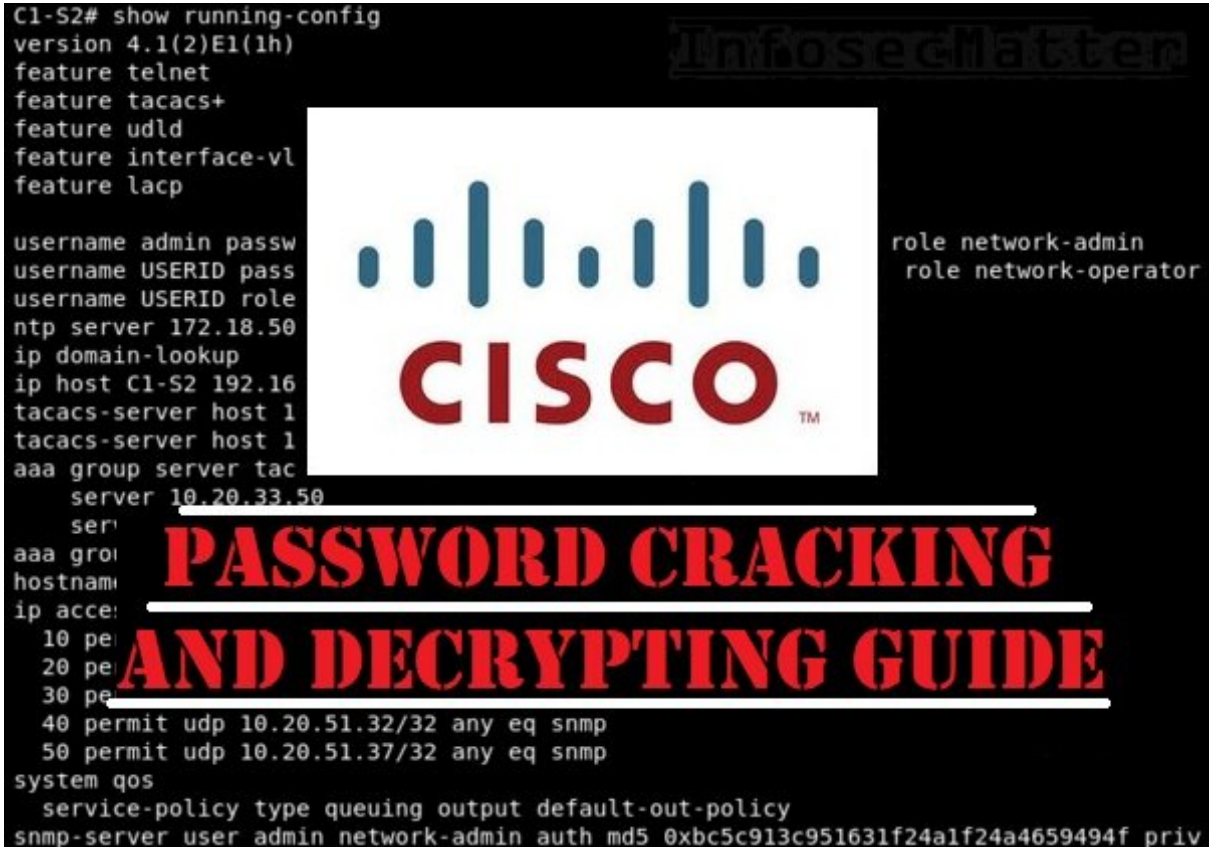
Cisco Password Cracking and Decrypting Guide

2020-03-16



In this guide we will go through Cisco password types that can be found in Cisco IOS-based network devices. We will cover all common Cisco password types (0, 4, 5, 7, 8 and 9) and provide

SEARCH OUR SITE

Search

FOLLOW US

Github | Twitter | Facebook

Enter your email address:

Subscribe

CATEGORIES

Exploitation (3)

Network Security (2)

Penetration Testing (14)

Tools (5)

Vulnerability Assessment (4)

instructions on how to decrypt them or crack them using popular open-source password crackers such as John the Ripper or Hashcat.

Table Of Contents [hide]

Introduction

During penetration tests, it is not uncommon to come across a configuration file of a Cisco network device. It may be a configuration backup found laying somewhere on some computer in the network. It may be a console log output (e.g. from PuTTY) containing Cisco configuration snippets. Or we may just flat out break into some Cisco device configured with default credentials.

The first thing attackers do after they gain access to a Cisco device is that they pull current configuration from the device either by running `show running` or `show running-config` command. The attackers are typically looking for sensitive information such as stored credentials, SNMP community strings, network configuration details and so on.

Credentials are naturally the most interesting thing to look for and over the years Cisco has developed number of different methods for storing passwords in their devices. Hence the name **Cisco password type**.

Common Cisco password types

In the following sections, we will go through all these password types by order from the **least secure** (most easiest to crack) to the **most secure** (hardest to crack):

Cisco Password	Crackability	Best speed	John the Ripper	Hashcat
Type 0	instant	instant	n/a	n/a

RCE on Windows from Linux Part 3: Pass-The-Hash Toolkit

Minimalistic SMB login bruteforcer

RCE on Windows from Linux Part 2: CrackMapExec

Type 7	instant	instant	n/a	n/a
Type 4	easy	26.4 million per second	`--format=Raw-SHA256`	`-m 5700`
Type 5	medium	1.2 million per second	`--format=md5crypt`	`-m 500`
Type 8	hard	11.6 thousand per second	`--format=pbkdf2-hmac-sha256`	`-m 9200`
Type 9	very hard	1.8 thousand per second	`--format=scrypt`	`-m 9300`

RCE on Windows from Linux Part 1: Impacket

Disclaimer: All examples and speed measurements in this article were produced on a standard modern laptop equipped with a GPU and 4 CPU cores.

Let's jump right to it.

Cisco type 0 password

Cisco password type 0 is basically clear text password. There is no encryption nor obfuscation. It is the oldest and the most insecure method of storing passwords in Cisco devices. It should never be used.

The following example shows type 0 password found in a Cisco configuration:

```
username admin privilege 15 password 0 P@ssw0rd
```

As you can see, there is really nothing to crack or decrypt. We can clearly see that the **admin** user has a password of **P@ssw0rd**.

Cisco type 7 password

This password type uses Vigenère cipher which is essentially a simple alphabetical substitution encryption. The algorithm is reversible and thus it can be deciphered instantly into a plain text without any need for cracking.

The following example shows type 7 password found in a Cisco configuration:

```
username admin privilege 15 password 7 0236244818115F3348
```

Decrypt Cisco type 7 password

There are number of freely available tools for decrypting type 7 password. Here are some examples:

- https://www.question-defense.com/2011/08/17/perl-script-to-decode-cisco-type-7-password-hash
- https://gist.github.com/jayswan/1927995
- https://github.com/theevilbit/ciscot7

For instance, to decrypt the above type 7 password using Ciscot7 Python script, simply run:

```
wget https://raw.githubusercontent.com/theevilbit/ciscot7/master/ciscot7.py
python ciscot7.py -d -p 0236244818115F3348
```

```
root@kali:~# python ciscot7.py -d -p 0236244818115F3348
Decrypted password: P@ssw0rd
root@kali:~#
```

We can instantly see that the password is **P@ssw0rd**.

There are also numerous decrypters online for this type of password. But we strongly discourage using any them in order to avoid disclosing sensitive customer information (credentials) to a third party.

Go back to top.

Cisco type 4 password

This password type was designed around 2013 and the original plan was to use PBKDF2 (Password-Based Key Derivation Function version 2) algorithm. But due to an implementation issue, it somehow ended up being a mere single iteration of SHA256 without salt.

The following example shows type 4 password found in a Cisco configuration:

```
username admin secret 4 ds4zcEBHQMiiscBff5JmSaUctdI8fVdmGU18HAtxOCw
```

Decrypt Cisco type 4 passwords with John

John the Ripper recognizes this password type as **Raw-SHA256**. To crack it, we have to first convert it to the following john friendly format and save it in a file:

```
admin:ds4zcEBHQMiiscBff5JmSaUctdI8fVdmGU18HAtxOCw
```

Then we can crack it like this using a dictionary, for example:

```
john --format=Raw-SHA256 --wordlist=/usr/share/wordlists/rockyou.txt --fork 4 hashes.txt
```

Note that since we have 4 CPU cores, we can run john in 4 instances using `--fork` parameter:

```
root@kali:~# grep -c 'processor' /proc/cpuinfo
4
root@kali:~# john --format=Raw-SHA256 --wordlist=/usr/share/wordlists/rockyou.txt --fork=4 hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 128/128 AVX 4x])
Node numbers 1-4 of 4 (fork)
Each node loaded 1/4 of wordfile to memory (about 33 MB/node)
Press 'q' or Ctrl-C to abort, almost any other key for status
1 0g 0:00:01:10 0.00% (ETA: 20:07:59) 0g/s 6568Kp/s 6568Kc/s 6568KC/s 18Davemia..18Dave4575
2 0g 0:00:01:09 0.00% (ETA: 20:17:17) 0g/s 6565Kp/s 6565Kc/s 6565KC/s 18Helima..18Heaven6989
3 0g 0:00:01:10 0.00% (ETA: 20:05:00) 0g/s 6587Kp/s 6587Kc/s 6587KC/s 18May11day..18Maxy6699
4 0g 0:00:01:09 0.00% (ETA: 20:11:55) 0g/s 6598Kp/s 6598Kc/s 6598KC/s 18Elmer0..18Eliafonseca
2 0g 0:00:01:20 0.00% (ETA: 20:11:58) 0g/s 6611Kp/s 6611Kc/s 6611KC/s 55Pansytas..55Panlarb
1 0g 0:00:01:21 0.00% (ETA: 20:00:14) 0g/s 6616Kp/s 6616Kc/s 6616KC/s 55K10378..55June241968
3 0g 0:00:01:21 0.00% (ETA: 20:23:29) 0g/s 6621Kp/s 6621Kc/s 6621KC/s 54Ljnh1234..54Ljbljc
4 0g 0:00:01:20 0.00% (ETA: 20:03:46) 0g/s 6663Kp/s 6663Kc/s 6663KC/s 54Reccute..54Rebirth165
!P@ssw0rd        (admin)
```

From the above screenshot we can see that the **average speed** is around **26.4 million** password attempts per second.

Decrypt Cisco type 4 passwords with Hashcat

Hashcat recognizes this password type as hash mode **5700**. To crack it, we can keep using the same john friendly format Then we can crack it like this using a dictionary, for example:

```
hashcat -m 5700 --username -O -a 0 hashes.txt /usr/share/wordlists/rockyou.txt
```

Note that by using the `-O` parameter (optimized kernels), we will greatly increase the speed. But it will also limit the password length to 31 characters.

```
root@kali:~# hashcat -m 5700 -O -a 0 --quiet --username hashes.txt /usr/share/wordlists/rockyou.txt
13882877p 0:00:00:30 0.00% (ETA: 2027-03-05 21:37) 461838p/s 0008/2074
Session..........: hashcat
Status...........: Running
Hash.Type........: Cisco-IOS type 4 (SHA256)
Hash.Target......: ds4zcEBHQMiiscBff5JmSaUctdI8fVdmGU18HAtxOCw
Time.Started.....: Mon Mar 16 12:14:47 2020 (20 secs)
Time.Estimated...: Mon Mar 16 12:15:07 2020 (0 secs)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Speed.#2.........:    95135 H/s (9.59ms) @ Accel:32 Loops:1 Thr:256 Vec:1
Speed.#3.........:  1047.1 kH/s (4.14ms) @ Accel:512 Loops:1 Thr:256 Vec:1
Speed.#*.........:  1142.2 kH/s
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 18552065
Rejected.........: 5377
Restore.Point....: 0
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#2....: 01Sandbugggys -> 01Rock you karen
Candidates.#3....: 01Ciara95 -> 01Sandbugs3

ds4zcEBHQMiiscBff5JmSaUctdI8fVdmGU18HAtxOCw:!P@ssw0rd
```


From the above screenshot we can see that the **average speed** is around **1.14 million** password attempts per second. Seems like cracking this hash with john is much faster in our case.

Go back to top.

Cisco type 5 password

This password type was introduced around 1992 and it is essentially a 1,000 iteration of MD5 hash with salt. The salt is 4 characters long (32 bits). For modern computers this is not difficult enough and thus in many cases it can be successfully cracked.

The following example shows type 5 password found in a Cisco configuration:

```
username admin secret 5 $1$jUfy$2TVVXJ8sy.KO8ZhAKfIHt/
```

Decrypt Cisco type 5 passwords with John

John the Ripper recognizes this password type as **md5crypt**. To crack it, we have to again first convert it to the following john friendly format and save it in a file:

```
admin:$1$jUfy$2TVVXJ8sy.KO8ZhAKfIHt/
```

Then we can crack it like this using a dictionary, for example:

```
john --format=md5crypt --fork=4 --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
```

```
root@kali:~# john --format=md5crypt --fork 4 --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 AVX 4x3])
Node numbers 1-4 of 4 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
4 0g 0:00:00:22 25.55% (ETA: 23:52:00) 0g/s 41307p/s 41307c/s 41307C/s shamsiahzibri..shamseha
2 0g 0:00:00:23 25.24% (ETA: 23:52:04) 0g/s 40860p/s 40860c/s 40860C/s shidake..shida606
3 0g 0:00:00:23 24.75% (ETA: 23:52:05) 0g/s 40140p/s 40140c/s 40140C/s sissybis..sissy916
1 0g 0:00:00:23 24.95% (ETA: 23:52:05) 0g/s 40421p/s 40421c/s 40421C/s silarijal..silang86
4 0g 0:00:00:46 49.86% (ETA: 23:52:06) 0g/s 37959p/s 37959c/s 37959C/s j1101395..j10m18enro
```

```
1 0g 0:00:00:47 49.26% (ETA: 23:52:08) 0g/s 37505p/s 37505c/s 37505C/s janjeffre..janjans1512
2 0g 0:00:00:47 49.77% (ETA: 23:52:07) 0g/s 37885p/s 37885c/s 37885C/s jEbUss..jENNY3744
3 0g 0:00:00:47 49.93% (ETA: 23:52:07) 0g/s 38005p/s 38005c/s 38005C/s izabel84..izaalvarez
!P@ssw0rd        (admin)
```

From the above screenshot we can see that the **average speed** is around **155 thousand** password attempts per second.

Decrypt Cisco type 5 passwords with Hashcat

Hashcat recognizes this password type as hash mode **500**. To crack it, we can keep using the same john friendly format. Then we can crack it like this using a dictionary, for example:

```
hashcat -m 500 --username -O -a 0 hashes.txt /usr/share/wordlists/rockyou.txt
```

Note that by using the -O parameter (optimized kernels), we will greatly increase the speed. But it will also limit the password length to 31 characters.

```
root@kali:~# hashcat -m 500 --quiet --username -O -a 0 hashes.txt /usr/share/wordlists/rockyou.txt
[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit => s

Session..........: hashcat
Status...........: Running
Hash.Type........: md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5)
Hash.Target......: $1$jUfy$2TVVXJ8sy.KO8ZhAKfIHt/
Time.Started.....: Mon Mar 16 23:07:06 2020 (10 secs)
Time.Estimated...: Mon Mar 16 23:07:17 2020 (1 sec)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:    58900 H/s (11.30ms) @ Accel:64 Loops:31 Thr:16 Vec:1
Speed.#3.........:  1207.0 kH/s (10.02ms) @ Accel:128 Loops:125 Thr:64 Vec:1
Speed.#*.........:  1265.9 kH/s
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 12875465/14344385 (89.76%)
Rejected.........: 227017/12875465 (1.76%)
Restore.Point....: 12475450/14344385 (86.97%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:186-217
```

```
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:875-1000
Candidates.#2....: 2242719 -> 1atusl
Candidates.#3....: 217578132 -> 1b2kfan

$1$jUfy$2TVVXJ8sy.KO8ZhAKfIHt/:!P@ssw0rd
```

From the above screenshot we can see that the **average speed** is around **1.2 million** password attempts per second. Much better than john in our case.

Go back to top.

Cisco type 8 password

This password type is a proper implementation of the failed password type 4. This time it really uses the PBKDF2 algorithm and 10 character salt (80 bits). Essentially it is 20,000 iterations of SHA256 and this makes it much harder to crack in comparison with the previous password types.

The following example shows type 8 password found in a Cisco configuration:

```
username admin secret 8 $8$dsYGNam3K1SIJO$7nv/35M/qr6t.dVc7UY9zrJDWRVqncHub1PE9UlMQFs
```

Decrypt Cisco type 8 passwords with John

John the Ripper recognizes this password type as **pbkdf2-hmac-sha256**. To crack it, we have to again first convert it to the following john friendly format and save it in a file:

```
admin:$8$dsYGNam3K1SIJO$7nv/35M/qr6t.dVc7UY9zrJDWRVqncHub1PE9UlMQFs
```

Then we can crack it like this using a dictionary, for example:

```
john --format=pbkdf2-hmac-sha256 --fork=4 --wordlist=/usr/share/wordlists/rockyou.txt hashes
```

```
root@kali:~# john --format=pbkdf2-hmac-sha256 --fork=4 --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PBKDF2-HMAC-SHA256 [PBKDF2-SHA256 128/128 AVX 4x])
Cost 1 (iteration count) is 20000 for all loaded hashes
Node numbers 1-4 of 4 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
4 0g 0:00:00:05 0.05% (ETA: 17:58:22) 0g/s 328.5p/s 328.5c/s 328.5C/s angel6..melania
1 0g 0:00:00:06 0.04% (ETA: 18:44:24) 0g/s 314.1p/s 314.1c/s 314.1C/s emilee..jason123
2 0g 0:00:00:06 0.05% (ETA: 18:35:04) 0g/s 327.4p/s 327.4c/s 327.4C/s boogie1..parkour
3 0g 0:00:00:06 0.04% (ETA: 18:46:17) 0g/s 311.4p/s 311.4c/s 311.4C/s wildfire..fifteen
2 0g 0:00:00:25 0.17% (ETA: 18:55:08) 0g/s 288.8p/s 288.8c/s 288.8C/s BOWWOW1..180189
4 0g 0:00:00:25 0.18% (ETA: 18:50:41) 0g/s 294.2p/s 294.2c/s 294.2C/s 251185..10231023
3 0g 0:00:00:26 0.17% (ETA: 19:06:51) 0g/s 286.0p/s 286.0c/s 286.0C/s misty2..kaklong
1 0g 0:00:00:26 0.17% (ETA: 19:08:31) 0g/s 283.9p/s 283.9c/s 283.9C/s 171190..020805
cisco            (admin)
```

From the above screenshot we can see that the **average speed** is around **1,200** password attempts per second. Not much.

Decrypt Cisco type 8 passwords with Hashcat

Hashcat recognizes this password type as hash mode **9200**. To crack it, we can keep using the same john friendly format. Then we can crack it like this using a dictionary, for example:

```
hashcat -m 9200 --username -a 0 hashes.txt /usr/share/wordlists/rockyou.txt
```

```
root@kali:~# hashcat -m 9200 --quiet --username -a 0 hashes.txt /usr/share/wordlists/rockyou.txt
[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit => s

Session..........: hashcat
Status...........: Running
Hash.Type........: Cisco-IOS $8$ (PBKDF2-SHA256)
```

```
Hash.Target......: $8$dsYGNam3K1SIJO$7nv/35M/qr6t.dVc7UY9zrJDWRVqncHub...0lMQFs
Time.Started.....: Tue Mar 17 15:05:22 2020 (27 secs)
Time.Estimated...: Tue Mar 17 15:25:58 2020 (20 mins, 9 secs)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:     1271 H/s (7.01ms) @ Accel:16 Loops:8 Thr:64 Vec:1
Speed.#3.........:    10347 H/s (10.55ms) @ Accel:64 Loops:32 Thr:64 Vec:1
Speed.#*.........:    11619 H/s
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 286720/14344385 (2.00%)
Rejected.........: 0/286720 (0.00%)
Restore.Point....: 24576/14344385 (0.17%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:7632-7640
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:4096-4128
Candidates.#2....: 10032004 -> desodorante
Candidates.#3....: desmond07 -> 14861486

$8$dsYGNam3K1SIJO$7nv/35M/qr6t.dVc7UY9zrJDWRVqncHub1PE9UlMQFs:cisco
```

From the above screenshot we can see that the **average speed** is around **11,600** password attempts per second. It's definitely faster than john in our case, but in overall it's not very fast.

Go back to top.

Cisco type 9 password

This password type uses Scrypt algorithm. Scrypt was specifically designed to make cracking very difficult even on large-scale cracking rigs with many GPUs or hardware ASICs. This is due to the fact that Scrypt requires large amount of memory to perform its function.

The following example shows type 9 password found in a Cisco configuration:

```
username admin secret 9 $9$nhEmQVczB7dqsO$X.HsgL6x1il0RxkOSSvyQYwucySCt7qFm4v7pqCxkKM
```

Decrypt Cisco type 9 passwords with John

John the Ripper recognizes this password type as **scrypt**. To crack it, we have to again first convert it to the following john friendly format and save it in a file:

```
admin:$9$nhEmQVczB7dqsO$X.HsgL6x1il0RxkOSSvyQYwucySCt7qFm4v7pqCxkKM
```

Then we can crack it like this using a dictionary, for example:

```
john --format=scrypt --fork=4 --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
```

```
root@kali:~# john --format=scrypt --fork=4 --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (scrypt [Salsa20/8 128/128 AVX])
Cost 1 (N) is 16384 for all loaded hashes
Cost 2 (r) is 1 for all loaded hashes
Cost 3 (p) is 1 for all loaded hashes
Node numbers 1-4 of 4 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
4 0g 0:00:00:22 0.19% (ETA: 19:08:13) 0g/s 360.9p/s 360.9c/s 360.9C/s parkway
1 0g 0:00:00:23 0.19% (ETA: 19:15:56) 0g/s 362.3p/s 362.3c/s 362.3C/s hockey14
3 0g 0:00:00:22 0.20% (ETA: 19:04:11) 0g/s 369.1p/s 369.1c/s 369.1C/s longlong
2 0g 0:00:00:22 0.20% (ETA: 19:03:29) 0g/s 370.8p/s 370.8c/s 370.8C/s double07
2 0g 0:00:02:11 1.04% (ETA: 19:27:10) 0g/s 338.0p/s 338.0c/s 338.0C/s 14031995
1 0g 0:00:02:11 1.04% (ETA: 19:27:11) 0g/s 338.0p/s 338.0c/s 338.0C/s 14091980
4 0g 0:00:02:11 1.06% (ETA: 19:23:26) 0g/s 344.0p/s 344.0c/s 344.0C/s sagarino
3 0g 0:00:02:11 1.05% (ETA: 19:24:58) 0g/s 341.7p/s 341.7c/s 341.7C/s thayne
cisco            (admin)
```

From the above screenshot we can see that the **average speed** is around **1,400** password attempts per second. Not much.

Decrypt Cisco type 9 passwords with Hashcat

Hashcat recognizes this password type as hash mode **9300**. To crack it, we can keep using the same john friendly format. Then we can crack it like this using a dictionary, for example:

```
hashcat -m 9300 --username -a 0 --force hashes.txt /usr/share/wordlists/rockyou.txt
```

Note that we have to provide `--force` parameter since the hash-mode 9300 is marked as unstable for our particular device.

```
root@kali:~# hashcat -m 9300 --quiet --username -a 0 --force hashes.txt /usr/share/wordlists/rockyou.txt
[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit => s

Session..........: hashcat
Status...........: Running
Hash.Type........: Cisco-IOS $9$ (scrypt)
Hash.Target......: $9$nhEmQVczB7dqsO$X.HsgL6x1il0RxkOSSvyQYwucySCt7qFm...qCxkKM
Time.Started.....: Tue Mar 17 16:14:44 2020 (52 secs)
Time.Estimated...: Tue Mar 17 18:26:04 2020 (2 hours, 10 mins)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:      859 H/s (3015.99ms) @ Accel:16 Loops:1 Thr:8 Vec:1
Speed.#3.........:      963 H/s (2152.81ms) @ Accel:16 Loops:1 Thr:8 Vec:1
Speed.#*.........:     1824 H/s
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 92160/14344385 (0.64%)
Rejected.........: 0/92160 (0.00%)
Restore.Point....: 88064/14344385 (0.61%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#2....: melissam -> theused!
Candidates.#3....: erica18 -> 020180

$9$nhEmQVczB7dqsO$X.HsgL6x1il0RxkOSSvyQYwucySCt7qFm4v7pqCxkKM:cisco
```


From the above screenshot we can see that the **average speed** is around **1,800** password attempts per second. Not much either.

Go back to top.

Cracking tips

Use KoreLogic custom rules

John the Ripper contains very useful ruleset for generating passwords called **KoreLogic**. This ruleset originated in DEFCON 2010 contest and it is a great way of generating passwords from patterns or when traditional dictionary attack fails.

To make use of it, simply create a dictionary with patterns like this e.g.:

```
infosec
infosecmatter
infosec matter
```

Save it into `patterns.txt` file. Then simply run john like this:

```
john --wordlist=patterns.txt --rules=korelogic ...
```

Chain John the Ripper with Hashcat

Although there has been some efforts to convert the aforementioned KoreLogic rules into Hashcat, the result is only partial. Fortunately, we can chain together John the Ripper with Hashcat to make it use KoreLogic rules in full.

Simply generate the passwords using John the Ripper on the stdout and feed them into Hashcat using pipe like this:

```
john --wordlist=patterns.txt --rules=korelogic --stdout | hashcat ...
```

The same method can be used for any ruleset that we have created in the john format.

Go back to top.

References

- https://learningnetwork.cisco.com/s/article/cisco-routers-password-types
- https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20130318-type4
- https://community.cisco.com/t5/networking-documents/understanding-the-differences-between-the-cisco-password-secret/ta-p/3163238

TAGS | Cisco | Cracking | Credentials | Hashcat | John the Ripper | Network devices | Password

RECENT POSTS

Pentesting 101: Working With Exploits

RCE on Windows from Linux Part 3: Pass-The-Hash Toolkit

Minimalistic SMB login bruteforcer

RCE on Windows from Linux Part 2: CrackMapExec

RCE on Windows from Linux Part 1: Impacket

LEAVE A COMMENT

Your email address will not be published. Required fields are marked *

Type here..

Name*

Email*

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment »