Welcome

Hashcat

Password Cracking

What is a password?

A password is one of the three different methods of authentication.

Scan for blog post on 2-factor authentication.

How to Store Passwords

There are multiple ways
to store passwords.

Plain Text Hash + Salt

<---------->

Level 00 Level 04

Scenario

When is password cracking relevant?

- Password cracking can be a step to privilege escalation.
- Examples:
  - A website or server is leaking password files to the public.
  - You have access to a lower privilege account on a system and want to gain higher privilege.
  - Multiple other scenarios.

For this workshop, imagine you are an attacker that has discovered daviscybersec.org is leaking passwords.

Level 00: Plain Text

Plain text passwords are worst way to store passwords. If you are a developer, never ever store passwords in plain text.

Challenge: Head to daviscybersec.org/passwords/00.txt
Can you figure out the password? *It's plain text so I hope you don't need a solution here...*

Plain Text Breach Example

2009 RockYou Breach

RockYou, which made widgets for MySpace had a breach resulting in over 32 million account's login information being exposed. On top of this, all the passwords were stored in plain text.

This was completely preventable...

Decompressing rockyou.txt

"Change Directory"

Open a terminal and run ***cd /usr/share/wordlists***
This folder holds useful wordlists you can use on your adventures. You can use ***ls*** to see the contents of the folder.

"List Directory Contents"

The passwords from the RockYou breach are included, but compressed because it is quite large.
Use ***sudo gzip -d rockyou.txt.gz*** to decompress.

This is a restricted folder, this runs a command with root permissions.

Name of program

Decompress

Name of file

Level 01: Encoded

Encoded passwords have about the same security level as plain text passwords, you just need an extra step.

Challenge: Head to daviscybersec.org/passwords/01.txt

01.txt:
cGV0dGluZ2NoZWV0bzI0Nw==

Can you figure out the password? Test at daviscybersec.org/login to check if you are right.

Level 01: Solution

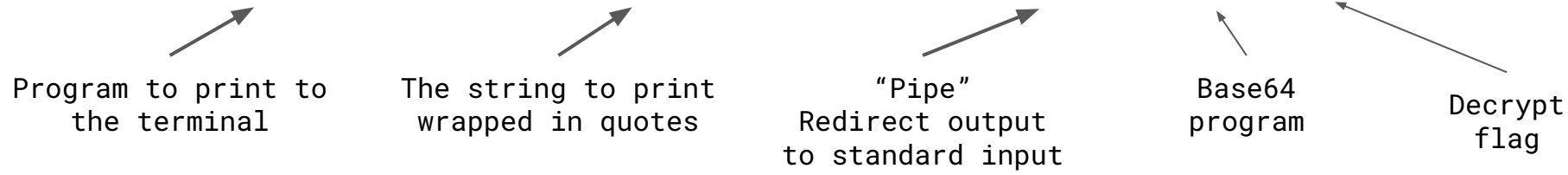
01.txt:
cGV0dGluZ2NoZWV0bzI0Nw==

This looks like it is encoded in Base64.

Solution 1: Copy/paste into a website like cryptii.com

Solution 2: Use command

echo "cGV0dGluZ2NoZWV0bzI0Nw==" | base64 -d



Level 02: Hashed

Hashes are strings of characters that have
been mathematically converted based on a specific algorithm.

We won't go into specifics on the math, but know:

- String -> Hash: Easy
- Hash -> String: Very Hard

Example Hash Types: MD4, MD5, SHA-1, etc.

Challenge: Download daviscybersec.org/passwords/02.txt
Can you figure out the password? Test if you are right.
(You are not expected to know this yet.)

Level 02: Solution

02.txt:
fc36b1efebc9d6f1f3b12382d6560fe5

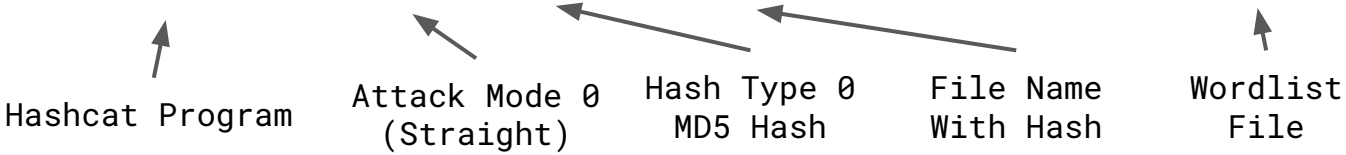
This is an MD5 type hash!
Solution: Use hashcat

Save 02.txt to your Downloads folder. (Ctrl + S on browser.)

Go to your Downloads by using ***cd ~/Downloads***

Run

hashcat -a 0 -m 0 02.txt /usr/share/wordlists/rockyou.txt



What is hashcat doing?

Hashcat is hashing every password in our wordlist to the given hash format, then checking it against our given hash.

Checking fc36b1efebc9d6f1f3b12382d6560fe5...

Wordlist		Hashed	Match
password5	-->	db0edd04aaac4506f7edab03ac855d56	❌
hacker	-->	d6a6bc0db10694a2d90e3a69648f3a03	❌
catdog1	-->	fc36b1efebc9d6f1f3b12382d6560fe5	✔️

Problems with MD5

1. Intersection Attack

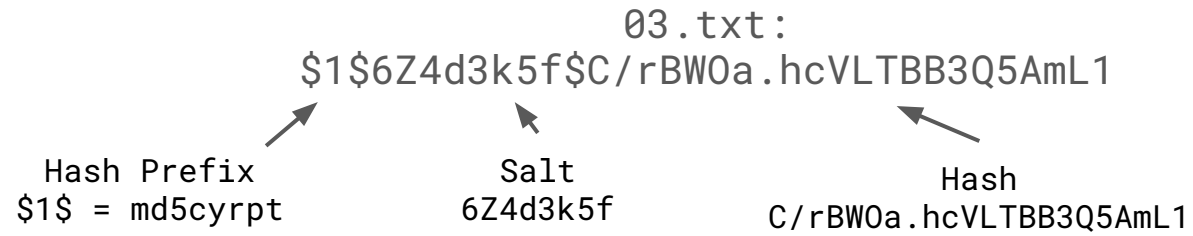
   MD5 is not entirely cryptographically secure, and should not be used for passwords. Simplified: multiple passwords can generate the same hash.

2. Rainbow Tables

   Pre-generated hashes make the cracking progress even quicker.

Level 03: Hash + Salt

Download daviscybersec.org/passwords/03.txt



This is an MD5 hash with a salt. A salt is an added string to the hashing process which makes it slower to use passwords lists or brute force. Can you figure out the password? Test it out!

Level 03: Solution

Let’s use John the Ripper this time!

Save 03.txt to your Downloads folder.

Run

john --wordlist=/usr/share/wordlists/rockyou.txt 03.txt

Hashcat & John

Both are great! Each one has strengths and weaknesses.

John is older but is compatible with system hashing. (Hence old flag format --wordlist=/location/) John will also automatically brute force after exhausted wordlist.

Hashcat it’s easy to use rules, use different attack types, and more.

Both are pretty smart! They can auto detect hash types and work with multiple hashes at a time.

```
user1:hash1
user2:hash2
user3:hash3
```

Level 04: Hash + Salt + Unique

Using password lists is great, but what if a password isn't in a password list?

Download daviscybersec.org/passwords/04.txt:

04.txt:
1mZyJXWKf$fkFteq9H1/BGFpNNPTfC80

This will not work with a password list! Can you figure out the password? Test it out!

Level 04: Solution

If you know the general format of a password, you can use rules to modify password lists. Examples: a to @, i to !, password + numbers

Let’s switch back to hashcat. There exists common rules lists you can search how to use. Let’s just stick to passwords + number. This is slow! Do it at home.

hashcat -a 3 -m 500 04.txt /usr/share/wordlists/rockyou.txt -1 ?d

md5crypt

Mask #1
?d = 01234567890

Prevention

As a developer, how can we prevent this?

- Use OAuth - No need to store passwords.
- Implement 2-factor authentication - Cracked passwords cannot be used despite being cracked.
- FOLLOW A RECENT AND RELEVANT TUTORIAL ON STORING PASSWORDS!!! USE A SECURE HASH METHOD!!! (yescrypt, bcrypt, etc.)

As a user, how can we prevent this?

- Use 2-factor authentication whenever possible.
- Have strong unique passwords on each website, use a password manager.

Level 05: Challenge Hash

Save daviscybersec.org/passwords/05.txt
Can you figure out the password? Test it out.

05.txt:
$2b$05$3EhiYUKKvZC74eH4UkkHnOasosRDWwZuMXX0FQ0m4TKXR/LHGWNj2