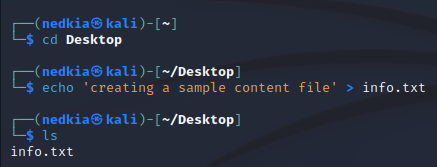
Ned Kiame

Tool: John the Ripper using Kali Linux

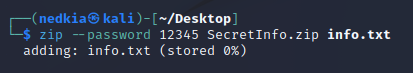
Contents:

1. Tool Overview
   1. **Capabilities**
      1. **Password Cracking:** Currently John the Ripper is being used in the industry to test the complexity of passwords. John The Ripper can crack most simple passwords, and it's generally noted that if a password can be easily cracked using John The Ripper, it is not a strong enough password.
      2. **Attack Modes:** John The Ripper is capable of performing different types of password attacks
         1. **Brute Force:** Brute force attacks rapidly tries every password combination until it is able to crack a password. Generally this attack is only successful under two conditions: The password is very short (which will make the number of guesses a feasible amount) or the CPU/GPU is so powerful that it is able to rapidly guess every combination until it reaches the password. For longer passwords, this can only be done using Quantum computing.
         2. **Rainbow Table:** A rainbow table attempts to solve some of the computational issues that a brute force attack has by doing their computations beforehand. Rainbow tables are large lists of passwords and their corresponding hashes. When a rainbow table attack is performed, the CPU/GPU quickly attempts to match the hash that it is given to a known hash in the rainbow table database. This drastically improves the time it takes to match a hash with a password because the hash is always calculated before the attack is attempted. The drawback to this is that the rainbow table can only crack hashes that it already has and if someone uses a password that hasn’t been calculated already, then the rainbow table will never find it.
         3. **Dictionary Attack**: Dictionary attacks use a predefined world list to crack passwords. These types of attacks are used to find common passwords, or they are compared to a list of leaked passwords to see if there are any matches.
         4. **Hybrid Attack:** This is a mixture of a dictionary attack and a Brute force attack. This attack works by first finding a sample password from a dictionary, and it then uses brute force techniques to generate variations of the password from the dictionary table. This is primarily used to catch passwords that have common words with little variation (Example: “Password197”)
         5. **Mask Attack:** This attack is comparable to a Hybrid attack but instead of taking the variation from words found in a dictionary, this attack will try to create variations based on what the user provides (using the –mask function).
      3. **Various File Format:** John The Ripper was designed to support and identify multiple types of file formats. This includes but is not limited to cracking zip files, database files (SQL) and hash files.
      4. **GPU integration and acceleration**: John The Ripper supports GPU power integration which means that an individual does not only need to rely on a CPU to crack passwords. Using a GPU for assistance can dramatically improve results
      5. **Tool and Software Integration:** John The Ripper supports multiple languages and platforms. It also has features which makes it easy for other software to work with John The Ripper. John The Ripper has been included in many Penetration testing and Red team frameworks to automatically test for weak passwords in professional environments.
      6. **Hash Generation and Password Evaluation:** Based on the speed in which John can/cannot crack a password, it will give a score to help Penetration testers evaluate the strength of a password, and this tool can even be used to generate hashes of these passwords.
2. How Passwords are stored
   1. **ClearText:** Storing a password in cleartext is simply storing the password strings in some location on the computer. This is the common way that passwords were stored before security became a concern but in modern times this is known as bad practice. Storing a password in cleartext means that we don’t even need to use a tool such as John the Ripper to crack it, because there is no encryption placed on it.
   2. **Hashing:** Hashing is done by manipulating a string so that it is unreadable to those who do not know what the manipulation was. Modern day algorithms are so advanced that even if you know the exact algorithm that was used, it is still impossible to mathematically reverse the operation. Most hashed passwords are cracked because the password was trivial and another computer was able to recreate the password and compute its hash. If a strong hashed password is cracked, it is usually because the hash or password was leaked and it gave malicious actors the opportunity to calculate the hash of the complex password.
   3. **Salting + Hashing:** Salting adds a new layer of security hashing process. This is done by further manipulating the hashed value with a unique, local function. This protects against leaked passwords because if a password is compromised on one website or database, the malicious actor will still need to compromise the specific salt value in order to find the password in cleartext. This is the current industry best practice.
3. Demo Examples:
   1. First I created a sample text files which will be used as our data content:

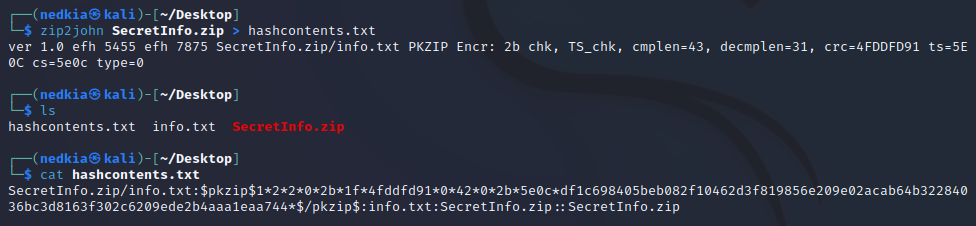


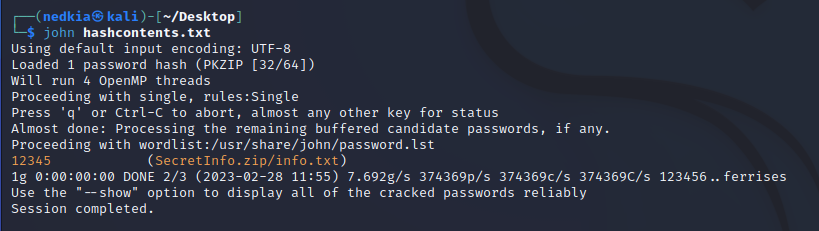
* 1. Then I use the zip linux command to create a password protected zip file:

In this example, our zip file password is “12345”, the zip file name is SecretInfo.zip and the contents within the zip file is the info.txt file.



* 1. Since I created a Zip File, I will use the “zip2John” command in order to retrieve the hash of the zip file and write the contents of the hash to a text file. It's important to note that John supports multiple files and hashes, and retrieving the hash value is usually a singular command.



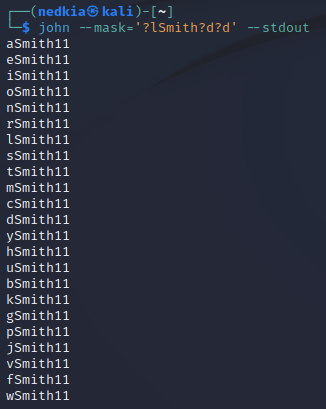
* 1. After the hash has been written to contents, the final step is to run the john command on the newly created file.

1. Masking
   1. When using John The Ripper, we must be very conservative of our resources and we should use techniques that preserve our resources when possible. Knowing even the smallest information about a password can exponentially impact how many resources we use, which also directly impacts the result of using John to crack the password. Knowing a small part of the password means that any resources used to guess passwords without that small piece of information is a complete waste of many to prevent the password from being cracked.

Masking helps us significantly by allowing us to perform a hybrid attack. For example, if we were attempting to brute force a school email, we may know some very important information. Assume that we are attempting to crack John Smith’s email with the following password policy:

1. Emails contain the user’s last name
2. Emails contain 2 numbers
3. All Emails precede with a random character at the start

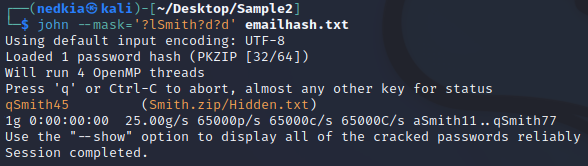
Attempting to crack John Smith’s email using John The Ripper without any specific mask could take a long time considering that John will by default use a known wordlist, rainbow table or simply attempt to brute force the password. If we know that “Smith” is in the password, and that there are specifically 2 numbers, then using the –mask command can come in handy.

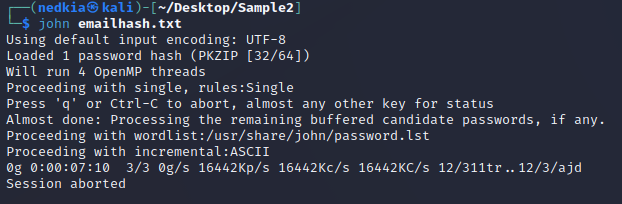
By using the –stdout flag, we can observe the character combinations that John is outputting.

Note: There are more flags that can be used for certain characters such as hexadecimal values. These are the most common and are used for most password hashes.

| Flags | John Output |
| --- | --- |
| ?I | ASCII Uppercase |
| ?u | ASCII Lowercase |
| ?d | Numbers |
| ?s | Special character not included in ?I,?u or ?d |
| ?[range\_start - range\_end] | [a-z] , [0-100], etc. |

Hash cracked instantly by using the –mask command



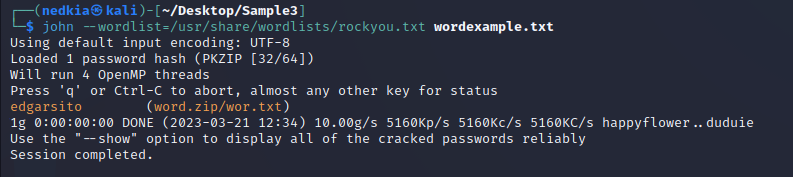
Hash aborted after 7 minutes (failed to crack) 

Wordlists:

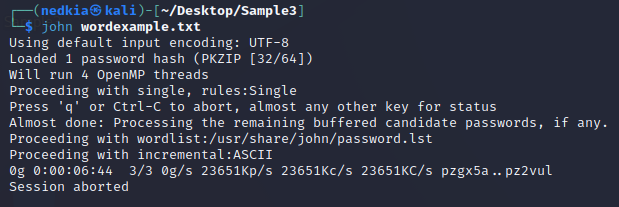
When databases are hacked into and breached it is almost always because the perpetrator wants to sell the information to some third party. Sensitive information such as financial and medical records are worth a lot more and are big targets for malicious actors. Some of these data breaches end up becoming open sourced and are available to many different actors who may or may not use this information to steal accounts. When usernames, passwords and hashes are leaked, hackers will create word lists containing these breached credentials. By using open source tools, a hacker will be able to find out where

John the Ripper capitalizes on this by allowing custom wordlists to be easily incorporated, which will enable penetration testers to test hashes on current data leaks. Even if the password is extremely obscure and would otherwise be considered safe, if it is found in a wordlist then it would only take seconds for someone to crack it.

Here we attempted to crack the hash of “edgarsito” using the “rockyou.txt” wordlist. Rockyou.txt is a common wordlist found on kali linux, and it contains many passwords that were previously breached. By specifying the wordlist, we were able to crack the hash of “edgarsito” instantly.



While we may eventually be able to crack the hash using regular brute force techniques, it is drastically longer.



(aborted after 7 minutes, hash was unable to be cracked)

**Preventing John the Ripper:**

Security is always done by looking at the perspective of both attackers and defenders. The better one side understands the other, the better they are able to adapt to their techniques. This tug of war is a continuous battle that the cybersecurity industry is involved in. When computers first came out, security was not a priority but rather just “making everything work”. During this time simple pieces of malware were able to create a lot of havoc on the Internet and eventually security implementations began to become more and more common. John the Ripper and the entire Kali linux distribution is the result of tools created to attack systems, and it serves as a way for defenders to harden their defense and for attackers to attack a weak defense.John the Ripper works by using computer resources to rapidly guess passwords, with the hopes of eventually guessing the right password. Since a human doesn't need to type a password in manually, John the Ripper is able to sit and keep guessing passwords faster than a human ever could. John relies on the probability of guessing a password, and as a result developers and security engineers have learned of various ways of stopping these techniques.

The main ways to prevent tools such as John the Ripper is with the following:

1. **Increasing Password Length & Complexity**

Possibly the most trivial solution, but this solution is extremely effective. Even if the password is very basic, increasing the length will exponentially slow down John the Ripper.In Fact, the general equation for calculating the password length isPossible combinations = possible number of character ^ (password length) This is very important considering that adding a few characters to a password will increase the time of cracking the password from a few seconds to many years.

When given the choice most people will choose an easy to remember password that they assume nobody will be able to guess. A password auditing tool cares more about the length and the character than the obscurity of the word.

Normal Brute Force attacks will begin with numbers -> lowercase letters -> Uppercase letters -> symbols.

Having a combination of all of these will force someone who uses John the Ripper to include all of these characters which means that the amount of guesses will also exponentially increase.

(Data from HowSecureismyPassword.net)

| # of characters | Numbers | Lowercase | Uppercase and Lowercase | Numbers, Uppercase, Lowercase | Numbers, UpperCase, Lowercase, Symbols |
| --- | --- | --- | --- | --- | --- |
| 4 | Instantly | Instantly | Instantly | Instantly | Instantly |
| 5 | Instantly | Instantly | Instantly | Instantly | Instantly |
| 6 | Instantly | Instantly | Instantly | 1 sec | 5 sec |
| 7 | Instantly | Instantly | 25 seconds | 1 min | 6 min |
| 8 | Instantly | 5 seconds | 22 min | 1 hour | 8 hours |
| 9 | Instantly | 2 minutes | 19 hours | 3 days | 3 weeks |
| 10 | Instantly | 58 mins | 1 month | 7 months | 5 years |
| 11 | 2 seconds | 1 day | 5 years | 41 years | 400 years |
| 12 | 25 seconds | 3 weeks | 300 years | 2000 years | 34,0000 years |
| 13 | 4 minutes | 1 year | 16,000 years | 100,000 years | 2 million years |
| 14 | 41 minutes | 51 years | 800,000 years | 9 million years | 200 million years |

1. **Forcing Multi-Factor authentication**

Regardless of the type of attack that we use, we still need to make many guesses. Although it is usually a Resource vs Complexity battle, forcing Two-factor authentication completely mitigated tools such as John the Ripper. Depending on the type of Authentication, it can be very difficult or impossible to create a large amount of guesses in a plausible time. Some forms of two factor authentication may require email/phone access, or even a physical item such as a smart key. With this said, John the Ripper excels at its flexibility. It is not out of the realm of possibilities for a security engineer to find a way to automate the multifactor authentication and incorporate it with John the Ripper’s capabilities.

When done correctly, the multi factor aspect of multifactor authentication is always different in nature. For example, a password is “Something you know” while an email code is “Something you have”. If a security engineer decided to use both a password and a PIN number, then they would have two instances of “Something you know” which decreases the overall security. The main reason for this is because an attacker will only need one target in order to obtain both instances of the information. Most pins that act as a form of authentication in conjunction with a password are usually hardware signed and will only work if done on the specific piece of hardware. Figuring out and understanding different forms of multi factor authentication will give the advantage

1. **Limiting guesses**

John the Ripper needs to be able to compute a large amount of guesses in order to correctly crack the hash. The goal for a penetration tester is to have access to the hashes locally so that they can freely test many passwords on their own computer. If they do not have direct access to the password that they aim to crack, then they are at the mercy of whatever mitigations are put in place on a login screen. Many login screens only allow a certain amount of password guesses within a given time period, which can make tools such as John the Ripper useless.

Some login screens will have incremental guess times, which means that after failing, the time one must wait in order to guess again will keep increasing. This means that no matter how patient an attacker is, it is unreasonable to break the password in this manner. Other Login screens may only allow a fixed number of guesses before notifying the user through another form of authentication. This strategy is to alert the user that they have guessed too many times and could possibly make someone aware that their password is currently being audited. For these cases, a user is normally asked to change and harden their password in order to prevent the malicious actor from future attempts.

1. **Access Control**

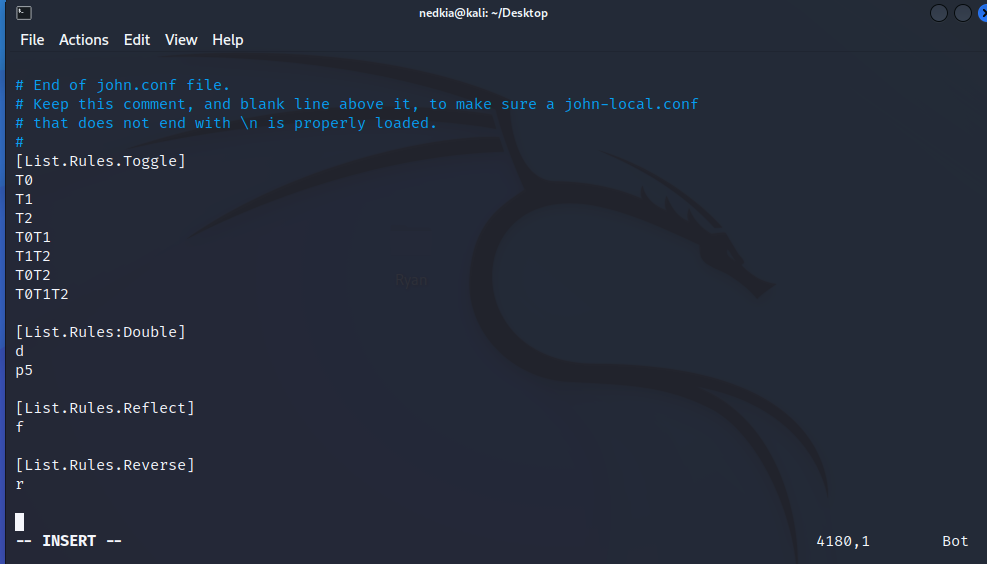
As previously mentioned, John the Ripper works best when we have access to the hash on a local machine. Storing and having access to hashes is extremely important for defensive security professionals as they want to ensure that nobody is able to run these commands and execute any auditing attacks on their hashes.

Access control is a very large process because it involves so many different attack vectors. A security professional will have to take into account how the entire company functions, as well as every single device and its connections. A single compromised smart refrigerator can be the reason why a company has a million dollar data breach.

Technically speaking, storing passwords is done differently depending on the database and/or the operating system. Older legacy software may just store its password hash in a folder within itself. Windows will store some of its hashes for authentication in the Security Account Manager (SAM) database. For a malicious actor to access this folder, they will need to somehow infect the file system, or find a vulnerability that allows them to exfiltrate this data. If the windows machine is on a domain environment, the system administrator may choose to store these hashes on a separate database located in the domain controller server itself (A NTDS database).

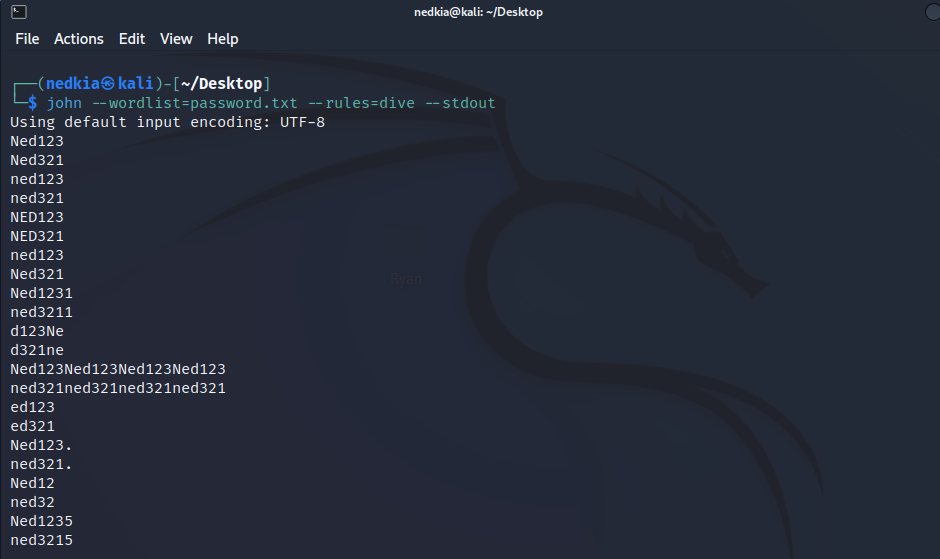
For Linux machines, we can find hashes in the /etc/shadow directory. This directory is only accessible by super or root users and works similar to the Windows directory. Being able to access these folders is an entire process in itself and will usually involve multiple forms of attacks. A tool like John the Ripper is only useful when we have access to a password hash, so in the real world, John the Ripper is only used for defensive hardening purposes, research purposes or at the later stages of a malicious attack.

**Rules**



Rules in John the Ripper act very similar to the –mask command. Rules are used when you know some information about the password and you want to specifically change how the variation of guesses works. Rules are added by looking at the /usr/share/john/john.conf file and manually adding a rule to this file. Rules can create custom variation such as adding specific characters to the end of a word from a wordlist, or doing things such as doubling and reversing the word. Many of these rules are situation based and similar to the –mask command, could be used to drastically improve the time that it takes to crack and audit a password.





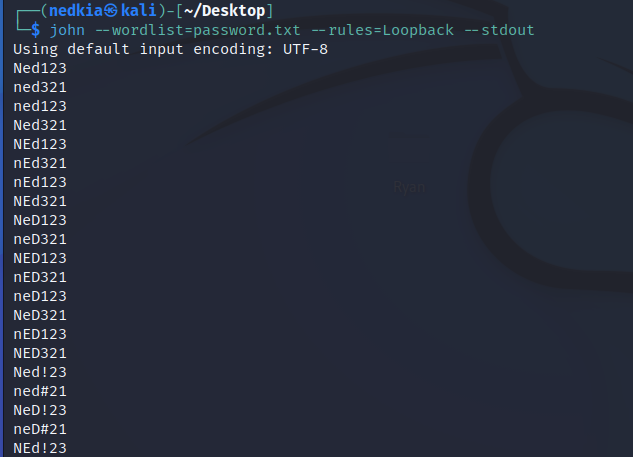
Fun fact: Both the dive and best64 rule are integrations from a similar tool known as Hashcat. Best64 is a set of rules that researchers consider to be very effective because it contains transformations that are optimized for many different types of passwords. Dive is another rule that uses wordlists as well as custom masking to attempt to create similar optimized variations. These rules are one of many things that shows how flexible John the Ripper is and how it is able to integrate aspects of other software into its own configurations.

More examples of potential rules:

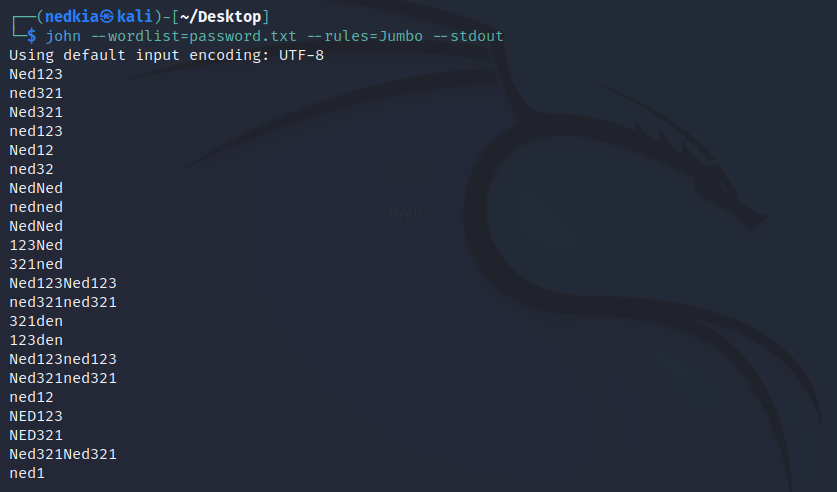
1. InsidePro



1. Loopback



1. Jumbo



<https://dl.acm.org/doi/10.1145/3314183.3324967>

**Integrations into other works (UPAT):**

John the Ripper excels in its flexibility and is able to be integrated into other tools pretty easily. The tool known as UPAT was an interface that attempted to integrate John the Ripper with various other tools into one, easy-to-use interface which would make password auditing much easier. This helps both defenders and attacks because passwords have the potential to be much stronger with a more convenient access to testing, but this tool can also be used to make password cracking attacks easier to do for malicious actors.

UPAT focused on improving John the Ripper in ways in which John has no control over improving upon. One thing that UPAT did was introduce custom wordlists by utilizing Web crawling techniques. UPAT used Open source intelligence (OSINT) to find information about targets, and then automatically generate passwords related to this information. This increases the likelihood of John The Ripper’s success in cracking a password because it makes many people relate their passwords to something about themselves or their life. UPAT is also able to perform file operations on these raw wordlists that are fed to John the Ripper. Many wordlists are just jumbles of information that take a lot of time for John the Ripper to parse through. UPAT can perform file operations to merge, split and extract words from the wordlists which will increase the efficiency of John the Ripper further. UPAT is also able to perform analysis on Wordlists that have a high success rate in order to learn more about why the wordlist is successful. If patterns emerge, then UPAT can use this information to choose or generate better wordlists.

UPAT is able to monitor and regulate GPU temperature which helps the user know if they are overworking their computer when using John The Ripper. John The Ripper loves computer resources and will try to optimize and maximize output from a Processor. Working with a tool like UPAT can prevent hardware and can solve the hardware risk of using John the Ripper. UPAT can also notify people if they are found in a common breach (using haveibeenpwned.com), which further promotes secure password awareness.UPAT is one of many different tools that is integrated and used with John the Ripper. John’s flexibility and useability makes it a very strong tool for many different organizations.

<https://bigdataanalyticsnews.com/artificial-intelligence-key-to-password-security/>

**Integrations into other work (PGAN)**

Artificial Intelligence has recently been affecting many different industries and areas, and password auditing is included. There have been researchers who have been using AI tools to increase their chances of cracking passwords when using tools like John the Ripper. PassGAN is an AI technology software which has been able to outperform other normal cracking tools by integrating AI into its cracking technique. PassGAN is able to find and guess passwords found on social media to further increase the likelihood of correctly guessing a password. AI excels in the way it is able to read and interpret data which makes PassGan very advantageous to use.

Artificial Intelligence and Machine learning can also be used to increase the strength of passwords by dynamically auditing a password strength when it is being created. Programs can be used to help remember the type of password that was used and solved in a breach which could then be used to notify users when their password is exposed or too weak.

Artificial Intelligence has been used on the defensive side of cybersecurity for quite some time now. Heuristic and behavioral based intrusion prevention and detections allows a security researcher to detect anomalies in an environment. If credentials are stolen, the malicious actor will most likely behave in some way that is different from the normal user and this will alert defenders to prevent further access. Artificial Intelligence is able to learn how a user behaves, and use this knowledge to authenticate them and detect abnormalities in a much better way.

PassGAN uses Markov chains, which is used in brute forcing in order to improve the likelihood of cracking a password. This is by choosing brute force combinations that are more likely to be chosen and incorporating this statistic into how the mutations of the brute forcing is done.

PassGAN contains a training model on their github page which allows people to use their own datasets to better create models. PassGAN has some pre-built models already available for use, mainly because the computational overhead for large models is quite significant and users may not have the resources or time available to generate their own models.

<https://github.com/d4ichi/PassGAN>

**Artificial Intelligence and Password Auditing tools:**

**LastPass:** LastPass is a password manager that is used to make remembering complex passwords an easy task. This is done by having one universal password key and then all other passwords are managed by this key. This allows users to automatically generate complex and nearly uncrackable passwords quite easily without having to remember the complex password. In order to generate these complex passwords, LastPass uses an Artificial Intelligence algorithm to create strong passwords and identify any weaknesses in their own generated passwords.

**KeeperSecurity:** Very similar to LastPass, the KeeperSecurity password manager uses a similar AI to determine if a password is weak or not. Keeper does this by having an interval which determines the strength of a password, and then a generator which generates unique and strong passwords that are secure based on the interval.

**PasswordPing:** This is a helpful tool for detecting if a password was compromised and it also attempts to identify any users who are using these compromised passwords. This is especially helpful in a large data breach because many of the breached passwords will be used across the internet.

**PasswordBoss:** Another password manager that uses AI to strengthen their password generation. In general most password managers contain some sort of AI built into them which allows them to generate the most ideal unique passwords for their users.

**Dashlane:** Dashlane is similar to the other password managers, but it works by assigning each generated password a score. The score is determined based on the password strength and this helps users create and generate a very strong unique password.

**Auth0:** This is an access management platform which has AI integrations which acts as an intrusion detection system. This algorithm works by analyzing patterns and potential threats and in real time they are able to detect these threats. Using AI to identify abnormal behavior is a strength of AI.

**Okta:** Another access management platform that has AI integrations in its system. Okta uses a heuristic algorithm to identify login patterns and use this data to detect abnormalities. This way if someone is attempting to brute force or perform another type of attack that causes abnormal behavior, Okta will detect and notify users accordingly.

**Azure Active Directory:** Microsoft’s Azure AD contains AI integrations which allows for the notification of potential security threats. Similar to Okta, Microsoft’s AI is able to detect abnormal behavior and use this to determine if an attack is occuring or not. These attacks can be anything from brute force attacks, password spraying attacks, rainbow table attacks and dictionary attacks.

**OneLogin:** Onelogin is another identity access management platform that allows domain users to log into various different password protected systems. OneLogin’s AI has capabilities to detect phishing attacks and other forms of cyber attacks such as Brute force attacks.

**SecureAuth:** This access management identity platform has its own AI integrations by implementing pattern matching and adapting accordingly. SecureAuth is able to provide various types of authentication security, such as a risk based security Authentication.

**Integrations in Kali Linux (nmap, metasploit, hashcat)**

John the Ripper is a tool that is pre-installed on most modern kali linux distributions. Kali Linux has many other open source tools that are already built in and contain integrations with John The Ripper. Nmap (Network Mapping) is a popular network mapping tool that is used by penetration testers to find information about networks.

1. Nmap is able to use John the Ripper to crack network related authentication hashes, included in services such as SSH, telnet and FTP. This makes network auditing much easier because a professional will be able to test most easy hashes right as they find it and not have to perform any extra steps when auditing.

1. Metasploit, which is a framework used for penetration testing and exploitation also has integrations with John the Ripper to make password auditing easier.
2. Hashcat, a similar tool to John the Ripper, uses some of its capabilities to improve its own password cracking.

There are many integrations of John the Ripper that stem beyond Kali Linux and this shows how flexible John the Ripper is.

Password Breach information: haveIbeenpwned.com

As we have seen, the effect that a breach has on a password is huge and heavily increases the chance of the password being compromised. With so much data being “out there” and with no real way of knowing for sure where our data is being sent to, there isn’t a completely certain way of knowing if your information is safely stored or if it had been leaked somewhere. There are rules and regulations that make it so that companies must inform their clients of data leaks, but this information does not always reach the clients and the fix is instant or perfect.

Some Security researchers decided to attempt to combat these leaks by compiling them and allowing people to search for their own information in these leaks. One such website is haveibeenpwned.com which is a search engine for leaked data. While the website doesn’t give someone access to the actual information, it notifies them if the input has been found in a leak and it also tells them which leak it was. This website encourages users to switch passwords frequently and to be more cautious of where they are putting their information.