Experiment no:1

Recon-ng tool

Aim : To perform active reconnaissance using reconnaissance tool using reco-ng

Description:

Recon-ng is free and open-source tool available on GitHub. Recon-ng is based upon Open-Source Intelligence (OSINT), the easiest and useful tool for reconnaissance. Recon-ng interface is very similar to Metasploit1 and Metasploit2. Recon-ng provides a command-line interface that you can run on Kali Linux. This tool can be used to get information about our target(domain). The interactive console provides a number of helpful features, such as command completion and contextual help. Recon-ng is a Web Reconnaissance tool written in Python. It has so many modules, database interaction, built-in convenience functions, interactive help, and command completion, Recon-ng provides a powerful environment in which open-source web-based reconnaissance can be conducted, and we can gather all information.

Features of Recon-ng :

- ✓ Recon-ng is free and open-source tool this means you can download and use it at free of cost.
- ✓ Recon-ng is a complete package of information gathering modules. It has so many modules that you can use for information gathering.
- ✓ Recon-ng works and acts as a web application/website scanner.
- ✓ Recon-ng is one of the easiest and useful tool for performing reconnaissance.
- ✓ Recon-ng interface is very similar to metasploitable1 and metasploitable2 that makes is easy to use.
- ✓ Recon-ng's interactive console provides a number of helpful features.
- ✓ Recon-ng is used for information gathering and vulnerability assessment of web applications.
- ✓ Recon-ng uses shodan search engine to scan iot devices.
- ✓ Recon-ng can easily find loopholes in the code of web applications and websites.
- ✓ Recon-ng has following modules Geoip lookup, Banner grabbing, DNS lookup, port scanning, These modules makes this tool so powerful.

- ✓ Recon-ng can target a single domain and can found all the subdomains of that domain which makes work easy for pentesters.

Uses of Recon-ng :

- ✓ Recon-ng is a complete package of Information gathering tools.
- ✓ Recon-ng can be used to find IP Addresses of target.
- ✓ Recon-ng can be used to look for error-based SQL injections.
- ✓ Recon-ng can be used to find sensitive files such as robots.txt.
- ✓ Recon-ng can be used to find information about Geo-IP lookup, Banner grabbing, DNS lookup, port scanning, sub-domain information, reverse IP using WHOIS lookup.
- ✓ Recon-ng can be used to detects Content Management Systems (CMS) in use of a target web application,
- ✓ InfoSploit can be used for WHOIS data collection, Geo-IP lookup, Banner grabbing, DNS lookup, port scanning, sub-domain information, reverse IP, and MX records lookup
- ✓ Recon-ng is a complete package (TOOL) for information gathering. This tool is free and Open Source.
- ✓ Recon-ng subdomain finder modules is used to find subdomains of a single domain.
- ✓ Recon-ng can be used to find robots.txt file of a website.
- ✓ Recon-ng port scanner modules find closes and open ports which can be used to maintain access to the server.
- ✓ Recon-ng has various modules that can be used to get the information about target.

<u>Procedures:</u>

Step 1: Open Terminal of your Kali Linux

Step 2: On Terminal now type command.

git clone https://github.com/lanmaster53/recon-ng.git

recon-ng has been installed on your Kali Linux ,now you just have to run recon-ng.

Step 3: To launch recon-ng on your kali Linux type the following the command and press enter.

recon-ng

Now Recon-ng has been downloaded and running successfully.

Step 4: Now to do Reconnaissance first you have to create a workspace for that. Basically, workspaces are like separate spaces in which you can perform reconnaissance of different targets. To know about workspaces just type the following command.

workspaces

workspaces create (name)

Step 5: You have created workspace for you, now you have to go to marketplace to install modules to initiate your Reconnaissance here we have created a workspace called GeeksForGeeks. Now we will Reconnaissance within GeeksForGeeks workspace. Now go to marketplace and install modules.

marketplace search

Step 6: As you can now see a list of modules and so many of them are not installed so to install those modules type following command.

marketplace install (module name)

marketplace install all

Step 7: Type the command

modules search hack

Step 8: As you can see that we have installed the module names recon/domains-hosts/hackertarget. Now we will load this module in our workspace.

modules load (module name)

Step 9: As you can see now we are under those modules. Now to use this module we have to set the source.

options set SOURCE (domain name)

Step 10: Type the command

run

We have set google.com as a source by command options set SOURCE google.com. Recon-ng is Open-Source Intelligence, the easiest and useful tool for reconnaissance.

RESULT

Active reconnaissance using reconnaissance tool using recon-ng is performed.

Enumerate Infrastructure and Application Admin Interface

AIM:

To famialiarize tools for enumerating Infrastructure and Application Admin Interface.

Description:

Administrator interfaces may be present in the application or on the application server to allow certain users to perform privileged activities on the site. Tests should be undertaken to reveal if and how this privileged functionality can be accessed by an unauthorized or standard user.

An application may require an administrator interface to enable a privileged user to access functionality that may make changes to how the site functions. Such changes may include:

•user account provisioning

•site design and layout

•data manipulation

•configuration changes

In many instances, such interfaces do not have sufficient controls to protect them from unauthorized access. Testing is aimed at discovering these administrator interfaces and accessing functionality intended for the privileged users.

There are a wide variety of administration interfaces for different technologies. For example:

•Remote management protocols such as SSH, PowerShell, RDP and VNC

•Browser-based management such as cloud-based web consoles and appliance configuration panels

•APIs that expose management functionality

•Thick clients - typically, software installed on a device that drives an administration protocol or API in the background

Some systems are managed through code and configuration files rather than the interfaces listed above. This can also be a form of system administration.

Administration interfaces are an attack surface. Only legitimate administrators should be able to communicate with them. You should isolate these interfaces using architectural controls and constrain who can connect to the system and from where. This will help to protect against attacks such as brute forcing administrator login, or using an exploit to gain access.

There are a number of ways that you can reduce the exposure of your management interfaces. For example:

- ✓ You could create a dedicated management network that only authorised administrators have physical access to.
- ✓ You could place your administration interfaces behind a VPN that only authenticated administrators and devices can use.
- ✓ You could implement an IP allow list to restrict the devices or networks that can access the administration interface.

Some architectural controls are stronger than others. For example, IP allow lists can be overcome if the attacker is able to share the IP address space, or spoof it. They can also be difficult to maintain.

Depending on your risk appetite, it may be necessary to combine multiple mitigations to adequately protect your administration interfaces.The mitigations that you choose will depend on a number of factors, including where the interface is located, what it allows access to and who the administrators are that need access to it.

Implementation guidance

- ✓ Only permit authorised devices. It should not be possible for untrusted devices to access your administration interfaces. Architectural controls should be implemented to ensure that the origin of administration activities is a trusted device. This helps to reduce the attack surface.
- ✓ Use browse down, not browse up. If an attacker compromises a less trusted device, they will inherit it's accesses. If that device can be used to browse up to a more critical system, the attacker can do so too. See 1 - Gain trust in your management devices for more information.
- ✓ Authenticate the administrator. Administrators must be authenticated before carrying out their duties. Authentication should be achieved using well known protocols.
- ✓ Utilise multi-factor authentication. Where available, MFA should be enabled on administrator accounts. This introduces a high hurdle for an attacker to jump, but has little effect on administrators. Remember that this isn't bulletproof: if an adversary has

access to the PAW, they can piggyback on an authenticated session after MFA has been completed.

- ✓ Protect your administration credentials. Discourage administrators from writing credentials on post-it notes and consider the use of a password manager. Also ensure that credentials are not hard-coded into software projects, as they could mistakenly be published to a code repository. If certificates are used for authentication, carefully protect them. See our password guidance for more information.
- ✓ Use Privilege Access Management. See 4 - Use privileged access management for more information.
- ✓ Use secure protocols. The protocol that is used to carry traffic from a Privileged Access Workstation to the administration interface should be encrypted. If it is not, an attacker could view the traffic and manipulate it. For example, APIs using HTTPS should be preferred over Telnet. You should investigate what protocol is used by any thick client and ensure that these connections are protected too.
- ✓ Update your administration infrastructure. Updates should be installed to ensure that your systems are patched against known vulnerabilities. These updates should be installed as soon as possible, because attackers put effort into understanding them, searching for the vulnerabilities that they patch.
- ✓ Monitor administration activities. See 5 - Log and audit administration activities for more information.
- ✓ Implement procedural policies governing system administrators. System administration policies should cover a wide number of topics. For example, the number of administrators should be bounded with upper and lower limits that are appropriate to your environment. Further, the principal of least privilege should be applied, and a robust 'joiners, movers and leavers' process should be defined.

<u>Procedure:</u>

✓ ***Dnsrecon***

DNSRecon is a Python script that provides the ability to perform:

- Check all NS Records for Zone Transfers.
- Enumerate General DNS Records for a given Domain (MX, SOA, NS, A, AAAA, SPF and TXT).
- Perform common SRV Record Enumeration.

- Top Level Domain (TLD) Expansion.
- Check for Wildcard Resolution.
- Brute Force subdomain and host A and AAAA records given a domain and a wordlist.
- Perform a PTR Record lookup for a given IP Range or CIDR.
- Check a DNS Server Cached records for A, AAAA and CNAME
- Records provided a list of host records in a text file to check.
- Enumerate Hosts and Subdomains using Google

type DnsRecon on the kali linux terminal.

✓ ***theHarvester***

theHarvester is a command-line tool included in Kali Linux that acts as a wrapper for a variety of search engines and is used to find email accounts, subdomain names, virtual hosts, open ports / banners, and employee names related to a domain from different public sources (such as search engines and PGP key servers). In recent versions, the authors added the capability of doing DNS brute force, reverse IP resolution, and **Top-Level Domain** (**TLD**) expansion.

✓ In the following example, theHarvester is used to gather information about zonetransfer.me:

✓ ***Finding subdomain info***

✓ ***Dig***

dig command stands for Domain Information Groper. It is used for retrieving information about DNS name servers. It is basically used by network administrators. It is used for verifying and troubleshooting DNS problems and to perform DNS lookups. Dig command replaces older tools such as nslookup and the host.

dig geeksforgeeks.org

- ✓ **Google Dorking**

  A Google Dork is a special search term. These terms, when used with regular search keywords, can help us discover hidden resources crawled by Google.

Here are some of the most common operators used in Google Dorking.

Intitle operator

The “**intitle**” operator searches for web pages with specific words or phrases in the title tag. For instance, if you’re looking for pages that contain the phrase “password” and have “index of” in the title, you would use the search term:intitle:”index of” password.

In title. Image by the author.

Inurl operator

The “**inurl**” operator searches for web pages that contain specific words or phrases in the URL. For example, if you’re looking for pages that contain “admin.php” in the URL, you would use the search term:inurl:admin.php.

In url. Image by the author.

Site operator

The “**site**” operator allows you to search within a specific website or domain. For instance, if you’re looking for pages on the example.com domain that contain the word “Steganography”, you would use the search term:site:yeahhub.com “Steganography”

In site. Image by the author.

Filetype operator

The “**filetype**” operator allows you to search for specific file types, such as PDFs or Word documents. For example, if you’re looking for PDF files that contain the phrase “confidential report”, you would use the search term:filetype:pdf "Advanced Network Security"

Filetype. Image by the author.

Intext operator

The “**intext**” operator searches for pages that contain specific words or phrases within the body of the page. For instance, if you’re looking for pages that contain both the words “login” and “password” within the body of the page, you would use the search term:intext:"about" contact.

In text. Image by the author.

Link operator

The "**link**" operator searches for web pages that link to a specific URL. For example, if you're looking for web pages that link to the example.com domain, you would use the search term:link:"example.com"

Link operator. Image by the author.

Cache operator

The "**cache**" operator is used to retrieve the cached version of a web page. When you search for a website using Google, Google creates a cached version of that page in its system. This version can be useful if the original website is temporarily down or if you want to view an older version of the website.

Here is the syntax to find the cached version of yahoo.com.cache:https://www.yahoo.com

Cached version of yahoo.com. Image by author.

Related operator

The "**related**" operator is used to find web pages that are related to a specific URL. Here is the syntax to use the "related" operator to find sites similar to yahoo.com.

Related operator. Image by author.

By combining these operators in creative ways, you can find specific types of information on the web that can be useful for penetration testing and other purposes.

Test for Configuration and deployment

Aim: To famializrize tools configuration and deployment testing

Description:

The intrinsic complexity of interconnected and heterogeneous web server infrastructure, which can include hundreds of web applications, makes configuration management and review a fundamental step in testing and deploying every single application. It takes only a single vulnerability to undermine the security of the entire infrastructure, and even small and seemingly unimportant problems may evolve into severe risks for another application on the same server. In order to address these problems, it is of utmost importance to perform an in-depth review of configuration and known security issues, after having mapped the entire architecture.

Proper configuration management of the web server infrastructure is very important in order to preserve the security of the application itself. If elements such as the web server software, the back-end database servers, or the authentication servers are not properly reviewed and secured, they might introduce undesired risks or introduce new vulnerabilities that might compromise the application itself.

For example, a web server vulnerability that would allow a remote attacker to disclose the source code of the application itself (a vulnerability that has arisen a number of times in both web servers or application servers) could compromise the application, as anonymous users could use the information disclosed in the source code to leverage attacks against the application or its users.

The following steps need to be taken to test the configuration management infrastructure:

- ✓ The different elements that make up the infrastructure need to be determined in order to understand how they interact with a web application and how they affect its security.
- ✓ All the elements of the infrastructure need to be reviewed in order to make sure that they don’t contain any known vulnerabilities.
- ✓ A review needs to be made of the administrative tools used to maintain all the different elements.

- ✓ The authentication systems, need to reviewed in order to assure that they serve the needs of the application and that they cannot be manipulated by external users to leverage access.
- ✓ A list of defined ports which are required for the application should be maintained and kept under change control.

Procedure:

tools:

1.WPScan

Wpscan is a WordPress security scanner used to test WordPress installations and WordPress-powered websites. This is a command line tool used in Kali Linux. This tool can be used to find any vulnerable plugins, themes, or backups running on the site. It is usually used by individual WordPress site owners to test their own websites for vulnerabilities and also by large organizations to maintain a secure website. This tool can also be used to enumerate users and perform brute-force attacks on known WordPress users. In this article, We are going to take you through different commands of wpscan tool, the most commonly used attacks on WordPress sites, and tips to defend against them. The below functionalities of this tool can be used from the point of view of a hacker or even just someone who wants to test if their WordPress site is secure enough.

- ***–url : (basic scan)***

    wpscan –url IP_ADDRESS_OF_WEBSITE

- **–help: (help options)**

```
wpscan --help
```

It’s a common practice in Linux to use the “–help” option to get the complete list of the usability of the tool using different switches for different functionalities.

- ***-e u: (enumerating website users)***
  wpscan –url IP_ADDRESS_OF_WEBSITE -e u

This lets the wpscan tool enumerate the WordPress site for valid login usernames. After the scan, it would give all the usernames the tool has enumerated which are valid users of the WordPress site and are often times brute forced to gain unauthorized access to the WordPress admin/author dashboard.

- ***-U -P: (brute-forcing password for the identified user)***

  wpscan –url IP_ADDRESS_OF_WEBSITE -U USER_NAME -P PATH_TO_WORDLIST

As we managed to enumerate some usernames for the WordPress site above, let's try to brute-force the user "kwheel".

  - **Brute forcing**: It is a technique where a wordlist is used against a tool that effectively finds the matching password for the given username.
  - **Wordlist**: The password cracking tool uses a wordlist, which is nothing but a text file containing a set of commonly used passwords. Ex: 12345678, qwerty, pizza123,etc.

Now let's try to brute force the user "kwheel".

rokyou.txt is a quite commonly used wordlist for brute force attacks.

2. OWASP ZAP

ZAP has installers for Windows, Linux, and Mac OS/X, as well as Docker images. Download the appropriate installer from the download page and install it on the machine where you will run the penetration test.

Java 8 or higher is required to run ZAP. The Mac OS/X installer includes the appropriate Java version, but Java 8+ must be installed separately for Windows, Linux, and cross-platform versions. The Docker version already includes Java.

When you start ZAP for the first time, you need to choose whether to make the ZAP session persistent. If you persist the session, it will be saved to a local HSQLDB. Otherwise, files will be deleted when you log out of ZAP.

Before proceeding, ensure you have permission from the web application owner to perform a penetration test.

Run a quick start auto scan:

- ➢ Start ZAP and click the Quick Launch tab in the workspace window.
- ➢ Click the Auto Scan button.
- ➢ In the Attack URL text box, enter the full URL of the web application.
- ➢ Select either Use traditional spider, Use ajax spider, or both (more details below)
- ➢ Click Attack.

1. Click **Attack**.

ZAP uses a crawler to go through the web application and scan pages it finds. It then uses the active scanner to attack every page, function, and parameter it finds.

ZAP-spiders

ZAP provides two spiders for scraping web applications, which you can select in the automated scan dialog:

- ▪ The traditional ZAP spider inspects HTML in a web application's response to detect links. Although this spider is fast, it is less effective when navigating AJAX web applications that use JavaScript to generate links.

- ▪ The ZAP AJAX spider is more effective for JavaScript applications. It navigates a web application by invoking a browser, rendering the full JavaScript of the page, and following any links on the resulting page. AJAX spiders are slower than traditional spiders and require additional configuration to be used in a headless environment. ZAP uses two forms of scanning:

- Passive scanning investigates all proxy requests and responses, but does not change the response in any way and is considered safe. It can be done on a background thread so it doesn't slow down the application. This can find some vulnerabilities and can help you understand the basic security posture of a web application.
- Active scanning attempts to find additional vulnerabilities using known attack vectors against the selected target. Do not use active scans against targets you don't have permission to test, as active scans are real attacks that might cause damage .

Identity Management Testing

Aim :

To familiarize tools for testing identity management using Burpsuite

Description:

Identity management (ID management) is the organizational process for ensuring individuals have the appropriate access to technology resources.This includes the identification, authentication and authorization of a person, or persons, to have access to applications, systems or networks. This is done by associating user rights and restrictions with established identities.

Managed identities can also refer to software processes that need access to organizational systems. Identity management can be considered an essential component for security. Identity management includes authenticating users and determining whether they're allowed access to particular systems. ID management works hand-in-hand with identity and access management (IAM) systems. Identity management is focused on authentication, while access management is aimed at authorization.

The main goal of identity management is to ensure only authenticated users are granted access to the specific applications, systems or IT environments for which they are authorized. This includes control over user provisioning and the process of onboarding new users such as employees, partners, clients and other stakeholders.

Identity management also includes control over the process of authorizing system or network permissions for existing users and the offboarding of users who are no longer authorized to access organization systems.

ID management determines whether a user has access to systems and sets the level of access and permissions a user has on a particular system. For instance, a user may be authorized to access a system but be restricted from some of its components.Identity governance, the policies and processes that guide how roles and user access should be administered across a business environment, is an important aspect of identity management. Identity governance is key to successfully managing role-based access management systems.

Identity management is an important part of the enterprise security plan, as it is linked to both the security and productivity of the organization.In many organizations, users are granted more access privileges than they need to perform their functions. Attackers can take advantage of compromised user credentials to gain access to organizations' network and data. Using identity management, organizations can safeguard their corporate assets against many threats including hacking, ransomware, phishing and other malware attacks.

Identity management systems add an additional layer of protection by ensuring user access policies and rules are applied consistently across an organization.

Procedure:

1. **Test-Role-Definitions**

   Burp Suite Enterprise Edition uses a role-based access control model. You manage permissions for users using roles and groups:

   - ✓ A user represents a person who has access to Burp Suite Enterprise Edition via the web interface, or a system that has access via one of the APIs.
   - ✓ A role is a set of permissions to perform specific actions, such as scheduling and deleting scans. You assign roles to groups of users.
   - ✓ A group is a collection of users with an assigned set of roles.

2. **Test User Registration Process**

   Verify that the identity requirements for user registration are aligned with business and security requirements:

   - ✓ Can anyone register for access?
   - ✓ Are registrations vetted by a human prior to provisioning, or are they automatically granted if the criteria are met?
   - ✓ Can the same person or identity register multiple times?
   - ✓ Can users register for different roles or permissions?
   - ✓ What proof of identity is required for a registration to be successful?
   - ✓ Are registered identities verified?

   Validate the registration process:

   - ✓ Can identity information be easily forged or faked?
   - ✓ Can the exchange of identity information be manipulated during registration?

3. **Testing for Account Enumeration and Guessable User Account**

   The scope of this test is to verify if it is possible to collect a set of valid usernames by interacting with the authentication mechanism of the application. This test will be useful

for brute force testing, in which the tester verifies if, given a valid username, it is possible to find the corresponding password.
Often, web applications reveal when a username exists on system, either as a consequence of mis-configuration or as a design decision. For example, sometimes, when we submit wrong credentials, we receive a message that states that either the username is present on the system or the provided password is wrong. The information obtained can be used by an attacker to gain a list of users on system. This information can be used to attack the web application, for example, through a brute force or default username and password attack.

The tester should interact with the authentication mechanism of the application to understand if sending particular requests causes the application to answer in different manners. This issue exists because the information released from web application or web server when the user provide a valid username is different than when they use an invalid one.

4.Testing for Weak or Unenforced Username Policy

- Determine the structure of account names.
- Evaluate the application's response to valid and invalid account names.
- Use different responses to valid and invalid account names to enumerate valid account names.
- Use account name dictionaries to enumerate valid account names.

Authentication and Authorization Testing

Aim:

To familiarize authentication and authorization using burpsuite

Description:

Authentication is the process of verifying who someone is, whereas authorization is the process of verifying what specific applications, files, and data a user has access to. The situation is like that of an airline that needs to determine which people can come on board. The first step is to confirm the identity of a passenger to make sure they are who they say they are. Once a passenger's identity has been determined, the second step is verifying any special services the passenger has access to, whether it's flying first-class or visiting the VIP lounge.

Authentication is used to verify that users really are who they represent themselves to be. Once this has been confirmed, authorization is then used to grant the user permission to access different levels of information and perform specific functions, depending on the rules established for different types of users.

While user identity has historically been validated using the combination of a username and password, today's authentication methods commonly rely upon three classes of information:

- What you know: Most commonly, this is a password. But it can also be an answer to a security question or a one-time pin that grants user access to just one session or transaction.
- What you possess: This could be a mobile device or app, a security token, or digital ID card.
- What you are: This is biometric data such as a fingerprint, retinal scan, or facial recognition.

Oftentimes, these types of information are combined using multiple layers of authentication. For example, a user may be asked to provide a username and password to complete an online purchase. Once that's confirmed, a one-time pin may be sent to the user's mobile phone as a second layer of security. Combining multiple authentication methods with consistent authentication protocols, organizations can ensure security as well as compatibility between systems.

Once a user is authenticated, authorization controls are then applied to ensure users can access the data they need and perform specific functions such as adding or deleting information—

based on the permissions granted by the organization. These permissions can be assigned at the application, operating system, or infrastructure levels. Two common authorization techniques include:

- Role-based access controls (RBAC): This authorization method gives users access to information based on their role within the organization. For example, all employees within a company may be able to view, but not modify, their personal information such as pay, vacation time, and 401K data. Yet human resources (HR) managers may be given access to all employees' HR information with the ability to add, delete, and change this data. By assigning permissions according to each person's role, organizations can ensure every user is productive, while limiting access to sensitive information.
- Attribute-based access control (ABAC): ABAC grants users permissions on a more granular level than RBAC using a series of specific attributes. This may include user attributes such as the user's name, role, organization, ID, and security clearance. It may include environmental attributes such as the time of access, location of the data, and current organizational threat levels. And it may include resource attributes such as the resource owner, file name, and level of data sensitivity. ABAC is a more complex authorization process than RBAC designed to further limit access. For example, rather than allowing all HR managers in an organization to change employees' HR data, access can be limited to certain geographical locations or hours of the day to maintain tight security limits.

Procedure:

1. Testing for Default Credentials

Authentication lies at the heart of an application's protection against unauthorized access. If an attacker is able to break an application's authentication function then they may be able to own the entire application.

The following tutorial demonstrates a technique to bypass authentication using a simulated login page from the "Mutillidae" training tool. The version of "Mutillidae" we are using is taken from OWASP's Broken Web Application Project. Find out how to download, install and use this project.

First, ensure that Burp is correctly configured with your browser.

In the Burp Proxy tab, ensure "Intercept is off" and visit the login page of the application you are testing in your browser.

Return to Burp.In the Proxy "Intercept" tab, ensure "Intercept is on".

In your browser enter some arbitrary details in to the login page and submit the request.

The captured request can be viewed in the Proxy "Intercept" tab.Right click on the request to bring up the context menu.Then click "Send to Intruder".

Go to the Intruder "Positions" tab.

Add the "username" and "password" parameter values as payload positions by highlighting them and using the "Add" button.

Change the attack to "Cluster bomb" using the "Attack type" drop down menu.

Go to the "Payloads" tab.
In the "Payload sets" settings, ensure "Payload set" is "1" and "Payload type" is set to "Simple list".
In the "Payload settings" field enter some possible usernames. You can do this manually or use a custom or pre-set payload list.

Next, in the "Payload Sets" options, change "Payload" set to "2".

In the "Payload settings" field enter some possible passwords. You can do this manually or using a custom or pre-set list.

Click the "Start attack" button.

In the "Intruder attack" window you can sort the results using the column headers.In this example sort by "Length" and by "Status".

The table now provides us with some interesting results for further investigation.

By viewing the response in the attack window we can see that request 118 is logged in as "admin".

To confirm that the brute force attack has been successful, use the gathered information (username and password) on the web application's login page.

Account Lock Out

In some instances, brute forcing a login page may result in an application locking out the user account. This could be the due to a lock out policy based on a certain number of bad login attempts etc.Although designed to protect the account, such policies can often give rise to further vulnerabilities. A malicious user may be able to lock out multiple accounts, denying access to a system.In addition, a locked out account may cause variances in the behavior of the application, this behavior should be explored and potentially exploited.

2.Testing for Weak Lock out Mechanism

Account lockout mechanisms should be present within an application to mitigate brute-force login attacks. Typically, applications set a threshold between three to five attempts. Many applications lock for a period of time before a re-attempt is allowed.

Penetration testers must test all aspects of login protections, including challenge questions and response, if present.

Account lockout mechanisms are used to mitigate brute force attacks. Some of the attacks that can be defeated by using lockout mechanism:

- ✓ Login password or username guessing attack.
- ✓ Code guessing on any 2FA functionality or Security Questions.

Account lockout mechanisms require a balance between protecting accounts from unauthorized access and protecting users from being denied authorized access. Accounts are typically locked after 3 to 5 unsuccessful attempts and can only be unlocked after a predetermined period of time, via a self-service unlock mechanism, or intervention by an administrator.

Despite it being easy to conduct brute force attacks, the result of a successful attack is dangerous as the attacker will have full access on the user account and with it all the functionality and services they have access to.

3.Testing for Browser Cache Weaknesses

Browser caching is provided for improved performance and better end-user experience. However, when sensitive data is typed into a browser by the user, such data can also be cached in the browser history. This cached data is visible by examining the browser's cache or simply by pressing the browser's back button.

The goal of poisoning the cache is to make the clients load unexpected resources partially or controlled by the attacker.

The poisoned response will only be served to users who visit the affected page while the cache is poisoned. As a result, the impact can range from non-existent to massive depending on whether the page is popular or not.

4. Testing for Weak Password Policy

Testing for Weak password policy is part of Web Application penetration testing and it is covered in OWASP authentication testing section.

Please refer OWASP testing guide "**Testing for Weak password policy (OTG-AUTHN-007)**" for password policy testing. In this article, i will show you how to automate item number 4.

1. What characters are permitted and forbidden for use within a password? Is the user required to use characters from different character sets such as lower and uppercase letters, digits and special symbols?

2. How often can a user change their password? How quickly can a user change their password after a previous change? Users may bypass password history requirements by changing their password 5 times in a row so that after the last password change they have configured their initial password again.

3. When must a user change their password? After 90 days? After account lockout due to excessive log on attempts?

4. How often can a user reuse a password? Does the application maintain a history of the user's previous used 8 passwords?

5. How different must the next password be from the last password?

6. Is the user prevented from using his username or other account information (such as first or last name) in the password?

Login into application using existing credentials.Capture the request in burp suite proxy

Send the request to intruder

I will be using sequential number with Password@1 as my passwords, from intruder select last character of password and select attack type as pitchfork.

From payload tab select payload type as numbers and fill number range as shown in fig.

Similarly select payload for second position, note the difference from and to number range.

Select payload for third position

This part is very important, from intruder options tab select number of thread as one and throttle it for 5 sec(5000 millisec). Without this configuration there will be a chaos and password will be changed in random orders and logic will break.

Now you can see intruder result with password payload in results

And password changed in response.

5. Testing for Weak Password Change or Reset Functionalities

For any application that requires the user to authenticate with a password, there must be a mechanism by which the user can regain access to their account if they forget their password. Although this can sometimes be a manual process that involves contacting the owner of the website or a support team, users are frequently allowed to carry out a self-service password reset, and to regain access to their account by providing some other evidence of their identity.

As this functionality provides a direct route to compromise the user's account, it is crucial that it is implemented securely. The first step is to gather information about what mechanisms are available to allow the user to reset their password on the application. If there are multiple interfaces on the same site (such as a web interface, mobile application, and API) then these should all be reviewed, in case they provide different functionality.

Once this has been established, determine what information is required in order for a user to initiate a password reset. This can be the username or email address (both of which may be obtained from public information), but it could also be an internally-generated user ID.

Input Validation Testing

Aim :

To familiarize sql injection testing for input validation using burpsuite

Description:

SQL injection testing checks if it is possible to inject data into the application so that it executes a user-controlled SQL query in the database. Testers find a SQL injection vulnerability if the application uses user input to create SQL queries without proper input validation. A successful exploitation of this class of vulnerability allows an unauthorized user to access or manipulate data in the database.

An SQL injection attack consists of insertion or "injection" of either a partial or complete SQL query via the data input or transmitted from the client (browser) to the web application. A successful SQL injection attack can read sensitive data from the database, modify database data (insert/update/delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file existing on the DBMS file system or write files into the file system, and, in some cases, issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

SQL Injection attacks can be divided into the following three classes:

- ✓ Inband: data is extracted using the same channel that is used to inject the SQL code. This is the most straightforward kind of attack, in which the retrieved data is presented directly in the application web page.
- ✓ Out-of-band: data is retrieved using a different channel (e.g., an email with the results of the query is generated and sent to the tester).
- ✓ Inferential or Blind: there is no actual transfer of data, but the tester is able to reconstruct the information by sending particular requests and observing the resulting behavior of the DB Server.

A successful SQL Injection attack requires the attacker to craft a syntactically correct SQL Query. If the application returns an error message generated by an incorrect query, then it may be easier for an attacker to reconstruct the logic of the original query and, therefore, understand how to perform the injection correctly. However, if the application hides the error details, then the tester must be able to reverse engineer the logic of the original query.

About the techniques to exploit SQL injection flaws there are five commons techniques. Also those techniques sometimes can be used in a combined way (e.g. union operator and out-of-band):

- ✓ Union Operator: can be used when the SQL injection flaw happens in a SELECT statement, making it possible to combine two queries into a single result or result set.
- ✓ Boolean: use Boolean condition(s) to verify whether certain conditions are true or false.
- ✓ Error based: this technique forces the database to generate an error, giving the attacker or tester information upon which to refine their injection.
- ✓ Out-of-band: technique used to retrieve data using a different channel (e.g., make a HTTP connection to send the results to a web server).
- ✓ Time delay: use database commands (e.g. sleep) to delay answers in conditional queries. It is useful when attacker doesn't have some kind of answer (result, output, or error) from the application.

Procedure:

Scanning for SQL injection vulnerabilities

If you're using Burp Suite Professional, you can use Burp Scanner to test for SQL injection vulnerabilities:

1. Identify a request that you want to investigate.
2. In **Proxy > HTTP history**, right-click the request and select **Do active scan**. Burp Scanner audits the application.
3. Review the **Issue activity** panel on the **Dashboard** to identify any SQL injection issues that Burp Scanner flags.

Manually fuzzing for SQL injection vulnerabilities

You can alternatively use Burp Intruder to test for SQL injection vulnerabilities. This process also enables you to closely investigate any issues that Burp Scanner has identified:

1. Identify a request that you want to investigate.

2. In the request, highlight the parameter that you want to test and select **Send to Intruder**.
3. Go to the **Intruder > Positions** tab. Notice that the parameter has been automatically marked as a payload position.
4. Go to the **Payloads** tab. Under **Payload settings [Simple list]** add a list of SQL fuzz strings.
    1. If you're using Burp Suite Professional, open the **Add from list** drop-down menu and select the built-in **Fuzzing - SQL wordlist**.
    2. If you're using Burp Suite Community Edition, manually add a list.
5. Under **Payload processing**, click **Add**. Configure payload processing rules to replace any list placeholders with an appropriate value. You need to do this if you're using the built-in wordlist:
    1. To replace the {base} placeholder, select **Replace placeholder with base value**.
    2. To replace other placeholders, select **Match/Replace**, then specify the placeholder and replacement. For example, replace {domain} with the domain name of the site you're testing.
6. Click **Start attack**. The attack starts running in a new dialog. Intruder sends a request for each SQL fuzz string on the list.
7. When the attack is finished, study the responses to look for any noteworthy behavior. For example, look for:
    - Responses that include additional data as a result of the query.
    - Responses that include other differences due to the query, such as a "welcome back" message or error message.
    - Responses that had a time delay due to the query.

   If you're using the lab, look for responses with a longer length. These may include additional products.

Client-side Testing

Aim :

To perform client side testing using burpsuite

Description:

There exist a variety of testing tools that can make it easy to implement client-side A/B testing. Many of them include a WYSIWYG editor that lets you easily change components in a visual editor without needing to reach into the code at all. This type of testing framework makes running tests on the client-side extremely easy and intuitive.

It also makes it possible for marketing teams to run experiments without needing to employ a front-end developer. Not a single line of code needs to be written, not a single actual deployment needs to happen until the experiment is complete. Once that happens, the developers only need to be brought in if the winning variation was one of the alternatives: otherwise, the alternate variations can simply be scrapped and a new experiment can begin.

Another benefit of client side testing is the additional user data available. Because the variation hasn't been decided until the page loads in the visitor's browser, more data can be gathered about the user to determine which variation to serve. On the other hand, server side testing has less user data to work on, so it's less able to segment users.

There are some drawbacks to client side testing, though. The most common is that, since the test is implemented using client side JavaScript, the user experience can suffer. Depending on the specific implementation, the page load time can get higher as it takes a second to determine what variation the user should see, or the user could see a "flickering" effect on the webpage as the original version is displayed before the test variation displays in its place. While load time issues are harder to fix, flickering can be tactically improved by only using client-side tests for elements below the fold.

A client side A/B test, like any other A/B test, begins with a hypothesis. "We think changing the color of this CTA button will improve conversion rate" is a classic example. Once the hypothesis is determined, the variations can be created using the visual editor and displayed to users using the testing tool.

After the test is complete, significance is calculated, and the winning variation is determined, it’s time to implement the winner. This is a key difference between client-side and server-side testing: when an alternate version wins in an experiment, the actual deployment process is slower than in client-side testing because the variations have not yet been built. With a server-side test, the variations have to be built in order to be tested, so the rollout process is extremely fast. However, on the converse, if a test fails to produce significant results, the variations that cost developer effort for a server-side test will have to simply be scrapped, whereas no developer effort went into creating variations in a client-side test. There is less cost to doing more experiments if they’re done on the client-side.

Procedure:

1.Testing for Cross-Site Scripting:

Reflected cross-site scripting vulnerabilities arise when data is copied from a request and echoed in to the application’s immediate response in an unsafe way. An attacker can use the vulnerability to construct a request which, if issued by another application user, will cause JavaScript code supplied by the attacker to execute within the user's browser in the context of that user's session with the application. The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.

First, ensure that Burp is correctly configured with your browser.With intercept turned off in the Proxy "Intercept" tab, visit the web application you are testing in your browser.

Visit the page of the website you wish to test for XSS vulnerabilities.

Return to Burp.In the Proxy "Intercept" tab, ensure "Intercept is on".

Enter some appropriate input in to the web application and submit the request.

The request will be captured by Burp. You can view the HTTP request in the Proxy "Intercept" tab.You can also locate the relevant request in various Burp tabs without having to use the intercept function, e.g. requests are logged and detailed in the "HTTP history" tab within the "Proxy" tab.Right click anywhere on the request to bring up the context menu.Click "Send to Repeater"

Go to the "Repeater" tab.

Here we can input various XSS payloads into the input field.We can test various inputs by editing the "Value" of the appropriate parameter in the "Raw" or "Params" tabs.

A simple payload such as **<s>** can often be used to check for issues.

In this example we have used a payload that attempts to perform a proof of concept pop up in our browser.

Click "Go".

We can assess whether the attack payload appears unmodified in the response. If so, the application is almost certainly vulnerable to XSS.You can find the response quickly using the search bar at the bottom of the response panel.The highlighted text is the result of our search.

Right click on the response to bring up the context menu.Click "Show response in browser" to copy the URL.You can also use "Copy URL" or "Request in browser".

In the pop up window, click "Copy".

Copy the URL in to your browser's address bar.
In this example we were able to produce a proof of concept for the vulnerability.

2. Testing for JavaScript Execution

If you use Burp Suite for testing applications then there are multiple ways to gather all the JavaScript files in an application. Navigate through an application while the traffic is being sent through Burp proxy. Once you are done with the navigation, you can use Burp's tool-set to extract all the JavaScript files.If you are using Burp Suite Community Edition, you can navigate to proxy > HTTP history and use the display filters to only display the JavaScript files used by the application. You can also copy the URLs for all the JavaScript files displayed.

Burp display filters to display only JavaScript files for a given application

Copy the URLs for all the JavaScript file displayed after filtering

If you are using Burp Suite Professional, You can not only copy the URLs for all the JavaScript files in an application but also export all the scripts. Under Target > Site map right click on the site of interest and select Engagement tools > Find scripts . Using this feature you can export all the scripts in that application and also copy URLs.

Burp “Find Scripts” to identify all the JS files on an application

Burp “Find scripts” can export all the scripts, not just URLs

3. Testing for HTML Injection

This vulnerability occurs when the user input is not correctly sanitized and the output is not encoded. An injection allows the attacker to send a malicious HTML page to a victim. The targeted browser will not be able to distinguish (trust) the legit from the malicious parts and consequently will parse and execute all as legit in the victim context.

There is a wide range of methods and attributes that could be used to render HTML content. If these methods are provided with an untrusted input, then there is an high risk of XSS, specifically an HTML injection one. Malicious HTML code could be injected for example via innerHTML, that is used to render user inserted HTML code. If strings are not correctly sanitized the problem could lead to XSS based HTML injection. Another method could be document.write()

When trying to exploit this kind of issues, consider that some characters are treated differently by different browsers. For reference see the DOM XSS Wiki.The innerHTML property sets or returns the inner HTML of an element. An improper usage of this property, that means lack of sanitization from untrusted input and missing output encoding, could allow an attacker to inject malicious HTML code.