SANS ANALYST PROGRAM

Sponsored by LogRhythm

Keys to the Kingdom: Monitoring Privileged User Actions for Security and Compliance

A SANS Whitepaper – May 2010

Written by: Dave Shackleford

Insider Threats and Controls

Monitoring and Managing Privileged Users

Better Reporting through Correlation

Privileged User Monitoring for Compliance

Introduction

For years, the information security community has debated whether the threat of internal attack or external attack is of the greatest concern for organizations. Security practitioners have generally come to the conclusion that the volume of external attacks is far greater than internally-based attacks, simply due to the number of probes and attacks pounding their networks every day. On the other hand, despite their smaller volume, inside attacks generally cause significantly more damage because the attackers already have access. Nowhere is this more applicable than with privileged users. A privileged user, by definition, is a "[u]ser who, by virtue of function, and/or seniority, has been allocated powers within the computer system, which are significantly greater than those available to the majority of users."[1]

One study that has tracked external and internal incidents is the annual CSI/FBI survey. In 2004, the number of incidents attributed to inside attacks was roughly the same as that from outside (52 percent of respondents stated that one to five incidents had likely occurred for each).[2] In the 2007 report, respondents indicated for the first time that insider abuse of network access and email were bigger problems than malware.[3] In the 2009 survey, a new category, "Unauthorized access or privilege escalation by insider," was created to try and capture more granular statistics on this threat. In the 2009 survey, 15 percent of respondents experienced this particular issue.

Internal attacks cost United States businesses $400 billion per year, according to a national fraud survey conducted by The Association of Certified Fraud Examiners. Of that, $348 billion can be tied directly to privileged users, according to a CSO Online article.[4] This statistic illustrates the percentage of actual losses attributed to privileged, as opposed to regular, computer users.

Monitoring the actions of these users is paramount for security and compliance reporting. However, monitoring at this level has been a challenge for many organizations. This paper explores some of the types of insider threats organizations face today and discusses monitoring and managing privileged user actions and the role this level of monitoring plays in today's compliance reporting efforts.

[1] www.yourwindow.to/Information-Security/gl_privilegeduser.htm
[2] http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2004.pdf
[3] www.darkreading.com/security/perimeter/showArticle.jhtml?articleID=208804727
[4] www.csoonline.com/read/040106/caveat041206.html

Insider Threats and Controls

There are many types of insider attacks, both malicious and unintentional, including:

- **System sabotage:** This type of attack is malicious in nature and usually consists of a disgruntled insider destroying data or rendering an operating system or applications unusable in some way. Two case examples of this type of administrative sabotage (a logic bomb and a network Denial of Service via lockout) follow as case studies at the end of this section.
- **Theft of assets or data:** Usually malicious, this attack can be very difficult to identify and may be one of the most damaging overall. In November of 2009, T-Mobile revealed that an employee in the United Kingdom had stolen and sold millions of customer records to data brokers, resulting in the UK's largest data breach to date.[5]
- **Introduction of "bad code:"** An attack of this nature may be deliberate or accidental (for example, by haplessly using poor coding practices or purchasing bad code). It is usually attributed to developers or other IT professionals who have access to code or scripts used by an organization. One example of this is the logic bomb coded into a program by a rogue developer at Fannie Mae, described later in this paper.
- **Introduction of malware:** This is an attack that may not be deliberate in nature; many malware infections are unintentional. The higher the user's privileges are, however, the more devastating the attack can be. One particularly insidious example the introduction of malware occurred in 2002, when the popular packet sniffer tcpdump was infected with a Trojan on the download site, infecting numerous network engineers' systems.[6]
- **Unauthorized hardware and software:** By introducing wireless access points, USB storage devices, and unapproved software into an organization, insiders may introduce new threats and vulnerabilities to the environment.
- **Social engineering:** It is often said that the weakest link in the chain of security is people, and by exploiting them, insiders can easily bypass policies and controls. Such attacks may range from the innocent (talking an administrator into installing a software package a user wants) to malicious (a help desk analyst talking a user out of her password to gain access to her files).
- **Accidents:** Even accidental actions by insiders cause tremendous damage. In the 2009 CSI/FBI survey, 25 percent of respondents felt that nonmalicious insider actions caused over 60 percent of their financial losses!

Two recent case examples of insider attacks help to illustrate the importance of privileged user monitoring. As was the case in these examples, other security practices were not being followed.

[5] www.darkreading.com/database_security/security/privacy/showArticle.jhtml?articleID=221900209
[6] www.cert.org/advisories/CA-2002-30.html

Case Study 1: The Fannie Mae Logic Bomb

On October 24, 2008, a UNIX engineer at Fannie Mae named Babubha Makwana was informed that he would be let go from the company at the end of the day. Rather than following best practice of immediately revoking all system access and escorting him from the building, Fannie Mae allowed Makwana to stay on site and finish the work day. During this time, he created a series of scripts that could have caused enormous damage to the company upon execution, first by disabling monitoring, then disabling all system access to Fannie Mae's 4,000 servers, and finally wiping all data from the servers and their backup systems. The code that launched the series of scripts, which was set to trigger on January 31, 2009, was embedded in a key script that ran every morning. Fortunately for Fannie Mae, another engineer found the embedded logic bomb before it went off and alerted authorities.[7]

In this particular attack, the engineer had extensive privileges to both systems and code. Logs and other audit trails were present, revealing Makwana's logins on that day. However, no alarms were triggered by his activities because he was a privileged user who was not being monitored. Fannie Mae was lucky: The logic bomb was discovered by accident.

Case Study 2: San Francisco's Rogue System Administrator

In July 2008, Terry Childs was put in jail after locking every member of San Francisco's IT staff out of all critical systems on the city's network. These systems, connected to a distributed network called FiberWAN, which Childs helped develop, were secretly locked down with a single password known only to Childs. When Childs' supervisors demanded the password, he refused to divulge the information, even after he was arrested and jailed. The refusal left most of San Francisco's working network unusable for days, until he finally gave them the master password.[8]

Childs had complete and unfettered access to the most critical systems in the network, and significantly more care should have been taken to monitor what he was doing on a day-to-day basis. A privileged user monitoring strategy could, potentially, have thwarted this attack, saving the city significant time and money. Privileged user monitoring is a large part of Carnegie Mellon Computer Emergency Response Team's (CERT),[9] recommendations for prevention of insider threats and for compliance reporting of administrator actions.

[7] www.wired.com/threatlevel/2009/01/fannie/
[8] www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2008/07/16/BA4011PFJP.DTL&feed=rss.bayarea
[9] www.sei.cmu.edu/library/abstracts/news-at-sei/securitymatters200702.cfm

According to CERT, mechanisms to prevent privileged insider abuse should include the following:

- **Enforce separation of duties and least privilege.** *Separation of duties* implies that no one employee can perform all privileged actions for a system or application. Least privilege implies that employees are granted only the bare minimum privileges needed to perform their jobs.
- **Implement strict password and account-management policies and practices.** This should be enforced for all users, including administrators and other privileged users.
- **Log, monitor, and audit employee online actions.** Organizations need to be vigilant about what actions privileged users are taking, and should use a variety of logging and monitoring techniques.
- **Use extra caution with system administrators and privileged users.** Because these users are often granted the "keys to the kingdom" in terms of access and capabilities, additional safeguards often need to be implemented to adequately monitor and manage their behavior.

Monitoring and Managing Privileged Users

The last two points on the list of CERT best practices for mitigating insider threats warrant closer attention. First, consider the importance of the previous point, "Use extra caution with system administrators and privileged users." This pertains to both managing and monitoring privileged users. Managing privileged users is usually accomplished by the following:

- **Creating and enforcing policies that forbid the use of single, "all powerful" accounts:** This is the first step to managing the actions privileged users can take. In order to define more granular roles and privileges for privileged users, each user must have a unique user account or user ID that can be tied to him or her.
- **Leveraging privilege control tools:** Tools such as sudo[10] can have policies defined for various users and groups that control what actions can be taken and what permissions are available to specific resources. For example, user accounts associated with managing particular applications or services can be granted explicit privileges only to those resources, and no others. In addition to controlling privileges, more detailed audit trails can be associated with much more specific actions taken on systems.

Monitoring privileged users relies heavily on log and event management. One of CERT's recommendations for reducing the risk of insider threats is to "Log, monitor, and audit employee online actions." Although this is intuitive from a security management perspective, putting this policy into practice is often much easier said than done. In order for security professionals to create alerts based on log events associated with privileged users, the information present in the logs needs to be clear and understandable. Unfortunately, this is usually not the case on most modern computing platforms. On Microsoft Windows platforms, for example, there are numerous events generated that relate to privileged use and users, most of them difficult to decipher, as shown in Figure 1:

Type	Date	Time	Source	Category	Event	User
Success Audit	4/24/2006	6:46:39 AM	Security	Detailed Tracking	593	Administrator
Success Audit	4/24/2006	6:46:13 AM	Security	Detailed Tracking	600	SYSTEM
Success Audit	4/24/2006	6:46:13 AM	Security	Detailed Tracking	592	SYSTEM
Success Audit	4/24/2006	6:45:33 AM	Security	Logon/Logoff	538	SYSTEM
Success Audit	4/24/2006	6:45:22 AM	Security	Logon/Logoff	540	SYSTEM
Success Audit	4/24/2006	6:45:22 AM	Security	Privilege Use	576	SYSTEM
Success Audit	4/24/2006	6:45:22 AM	Security	Logon/Logoff	540	SYSTEM
Success Audit	4/24/2006	6:45:22 AM	Security	Privilege Use	576	SYSTEM

Figure 1: Windows Event Logs Related to Privileged Use and Users

[10] www.gratisoft.us/sudo

Many of these events are vague, even with more detailed information provided. For example, the event shown in Figure 2 doesn't tell us if anything is really happening that should be reported.

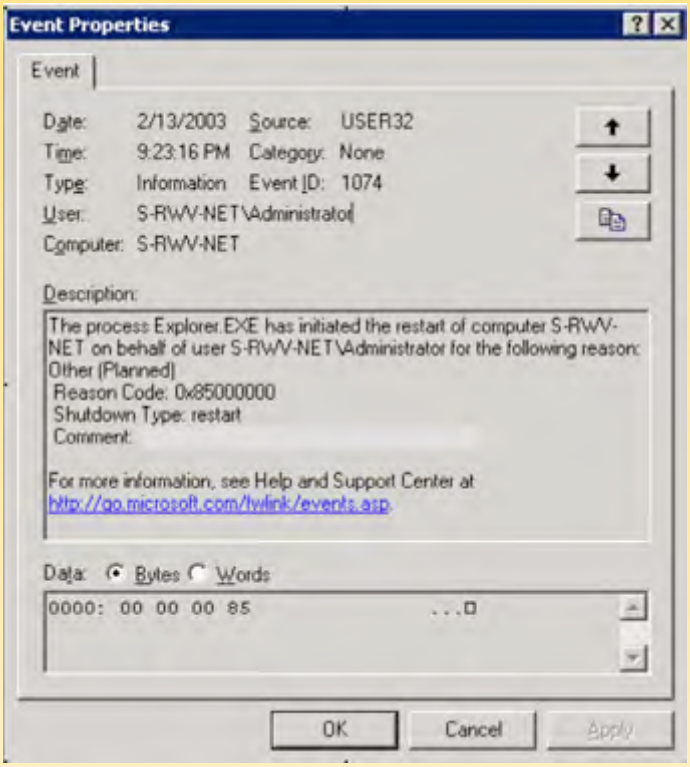

Figure 2: A Vague Windows Event Log

In Figure 2, what appears to be a local Administrator account (S-RWV-NET\Administrator) caused the computer to restart (Event ID 1074) by way of the **Explorer.exe** process. This occurred at 9:23PM. What does this actually mean, though? What action did the Administrator take to cause this event to occur? If manually shut down (a possibility here, as the reason stated is shown as Planned), was this an approved change? In addition, who was this Administrator? No specific user name is given, nor is any additional context provided explaining where the Administrator logged in from. Starting with Windows Vista, the event IDs and other identifying information changed for all Windows security events, making logging and alerting for privileged user events even more confusing.

What about UNIX logging? Although UNIX logging has traditionally been easier to accomplish on a wider scale (primarily due to native Syslog integration for quite some time), the information found in UNIX logs is often vague as well, particularly related to privileged user activities. One of the reasons for this is the widespread use of a shared root user account that is employed by multiple administrators for controlling servers. This prevents any log events from describing what activities were carried out by a particular user ID, as shown in Figure 3:

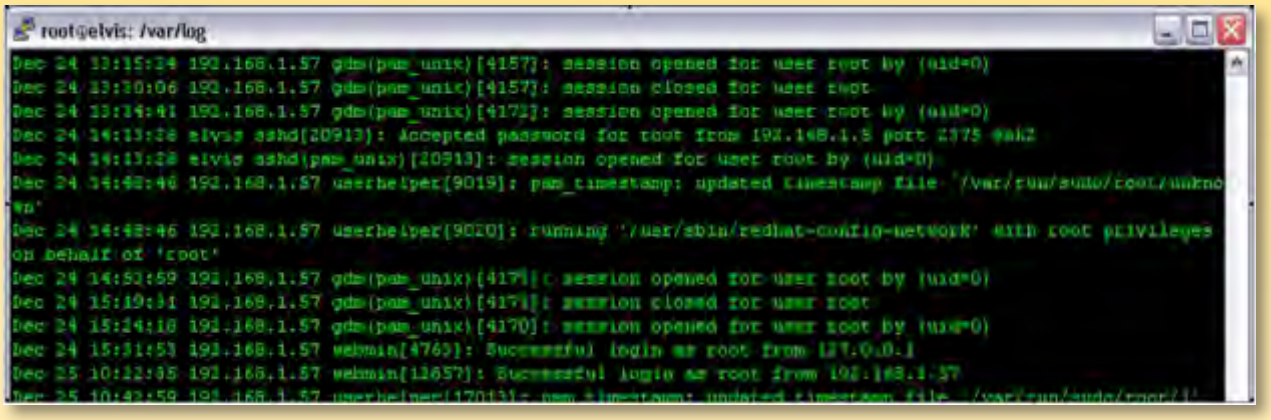

Figure 3: Generic UNIX Log Entries

In addition to the lack of specificity found in many of these logs (the exceptions being those logs generated by privileged user management software like sudo), there is rarely any good way to relate logs from one system to another. Consider the logs shown in Figure 4:

Log Type	Log Entry
Firewall	[00001] 2006-05-25 13:34:33 [Root]system-alert-00008: IP spoofing! From 10.1.1.238:80 to a.b.c.d:49807, proto TCP (zone Untrust, int ethernet3). Occurred 1 times.
RADIUS	05/25/2006,13:34:55,Authen failed,dave,admin,122.55.32.13,External DB user invalid or bad password,,,16,10.2.3.1
Antivirus	240801012128,5,1,720997,RBLWAP,SYSTEM,Trojan Horse, C:\WINDOWS\TEMP\win1C.tmp,5,4,4,256,570441764,"","",0,,0,,0,25464,0,0,0,0,0,0,20060830.022,58100,2,4,0,acme-AVSRV,{579642AA-5A5E-46E1-8613-2289349D1F27},,(IP)-192.168.100.237,acmeav,acme,,8.1.825

Figure 4: Various System Log Events (Highlighted Areas Provide Some Helpful Information)

These logs tell us more about potential events, but don't tell us if these were all related? For example, what if an administrator had connected via VPN to the corporate network and her system at home was infected with malware that tried to spread? This is only one simple case where logs may be generated, but the logs don't tell the full story without some additional correlation. Consider the next hypothetical case study.

Hypothetical Case Study: A Competitor Has Too Much Information

ABC Corp. is a Fortune 500 manufacturing firm with over 50,000 employees worldwide. Their central headquarters is in New Orleans, with all major data center facilities there or close by. The IT infrastructure is comprised of both Windows and Linux servers; remote access is available via VPN; and there are numerous systems administrators who enjoy relatively unfettered access to the systems they manage. Through various channels, the company realizes that a competitor has gained access to sensitive intellectual property, namely development plans for a new product. Was their system hacked? Was this a case of corporate espionage?

The first steps taken by the security team involved reviewing the IDS events and firewall logs. Nothing unusual appeared to be there, other than the predictable scanning and routine attacks from the Internet in general. If the attack came from the outside and involved exploits or malicious code, it must have been very stealthy, indeed. The team then turned to the system-level logs, particularly from databases and several related application servers that stored sensitive data related to the project. The application logs reveal the following:

- A large volume of "success" logs that correspond to service account logins from Web servers
- Database access logs that align with the application server logs
- Occasional logins from developers for maintenance or changes
- System logs with many instances of root and *Administrator* logins

The last point is the most difficult: How do you determine which root and Administrator users logged in? And, what did they do once logged in? The security team had trouble differentiating between actual users sharing the generic admin accounts and mapping this activity to location. As many members of the IT staff used the VPN to access the network remotely, this access could really have been from anywhere. In this scenario, it turns out that one rogue admin had logged in from a VPN connection, accessed these systems by logging in as root or Administrator directly, and copying sensitive files elsewhere. How could this have been detected? Let's explore a strategy for monitoring privileged users and look at several logging and correlation examples in the following sections.

Better Reporting through Correlation

First and foremost, all organizations need a policy and approach to privileged user management and monitoring. By examining the levels of access needed to perform most daily activities and crafting policy that supports these job functions while still maintaining separation of duties and the principle of least privilege, organizations are getting off to a good start. Processes for granting access should be in place and in line with the policies, and periodic audits should also be a part of the program. The next step is to implement traditional security controls, such as firewalls and network access controls, limiting access to systems, intrusion detection systems, identity and access management solutions, and so on. All of these systems should have logging enabled, and system logging for all user activity should be in place, especially for privileged actions and logon/logoff events. Implementation of a privileged user management product or tool like sudo can go a long way to lending context and control to the program, as well.

Even with all of these aspects in place, monitoring privileged user activity is still likely to be challenging. The reason for this is simple: There is no native correlation between any of the events and logs being generated. Returning to our case study, consider the logs shown in Figure 5, in which we've highlighted the separate activities that are not correlated:

Log Type	Log Entry
VPN	01 21 22:31:43 VPN.ABC.COM %ASA-6-113004: AAA user authentication Successful : server =1.2.3.4: user = BSMITH
DHCP Server	Jan 21 22:31:54 dhcp.abc.com dhcpd: DHCPDISCOVER from 00:0c:76:8b:c4:16 via eth0 Jan 21 22:31:58 dhcp.abc.com dhcpd: DHCPOFFER on 172.16.23.21 to 00:0c:76:8b:c4:16 (Smith_HomePC) via eth0
Server	Jan 21 22:33:08 AppServer sshd[8813]: Accepted password for root from 172.16.23.21 port 1066 ssh2
Firewall	Jan 21 23:07:56 firewall: NetScreen device_id=firewall [Root]system-notification-00257(traffic): start_time="2010-01-21 23:07:56" duration=14 policy_id=119 service=tcp/port:22 proto=6 src zone=Trust dst zone=Untrust action=Permit sent=23402 rcvd=23402 src=172.16.99.99 dst=1.2.3.4 src_port=3036 dst_port=22

Figure 5: Separate System Logs Related to Hypothetical Data Theft

It's hard to gather from the above four log reports what was really happening: An authorized user is logging in over VPN, escalating privileges, and shipping data out over SSH. Figure 6 is a diagram of the steps the admin took.

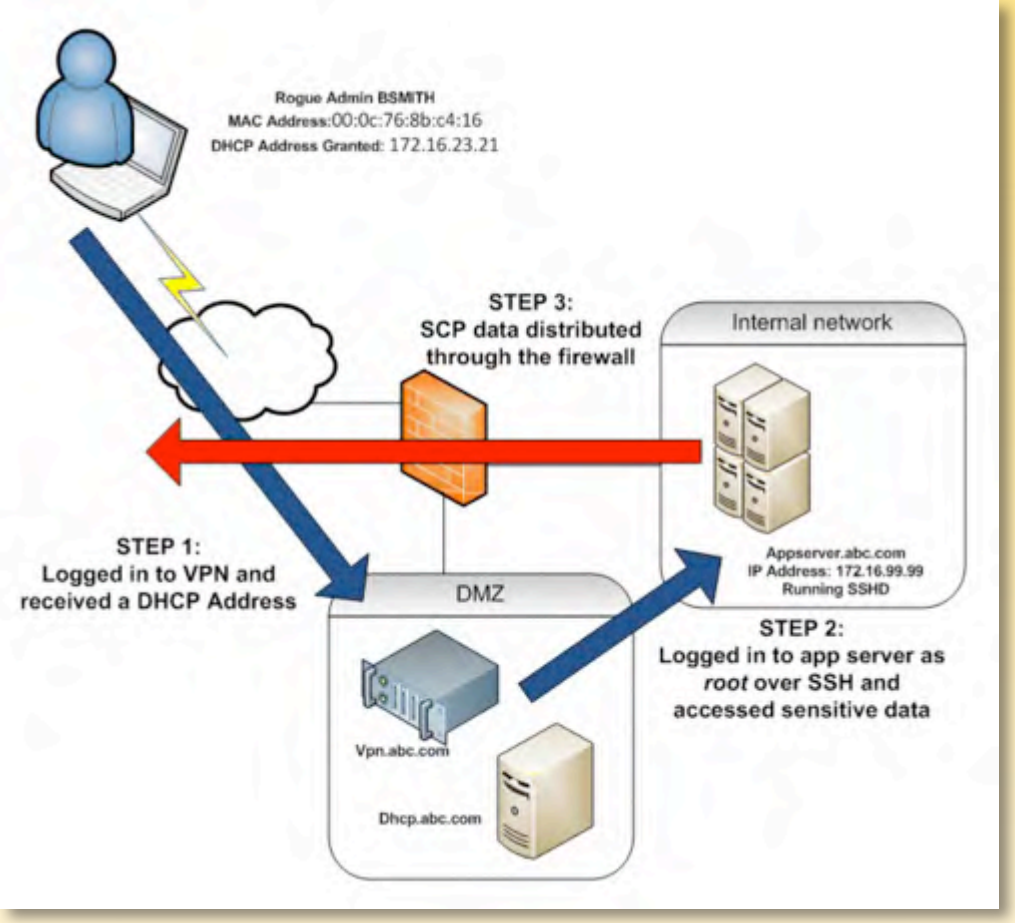

Figure 6: A Rogue Administrator in Action

The administrator connected to the VPN from her home machine and logged in with her normal User ID of *Bsmith*. She was then granted a DHCP address of 172.16.23.21 to her home system's MAC address of 00:0c:76:8b:c4:16. She then logged in as root, using SSH to the application server at IP address 172.16.99.99. A short time later, the firewall logged a successful outbound SSH/Secure Copy (SCP) session to address 1.2.3.4 from the app server, where the admin was using SCP to send data outbound.

Sounds simple enough to catch, right? If the proper correlation and monitoring capabilities are in place, it could be. Usually, however, this is not the case.

Privileged User Monitoring for Compliance

Many organizations are facing both government and industry compliance requirements that involve implementing policies, audit processes, and security controls. Several of these call out privileged user management and monitoring specifically. Two examples are the Payment Card Industry Data Security Standard (PCI DSS) and the Federal Financial Institutions Examination Council (FFIEC) Information Security Booklet.

The PCI DSS is comprised of 12 sections, each focusing on a major aspect of information security programs. Section 10 is labeled "Track and monitor all access to network resources and cardholder data," and contains two subsections that require privileged user monitoring:

- **Section 10.1:** "Establish a process for linking all access to system components (especially access done with administrative privileges such as root) to each individual user"
- **Section 10.2.2:** "Implement automated audit trails . . . [for] all actions taken by privileged users"

These requirements directly follow best practices, namely, to disallow the use of generic privileged user accounts such as root and Administrator directly (with tools like su[11] and sudo), and also to generate and maintain logs related to all privileged user activity.

For financial and banking applications, the FFIEC Information Security Booklet is a primary source of guidance that includes privileged user monitoring in its best practices guidelines. The guidelines specify that "[a]uthorization for privileged access should be tightly controlled." Privileged access refers to the ability to override system or application controls. Good practices for controlling privileged access include:

- Identifying each privilege associated with each system component
- Implementing a process to allocate privileges and allocating those privileges either on a need-to-use or an event-by-event basis
- Documenting the granting of and administrative limits on privileges
- Finding alternate ways of achieving the business objectives
- Assigning privileges to a unique user ID where different from normal business use
- Logging and auditing the use of privileged access
- Reviewing privileged access rights at appropriate intervals and regularly reviewing privilege access allocations
- Prohibiting shared privileged access by multiple users

These guidelines address the same best practices as PCI DSS does in terms of unique IDs and logging; and, they go further, by specifying policies and auditing requirements for privileged user activities. Most other compliance mandates have either direct or implied requirements to manage and monitor privileged user access, as well.

[11] http://en.wikipedia.org/wiki/Su_%28Unix%29

Conclusion

There are many reasons to pay attention to privileged user activity. Aside from the risk of malicious behavior from insiders, even accidental activities can have disastrous consequences due to excessive privilege use.

Many compliance mandates are now also stipulating the management and monitoring of privileged user activities, ranging from policy definition to implementation of least privilege and logging requirements.

For these reasons, it's critical to restrict the range of activities privileged users can perform and monitor all activity closely via logging and other means. Even then, your procedures may not be enough to truly identify all aspects of privileged user behavior, because numerous events can occur at different times on different systems.

To truly monitor privileged users, organizations will also need to correlate these events, providing deeper insight into how they relate and which users are performing the actions. If organizations don't take such steps, they not only risk losing the keys to their kingdoms, but also risk violation of privileged user monitoring and correlation requirements as specified by numerous best practices and regulatory requirements.

About the Author

Senior SANS Analyst, Dave Shackleford, is Director of Security Assessments and Risk & Compliance at Sword & Shield Enterprise Security, a SANS instructor and GIAC technical director. He has consulted with hundreds of organizations in the areas of regulatory compliance, security, and network architecture and engineering. He has worked as Chief Security Officer for Configuresoft, Chief Technology Officer for the Center for Internet Security, and has also worked as a security architect, analyst, and manager for several Fortune 500 companies.

SANS would like to thank this paper's sponsor: