



Main Challenge

Badge Manipulation

✓ 6) Badge Manipulation

Difficulty: 🌲🌲🌲🌲🌲

Bypass the authentication mechanism associated with the room near Pepper Minstix. A sample employee badge is available. What is the access control number revealed by the door authentication panel? For hints on achieving this objective, please visit Pepper Minstix and help her with the **Yule Log Analysis** Cranberry Pi terminal challenge.



Hint Challenge

The Yule Log Analysis

Cranberry Pi terminal challenge



Hint Challenge

The Yule Log Analysis

Cranberry Pi terminal challenge

- Pepper Minstix at 2nd floor go right into corridor until end then left continue forward until you find him

Hi, I'm Pepper Minstix.

Have you heard of **password spraying**? It seems we've been victim.

We fear that they were successful in accessing one of our Elf Web Access accounts, but we don't know which one.

Parsing through .evtx files can be tricky, but there's a **Python script that can help you convert it into XML for easier grep'ing.**



Password Spraying

<https://securityweekly.com/2017/07/21/tsw11/>

if video didn't work go here :

https://www.youtube.com/watch?v=ZIOw_xfqKM



Terminal Screen

```
lXMMMx;.....XMMMMK,,coddec10kxoc,.
lk:oNMMMx;.....XMMWN00o:.....MMMMMMoc;'
.0l,,dNMMx;.....XNNWMMmk;.....MMMMMx;.....:;
.K;.....xWMX;.....Kx:kwMMMk;.....MMMM0;.....:k'
.XklooooddolckwN:l0;.....;kWM0;.....MMMN;.....c0wMMd
;oooc;.....cMMMMMMxk00;.....:0MM0;.....MMWc;.....lKMMMMWko
;0MMWl;.....cMMMMM0;.....cc;.....:0M0;.....MMd;.....oXMMWkxc;.....c
c0dXMMWl;.....cMMMx;.....:xxo;.....cK0;.....M0;.....xNWkxc;.....:
.0l,,oNMMWl;.....cMMW;.....dXMMWNWx0dc;.....lxcX:x0xc;.....:
;0;.....dNMW0;.....cMMl;.....;xNMMMW0kkkkkkddxdddxxxxxxxxxxxxxxxxxxo
.wl;.....dWMO;.....cMMx;.....:0wMMW0xc;.....c;.....d0kcK;.....kc:oK0NMMMMMMMMMMd
KMMWx0dl;.....;xWd;.....cM0;.....l0MW0dc;.....lkWWk;.....0W;.....x0;.....;ld0xWMMMM'
'MMMMMMMMMMM0ko;.....;kdcN;o00dc;.....0x;.....dMW;.....XWk;.....:okk
cNKKKKKKKKKKKKKKkoodxxdccccccccccccccco;.....wMW;.....XWk;.....l
x;.....;cdko0ldld0KwMMMMMMMMMMMMMMx;.....xMMW;.....;XMMWx;.....;c
.K;.....;cd0WKL;xN,oXo;.....:ok0NMMMMMMc;.....0MMW;.....;KMMMNd;l'
dl;.....cx0WMM0c;.....lMN;.....oMxL;.....;ld0X0';.....dMMMMW;.....;KMMMK;
OoxKwMMMMWk;.....NMN;.....lWMKc;.....ldcLwMMMMW;.....:o0L.
0MMMMNx;.....KMMN;.....lwMM0c;.....l. ....cdk000ccc;.....
cwXo;.....KMMN;.....cwMM0;.....c;
.KC;.....:MMMMN;.....dMMMMWk'
```

I am Pepper Minstix, and I'm looking for your help.
Bad guys have us tangled up in pepperminty kelp!
"Password spraying" is to blame for this our grinchy fate.
Should we blame our password policies which users hate?

Here you'll find a web log filled with failure and success.
One successful login there requires your redress.
Can you help us figure out which user was attacked?
Tell us who fell victim, and please handle this with tact...

Submit the compromised webmail username to
runtoanswer to complete this challenge.
elf@9df3301ba913:~\$





Solution

1. Recommended watch KringleCon - Beau Bullock' talk about password spraying:

<https://www.youtube.com/watch?v=khwYjZYpzFw>

2. Let's list all files and directories using `ls` command :

```
ls
elf@5fd6242222c:~$ ls
evtx_dump.py  ho-ho-no.evtx  runtoanswer
elf@5fd6242222c:~$
```

Here you can see two files :

`evtx_dump.py` Python script that can help you convert evtx into XML for easier grep'ing

`ho-ho-no.evtx` Web log filled with failure and success in .evtx format

3. Let's convert `ho-ho-no.evtx` to readable xml format using python script `evtx_dump.py`:

```
python2 evtx_dump.py ho-ho-no.evtx > ho-ho-no.xml
```

4. Use `ls` command again to check if the file converted :

```
elf@5fd6242222c:~$ ls
evtx_dump.py  ho-ho-no.evtx  ho-ho-no.xml  runtoanswer
elf@5fd6242222c:~$
```

5. View `ho-ho-no.xml` file using `cat` tool :

```
cat ho-ho-no.xml
```

```
elf@fb2d5a3489f7:~$ cat ho-ho-no.xml
<?xml version="1.1" encoding="utf-8" standalone="yes" ?>

<Events>
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event"><System><Provider Name="Microsoft-Windows-Security-Auditing" Guid="{54849625-5478-4994-a5ba-3e3b0328c30d}"></Pr
ovider>
<EventID Qualifiers="">4647</EventID>
<Version>0</Version>
<Level>0</Level>
<Task>12545</Task>
<Opcode>0</Opcode>
<Keywords>0x8020000000000000</Keywords>
<TimeCreated SystemTime="2018-09-10 12:18:26.972103"></TimeCreated>
<EventRecordID>231712</EventRecordID>
<Correlation ActivityID="{fd18dc13-48f8-0001-58dc-18fdf848d401}" RelatedActivityID=""></Co
rrelation>
<Execution ProcessID="660" ThreadID="752"></Execution>
<Channel>Security</Channel>
<Computer>WIN-KCON-EXCH16.EM.KRINGLECON.COM</Computer>
<Security UserID=""></Security>
</System>
<EventData><Data Name="TargetUserSid">S-1-5-21-25059752-1411454016-2901770228-500</Data>
<Data Name="TargetUserName">Administrator</Data>
<Data Name="TargetDomainName">EM.KRINGLECON</Data>
<Data Name="TargetLogonId">0x00000000000969b09</Data>
</EventData>
</Event>
```

Copy the xml text to notepad for easier lookup .



6. The evtx is a Event Viewer file so we will look for windows login event codes:

4624 An account was successfully logged on

4625 An account failed to log on

You can find more about windows login event codes here :

> <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx>

> <https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4625>

> <https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4624>

7. Now let's use **grep** command to filter results , Look for failed attempts with code 4625 and export it to new file for easier analysis

```
grep -A 35 "4625" ho-ho-no.xml > 4625.xml
```

-B, --before-context=NUM print NUM lines of leading context

-A, --after-context=NUM print NUM lines of trailing context

8. Let's filter IP address of machine from which failed login attempts was performed :

```
grep "IpAddress" 4625.xml
```

```
elf@20388254ab49:~$ grep "IpAddress" 4625.xml
<Data Name="IpAddress">10.158.210.210</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
<Data Name="IpAddress">172.31.254.101</Data>
```

We will notice this IP "172.31.254.101" as main source of failed logins, so we will mark this as the attacker ip .

9. Let's looking for successful attempts with code 4624 and export it to separated xml file for easier analysis

```
grep -A 43 "4624" ho-ho-no.xml > 4624.xml
```

10. Let's filter events by attacker ip address "172.31.254.101" :

```
grep -B 13 "172.31.254.101" 4624.xml
```

```
elf@20388254ab49:~$ grep -B 13 "172.31.254.101" 4624.xml
<Data Name="TargetUserName">minty.candycane</Data>
<Data Name="TargetDomainName">EM.KRINGLECON</Data>
<Data Name="TargetLogonId">0x00000000114a4fe</Data>
<Data Name="LogonType">8</Data>
<Data Name="LogonProcessName">Advapi </Data>
<Data Name="AuthenticationPackageName">Negotiate</Data>
<Data Name="WorkstationName">WIN-KCON-EXCH16</Data>
<Data Name="LogonGuid">{d1a830e3-d804-588d-aea1-48b8610c3cc1}</Data>
<Data Name="TransmittedServices">--</Data>
<Data Name="LmPackageName">--</Data>
<Data Name="KeyLength">0</Data>
<Data Name="ProcessId">0x00000000000019f0</Data>
<Data Name="ProcessName">C:\Windows\System32\inetssrv\w3wp.exe</Data>
<Data Name="IpAddress">172.31.254.101</Data>
```

So we have successfully identified which user was attacked : **minty candycane**

11. Enter the name “**minty candycane**” into runtoanswer

```
MMN0d1110MMklxMMMMMMMMMMMMMMN0x01110kkWMMMMMMMMMMMMMM011KMMkl111x0NMM
MW0x01110lx0x1lxMMNxd0MMMMMMMMMMMMMMx0MMMMMMMMMMMMMMWkdKMM01100d101110KMM
M011dkkWMNkl111dNMKlloMMMMN01ok0NMx10MX0x0lxMMMMX111NMX01100NMMNKK010XM
MMWMMWMMW0d1110kdlx110WMMX1111110010011111WMMX111x0lx0111x0NMMNMMWMM
MMMN0k0111x0NMMW001110NMKlloN0k01110Kkl111WMMXkl111dKMMW0d1110KkWMM
MM0111d0KwMMMMMk0110x00d1dx110MMMMx10MMMMN111x00x00x11100MMMMWKK0111KMM
MMW0KNMMMMMMMMMMKk0XWMMW001110NMMx10MWXkl111dXMMMMWKKkXMMMMMMMMMMX0KwMM
MMMMMMMMMMMMMMMMMMMMMMW0x0110x00d10kdlx00x00x1110KkWMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMW01110WMMMMMMNkl1110WMMMMMMN11111xMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMN0x1110Kk0x00kdlx00kK0x0110KkWMMMMMMMMMMMMMMMMMMMM
MMWKKWMMMMMMMMMMKk0XMMMMMMW01110XMMx10MWKkl111dKwMMMMW00XMMMMMMMMMMNKKMM
MMkl11d0XWMMMMMkl110k00x00d110MMMMx10MMMMN111x00k00x01100MMMMWKKd111KMM
MMMN0x0110x0NMMW001110NMKlloNKK0111d0Kkl111WMMXkl111dKwMMX0x1110k0NMM
MMWMMWMMWKK0111dKx10d110WMMX1111110010011111WMMX111x00x0111d0XMMMMWMM
M011d0XWMMNkl111dNMKlloMMMMN010x0XMx10W0X110MMMMX111NMX01100WMMWKKd10XM
MW0x1110d1d0x1lxMMNxd0MMMMMMMMMMMMMMx10MMMMWMMWMMWxdxWMM011kk0d1110KwM
MMN0x11110MMklxMMMMMMMMMMMMMMMMMMNk01110KkWMMMMMMMMMMMMMMMMMM011KMMkl111KwMM
Mkl1d0X011KMMklxMMMMMMMMMMMMMMMMMMx111001001110MMMMMMMMMMMMMMMMMM011KMMkl1xKk010M
MMWMMMMd11KMMklxMMMMMMMMMMMMMMMMMMX00XMx10W000NMMMMMMMMMMMMMM011KMM011KwMMWMM
MMMMMMMMMMNKKMMklxMMMMMMMMMMMMMMMMMMN01dKwMMMMMMMMMMMMMMMMMM011KMMKKWMMMMMM
MMMMMMMMMMMMMMXkxXMMMMMMMMMMMMMMWKK0111111d0XMMMMMMMMMMMMMM0xkWMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMX0x1110k0x1k0x0110x0NMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMX0111d0XMMx10MMW0d111dWMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMW00KwMMWKK0111d0XWMMN0KkMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMkl11001001100MMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMX00XMx10WKOONMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMkl0MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMXMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
```

Silly Minty Candycane, well this is what she gets.
 "Winter2018" isn't for The Internets.
 Passwords formed with season-year are on the hackers' list.
 Maybe we should look at guidance published by the NIST?

Congratulations!





Well, that explains the odd activity in Minty's account. Thanks for your help!

All of the Kringle Castle employees have these cool cards with QR codes on them that give us access to restricted areas.

Unfortunately, the badge-scan-o-matic said my account was **disabled** when I tried scanning my badge.

I really needed access so I tried scanning several QR codes I made from my phone but the scanner kept saying "User Not Found".

I researched a **SQL database error from scanning a QR code with special characters in it** and found it may contain an injection vulnerability.

I was going to **try some variations I found on OWASP** but decided to stop so I don't tick-off Alabaster.



Barcode Creation > Creating QR barcodes

<https://www.the-qrcode-generator.com/>



SQL Injection

https://www.owasp.org/index.php/SQL_Injection_Bypassing_WAF#Auth_By-pass





Main Challenge

Badge Manipulation

📍 2nd floor from Pepper Minstix go to the end of the corridor then left until you reach the door on your right.



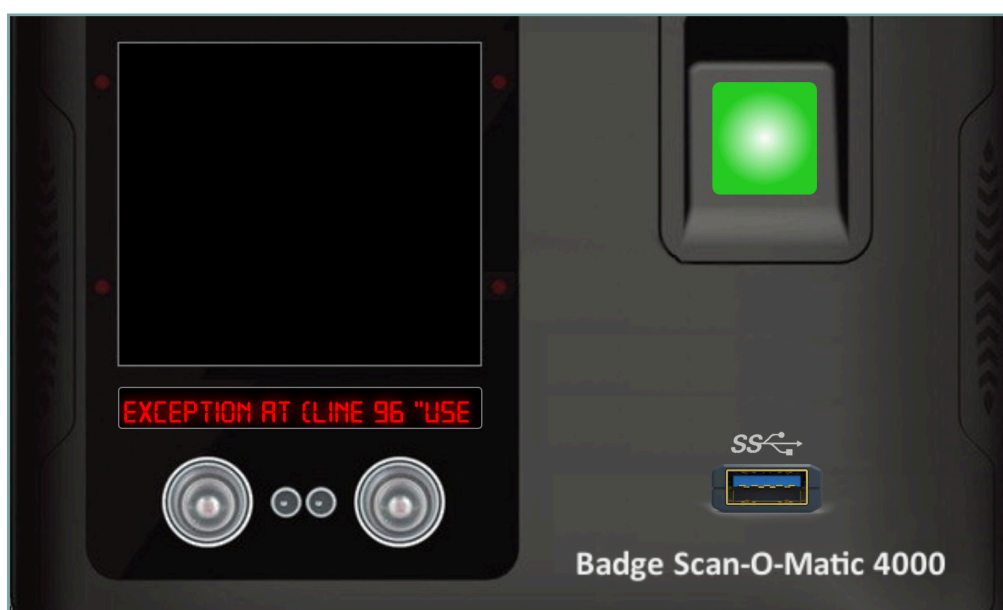
Solution

1. As OWASP website on how to bypass panels , Let's try some code variations :
or 1-- -'
2. Let's create the QR barcode badge :
 - Go to the QR barcode generator : <https://www.the-qrcode-generator.com/>
 - Write the sample code in Free Text box > Export the QR to PNG file and save it :

The screenshot shows the 'the-qrcode-generator.com' interface. The 'FREE TEXT' tab is active. The text input field contains 'or 1-- -''. To the right, a 'SAVE' button is visible. Below the input, the 'Static QR Code' is displayed as a black and white QR code.

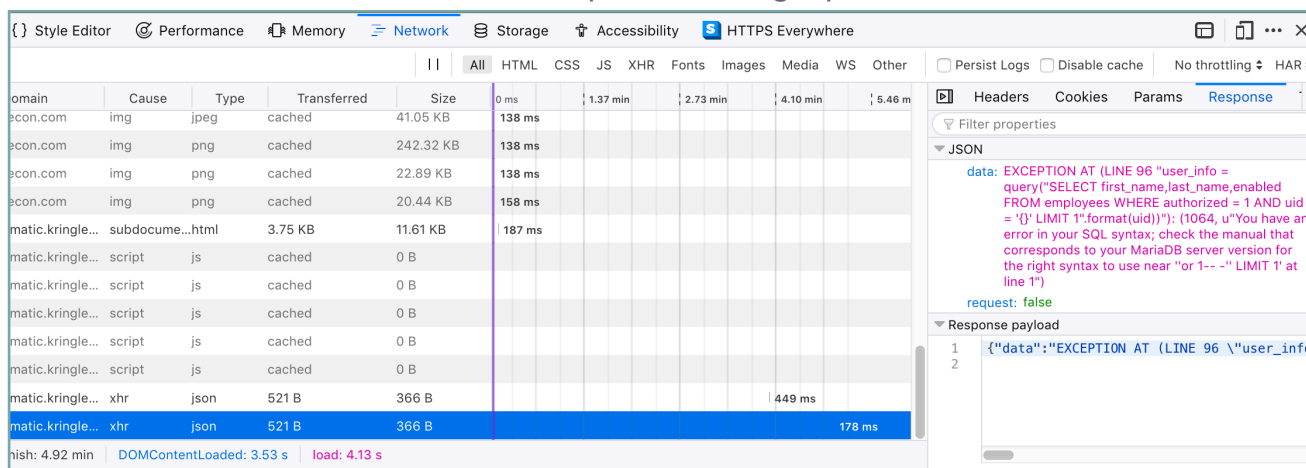
3. Go to the authentication panel then click usb port and upload the created qr png file :
 - > if you get this error message "resource_id not set in cookie" , try to login from different browser where third party cookies enabled.

After you upload the badge png file , you will get this error message :



To view the full message using the following method :

- In Firefox : Right click > inspect element > network tab > reload button
- Then re-upload the qr file again
- Select last loaded item > Select response from right panel



EXCEPTION AT (LINE 96 "user_info = query("SELECT first_name,last_name,enabled FROM employees WHERE authorized = 1 AND uid = '{ }' LIMIT 1".format(uid))"): (1064, u"You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'or 1-- -' LIMIT 1' at line 1")

4. This error message give us the query used to validate the badge :

```
SELECT first_name,last_name,enabled FROM employees WHERE authorized = 1
AND uid = '{ }' LIMIT 1
```

Where `uid` is our badge code and the interesting `enabled`, `authorized` variables.

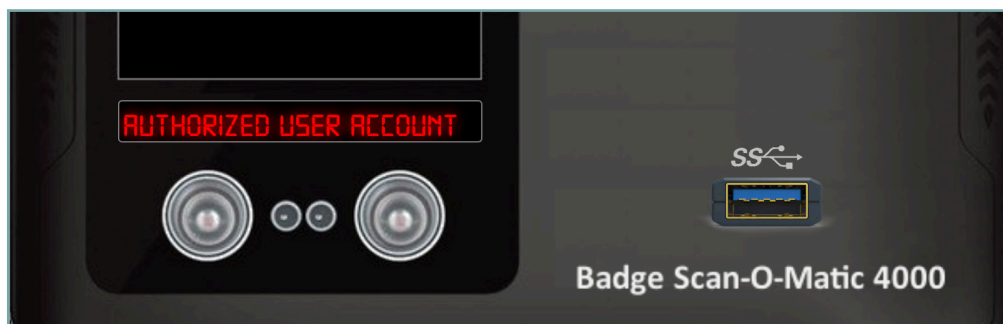
5. Let's reshape our code and regenerate our qr badge :

```
` OR 1 = 1 #
```

We will use `or` to select authorized accounts regardless the `uid`, also will use `#` at the end to inline comment the rest of the code because we need to ignore formatting of `uid`.

As you can see will get the following error message :

Authorized User Account Has Been Disabled!



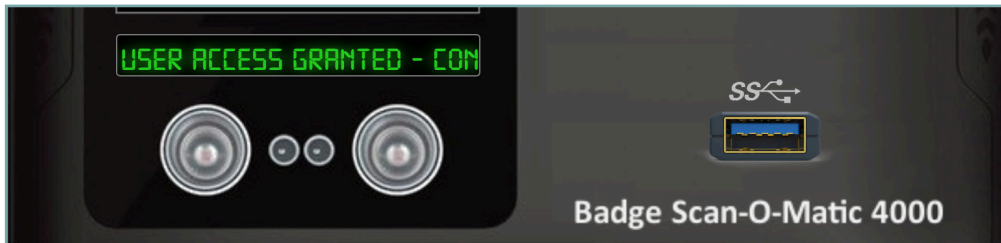
Badge Manipulation



6. Let's reshape our code again and regenerate our qr badge :

```
` OR enabled = 1 #
```


We will use or to select enabled accounts regardless the uid or authorized accounts , Also will try enabled with 1 then true to test the values .





Successfully opened the door and also got the access control number :

User Access Granted - Control number 198807

 Go to your Badge > Objectives > Enter 19880715



6) Badge Manipulation

Difficulty:  

Bypass the authentication mechanism associated with the room near Pepper Minstix. A sample employee badge is available. What is the access control number revealed by the door authentication panel? For hints on achieving this objective, please visit Pepper Minstix and help her with the **Yule Log Analysis Cranberry Pi terminal challenge**.



