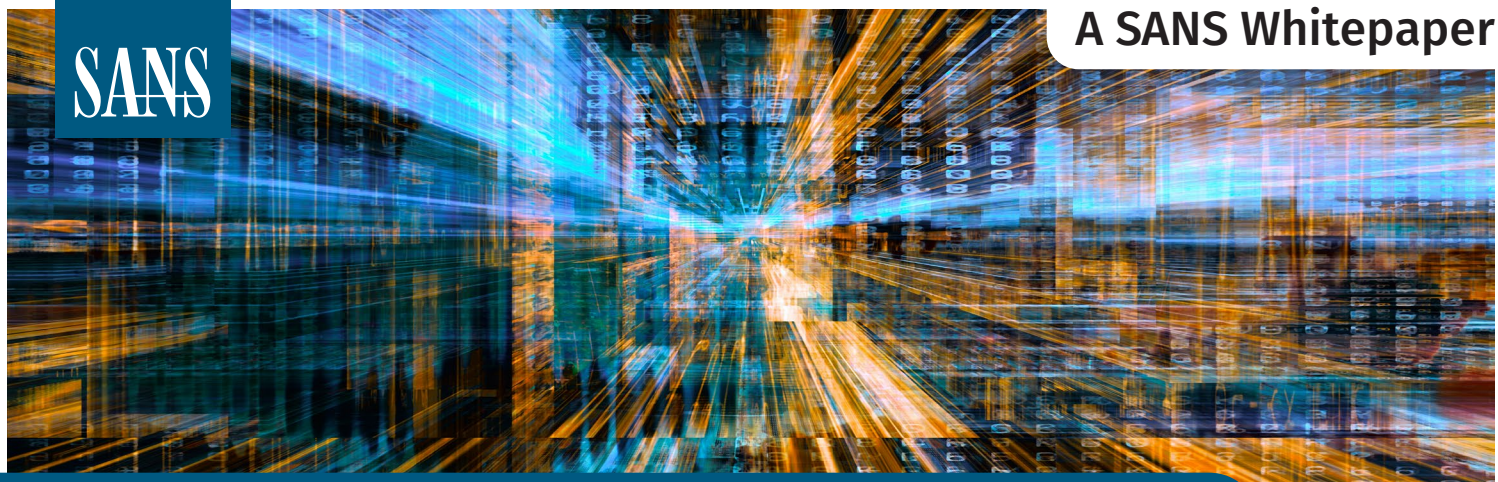
SANS

A SANS Whitepaper



Taming the Wild West: Finding Security in Linux

Written by **Matt Bromiley**

November 2019

Sponsored by:

Cmd

Getting Started

Defensive concepts within information security can oftentimes feel slanted towards one particular operating system or security model. However, in today's modern enterprises, we're seeing this "single-track defense" as highly inadequate simply because enterprise environments today are complex, global and multi-platform networks. Many organizations are increasingly adding more Linux-based platforms to their networks, largely due to the ease of deployment both on-premises and in the cloud. Many users are now familiar with using Linux systems, and this growing trend only adds to the complexities of enterprise defense.

In this paper, we want to discuss modern security concerns within the Linux systems in your environment. Linux presents an interesting juxtaposition: With an API key or a credit card, it's fairly trivial to become a privileged user, perhaps even root, on a Linux system. As today's enterprises see more and more Linux in their environment, it's important that users are provided mechanisms to help secure their accounts while providing the information security team the level of visibility needed to combat threats to the environment.

Linux within the enterprise is not a new phenomenon. Linux has been around for decades and has been implemented heavily in distributed systems deployments and server architecture. The recent growth of Linux-based operating systems within the enterprise has been accelerated sharply by cloud providers, the explosion of containerization and user endpoint freedoms.

As you read through this paper, we challenge you to think about a few key concepts:

- Does your information security team have visibility and insight into the use of Linux systems within your environment?
- What level of permissions are your users given to these systems and how are they using those permissions?
- Do you have a Linux security policy that governs how accounts can be used, ensuring that privileged accounts are not easily abused?

We'll be exploring security of the root user and why this topic is more pertinent than ever to modern enterprise environments. In fact, weeks before the release of this paper, a bug (CVE-2019-14287[1]) was announced that allowed for specific user configurations—meant to prevent the running of a command as root—to be abused to accomplish quite the opposite.

Linux? Security? Why?

Unfortunately, in many organizations Linux is still incorrectly regarded as an "advanced user only" operating system. While Linux fans and longtime users will not disagree with that statement, this perception has caused a natural rift in how Linux systems are monitored and managed within the environment. The larger user base has become more familiar with other operating systems or platforms, causing lopsided interest and investment that ultimately ignores Linux systems. These "advanced users" are simply given trust, without verification, that they are doing the right thing. In today's increasingly cross-platform enterprises, there's really no excuse to disregard any piece of the environment, regardless of perceived relevance or security applications. Threat actors certainly aren't making the same distinctions.

Of course, this stereotypical attitude towards Linux couldn't be further from the truth. While Linux users may consider themselves power users, they share the same security concerns:

- How are user accounts and access mechanisms, such as SSH keys, monitored within the environment? Are they being abused?
- It's trivial for a user to stand up their own system and become root. Are these privileges being treated properly?
- Are my systems vulnerable to any known or unknown exploits? How are we managing patching and visibility into the environment?

Linux in the Cloud: Friend or Foe?

The growing deployment of cloud services is one means by which Linux is growing heavily inside of modern enterprises. Use of cloud infrastructure, however, can present a double-edged sword to many organizations: It can be a fast way to get a concept up and running, but misconfiguration can also create unnecessary (or even previously unknown!) threat vectors for attackers to utilize. Some scenarios to consider:

- **API misuse—**Cloud operations are typically set up via automated scripts and/or helpers that utilize API keys. API keys carry certain privileges, sometimes equivalent to the root of an account, and should be guarded carefully.
- **"Forgotten" resources—**Given the rate that some developers spin applications and services up and down, it's entirely possible to forget that a certain resource was or is still deployed. Without proper maintenance and visibility, this oversight can lead to vulnerable software providing easy-to-exploit access into the environment.
- **External attacks—**Cloud provider IP ranges are well known, and they are equally utilized and targeted by threat actors. Utilizing a cloud provider does not transfer security risk to the provider; instead, it means your information security team must be aware and vigilant.

[1] CVE – CVE-2019-14287, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-14287

In general, many Linux-focused security concerns are similar to other operating systems and should be considered in the risk profile of the overall organization. With respect to Linux systems, two high-level risks to consider are:

- **Operational risks—**These secure assets so users don't accidentally bring environments down.
- **Threat risks—**These assess the likelihood of an internal or external threat actor affecting the environment and are also omnipresent. Later in this section, we'll discuss a recent example of threats maturing against insecure operating systems.

The information security team doesn't lose responsibility for these risks just because an operating system changes. When we consider the impact that a compromised root account could have when layered over either of the above, it's easy to see why we need to protect root accounts just like we do domain administrator accounts.

This brief example is one among many of how root accounts have suffered abuse in years past. With an increase in the usage of Linux-based operating systems and "new" technologies, a development team can spin up a new instance—or *cluster*—of a technology that may have not been vetted or tested by the information security team. And yet, their job is to immediately assume responsibility for securing this data.

Case Study

There's an easy parallel from recent history that illustrates the dangers of unmonitored (or unknowingly acting as) root. Popular NoSQL technologies, such as MongoDB or CouchDB, used to ship "out-of-the-box" bound to the IP address 0.0.0.0, serving on commonly known ports. This delivery method essentially allowed an applicable, deployed NoSQL database to be accessible from any IP address as soon as it was spun up. Behind a strong firewall or inside a closed network, this configuration still meant anyone with a route could access the instance.

Furthermore, administrator accounts were either enabled by default (with publicly-known or blank passwords) or not needed—every user was an admin by default. Database administrators of yesteryear would have been astonished at these configurations. Yet, in an effort to build or try new products, find efficiencies and implement new development practices, these unsafe, misconfigured databases were deployed by the tens of thousands and populated with very real data. And in many cases, they were also **deployed as root**.

It was only a matter of time before threat actors and opportunists realized the combination, and in 2017 attacks came in waves that hit *tens of thousands* of instances nearly simultaneously. Some of the most popular and prolific attacks[2] [3] were waves of product-based ransomware attacks that saw attackers moving from one database product to another, locking up terabytes at a time. With their new web applications unable to access and resource corporate data, some organizations were forced to pay up or cut their losses.

Advanced threat actors, who are not above utilizing well-known vulnerabilities to break into an environment, were paying attention to the technical details of these attacks, as shown in Figure 1.

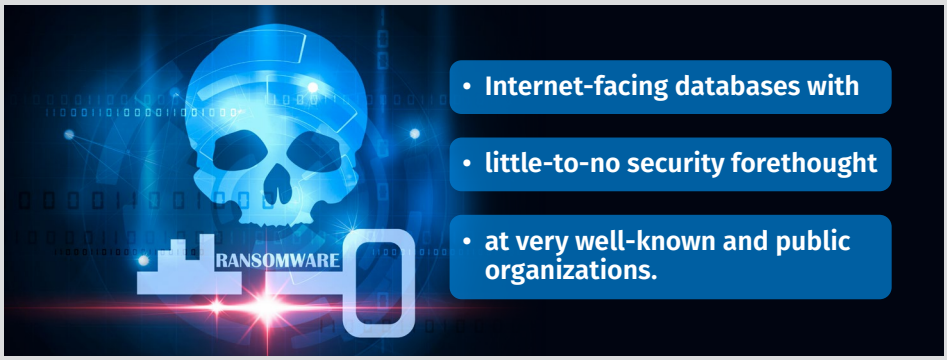


Figure 1. Common Features of Product-Based Ransomware Attacks[4]

It was only a matter of time before formidable exploits for these products were discovered, weaponized and used as an entry-vector into a target organization's environment. Web and database server-side exploits often run with the permissions of the service itself, and these threat actors were pleased to find that in many cases, their instances had been deployed as root. Game over.

[2] "After MongoDB attack, ransomware groups hit exposed Elasticsearch clusters," www.computerworld.com/article/3157770/after-mongodb-attack-ransomware-groups-hit-exposed-elasticsearch-clusters.html

[3] "Elasticsearch Servers Latest Target of Ransom Attacks," www.securityweek.com/elasticsearch-servers-latest-target-ransom-attacks

[4] It is worth noting that since these incidents, many NoSQL database vendors have improved their "out-of-the-box" security defaults, including locking access down to localhost and installing as a least-privileged user. Developers, however, have written and shared scripts that reverse these changes immediately, removing any security forethought.

Perhaps your organization already has account management policies in place. If so, fantastic! The progress bar is far from complete, though. The next step is to ensure they are being implemented and followed.

Modern Day Linux Security Practices

So, where do we begin to secure our Linux-based environments? Fortunately, enterprise account monitoring has received significant attention from the information security community in recent years, and we can lean on many of these principles to define steps forward. It's also helpful to examine the pain points that many daily Linux users encounter and see how we can provide security controls around those.

Don't Run as Root!

The first, and perhaps the most legacy of all Linux security practices, is not to run *anything* unnecessarily as root. Traditionally, administrative accounts have been used for "quick-fire" actions: installing software, killing hung processes, setting up new accounts or cycling a system. Long-running services and daily users should not be performing actions as root—it's simply not needed. An even more effective approach would be that a least-privilege user wouldn't even have the option to escalate to root. We'd expect an escalation and a business justification for this action in other circumstances/systems.

Running services as root is a common "accident" that threat actors love to take advantage of. Many well-known data breaches have started with the compromise of a web-facing Linux system, often running an application that has been deployed to run as root. By running an exploit that provides for Remote Code Execution (RCE) on that system, the threat actor is effectively running their commands as root—without ever compromising the account! This is only one, but perhaps the most public, of the reasons why the root account should be used sparingly.

Unfortunately, this concept quickly enters a tricky area with some organizational setups. In BYOD environments, for example, any user with a credit card can become root by simply buying their own device and using it to connect to the enterprise. After data is brought down to a user's device, any policies wrapped around that data may be downgraded or removed as the organization no longer has control over the endpoint. We are not advocating for the removal of this dynamic, which would be quite impossible. Instead, we encourage you to educate users on the importance of protecting their most important account.

Command Monitoring and Enforcement

Within the Linux operating system, there are certain commands and binaries that are only available to root users—this is by design of the operating system. Any user attempting to run them should cause suspicion, especially if patterns or repeated activity are detected.

Catching command execution in a *historical* fashion, however, still provides the attacker the advantage of time. Live monitoring of command execution in Linux systems can help the security team gain real insight into what users are up to and help build better detections based on suspicious activities. Simply reviewing command history is only half the battle—real-time insight allows for faster account detection.

The other practice to consider for your root users is what you *don't* want them to be able to do easily. Certain command executions, such as those that may move laterally, spawn malicious processes or access sensitive data, might need to be restricted to *only the relevant users*. But wait, how can we limit the root user? The goal is not to limit root but to guardrail what you would expect the account to do. Thus, misuse of root sticks out even more.

Simply put, there are things that the root account shouldn't be able to do. Some command executions are high-fidelity signs that an account may have been compromised and should be monitored, blocked, and/or require additional authentication to be performed.

Profiling Account Usage and Privileged Access

As we've presented, root accounts are typically used for "quick-fire" activities. This usage provides an excellent opportunity to model what the root user should be doing and observe what the user is actually doing. One would hardly expect root users to have flurried, consistent activity across an environment. Session lengths and process execution times (think long sessions and time windows) are ways to look for potentially suspicious root user activity, or at least an abuse of account privileges.

Additionally, information security teams need to be aware of how accounts are accessing the environment. Previously, we could easily wrap monitoring around SSH protocol usage. These days, administrations may use a Bastion host, a browser-based terminal emulator or another web application to gain access into an environment. Again, we can consider account norms and expectations when analyzing connectivity into the environment.

It's worth noting that account management should not only be limited to root. Users within Linux operating systems are still able to affect systems, create files and execute commands (albeit with fewer privileges)—all activities that could still unknowingly provide benefits to a willing threat actor.

Monitoring Workflows in the Environment

A perhaps largely disregarded (or perhaps, assumed-safe), but equally important consideration for your information security teams, is to consider how workflows are utilized within the environment. The usage of automation software in recent years has exploded, and we're seeing tools such as Ansible, Chef, Salt and Puppet be used to rapidly deploy and configure systems at significant scale. Of all the security concerns we've discussed, automation and workflows can potentially open up exposure and/or misconfigurations faster than a team may be able to keep up with.

If they are a critical component of unfolding parts of the organization, then the information security team must be aware of how these workflows are implemented. Similar to policy adherence, workflows should be built with the exact principles we've previously outlined: don't overuse (or even use, if you can avoid it!) the root account when building out the organization.

Monitor Linux Like an Attacker

As we've mentioned several times, there have been significant improvements in understanding how process and accounts can be abused for the benefit of a threat actor. Mitre's ATT&CK Matrix, for example, provides dozens of techniques that are available to an attacker under the right circumstances—all within a Linux-based operating system.

Perhaps one of the first concrete steps to achieving this goal would be to examine the Linux ATT&CK Matrix. If an organization can understand how threat actors have been observed to abuse various operating systems in the past, they can begin to build detections and account protections to counter those. Furthermore, the ATT&CK Matrix provides insight into activities *before* and *after* account compromise. This data can be roped into previously discussed security practices.

Closing Thoughts

The usage of Linux-based operating systems in enterprise environments is not a new concept. With the recent growth of "newer" technologies and a flurry of new developers and development practices, however, the usage of Linux by more users is growing daily. Furthermore, most cloud platforms will default to Linux-based operating systems, meaning as organizations start to expand, we can expect to see more diverse enterprise makeups.

In this paper, we explored the need to secure the root account within your environment. Drawing parallels from case studies where administrative or root accounts have been stolen, it's easy to see how a misused or compromised root account could quickly wreak havoc. Both publicly and privately, we've seen multiple examples of enterprise and data breaches that have either started with or relied heavily on the usage of compromised Linux root users. We also looked at some modern best practices that implement key security controls around Linux instances and their users.

Security, however, isn't something that can simply be turned on, regardless of operating system. It begins with knowledge of the environment, visibility into the systems and processes and an understanding of how those systems contribute to business operations. Users must also be educated on the power of their account privileges, the potential impact of account misuse and how to best safeguard their access. With these pieces in place, the information security team is ready for any technology on the horizon.

About the Author

Matt Bromiley is a SANS Digital Forensics and Incident Response instructor, teaching FOR508: Advanced Digital Forensics, Incident Response, and Threat Hunting and FOR572: Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response. He is also an IR consultant at a global incident response and forensic analysis company, combining his experience in digital forensics, log analytics, and incident response and management. His skills include disk, database, memory, and network forensics, incident management, threat intelligence, and network security monitoring. Matt has worked with organizations of all shapes and sizes, from multinational conglomerates to small, regional shops. He is passionate about learning, teaching, and working on open source tools.

Sponsor

SANS would like to thank this paper's sponsor:

>_cmd