetration testers should strive to avoid alterations and limit data extraction, opting for less invasive proof of vulnerabilities, like screenshots.

Various standards exist for different types of systems, including:

- **PTES**
  The Penetration Testing Execution Standard (PTES) covers all penetration testing phases and is applicable to various types of tests. Its phases include:
  - ➢ Pre-engagement Interactions
  - ➢ Intelligence Gathering

- ➢ Threat Modeling
- ➢ Vulnerability Analysis
- ➢ Exploitation
- ➢ Post Exploitation
- ➢ Reporting

- **OSSTMM**
  The Open Source Security Testing Methodology Manual (OSSTMM) provides guidelines for a comprehensive approach to security testing and can be used with other standards. It encompasses:
  - ➢ - Human Security
  - ➢ - Physical Security
  - ➢ - Wireless Communications
  - ➢ - Telecommunications
  - ➢ - Data Networks

- **NIST**
  The National Institute of Standards and Technology (NIST) is known for the NIST Cybersecurity Framework and also offers a Penetration Testing Framework, which includes:
  - ➢ Planning
  - ➢ Discovery
  - ➢ Attack
  - ➢ Reporting

- **OWASP**
  The Open Web Application Security Project (OWASP) is a reference point for web application security standards and risk classification. OWASP maintains several standards and guides, such as:
  - ➢ Web Security Testing Guide (WSTG) (OWASP Web Security Testing Guide)
  - ➢ Mobile Security Testing Guide (MSTG)(OWASP Mobile Application Security)
  - ➢ Firmware Security Testing Methodology (https://github.com/scriptingxss/owasp-fstm)

  These frameworks and standards provide structured methods to ensure thorough and responsible penetration testing and vulnerability assessments.

4.11 Common Vulnerability Scoring System (CVSS)

In the domain of cybersecurity, assessing the gravity of vulnerabilities is essential, and there exist multiple methodologies to quantify the severity levels. The Common Vulnerability Scoring System (CVSS) represents the standardized framework widely adopted in the industry for such

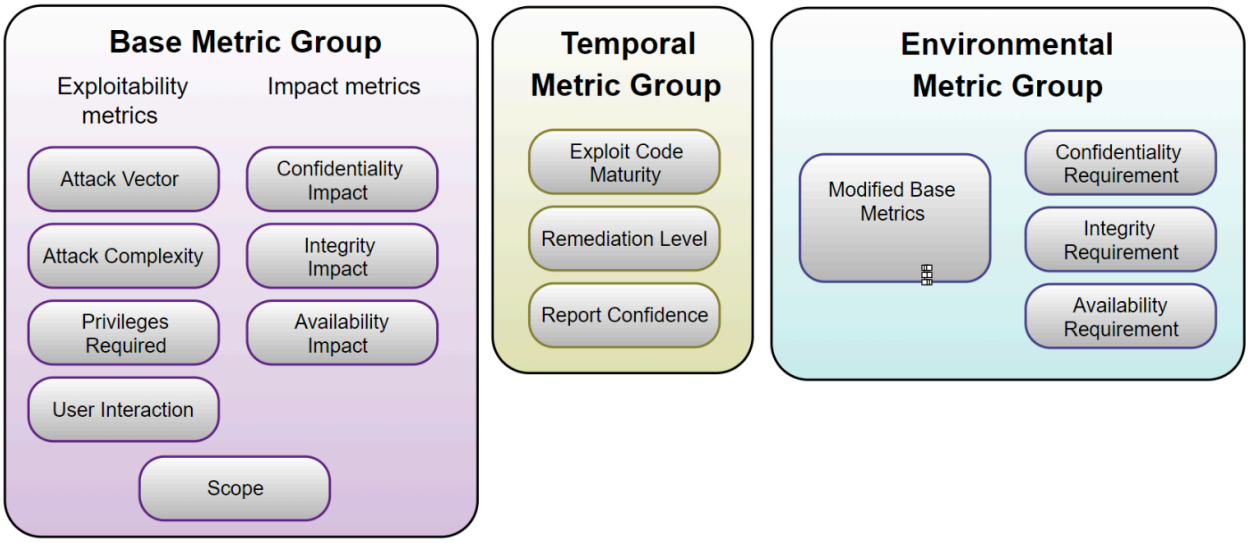
evaluations. Scanning solutions frequently integrate CVSS scores within their output, attributing a severity rating to each identified vulnerability. Comprehending the origination of these scores is pivotal, enabling manual computations or substantiation of assigned severity levels when necessary. Alongside CVSS, the Microsoft-developed DREAD model serves as an additional risk assessment tool. This system aids IT security experts in scrutinizing and quantifying the seriousness of security threats. DREAD executes risk analysis by applying a decuple-point metric, which systematically appraises the threat or vulnerability in question across five principal criteria.

- Damage Potential
- Reproducibility
- Exploitability
- Affected Users
- Discoverability

The DREAD(https://en.wikipedia.org/wiki/DREAD_(risk_assessment_model)) model is a cornerstone of Microsoft's approach to security, integral for the continuous monitoring, evaluation, and response to security concerns within its product range. It also functions as a guiding framework for IT security experts and organizational leaders, assisting in the risk analysis and the hierarchical ordering of security challenges. This model informs decision-making processes in managing and mitigating potential security incidents.

4.11.1.Risk Scoring

The CVSS system helps categorize the risk associated with an issue and allows organizations to prioritize issues based on the rating. The CVSS scoring consists of the exploitability and impact of an issue. The exploitability measurements consist of access vector, access complexity, and authentication. The impact metrics consist of the CIA triad, including confidentiality, integrity, and availability



Picture from (https://www.first.org/cvss/v3-1/media/MetricGroups.svg)

4.11.2 Base Metric Group

The CVSS base metric group represents the vulnerability characteristics and consists of exploitability metrics and impact metrics.

Exploitability Metrics
The Exploitability metrics are a way to evaluate the technical means needed to exploit the issue using the metrics below:

- Attack Vector
- Attack Complexity
- Privileges Required
- User Interaction

Impact Metrics
The Impact metrics represent the repercussions of successfully exploiting an issue and what is impacted in an environment, and it is based on the CIA triad. The CIA triad as we seen in chapter 1 is an acronym for Confidentiality, Integrity, and Availability.

- Confidentiality pertains to the safeguarding of sensitive data and ensuring that only those with proper authorization can access it. For instance, it would be considered a high-impact scenario if an intruder managed to exfiltrate passwords or decryption keys. Conversely, a low-impact scenario might involve the unauthorized retrieval of data that is not critical to the organization's operations.
-
- On integrity involves ensuring that data remains unaltered and trustworthy. A situation where an intruder successfully corrupts essential operational files would be deemed

high-impact. On the other hand, a scenario where the intruder lacks the ability to precisely control or predict the extent of data alteration would be considered low-impact.

- On availability is concerned with the accessibility of information necessary for the organization's operational needs. An attacker rendering the entire computing environment inoperable would represent a high-impact event. In contrast, a low-impact event would be characterized by an attack that partially disrupts access, where users continue to retain access to certain organizational resources.

4.11.3 Temporal Metric Group

The Temporal Metric Group encompasses the current status of exploits and patches related to a vulnerability.

Exploit Code Maturity
This metric assesses the likelihood of a vulnerability being actively exploited, based on the complexity of the required exploitation techniques. The associated values for this metric range from 'Not Defined' to 'High', 'Functional', 'Proof-of-Concept', and 'Unproven'.

Omitting this metric is signified by a 'Not Defined' status. A 'High' value indicates that reliable and straightforward exploitation methods are available, often with the support of automated tools. A 'Functional' status means publicly accessible exploit code exists. A 'Proof-of-Concept' level shows the availability of an initial exploit requiring adaptation for effective use.

Remediation Level
This metric signifies the urgency and availability of fixes for a vulnerability. It spans from 'Not Defined' to 'Unavailable', 'Workaround', 'Temporary Fix', and 'Official Fix'.

'Not Defined' suggests this metric is disregarded. 'Unavailable' means there is no remediation yet available. 'Workaround' refers to a temporary, non-official solution, while 'Temporary Fix' denotes an interim official remedy pending a permanent patch. 'Official Fix' confirms the release of a definitive vendor patch.

Report Confidence
Report Confidence evaluates the certainty of a vulnerability's existence and the precision of its technical details. Values range from 'Not Defined' to 'Confirmed', 'Reasonable', and 'Unknown'.

Skipping this metric is indicated by 'Not Defined'. 'Confirmed' reveals that detailed, corroborative information from multiple sources verifies the vulnerability. 'Reasonable' suggests that the vulnerability is documented, but there may be some uncertainty due to incomplete details for reproducing the exploit.

Environmental Metric Group
This group measures the relevance of a vulnerability to a particular organization by considering the principles of confidentiality, integrity, and availability.

Modified Base Metrics
These metrics reflect adjustments made when an organization assesses the vulnerability as posing a different level of risk than the standard CVSS base metrics suggest. The possible values are 'Not Defined', 'High', 'Medium', and 'Low'.

Choosing not to use this metric is denoted by 'Not Defined'. A 'High' rating implies a profound impact on the organization's operations or clientele if compromised. A 'Medium' level indicates a substantial, though not critical, effect, and a 'Low' value suggests only a marginal impact on the organization and its customers.

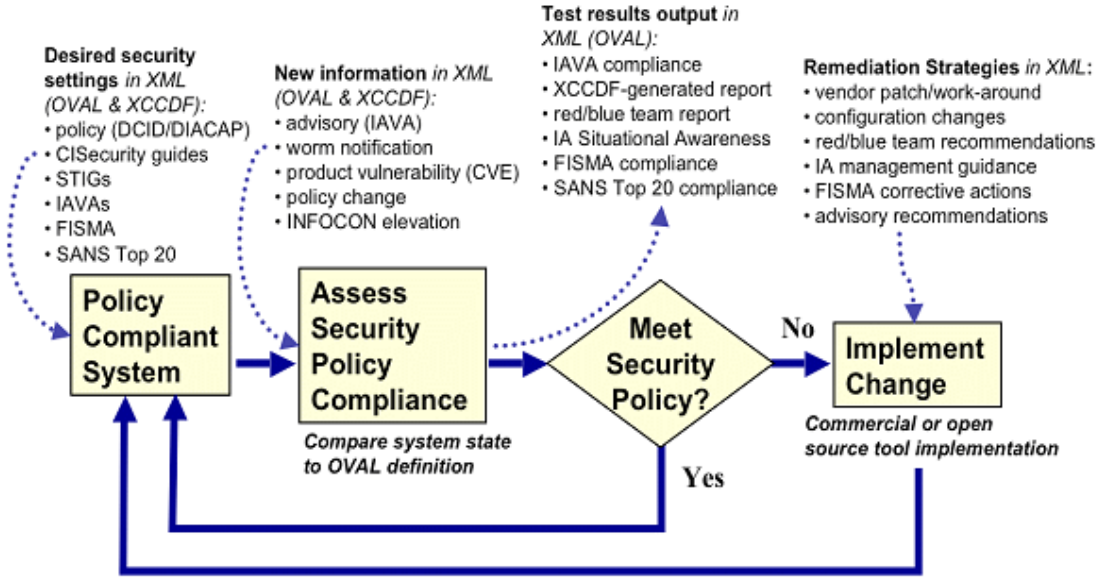
4.12 Calculating CVSS Severity

In order to calculate CVSS v3.1 score we can use this calculator from the National Vulnerability Database: (https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator)

4.12.1 Open Vulnerability Assessment Language (OVAL)

The Open Vulnerability Assessment Language (OVAL) is an open and internationally recognized security standard designed to chronicle and appraise the security posture and configuration of computer systems. The standard, which is also endorsed by the Cybersecurity and Communications division of the U.S. Department of Homeland Security, furnishes a schema for articulating the characteristics of systems and cataloging security advisories. The OVAL repository, boasting in excess of 7,000 definitions, serves as a resource freely accessible to the security community at large. Furthermore, OVAL plays an integral role in the Security Content Automation Protocol (SCAP) framework established by the U.S. National Institute of Standards and Technology (NIST). SCAP synthesizes community-contributed strategies to streamline the processes of vulnerability management, compliance measurement, and policy conformity enforcement across systems.

Oval Process



Picture taken from (https://oval.mitre.org/documents/docs-05/extras/0505Martin_f3.gif)

Captured in XML, OVAL definitions allow for the detection of software vulnerabilities, configuration errors, installed software, and other system details without the necessity for active exploitation. This preemptive identification enables organizations to determine which systems require updates across their networks.

There are four primary categories of OVAL definitions:

- OVAL Vulnerability Definitions: Pinpoint vulnerabilities within a system.
- OVAL Compliance Definitions: Check whether system settings align with established policies.
- OVAL Inventory Definitions: Ascertain the presence of certain software on a system.
- OVAL Patch Definitions: Verify the application of necessary software patches on a system.

The OVAL identification system uses a distinct format, delineated as "oval:Organization Domain Name:ID Type:ID Value". The ID Type varies across several classifications, such as definition (def), object (obj), state (ste), and variable (var). For instance, an identifier might appear as oval:org.mitre.oval:obj:1116.

Security scanners, like Nessus, can incorporate OVAL configurations to construct compliance scanning protocols. (*Oval*, n.d.) **[16]**

The Common Vulnerabilities and Exposures (CVE) system is a publicly accessible inventory of known security vulnerabilities, backed by the United States Department of Homeland Security (DHS). Each recorded security vulnerability is allocated a distinct CVE ID by the CVE

Numbering Authority (CNA). The rationale behind assigning a unique CVE ID is to establish a uniform reference for a vulnerability or exposure as identified by a researcher. A CVE entry comprises crucial details about the vulnerability or exposure, encompassing its description and relevant references. This information equips an organization's IT department with the necessary insights to gauge the potential impact of the problem on their systems.

The criteria for assigning a CVE ID to a vulnerability include independent rectifiability of the issue, its restriction to a single codebase, and formal recognition and documentation by the concerned vendor.

4.13 Penetration Tester Tools

There are many different tools for a penetration tester to do their job, each one has specific uses and can be used for a specific task according to the needs. Some common categories are listed here. **(*7 Pentesting Tools You Must Know About*, n.d.) [23]**

- **Port scanners**—Port scanners are tools that detect open ports on a system, aiding testers in identifying the operating system and active applications on a targeted network. These scanners are essential during the reconnaissance phase, offering insights into possible avenues for attacks.
- **Vulnerability scanners**—Vulnerability scanners probe for existing vulnerabilities in servers, operating systems, and applications, including configuration errors that might be exploitable. The reports generated by these scanners are crucial for penetration testers, guiding them in selecting vulnerabilities that could provide initial system access.
- **Network sniffer**—Network sniffers are used to observe network traffic details, including the origin, destination, involved devices, and the protocols and ports in use. They are particularly useful for verifying data encryption and pinpointing communication channels that might be susceptible to exploitation in penetration testing.
- **Web proxy**—Web proxies enable penetration testers to intercept and alter the traffic exchanged between their browser and a company's web servers. This capability is key in uncovering hidden HTML elements like form fields, which could be exploited for attacks such as cross-site scripting (XSS) or cross-site request forgery (CSRF).
- **Password cracker**—Password crackers focus on decrypting password hashes, a common target for attackers aiming to gain elevated privileges in a system or network.

By using these tools, penetration testers can assess the strength of passwords used within an organization, identifying weak passwords that could be easily compromised.

4.13.1 Port Scanners, Enumeration and Nmap

Enumeration is a pivotal process. It's not merely about accessing a target system but rather methodically identifying all potential attack avenues. This phase goes beyond the simple use of tools; it’s more about the strategic interaction with services to extract useful information and discover possible vulnerabilities.

The essence of enumeration lies in understanding how various services operate and the specific syntax they utilize for communication. The goal is to augment knowledge about technologies and protocols, adapting this newfound information to previously acquired knowledge. Enumeration involves gathering extensive data to uncover attack vectors, with the more information gathered, the easier it becomes to identify these vectors. Accessing a target system becomes straightforward once the method is known. Generally, this can be narrowed down to functionalities or resources that allow interaction with the target or provide additional information, and information that offers further critical insights for system access. During scanning, the focus is on these aspects, with most vulnerabilities arising from misconfigurations or security oversight. Often, reliance solely on firewalls, Group Policy Objects (GPOs), and updates is insufficient for robust network security. Enumeration is crucial, yet it is often misunderstood. The challenge usually isn't about trying all tools, but understanding how to interact with the service and discern what's relevant. Investing time in learning about the service can significantly expedite achieving the objective of system access. Manual enumeration plays a vital role here. While automated scanning tools can expedite the process, they may not always circumvent the security measures of services. This is where manual efforts become indispensable.

Network Mapper (Nmap) is an exemplary tool in this context. It's an open-source network scanner used for security auditing and network discovery. Written in C, C++, Python, and Lua, Nmap excels in identifying available hosts on a network, along with their services, applications, operating systems, and versions. It can also evaluate firewalls and intrusion detection systems. Nmap is utilized by network administrators and IT security professionals for various purposes

including network security audits, penetration testing simulations, firewall and IDS configuration checks, network mapping, response analysis, open port identification, and vulnerability assessments.Nmap's architecture allows for a variety of scans such as host discovery, port scanning, service and OS detection, and scriptable interaction through the Nmap Scripting Engine. Its diverse scanning techniques include TCP SYN/Connect()/ACK/Window/Maimon scans, UDP Scan, TCP Null, FIN, and Xmas scans, IP protocol scans, and more. For instance, the TCP-SYN scan (-sS), a default and popular method, can scan thousands of ports per second without establishing a full TCP connection, thereby offering effective and discreet scanning capabilities.

4.13.2 Vulnerability Scanning

Vulnerability scanning, a crucial aspect of network security, aims to detect vulnerabilities in various network components like routers, firewalls, switches, and also in servers, workstations, and software applications. This process is mostly automated and seeks out potential or known vulnerabilities at both network and application levels. While vulnerability scanners generally don’t exploit the found vulnerabilities, human intervention is necessary to confirm whether the issues detected are actual vulnerabilities requiring attention or mere false positives that should be disregarded in subsequent scans. Distinct from standard penetration testing, vulnerability scanning is often a component of such tests but differs in scope and depth. It can extend the range of a penetration test or expedite the testing process within limited timeframes. Penetration testing, in contrast, encompasses a broader range of activities beyond mere scanning.The variety of scans conducted depends on the specific tool in use, with most tools conducting a mix of dynamic and static analyses based on the target and identified vulnerabilities. Static analysis might flag vulnerabilities linked to be publicly known Common Vulnerabilities and Exposures (CVEs) based on the version of an asset. However, this may not always be reliable due to applied patches or specific non-susceptibilities to the CVE. Dynamic analysis, conversely, tests for vulnerabilities by applying harmless payloads, such as testing for weak credentials or attempting SQL or command injections on a target, typically a web application. A successful response from these payloads often indicates vulnerability.For comprehensive security, organizations are advised to conduct both unauthenticated and authenticated scans regularly. This practice helps ensure timely patching of assets as new vulnerabilities emerge and checks for any issues in newly added network assets. Effective vulnerability scanning is a critical element in an organization’s patch management

strategy.Regarding tools, Nessus, Nexpose, and Qualys are prominent vulnerability scanning platforms, offering community editions at no cost. Additionally, open-source options like OpenVAS are available for those seeking alternatives.

4.13.3 Web Proxies

Web proxies play a critical role in the security testing of web and mobile applications, especially given the reliance of these applications on back-end server processing. In Web Application Penetration Testing, which applies to both web and mobile apps, testing web requests to back-end servers is a key focus. Web proxies are employed to capture and manipulate the traffic between applications and back-end servers for testing purposes.

Understanding Web Proxies: Web proxies act as intermediaries between a user's browser or mobile application and a back-end server. They capture and display all web requests and responses exchanged, functioning as man-in-the-middle (MITM) tools. This is in contrast to network sniffing tools like Wireshark, which analyze all traffic on a network, including non-web-related data. Web proxies specifically target web traffic, often focusing on standard web ports like HTTP/80 and HTTPS/443.

These tools are indispensable for web penetration testers, as they simplify the process of capturing and modifying web requests. Once set up, a web proxy allows visibility into all HTTP requests from an application and responses from the server. Importantly, they enable the interception and alteration of specific requests, a crucial aspect of web penetration testing.

Applications of Web Proxies : Web proxies are versatile, with functionalities extending beyond just capturing and replaying HTTP requests. They are used for a variety of tasks such as:

- Web application vulnerability scanning
- Web fuzzing
- Web crawling and mapping
- Analysis of web requests
- Testing web configurations

- Conducting code reviews

Introduction to Burp Suite: Burp Suite, often simply called Burp, stands out as a leading web proxy tool for web penetration testing. It offers a user-friendly interface along with a range of features, including a built-in Chromium browser for testing web applications. While Burp Suite has a commercial version with advanced features like an active web app scanner and fast Burp Intruder, its free community version is a potent tool for most penetration testing needs. The community version is sufficient for many testers, with the Pro version being more suitable for advanced web application penetration testing. Burp Suite excels in various functionalities, some of which are available in the free version, while others, like the Active Web App Scanner, are part of the Pro offering.

4.13.4 Password Crackers

Brute Force attacks are a technique used to guess passwords or keys through systematic trial and error. This method, often employed for password cracking, relies on the fact that passwords are typically stored as hash values, not in plain text.

Files Containing Hashed Passwords:
For Windows, such files include unattend.xml, sysprep.inf, and SAM. On Linux systems, these files are shadow and shadow.bak, among others.

In brute force attacks, since the original password cannot be deduced from its hash, the approach involves generating hash values from a series of guessed passwords until one matches the stored hash, thereby revealing the password. This is known as offline brute-forcing. The focus here, however, will be on online brute-forcing, particularly targeting login forms on websites.

Tools for Brute Force Attacks:
Several tools facilitate brute-forcing login attempts, including Ncrack, wfuzz, medusa, patator, and hydra. These tools are designed to test various authentication mechanisms, like Basic HTTP AUTH, commonly found in web servers.

Hydra - A Versatile Brute Force Tool:
Hydra stands out as a proficient tool for login brute-forcing, covering a wide range of attacks and services. It's known for its speed and efficiency in testing credential pairs. Hydra is pre-installed in environments like Pwnbox and can also be installed on personal machines.

Methodology and Types of Password Attacks:
Brute Force attacks can be comprehensive, testing all possible character combinations. However, this method becomes impractical with longer passwords or those including mixed cases, numbers, and special characters. Therefore, Dictionary Attacks, which use lists of known passwords, are often employed. These attacks are more efficient, especially when passwords are not exceedingly complex or unique.

Wordlists for Brute Forcing:
Commonly used word lists such as **rockyou.txt**, which contains millions of passwords from online database leaks, are crucial for these attacks. Such wordlists are available in directories like /opt/useful/SecLists/Passwords/ in Pwnbox or can be downloaded from repositories like Hashcat's GitHub.

Approaches in Brute Force Attacks:
Various methods exist for carrying out brute force attacks, including online attacks on live applications, offline cracking of password hashes, reverse brute-forcing using common passwords with multiple usernames, and hybrid attacks that create customized wordlists based on known user information. Default passwords, often overlooked or forgotten, present a significant vulnerability and are thus a primary target in brute force attacks.

Overall

Having understood the role of an ethical hacker, we can outline a pathway to become one. As advised by John Hammond, a renowned Ethical Hacker and security expert, the journey begins with mastering Linux. Following this, engaging in "wargames" on Overthewire.org is recommended to sharpen practical skills. Python, a popular language within the hacking community, is the next essential skill to acquire. After gaining proficiency, one should venture into CTFTime.org to participate in 'Capture the Flag' challenges, particularly in a Red Team vs. Blue Team setup. For beginners, Tryhackme is an excellent, user-friendly training resource, along with Hackthebox Academy. Progressing from these stages, one can start participating in real bug bounty programs. To be considered a professional penetration tester, obtaining the OSCP (Offensive Security Certified Professional) certification, which is challenging yet essential, is the minimum requirement. (*Learn to Hack (the Best Way) // Ft. John Hammond*, 2020)

Conclusion

This paper delves into the realms of server security and ethical hacking, highlighting the critical need for maintaining a dynamic balance between solid defensive strategies and ongoing research into the ever-changing landscape of cyber threats. The synergy of theoretical knowledge and practical application is imperative for reinforcing server security effectively. As the nature of cybersecurity threats constantly transforms, so too must our approach to server security, through relentless education, adaptation, and proactive measures. Consider this paper as an initial attempt into the expansive and complex world of cybersecurity.

References

[1] Abdelhakim Hannousse, Salima Yahiouche, & Mohamed Cherif Nait-Hamoud. (2022). *Twenty-two years since revealing cross-site scripting attacks: a systematic mapping and a comprehensive survey*. 10.48550/arXiv.2205.08425

[2] Al-Sabaawi, A., & Alrowidhan, T. A. (2023, December 17). *Detecting network security vulnerabilities and proactive strategies to mitigate potential threats*. arXiv. Retrieved January 10, 2024, from https://doi.org/10.48550/arXiv.2212.11449

[3] Christen, M., Gordijn, B., & Loi, M. (Eds.). (2020). *The Ethics of Cybersecurity*. Springer International Publishing.

[4] *Confidentiality, Integrity, and Availability: The CIA Triad | Office of Information Security | Washington University in St. Louis*. (n.d.). Office of Information Security. Retrieved January 7, 2024, from https://informationsecurity.wustl.edu/items/confidentiality-integrity-and-availability-the-cia-triad/

[5] *CSRF Attack: Cross-Site Request Forgery Definition & Defence*. (2023, February 14). Okta. Retrieved January 10, 2024, from https://www.okta.com/au/identity-101/csrf-attack/

[6] *Cyber Kill Chain®*. (2011). Lockheed Martin. Retrieved January 7, 2024, from https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[7] *CyberSecurity Definitions*. (n.d.). Cybersecurity Enthusiast. Retrieved January 10, 2024, from https://csenthusiast.com/cybersecurity-definitions

[8] Etienne Boespflug,, Abderrahmane Bouguern, Laurent Mounier, & Marie-Laure Potet. (2023). *A tool assisted methodology to harden programs against multi-faults injections*. University of Grenoble Alpes (UGA), Grenoble, France.

[9] Fredriksson, J. (2022, April 10). *Network Architecture - Server Network Segmentation — WIRES AND WI.FI*. WIRES AND WI.FI. Retrieved January 10, 2024, from https://www.wiresandwi.fi/blog/server-network-segmentation

[10] *HackTheBox*. (n.d.). HTB Academy: Best Online Cybersecurity Courses & Certifications. Retrieved January 10, 2024, from https://academy.hackthebox.com

[11] Hassan, M. H., Nipa, S. S., Akter, M., Haque, R., Deepa, F. N., Rahman, M., Siddiqui, M., & Sharif, M. H. (2018). *Broken Authentication and session Management vulnerability: A case study of web application*. International Journal of Simulation: Systems, Science and Technology. 10.5013/IJSSST.a.19.02.06

[12] Johnson, G. (n.d.). *Remote and Local File Inclusion Explained*. Repository [Root Me. Retrieved January 10, 2024, from https://repository.root-me.org/Exploitation%20-%20Web/EN%20-%20Remote%20File%20Inclusion%20and%20Local%20File%20Inclusion%20explained.pdf

[13] Ke Li, Heng Yang, & Willem Visser. (2022). *Evolutionary Multi-Task Injection Testing on Web Application Firewalls*. 1Department of Computer Science, University of Exeter UK. https://arxiv.org/pdf/2206.05743.pdf

[14] Md. Maruf Hassan, Touhid Bhuiyan, & Saikat Biswas. (2016, November). *An Investigation of Educational Web Applications in Bangladesh: A Case Study on Local File Disclosure Vulnerability*.

[15] *National Vulnerability Database*. (n.d.). *National Vulnerability Database*

[16] *Oval*. (n.d.). Open Vulnerability and Assessment Language: OVAL. Retrieved January 4, 2024, from https://oval.mitre.org

[17] Rajat Tandon. (2020). *A Survey of Distributed Denial of Service Attacks and Defences*. 10.48550/arXiv.2008.01345

[18]Ralph Langner. (2013, 11). *To Kill a Centrifuge*. To Kill a Centrifuge. Retrieved January 7, 2024, from https://www.langner.com/wp-content/uploads/2017/03/to-kill-a-centrifuge.pdf

[19] *Saltzer and Schroeder's design principles*. (n.d.). Wikipedia. Retrieved January 7, 2024, from https://en.wikipedia.org/wiki/Saltzer_and_Schroeder%27s_design_principles

[20] Schneier, B. (2012, December 17). *News: Complexity the Worst Enemy of Security*. Schneier on Security. Retrieved January 7, 2024, from https://www.schneier.com/news/archives/2012/12/complexity_the_worst.html

[21] *Securing Network Servers*. (2000, November 9). Retrieved January 10, 2024, from https://apps.dtic.mil/sti/pdfs/ADA379469.pdf

[22] Sen, K. (2023, July 24). *What is a Vulnerability? Definition + Examples*. UpGuard. Retrieved January 7, 2024, from https://www.upguard.com/blog/vulnerability

[23] *7 Pentesting Tools You Must Know About*. (n.d.). HackerOne. Retrieved January 10, 2024, from https://www.hackerone.com/knowledge-center/7-pentesting-tools-you-must-know-about

[24] Tevault, D. A. (2020). *Mastering Linux Security and Hardening: Protect Your Linux Systems from Intruders, Malware Attacks, and Other Cyber Threats, 2nd Edition*. Packt Publishing.

[25] Tracy, M. C., Jansen, W., Scarfone, K., & Winograd, T. (2007). *Guidelines on Securing Public Web Servers*. NIST SP 800-44 Version 2, Guidelines on Securing Public Web Servers. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-44ver2.pdf

[26] Ware, W. (n.d.). *Penetration test*. Wikipedia. Retrieved January 10, 2024, from https://en.wikipedia.org/wiki/Penetration_test

[27] *Web Server Security Guidelines - Information Security Office - Computing Services*. (n.d.). Carnegie Mellon University. Retrieved January 5, 2024, from https://www.cmu.edu/iso/governance/guidelines/web-server.html

[28] Zetter, K. (2016, March 3). *Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid*. WIRED. Retrieved January 7, 2024, from https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/