SecTalks London 0x18
sectalks.slack.com
(#sectalks_lon)

Shortname Disclosure Vulnerability

About us

Different skill sets welcome: ops, devs, sysadmins, security researchers etc.
{Perth,Sydney,Brisbane,Melbourne,Canberra,Adelaide,Hobart}, **Australia**; São Paulo, **Brazil**; Beijing, **China**; Ljubljana, **Slovenia**; Christchurch, **New Zealand**

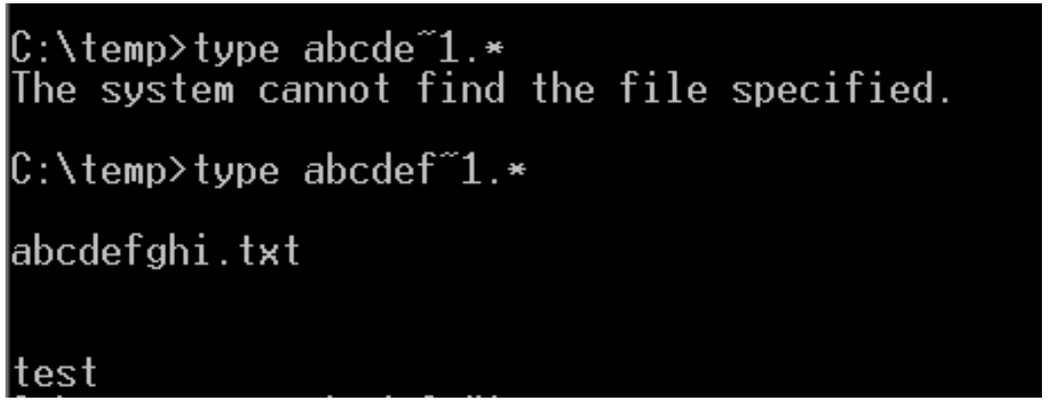
Strictly vendor neutral, no bullshit policy

About me

- Sysadmin, Security Engineer & Internal Penetration Tester
- n00b
- @egre55

Short (8.3) file/folder names

- Windows maintains a shortname (SFN) for every long filename (≥ 9 chars)
- 8.3 convention states that file names can be up to 8 chars, followed an extension of up to 3 chars
- In Windows, the SFN is truncated to 6 chars, followed by a tilde
- Convention used by DOS, but supported in modern Windows OS.



Short (8.3) file/folder names

- The kernel also maintains a record of short folder names for every long folder

  C:\Program Files (x86)\ can be referred to as C:\Progra~2

- ~1, ~2 notation allows for duplicate SFNs to be uniquely identified
- ? is used to substitute a single character, * is used to substitute one or more proceeding or following characters

```
C:\temp>cd thisi?~1

C:\temp\thisisareallylongfoldername>cd ..

C:\temp>cd th?sis~1

C:\temp\thisisareallylongfoldername>cd ..
```

The vulnerability

- Discovered by Soroush Dalili (@irsdl)
- When a query for a shortname is sent in an OPTIONS request, IIS will reply 404 for an existing file and 200 for a non-existing file
- The wildcard can reveal the presence of proceeding characters, which allows for SFNs in IIS to be brute forced

```
root@kali:/# curl -v -X OPTIONS "http://172.16.249.128/idontexist*~2.*" 2>&1 | grep "HTTP/
> OPTIONS /idontexist*~2.* HTTP/1.1
< HTTP/1.1 200 OK
root@kali:/# curl -v -X OPTIONS "http://172.16.249.128/inde*~1.*" 2>&1 | grep "HTTP/1.1"
> OPTIONS /inde*~1.* HTTP/1.1
< HTTP/1.1 404 Not Found
```

https://soroush.secproject.com/downloadable/iis_tilde_shortname_disclosure.txt

The vulnerability

- This only applies to specific IIS parsable file types, but there are quite a few:

asa, asax, ascx, ashx, asmx, asp, aspx, browser, compile, config, cs, csproj, disco, dsdgm, dsprototype, htm, html, licx, master, msgx, resources, resx, sdm, sdmDocument, sitemap, skin, soap, vsdisco, webinfo, etc…

- This vulnerability is useful, because when dirbusting, typically only common file extensions are used (asp, apsx, htm, html, txt, bak), along with known words
- The IIS tilde enumeration vulnerability can be used in conjuction with normal dirbusting in order to achieve better enumeration
- It can be used to detect unusual and interesting files, folders and extensions that dirbusting would miss, e.g. _secre~1.asp, z_uplo~1

Exploit development

- There are a few IIS shortname scanners available, but no Metasploit module
- I come from a mostly scripting background, wanting to improve my programming
- My buddy @MinatoTW already started work on a Metasploit module (Ruby), and let me tag along - thanks Minato for your patience / guidance! :)

The problem

- Assuming a charset of 50 (alpha, numbers, special chars), using Ruby's repeated permutations algorithm to identify just **one** folder (privat~1) would require 50^6 requests (15625000000)
- Ok, that's not possible, we need a way to optimise this process and reduce the number of requests
- We can use traversal, for each found char, send another 50 requests to find next char, append found char, repeat. This results in 50*6 requests, better!
- Reduction of charset (/*c*~1.*) is another good technique to minimise requests: 404 if there is a "c" is present. Do the same for extension (/*~1.*c*) and duplicates (/*~c.*).
- Assuming charset is reduced to 15 chars, this results in just 90 requests per found folder, much better!

The problem

- Threading: Ruby has a limit to the number of threads and is not thread safe
- For a variable *X* if there are multiple threads accessing the same variable this causes confusion
- To solve this we used queues to provide syncronization
- We can add our items to it, add a thread to process it, and this won't be shared by any other thread

Metasploit module

- Run msftidy.rb before submitting a module to ensure space type consitency, remove extraneous spaces
- Add documentation along with exploit in the initial PR
- The r7 guys are really knowledgable and advise on improvements
- Hoping to land it soon!

```
msf > use auxiliary/iis_shortname_scanner
msf auxiliary(iis_shortname_scanner) > set RHOST 172.16.249.128
RHOST => 172.16.249.128
msf auxiliary(iis_shortname_scanner) > check
[+] 172.16.249.128:80 The target is vulnerable.
msf auxiliary(iis_shortname_scanner) > run

[*] Scanning in progress...
[+] Directories found
http://172.16.249.128/aspnet~1
http://172.16.249.128/secret~1
[+] Files found
http://172.16.249.128/web~1.con
http://172.16.249.128/index~1.htm
http://172.16.249.128/upload~1.asp
http://172.16.249.128/upload~2.asp
[*] Auxiliary module execution completed
msf auxiliary(iis_shortname_scanner) >
```

https://github.com/rapid7/metasploit-framework/pull/10558

Remediation

- Disable creation of 8.3 short names on IIS server

  At registry key HKLM\SYSTEM\CurrentControlSet\Control\FileSystem, set NtfsDisable8dot3NameCreation DWORD with a value of 1

  OR

  fsutil.exe behavior set disable8dot3 1

https://support.microsoft.com/en-us/help/121007/how-to-disable-8-3-file-name-creation-on-ntfs-partitions

The good

- Git is pretty fun after initial learning curve
- Dev / making stuff is awesome
- Metasploit supports external python modules – next challenge!

The bad

- Coding 5 minutes when tired = 50 mins the next day fixing mistakes
- Slightly over-aggressive threading, meant some versions were effectively Nation State DoS cyber weapons