Hybrid Attacks

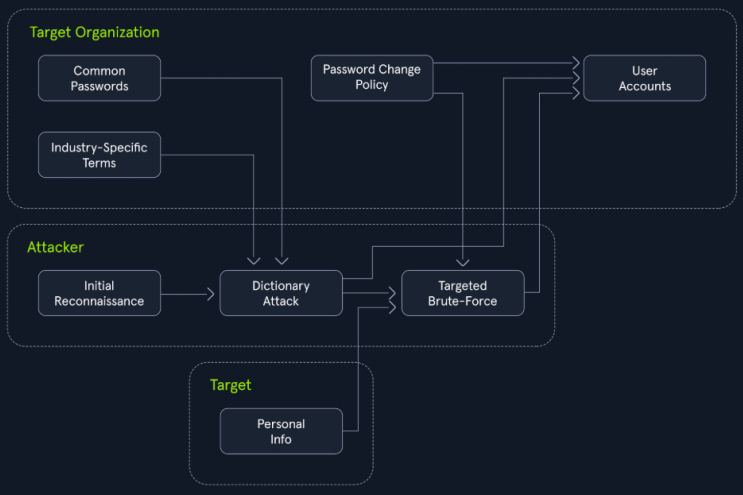
Many organizations implement policies requiring users to change their passwords periodically to enhance security. However, these policies can inadvertently breed predictable password patterns if users are not adequately educated on proper password hygiene.

Unfortunately, a widespread and insecure practice among users is making minor modifications to their passwords when forced to change them. This often manifests as appending a number or a special character to the end of the current password. For instance, a user might have an initial password like "Summer2023" and then, when prompted to update it, change it to "Summer2023!" or "Summer2024."

This predictable behavior creates a loophole that hybrid attacks can exploit ruthlessly. Attackers capitalize on this human tendency by employing sophisticated techniques that combine the strengths of dictionary and brute-force attacks, drastically increasing the likelihood of successful password breaches.

Hybrid Attacks in Action

Let's illustrate this with a practical example. Consider an attacker targeting an organization known to enforce regular password changes.



The attacker begins by launching a dictionary attack, using a wordlist curated with common passwords, industry-specific terms, and potentially personal information related to the organization or its employees. This phase attempts to quickly identify any low-hanging fruit - accounts protected by weak or easily guessable passwords.

However, if the dictionary attack proves unsuccessful, the hybrid attack seamlessly transitions into a brute-force mode. Instead of randomly generating password combinations, it strategically modifies the words from the original wordlist, appending numbers, special characters, or even incrementing years, as in our "Summer2023" example.

This targeted brute-force approach drastically reduces the search space compared to a traditional brute-force attack while covering many potential password variations that users might employ to comply with the password change policy.

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left

The Power of Hybrid Attacks

The effectiveness of hybrid attacks lies in their adaptability and efficiency. They leverage the strengths of both dictionary and brute-force techniques, maximizing the chances of cracking passwords, especially in scenarios where users fall into predictable patterns.

It's important to note that hybrid attacks are not limited to the password change scenario described above. They can be tailored to exploit any observed or suspected password patterns within a target organization. Let's consider a scenario where you have access to a common passwords wordlist, and you're targeting an organization with the following password policy:

- Minimum length: 8 characters
- Must include:
  - At least one uppercase letter
  - At least one lowercase letter
  - At least one number

To extract only the passwords that adhere to this policy, we can leverage the powerful command-line tools available on most Linux/Unix-based systems by default, specifically `grep` paired with regex. We are going to use the darkweb2017-top10000.txt password list for this. First, download the wordlist

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ wget https://raw.githubusercontent.com/danielmiessler/SecLists/refs/h
```

Next, we need to start matching that wordlist to the password policy.

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ grep -E '^.{8,}$' darkweb2017-top10000.txt > darkweb2017-minlength.tx
```

This initial `grep` command targets the core policy requirement of a minimum password length of 8 characters. The regular expression `^.{8,}$` acts as a filter, ensuring that only passwords containing at least 8 characters are passed through and saved in a temporary file named `darkweb2017-minlength.txt`.

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ grep -E '[A-Z]' darkweb2017-minlength.txt > darkweb2017-uppercase.txt
```

Building upon the previous filter, this `grep` command enforces the policy's demand for at least one uppercase letter. The regular expression `[A-Z]` ensures that any password lacking an uppercase letter is discarded, further refining the list saved in `darkweb2017-uppercase.txt`.

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ grep -E '[a-z]' darkweb2017-uppercase.txt > darkweb2017-lowercase.txt
```

Maintaining the filtering chain, this `grep` command ensures compliance with the policy's requirement for at least one lowercase letter. The regular expression `[a-z]` serves as the filter, keeping only passwords that include at least one lowercase letter and storing them in `darkweb2017-lowercase.txt`.

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ grep -E '[0-9]' darkweb2017-lowercase.txt > darkweb2017-number.txt
```

This last `grep` command tackles the policy's numerical requirement. The regular expression `[0-9]` acts as a filter, ensuring that passwords containing at least one numerical digit are preserved in `darkweb2017-number.txt`.

Hybrid Attacks

```
MisaelMacias@htb[/htb]$ wc -l darkweb2017-number.txt

89 darkweb2017-number.txt
```

As demonstrated by the output above, meticulously filtering the extensive 10,000-password list against the password policy has dramatically narrowed down our potential passwords to 89. This drastic reduction in the search space represents a significant boost in efficiency for any subsequent password cracking attempts. A

smaller, targeted list translates to a faster and more focused attack, optimizing the use of computational resources and increasing the likelihood of a successful breach.

Credential Stuffing: Leveraging Stolen Data for Unauthorized Access

Credential stuffing attacks exploit the unfortunate reality that many users reuse passwords across multiple online accounts. This pervasive practice, often driven by the desire for convenience and the challenge of managing numerous unique credentials, creates a fertile ground for attackers to exploit.

It's a multi-stage process that begins with attackers acquiring lists of compromised usernames and passwords. These lists can stem from large-scale data breaches or be compiled through phishing scams and malware. Notably, publicly available wordlists like `rockyou` or those found in `seclists` can also serve as a starting point, offering attackers a trove of commonly used passwords.

Once armed with these credentials, attackers identify potential targets - online services likely used by the individuals whose information they possess. Social media, email providers, online banking, and e-commerce sites are prime targets due to the sensitive data they often hold.

The attack then shifts into an automated phase. Attackers use tools or scripts to systematically test the stolen credentials against the chosen targets, often mimicking normal user behavior to avoid detection. This allows them to rapidly test vast numbers of credentials, increasing their chances of finding a match.

A successful match grants unauthorized access, opening the door to various malicious activities, from data theft and identity fraud to financial crimes. The compromised account may be a launchpad for further attacks, spreading malware, or infiltrating connected systems.

The Password Reuse Problem

The core issue fueling credential stuffing's success is the pervasive practice of password reuse. When users rely on the same or similar passwords for multiple accounts, a breach on one platform can have a domino effect, compromising numerous other accounts. This highlights the urgent need for strong, unique passwords for every online service, coupled with proactive security measures like multi-factor authentication.