Sandfly Security™

Agentless Linux Password Auditing

2024-08-26

Weak and default passwords are a major way to compromise Linux. Not only do many systems have default credentials that can be targeted by attackers, users often select bad passwords for their own use. Sandfly addresses this risk with a very unique feature:

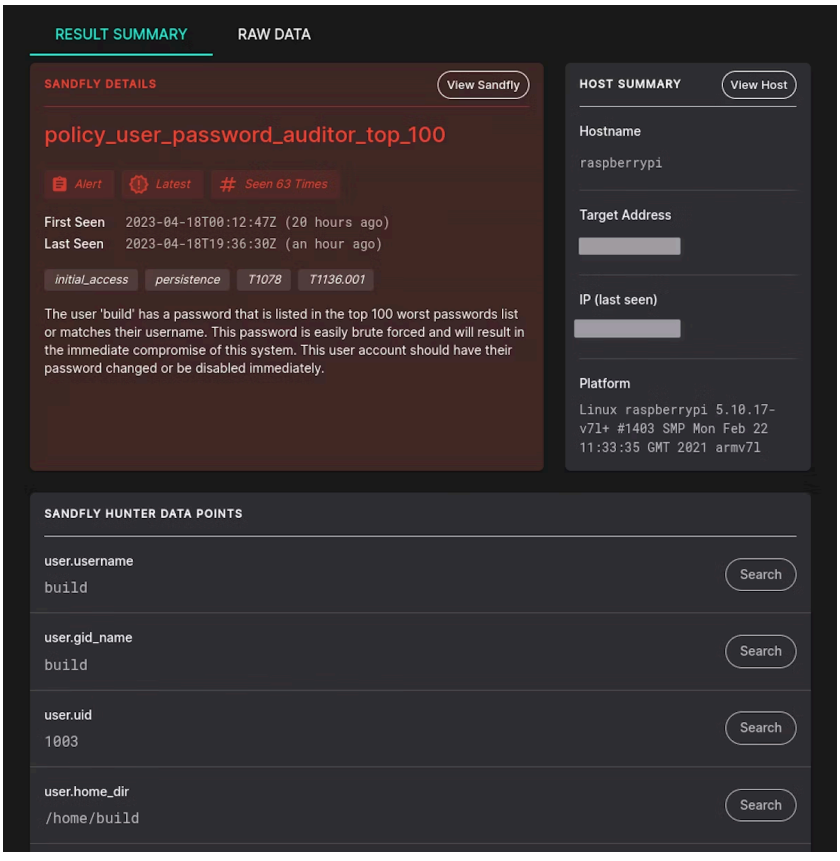
Agentless Password Auditing

Sandfly's agentless password auditor protects all types of Linux systems - whether in the cloud, on-prem, modern, or legacy systems. We even work on embedded and appliance devices. Most importantly, we work without loading intrusive and frequently incompatible agents on endpoints.

Fast and Safe Password Auditing

While we know weak passwords on Linux are a threat, the challenge lies in effectively finding them across all Linux distributions, versions, CPU types, and more. It is especially difficult to check for weak passwords without centrally storing password hashes and risking exposure.

Sandfly solves these problems with password audits conducted directly on systems we monitor agentlessly. Sandfly is able to determine if any accounts on a system have an easily guessed password in seconds.



External Brute Force Attacks are Common

Many attackers use external password brute forcing in order to gain access to systems. As a result, brute force attempts are a common part of background noise and are often overlooked so as not to overload security personnel with false alarms. However, if an attacker happens to hit a legitimate username/password combination, they will look like any other user and many organizations may not notice.

External Brute Force Limitations

External brute force attacks are not limited to intruders, they are also commonly done by security auditors. However, they both share the same limitations that reduce their effectiveness:

- Time delays due to network latency and invalid authentication attempts.
- Auto-banning of brute force attempts.
- Can not see all the accounts on a system that may have weak passwords.

Given these limitations, brute force attacks and external audits are often limited to dozens or, at most, hundreds of attempts before the need to move on. It is simply too slow to run massive lists of passwords, or attackers may get blocked by the local system with automated defenses. For instance, a login delay of five seconds would limit a password audit to at most 12 attempts a minute. This is very optimistic as many systems disconnect after a few bad attempts, and some even block the IP address if repeated attempts are made.

As a result, attackers and security teams typically use small and focused password lists for brute force attacks. They do not launch millions, thousands, or even hundreds, of attempts in most cases. Further, for security teams doing audits, a new problem rears its head as well:

Security teams may miss accounts on systems when doing external audits as they cannot see all usernames present.

For instance, an account named “*adminbackup*” is not a common default user, but if an attacker were to discover this account during their reconnaissance they may attempt to login with it. However external audits may be completely unaware of this risk and not check for this account.

A New Approach: On-Host Password Auditing

As opposed to external audits, Sandfly’s on-host auditing runs on the actual system being monitored as part of our core functionality. This is significantly faster and more accurate than external auditing. It can also be done far more frequently ensuring that bad passwords are rapidly found without risk of banning or generating alerts for security teams. The result:

Security teams get instant password security feedback with virtually no downside.

Sandfly's on-host password auditing for Linux offers numerous benefits, which include:

- Targets highest risk passwords that will lead to immediate compromise.
- Compliance with security policies such as GDPR or PCI, by avoiding the transfer of password hashes off-host for auditing which exposes them to theft[1].
- Compatibility with a wide range of Linux distributions, including legacy and embedded systems.
- Custom password lists.
- Auditing of all user accounts regardless of what they are named.
- Not subject to latency delays or auto-banning.
- Will not flood security teams with brute force false alarms.

By employing on-host password auditing, organizations can enhance security, maintain compliance, and ensure comprehensive password assessments across all infrastructure.

On-Host Auditing Solves Multiple Technical Issues

Sandfly's on-host auditing solves multiple problems discussed above.

Targets Highest Risk Passwords

On-host auditing targets the highest risk passwords likely to be tried by external brute force attempts. It shuts down easy low-hanging fruit attack paths by making sure common passwords are found before attackers get to them.

Doesn't Expose Sensitive Data

Password hashes are never taken off the remote system to audit. This eliminates a potential leakage risk and ensures sensitive data is better protected.

Works on All Linux Distributions

The password auditor works on any system Sandfly can monitor. This includes modern to legacy systems as well as embedded systems and appliances. It also includes most CPU types used on servers and embedded systems.

[1] Under GDPR, hashed passwords are considered personally identifying information (PII). Under US laws such as CCPA, it depends on whether compromise of a hashed password allows access to PII - if an attacker is able to brute-force a stolen password hash and access PII with these credentials, the hash will be considered PII.

Custom Passwords

Many organizations have to deal with users sharing passwords and using common accounts with shared passwords. Sandfly makes it extremely simple to audit for these passwords and eliminate them.

Sees All Accounts

Sandfly directly looks at user accounts on the target system and all users with password hashes are audited regardless of what they are named. Unknown accounts are identified and audited without any gaps in coverage.

Immune to Latency Issues

By moving password auditing to on-host, we eliminate all network and authentication latency issues. Checking for weak passwords directly on a host takes only a few seconds per user on most systems and can cover far more passwords than external auditing ever can.

Won't Cause Banning or False Alarms

Sandfly's password auditing process examines user password hashes directly on the host, rather than conducting brute force attacks against authentication services. As a result, external auto-banning mechanisms, such as *fail2ban*, *IPtables,* and network monitoring systems, are not triggered by excessive login attempts. Users can employ Sandfly's password auditing without concerns of being locked out of their hosts or overwhelming security teams with alerts.

Password Auditing Sandfly Modules

Sandfly has multiple built-in password auditing modules:

user_password_auditor_password_is_username - Username is the password. For example, *root/root, admin/admin*, etc. This is a major and common risk and is enabled by default.

policy_user_password_auditor_top_worst_small_list - User has a password in the approximately top 100 worst passwords. Enabled by default.

policy_user_password_auditor_linux_common - User has a password that is a common Linux user or service name. For example: *apache, nagios, nginx, ansible,* etc. Enabled by default.

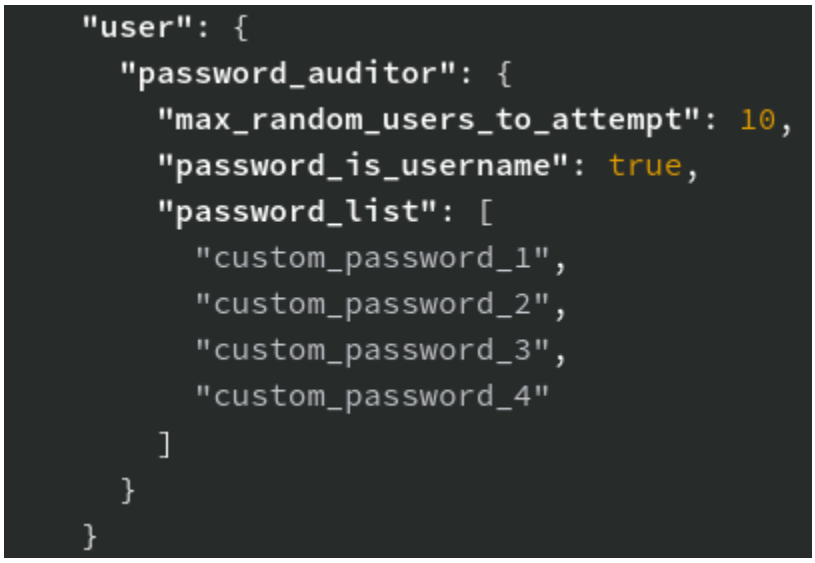
policy_user_password_auditor_top_worst_big_list - User has a password in the approximately top 500 worst passwords. This module is disabled by default, but can be run manually or enabled by the customer for automated scheduling if they desire.

policy_user_password_auditor_custom_password_check - Customer defined list of passwords they want to make sure are not being used anywhere. Ideal for finding notorious shared corporate passwords that linger for years. Can be enabled by the customer as needed.

All password modules can be run on demand or in an automated schedule.

Custom Password Auditing

As with all Sandfly modules, the password auditing checks can be cloned and modified to search for custom threats. For example, many organizations have shared passwords across accounts (the infamous corporate login credential). These passwords pose a particularly high risk as they are rarely changed and they often persist even after employees depart.

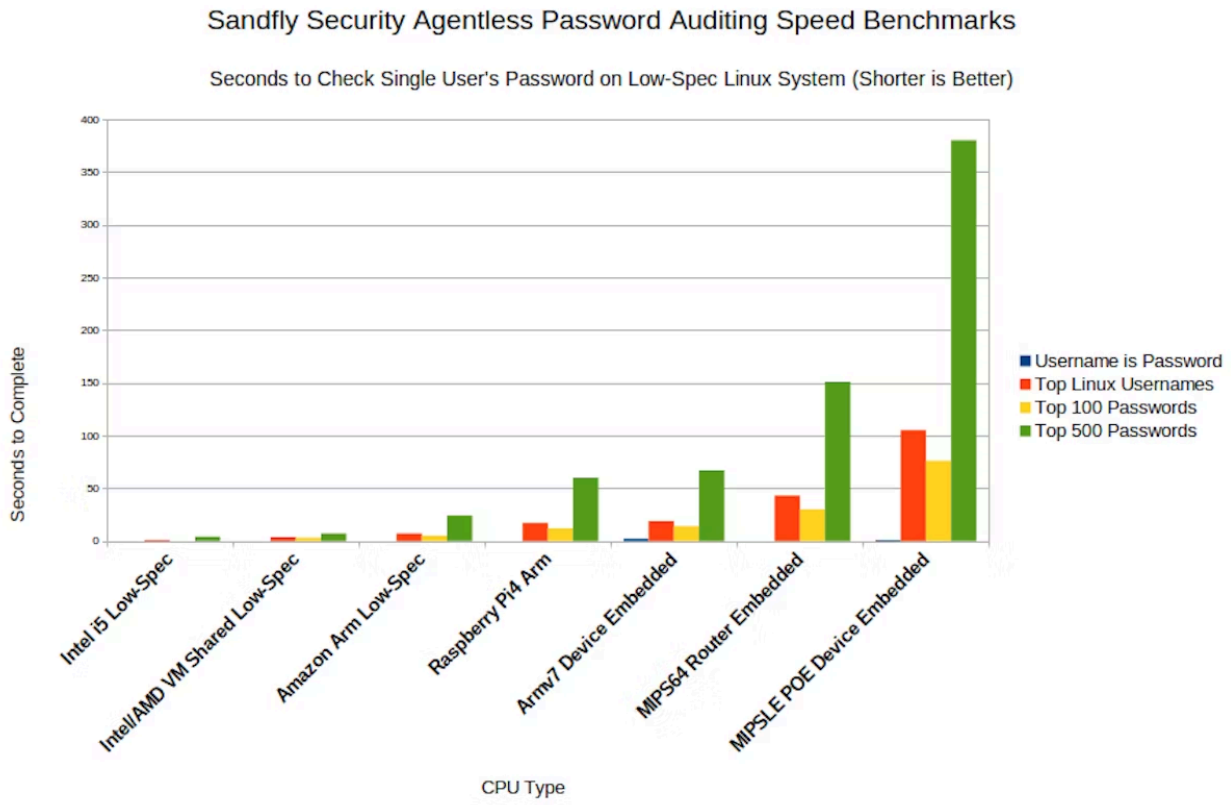


Searching for custom passwords is easy and fast for organizations dealing with this common problem.

Performance Benchmarks

When performing password auditing on the host, it is natural to inquire about potential CPU impacts. Sandfly is not trying millions of passwords, we are instead using small lists that are likely what external attackers would use to gain initial access. Using small lists of high risk passwords gets maximum effect with minimal impacts.

To assess worst-case impacts, the auditing modules were tested on various low-specification Linux systems. The highest-performing system was a low-end Intel i5. Additional tests were conducted on low-end shared CPU cloud VMs, including Amazon ARM EC2 cloud instances. Lastly, the modules were tested on embedded ARM and MIPS Power-Over-Ethernet (POE) network devices (IP Cameras) with the lowest processing power.



The password auditing modules were tested to determine the time required to audit each user across the host performance spectrum. The chart provided illustrates the time taken for each module to audit a single user. For instance, if an audit took approximately 5 seconds, that is for one user; for 10 users, the total audit time would be around 50 seconds.

The username is the password check is nearly instantaneous, even on low-powered embedded devices. Due to its low impact and critical importance, this check is enabled by default.

Top 100 worst passwords and Linux usernames checks exhibit similar performance metrics as the list sizes are nearly the same (about 100 passwords). On modern CPUs, they take a few seconds per user. On low-power devices, such as some embedded systems, each check takes approximately 10 seconds per user.

On Power-Over-Ethernet (POE) network devices, the impact on speed becomes more noticeable as these devices have relatively low processing power. Audits per user can take over 100 seconds for the Top 100 and Top Linux usernames. However, conducting periodic audits on these devices is recommended, as they often have limited users and the default credential risk they pose is significant. **Spending a minute of CPU time to audit embedded devices can save countless hours, days, or weeks of incident response work later.**

Regarding the Top 500 worst passwords, the time taken is still reasonable for bare metal and VMs (even low-end). For embedded devices, there is a time penalty that scales with the number of users. Despite this, occasional manual audits of embedded systems may still be worthwhile if the number of users checked is kept low. For modern high spec CPU systems, the Top 500 checks are fast and a good additional auditing module to enable for most organizations.

Custom password checks would depend on the size of the list, but can be judged by the metrics above for scale.

Limiting Performance Impacts

The password auditing engine incorporates a parameter called *'max_random_users_to_attempt*,' which determines the maximum number of users to be audited in a random manner. By default, this parameter is set to 10 users.

For example, if there are five users with a password hash, all of them will be audited. However, if a system has 21 users, Sandfly will randomly select 10 users for auditing. In subsequent audits, the random selection process is repeated, ensuring that all users are eventually audited over time. This value can be reduced to lower potential impacts even further if desired (e.g. setting it to 2 users max if checking embedded systems).

Another valuable feature is the *'password_is_username'* parameter, which instructs Sandfly to check if the username and password are identical. This check takes only a split second to run, and it is recommended to keep it enabled on all audit modules for optimal security.

The password auditing engine incorporates standard timeout values, ensuring that audits do not exceed a specified duration (default of 360 seconds). This feature guarantees that the CPU will not be overwhelmed with auditing tasks for an indefinite period.

Additionally, since the password auditing process operates on a single core and at low priority, it is unlikely to have a significant impact on the vast majority of Linux systems where it is employed, even in embedded systems.

Password are Still Everywhere

Using Sandfly's password auditing checks is highly recommended for Linux deployments, even when password authentication is not believed to be in use. We have seen instances of customers predominantly employing SSH keys, while password authentication was still present on some systems - for example, being enabled by users without the security team’s knowledge.

For embedded Linux systems and appliances, the likelihood of password usage is considerably higher. It is crucial to perform regular checks to maintain security on these devices.

Auditing for Passwords is Fast and High Value

Sandfly's password auditing feature is very fast and provides instant value. Identifying weak passwords happens in seconds and gives security team feedback on what systems need attention immediately. Denying attackers easy access to systems using brute force methods is critical in defending Linux infrastructure.