Monitoring up the Stack: Adding Value to SIEM

Version 1.2
Released: October 29, 2010

Securosis, L.L.C. 20930 North Tatum Blvd Suite #110-116 Phoenix, AZ 85050 **T** 602-412-3051 info@securosis.com www.securosis.com

Author’s Note

The content in this report was developed independently of any sponsors. It is based on material originally posted on the Securosis blog <http://securosis.com>, but has been enhanced, reviewed, and professionally edited.

Special thanks to Chris Pepper for editing and content support.

Licensed by ArcSight

ArcSight (NASDAQ: ARST) is a leading global provider of cybersecurity and compliance solutions that protect organizations from enterprise threats and risks. Based on the market-leading SIEM offering, the ArcSight Enterprise Threat and Risk Management (ETRM) platform enables businesses and government agencies to proactively safeguard digital assets, comply with corporate and regulatory policy and control the internal and external risks associated with cybertheft, cyberfraud, cyberwarfare and cyberespionage. For more information, visit www.arcsight.com.

Contributors

The following individuals contributed significantly to this report through comments on the Securosis blog and follow-on review and conversations:

Rick Caccia — ArcSight
Andrew van der Stock

Copyright

Table of Contents

Introduction

“How can I derive more value from my SIEM installation?” It comes up over and over again. Significant investments have been made in SIEM and Log Management platforms, and the evolving nature of attacks means end users are looking for more ways to leverage their security investments. SIEM/Log Management does a good job of collecting data, but extracting actionable information remains a challenge. In part this is due to the “drinking from the fire hose” effect, where the speed and volume of incoming data make it difficult to keep up. Additionally, the data needs to be pieced together with sufficient reference points from multiple event sources to provide context for the threat. But we found that the most significant limiting factor is often a network-centric perspective on data collection and analysis. As an industry we look at network traffic rather than transactions. We look at packet density instead of services. We look at IP addresses instead of user identity. We lack *context* to draw conclusions about the amount of real risk any specific attack presents.

> *How do I derive more value from my SIEM installation?*
>
> *That’s the question we aim to answer in this report.*

Historically, compliance and operations management have driven most investment in SIEM, Log Management, and other complimentary monitoring investments. SIEM can provide *continuous* monitoring, but most SIEM deployments are not set up to provide timely threat response to application attacks. And we all know that a majority of attacks (whether it’s 60% or 80% doesn’t matter) focus directly on applications. To support more advanced policies and controls we need to peel back the veil of network-oriented analysis and *climb the stack*, looking at applications and business transactions. In some cases, this just means a new way of looking at existing data. But that would be too easy, wouldn’t it? To monitor up the stack effectively we need to look at how the architecture, policy management, data collection, and analysis of an existing SIEM implementation must change.

What kind of changes are we talking about here? We can take some clues from existing capabilities such as business process analytics and fraud detection which require different policies, additional data, and additional analysis techniques. But that’s just technology, and pretty much misses the point. What’s *really* different of monitoring at the application layer? Application awareness and context.

When looking for ways to derive more value from SIEM, our research showed that as SIEM moves from network and security monitoring to monitoring people, applications and information, understanding the application context was necessary in order to implement policies and perform more advanced analysis. This is a key theme throughout this document, regardless of the specific technology or application type being monitored.

To highlight the differences in why network and security event monitoring are inherently limiting for these enhanced monitoring use cases, consider that devices and operating systems *do not represent business processes*. In some cases their events lack the information needed to perform useful analysis, but more often the policies and analysis engines are just not set up to detect fraud, spoofing, repudiation, and injection attacks. By the way, this isn't to damn SIEM — the offerings weren't originally designed to monitor up the stack. SIEM was designed to reduce events coming from network and security devices. But now the problem is less about event reduction and more about application attack defense, so the tools must evolve.

To highlight this disconnect from the application perspective, network identity and user identity are *extremely* different. Analysis, performed in context of the application, provides contextual data which is not available from only network and device data. It also provides an understanding of *transactions*, which are much more useful and informative than pure events because you can understand why something is happening, not just that it happened. Evolving threats target data and application functions, and we need that perspective to understand and keep up.

Ultimately we want to provide business analysis and operations management support when parsing event streams, which are the areas traditional SIEM platforms struggle with. And for compliance we need to implement controls and substantiate both effectiveness and appropriateness. To accomplish these goals we must employ additional tactics for baselining behavior, advanced forms of data analysis, policy management and (perhaps most importantly) a better understanding of user identity and authorization. Sure, for security and network forensics, network and security event analysis do a good job of piecing together related events across a network. But monitoring up the stack is more applicable for detecting misuse and more subtle forms of data theft — the tactics we see attackers choosing every day. And depending upon how the monitoring devices are deployed in your environment, you can *block* attacks, as well as report problems.

The problem is less about event reduction and more about application attack defense, so the tools must evolve.

Through the remainder of this document we will go into detail on on how to monitor "*up the stack*", i.e. monitoring the business instead of the network. We'll cover file integrity, identity, database, application and user activity monitoring, and how these features leverage different data and analysis techniques to address evolving threats to your business. Some of these sections may seem daunting, but we included the highlights -- and the technical pitfalls -- uncovered during our research to help you understand both the good and the bad. For the less technically inclined, we recommend you jump to the "Climbing the Stack" section, where we summarize the types of problems that can reasonably be addressed by advanced monitoring and in what order to add the additional data types. We deal with solving problems such as *Getting More from SIEM*, *Responding to Threats*, and *Tracking Privileged Users*. Perhaps most importantly, as time is your most limited resource, we provide a basic roadmap on how to deploy these features and avoid some of the internal politics to get the job done.

Threats

Whether looking at more types or data or responding to new threats, end users can derive more value from their SIEM/ Log Management platform by moving up from the network layer to focus on additional data types to analyze. In essence you'll *climb the stack* by adding database, application, and identity information into your analysis engine. Part of the dissatisfaction we hear from end users is the challenge of turning collected data into actionable information for operational efficiency and compliance requirements. This is compounded by the growing prevalence on application-oriented attacks, so monitoring only the network and servers doesn't help when the attackers are working above that level. It's kind of like repeatedly missing the bad guys because they are flying at 45,000', when you cannot climb past 20,000'. Technically nothing precludes SIEM from working at the 45,000' level and monitoring application components to pinpoint attacks, but it takes real work to get there. Given the rapid pace of evolution in the attack space to focus on applications and data, we don't believe keeping monitoring focused on infrastructure is viable, even over the medium term.

But before we reach any conclusions let's analyze the application threats we see in the field. It's not brain surgery and you've seen all these examples before, but they warrant another mention because we continue to miss opportunities to focus on detecting them. For example:

- **Email**: You click a link in a 'joke-of-the-day' email your spouse forwarded, which installs malware on your system, and then tries to infect every machine on your corporate network. A number of devices get compromised and become latent zombies waiting to blast your network and others.

- **Databases**: Your database vendor offers a new data replication feature to address failover requirements for your financial applications, but it's installed with public credentials. Any hacker can now replicate your database, without logging in, just by issuing a database command. Awesome!

- **Web Browsers**: Your marketing team launches a new campaign, but the third party content provider site got hacked. As your customers visit your site, they are unknowingly attacked using cross-site request forgery and then download malware. The customer's credentials and browsing history leak to Eastern Europe, and fraudulent transactions are submitted from customer machines without their knowledge. Yes, that's a happy day for your customers and also for you — and you cannot just blame the third party content provider. *You* own the problem.

- **Web Applications**: Your web application development team, in a hurry to launch a new feature on time, fails to validate some incoming parameters. Hackers exploit the database through a common SQL injection vulnerability to add new administrative users, copy sensitive data, and alter database configuration — all through normal SQL queries. By the way, as simple as this attack is, a typical SIEM won't catch it because all the requests are authorized and look normal. It's an application failure that causes security failure.

- ***Ad-hoc* applications**: The video game your kid installed on your laptop has a keystroke logger that records your activity and periodically sends an encrypted copy to the hackers who bought the exploit. They replay your last session, logging into your corporate VPN remotely to extract files and data under your credentials. It's great fun when the corporate investigators show up in your office to ask why **you** sent the formula for your company's most important product to China.

The power of distributed online systems to deliver services quickly and inexpensively cannot be denied, which means we security folks *cannot* stop these trends — no matter the risk. But we do have both a capability and responsibility to ensure these services are delivered as securely as possible, and to watch for bad behavior. Many of the events we discussed are not logged by traditional network security tools, and to casual inspection each transaction looks legitimate. Logic flaws, architectural flaws, and misused privileges look like normal operation to a router or an IPS. Browser exploits and SQL injection are difficult to detect without understanding application functionality. More problematic is that damage from these exploits occurs quickly, requiring a shift from after-the-fact forensic analysis to real-time monitoring for a chance to interrupt the attack.

> *We believe all organizations need to continue broadening how they monitor IT resources and incorporate technologies that are designed to look at the application layer, providing detection of application attacks in near real time.*

End users have all sorts of complaints about SIEM: it's too slow to keep up with multi-stage attacks, code substitution, etc.; it's ill suited to stopping SQL injection, rogue applications, data leakage, etc.; it's simply ineffective against cross-site scripting, hijacked privileges, etc. — we keep hearing that current tools to have no chance against these new attacks. *We don't think it's a flaw in SIEM.* It's a more fundamental problem of looking for attacks in the wrong places. We believe the answer entails broader monitoring capabilities at the application layer, and related technologies.

But the reality is that the tools and techniques used for application monitoring do not always fit existing SIEM architectures. Unfortunately this means some of the existing technologies you may have — and more importantly the way you've deployed them — may not fit into this new reality. We believe all organizations need to continue broadening how they monitor IT resources and incorporate technologies that are designed to look at the application layer, providing detection of application attacks in near real time. But to be clear, adoption is still very early and the tools are largely immature. The following is an an overview of the technologies designed to monitor at the application layer, which we will focus on in this paper:

- **File Integrity Monitoring:** This is real-time verification of applications, libraries, and patches on a platform. It's designed to detect replacement of files and executables, code injection, and the introduction of new and unapproved applications.

- **Identity Monitoring:** Designed to identify users and user activity across multiple applications, or when using generic group or service accounts. Employs a combination of location, credential, activity, and data comparisons to 'de-anonymize' user activity and identity.

- **User Activity Monitoring:** Examination of inbound and outbound user activity, application usage, and data. Commonly applied to email, web browsing, and other user initiated activity; as well as malware detection, botnets, and other types of *ad hoc* applications operating unbeknownst to the user.
- **Database Monitoring:** Designed to detect abnormal operation, statements, and user behavior; including both end users and database administrators. Monitoring systems review database activity for SQL injection, code injection, escalation of privilege, data theft, account hijacking, and misuse.
- **Application Monitoring:** Protects applications, web applications, and web-based clients from man-in-the-middle attacks, cross site scripting (XSS), cross site request forgery (CSRF), SQL injection, browser hacking, and data leakage. Commonly deployed as an appliance that monitors inbound application traffic.

Each of these enhanced monitoring systems works a bit differently, requiring slightly different integration requirements with existing SIEM and Log Management technology. And each new form of data collection and analysis will be prioritized according to your organization's use cases and attack scenarios. Each additional data type offers specific advantages, and many overlap with each other, so you'll have plenty of options for how to phase in these additional monitoring capabilities.

File Integrity Monitoring

We kick off our discussion of enhanced monitoring technologies with file integrity. As the name implies, it detects changes to files — whether text, configuration data, programs, code libraries, critical system files, or even Windows registries. Files are a common medium for delivering viruses and malware, and detecting changes to key files can indicate compromise.

File integrity monitoring works by analyzing changes to individual files. Any time a file is changed, added, or deleted, it's compared against a set of policies that govern file use, as well as signatures that indicate file tampering. Policies are as simple as a list of operations on a specific file that are not allowed, or might be more complicated — including comparisons of file contents and correlation against the user who made the change. When a policy is violated an alert is generated.

Changes are detected by examining file attributes — specifically name, date of creation, time of last modification, ownership, size in bytes, a hash to detect tampering, permissions, and type. Most file integrity monitors can also '`diff`' the contents of the file, comparing before against after to identify exactly what changed (for text-based files, anyway). All these comparisons are against a stored reference set of attributes that designates what state the file *should* be in. Optionally the contents can be stored for comparison, and to provide a rollback option.

File integrity monitoring can be periodic, at intervals from minutes to days. Some solutions offer real-time threat detection that inspects as files are accessed. The monitoring can be performed remotely — periodically accessing the system with user credentials and instructing the operating system to collect relevant information — or an agent can be installed on the target system that performs the data collection locally, and returns data upstream to the monitoring server. So there is considerable implementation flexibility.

As you can imagine, even a small company changes files a lot, so there is a lot to look at. And there are many files on many machines — for a typical large enterprise tens of thousands (if not more). Vendors of file integrity monitoring products provide the basic list of critical system files and policies, but you need to configure the file monitoring service to protect the rest of your environment. Keep in mind that some attacks are not fully encompassed by policy, and verification/investigation of suspicious activity must be performed manually. Administrators need to balance performance against coverage and policy precision against adaptability. Specify too many policies and track too many files, and the monitoring software consumes tremendous resources. File modification policies designed for maximum coverage generate many false positive alerts that must be manually reviewed. Rules must balance catching specific attacks against detecting broader classes of threats.

These challenges are mitigated in several ways. First, monitoring can be limited to just those files that contain sensitive information or are critical to the operation of the system or application. Second, policies can specify priority, so changes to key infrastructure or matches against known attack signatures get the highest priority. The vendor will help by supplying rules for known threats *and* to cover compliance mandates such as PCI-DSS. Suspicious events that indicate an attack policy violation are the next priority. Finally, permitted changes to critical files are logged for manual review at a lower priority to help reduce administrative burden.

File integrity monitoring has been around since the mid-90s, and has proven very effective for detection of malware and system compromise. Changes to Windows registry files and open source libraries are common hacks, and *very* difficult to detect manually. While file monitoring does not help with many of the web and browser attacks that use injection or alter programs in memory or take advantage of the user, it does detect many types of persistent threats, and so is a very logical extension to existing monitoring infrastructure.

Database Activity Monitoring

Database Activity Monitoring (DAM) looks specifically at database transactions, and integration of DAM data into SIEM and Log Management platforms is becoming more prevalent. Securosis is particularly interested in DAM, and has gone into gory technical detail on the product category. You can check out the database security page in the Securosis Research Library. Here we will give the "Cliff notes" version, describing the technology and some of the problems it solves. We will explain how DAM augments SIEM and Log Management analysis.

So what is Database Activity Monitoring? It's a system that captures and records database events — at minimum all Structured Query Language (SQL) activity, in real-time or near-real-time, including database administrator activity, across multiple database platforms, in order to generate alerts on policy violations.

For people already familiar with SIEM, DAM is very similar in many ways. Both follow a similar process of collecting, aggregating, and analyzing data. Both provide alerts and reports, and integrate into workflow (trouble ticket) systems to leverage the analysis. Both collect different data types, in different formats, from heterogenous systems. And both rely on correlation (and in some cases enrichment) to perform advanced analytics.

How are they different? The simple answer is that they collect different events and perform different analyses. But there is another significant difference: context. Database Activity Monitoring is tightly focused on database activity and how applications use the database (for good and not-so-good purposes). With specific knowledge of appropriate database use and operations, and a complete picture of database events, DAM is able to analyze database statements with far greater effectiveness.

In a nutshell, DAM provides focused and deep monitoring and analysis of one single important resource in the application chain, while SIEM provides great breadth of analysis across all devices.

Why is this important?

- **SQL injection protection:** Database activity monitoring can filter and protect against many SQL injection variants. It cannot provide complete prevention, but statement and behavioral analysis techniques catch many known and unknown database attacks. By white listing specific queries from specific applications, only authorized transactions can be allowed. This allows DAM to detect malicious and corrupted queries, as well as queries from unapproved applications, which are rarely a good sign. Additionally, DAM can transcend monitoring and actually block SQL injection before the statement reaches the database.

- **Behavioral monitoring:** DAM systems capture and record activity profiles, both of generic user accounts, and, specific database users. Changes in a particular user's behavior might indicate disgruntled employees, hijacked accounts, or even oversubscribed permissions.

- **Compliance purposes:** With DAM's complete view of database activity and ability to enforce policies at both statement and transaction/session levels, it is a proven tool for substantiating controls for regulatory requirements such as Sarbanes-Oxley. DAM can verify the controls are both in place and effective.

- **Content monitoring:** A couple DAM offerings additionally inspect content, so they are able to detect both SQL injection — as mentioned above — and also content injection. It's common for attackers to abuse social networking and file/photo sharing sites to store malware. When 'friends' view images or files their machines become infected. By analyzing the blob of content prior to storage, DAM can prevent some 'drive-by' injection attacks.

- **Blocking:** DAM can sit inline between the database and the application server and block malicious queries or perceived misuse. Today this option is being called virtual patching, providing protection in the same way that a firewall does, before a patch is available. Blocking is an advanced feature applied to many different use cases and threat models.

That should provide enough of an overview to start to think about whether and how you should add DAM to your monitoring strategy.

Why DAM?

The odds are, if you already have a SIEM/Log Management platform in place, you already look at some database audit logs. So why would you consider DAM in addition? The real question when thinking about how far up the stack (and where) to go with your monitoring strategy is whether adding database activity monitoring data will help with threat detection and other security efforts. To answer that question, consider that DAM collects important events which are *not* in log files (parsing database memory, collecting OS and/or protocol traffic, intercepting database library calls, undocumented vendor APIs, and stored procedures & triggers), provides real-time analysis and detection of database attacks, and *blocks* dangerous queries from reaching the database. These three features together are greater than the sum of their parts.

Over and above the attribute analysis (who, what, where, and when) that SIEM uses to analyze events, DAM uses lexical, behavioral, and content analysis techniques. By examining the components of a SQL statement, such as the `where` and `from` clauses, and the type and number of parameters, SQL injection and buffer overflow attacks can be detected. By capturing normal behavior patterns by user and group, DAM effectively detects system misuse and account hijacking. By examining content as it is both stored and retrieved, injection of code or leakage of restricted data can be detected as it occurs.

Once you have these two capabilities blocking is possible. If you need to block unwanted or malicious events, you must react in real time and deploy the technology in such a way that it can stop the query from being executed. Typical SIEM/LM deployments are designed to efficiently analyze events, which means only *after* data has been aggregated, normalized, and correlated. This is too late to stop an attack. By detecting threats *before* they hit the database, you have the capacity to block or quarantine the activity and take corrective action. DAM, deployed in line with the database server, can block or provide 'virtual database patching' against known threats.

Those are the reasons to consider augmenting SIEM and Log Management with Database Activity Monitoring.

Getting to DAM

What needs to be done to include DAM technology within your SIEM deployment? There are two options: leverage a standalone DAM product to submit alerts and events, or select a SIEM/Log Management platform that embeds these capabilities. All the standalone DAM products have the capability to feed collected events to third party SIEM and Log Management tools. Some can normalize events so SQL queries can be aggregated and correlated with other network events. In some cases they can also send alerts as well, either directly or by posting them to `syslog`.

Fully integrated systems take this a step further by linking multiple SQL operations together into logical transactions, enriching the logs with event data, or performing subsequent query analysis. They embed the analysis engine and behavioral profiling tools — allowing for tighter policy integration, reporting, and management. In the past, most database activity monitoring within SIEM products was 'DAM Light' — monitoring only network traffic or standard audit logs, and performing very little analysis. Today full-featured options are available within SIEM and Log Management platforms.

To summarize, DAM products offer much more granular inspection of database events that SIEM because DAM includes many more options for data collection, and database-specific analysis techniques, as well as the ability to block transactions violating policy. The degree to which you extract useful information depends on whether DAM is fully integrated with SIEM, and how much analysis and event sharing are established. If your requirement is to protect the database, you should consider this technology.

Application Monitoring

We now turn to applications. At first glance, many security practitioners may think applications have little to offer SIEM and Log Management systems. After all, applications are built on mountains of custom code, and security and development teams often lack a shared collaborative approach for software security. However, application monitoring for security should not be dismissed out of hand. Closed-minded security folks miss the fact that applications offer an opportunity to resolve some of the key challenges to monitoring. How? It comes back to a key point we have been making: the **need for context**. If knowing that Node A talked to Node B helps pinpoint a potential attack, then network monitoring is fine. But both monitoring and forensics efforts can leverage information about what transaction executed, who signed off on it, who initiated it, and what the result was — and you need to tie into the application to get that context.

This proximity to valuable assets makes the application an ideal place to see and report on what is happening at the level of user and system behavior, which can (and does) establish patterns of good and bad behavior that can provide additional indications of attacks.

In real estate, it's all about location, location, location. By climbing the stack and monitoring the application, you collect data closer to core enterprise assets such as transactions, business logic, rules, and policies. This proximity to valuable assets makes the application an ideal place to see and report on what is happening at the level of user and system behavior, which can (and does) establish patterns of good and bad behavior that can provide additional indications of attacks.

And remember that the location of the application monitor is critical for tracking both authorized users and threats — excerpting from the Threats section of this paper:

> *This is compounded by the growing prevalence on application-oriented attacks, so monitoring only the network and servers doesn't help when the attackers are working above that level. It's kind of like repeatedly missing the bad guys because they are flying at 45,000', when you cannot climb past 20,000'.*

Effective monitoring **requires** access to the app, the data, and the system's identity layers. They are the core assets of interest for **both** legitimate users and attackers trying to compromise your data.

So how do we get there? We can look to software security for some clues. The discipline of software engineering has made major strides at building security into applications over the last ten years. From static analysis, to threat modeling, to defensive programming, to black box scanners, to stronger identity standards like SAML, we have seen the software engineering community make real progress on improving overall application security. From the current paradigm of building security in, the logical next step is building **visibility** in, meaning the next step is to instrument applications with monitoring capabilities that collect and report on application use and abuse.

Application monitoring delivers several essential layers of visibility to SIEM and Log Management:

- **Access control:** Access control protects applications (including web applications) from unauthorized usage. But the access control container itself is often attacked via methods such as Cross Site Request Forgery (CSRF) and spoofing. Security architects rely heavily on access control infrastructure to enforce security at runtime and this data should be pumped into the SIEM/Log Management platform to monitor and report on its efficacy.

- **Threat monitoring:** Attackers specialize in crafting unpredictable SQL, LDAP, and other commands that are injected into servers and clients to troll through databases and other precious resources. The attacks are often not obviously attacks, until they are received and **processed** by the application — after all, "`DROP TABLE`" is a valid string. The *Build Security In* school has led software engineers to build input validation, exception management, data encoding, and data escaping routines into applications to protect against injection, but it's crucial to collect and report on each possible attack, even as the application is working to limit its impact. Yes, it's best to repel the attack immediately from within the application, but you also need to know about it, both to provide a warning to more closely monitor other applications, and in case the application is successfully compromised — the logs must be securely stored elsewhere, so even in the event of a complete platform compromise, the alert is still received.

- **Transaction monitoring:** Applications are increasingly built in tiers, components, and services, where the application is composed dynamically at runtime. The transaction messages' state is assembled from a series of references and remote calls, which obviously can't be monitored from the infrastructure level. The solution is to trigger an alert within the SIEM/Log Management platform when the application hits a crucial limit or other indication of malfeasance in the system; then by collecting critical information about the transaction record and history, the time required to investigate potential issues can be reduced.

- **Fraud detection:** In some systems, particularly financial systems, the application monitoring practice includes velocity and throttles to record behaviors that indicate likelihood of fraud. In more sophisticated systems, the monitors are active participants (not strictly monitors) and change the data and behavior of the system, such as through automatically flagging accounts as untrustworthy and sending alerts to the fraud group to start an investigation based on monitored behavior.

Application monitoring represents a logical progression from "build security in" practices. For security teams actively building in security, the organizational contacts, domain knowledge, and tooling should already be in place to execute on an effective application monitoring regime. In organizations where this model is still new, building visibility in through application monitoring can be an effective first step, but more work is required to set up people, process, and technologies that will work in the environment.

Getting Started with Application Monitoring

As with any new IT effort, it is important to remember that it's People, Process, and Technology — in that order. If your organization has a *build security in* software security regime in place, then you can leverage those resources and tools to

build visibility in. If not, application monitoring provides a good entree into the software security process, so here are some basics for getting started with application monitoring.

Application Monitors can be deployed as off the shelf products (like Web Application Firewalls) or delivered as custom code. However they are delivered, the design of the application monitor must address these issues:

- **Location:** Where application monitors may be deployed; what subjects, objects, and events are to be monitored.
- **Audit Log Messages:** How the Audit Log Observers collect and report events; these messages must be useful to the human(!) analysts who use them for incident response, event management, and compliance.
- **Publishing:** The way the Audit Log Observer publishes data to a SIEM/Log Manager must be robust and provide the analyst with high-quality data to review, and to avoid creating YAV (Yet Another Vulnerability).
- **Systems Management:** Making sure the monitoring system itself is working and can respond to faults.

Process

The process of integrating your application monitoring data into the SIEM/Log Management platform has two parts. First identify where and what type of application monitor to deploy. Similar to the discovery activity required for any data security initiative, you need to figure out what needs to be monitored before you can do anything else. Second, select the communications scheme from the application monitor to the SIEM/Log Management platform. This involves tackling data formats and protocols, especially for homegrown applications where the communication infrastructure may not exist.

The most useful application monitors provide event data not available elsewhere. Identify key interfaces to high priority assets such as message queues, mainframes, directories, and databases. For those interfaces, the application monitor should provide visibility into the message exchanges to and from the interfaces, session data, and the relevant metadata and policy information that guides its use. For applications that pass user content, the interception of messages and files provides the visibility you need. In terms of application monitor packaging, deployment (in specialized hardware, in the application itself, or in an access manager), performance, and manageability are key aspects — but less important than which subjects, objects, and events the monitor can access to collect and verify data.

Typically the user for an application monitor is a security incident responder, an auditor, or other operations staff. The application monitor domain model described below provides guidance on how to communicate in a way that enables the customer to quickly and reliably receive and respond to application monitor information.

Application Monitor Domain Model

The application monitor model is fairly simple. The core parts of the application monitor include:

- **Observer:** A component that listens for events
- **Event Model:** The set of events the Observer listens for, such as Session Created and User Account Created
- **Audit Log Record Format:** The data model for messages that the Observer writes to the SIEM/Log Manager, based on event type
- **Audit Log Publisher:** The message exchange mechanisms, such as publish and subscribe, that are used to communicate the Audit Log Records to the SIEM/Log Manager

These areas should be specified in some detail with the development and operations teams to make sure there is no confusion during the build process (*building visibility in*), but the same information is needed by off-the-shelf monitoring products. For the Event Model and Audit Log Record, there are several standard log/event formats which can be leveraged, including CEE (from Mitre), XDAS (from Open Group), and PCI DSS (from the PCI Security Standards Council). CEE and XDAS provide general purpose frameworks for types of events the observer should listen for and which data should be recorded; the PCI DSS standard is more specific to credit card processing. All these models are worth reviewing to find the most cost-effective way to integrate monitoring into your applications, and to make sure you aren't reinventing the wheel.

To tailor the standards to your specific deployment, avoid "drinking from the fire hose", where the speed and volume of incoming data make the signal-to-noise ratio unacceptable. As we like to say at Securosis: just because you *can* doesn't mean you *should*. Or think about phasing in application monitoring — collecting the most critical data initially and then expand the monitoring scope over time to gain a broader view of application activity.

The Event Model and Audit Records should collect and report on the areas described previously (Access Control, Threats, Compliance, and Fraud). But if your application is smart enough to detect malice or misuse, why wouldn't you just block it in the application anyway? Aye, there's the rub. *The role of the monitor is to collect and report, not to block.* This gets into a philosophical discussion beyond the scope of this report, but for now suffice it to say that figuring out whether and what to block is a key next step beyond monitoring.

The Event Model and Audit Records collected should be configureable (not hard-coded) in a rule or other configuration engine. This enables the security team to flexibly adjust logging thresholds, tweak data gathering, and take other actions as needed without recompiling and redeploying the application.

The two main areas the standards do not address are the Observer and the Audit Log Publisher. The optimal position for the Observer is often a choke point with visibility into a boundary's inputs and outputs — for example, watching technical boundaries like Java to .NET or web to mainframe. Choke points can be organizational (B2B connection), zone (DMZ to internal network), or state-based (account upgrade, transaction execution). The goal in selecting a location for the application monitor is to identify areas where valuable assets need not just protection, but also detection. A choke point in an application provides a centralized location to collect and report on inbound and outbound access. This can mean a WAF at the boundary of web applications, or it could be further down the stack, but the choke point must have access to the message payload data and be able to parse and make sense of the data to be useful to security analysts.

The Audit Log Publisher must be able to communicate messages to the SIEM/Log Management platform using secure enterprise-class messaging. This requires guaranteed delivery, and that policies can specify (and enforce) that messages get delivered exactly once and in order. Examples include JMS and MQ Series. The messages must also be signed and hashed for authentication and integrity.

Where to Go Next

As with many application security efforts, security must plan an integration strategy. After all, to build security in and monitor applications, the 'in' means **integration**. This can be done at the edge of the application, such as a Web Application Firewall or Filter (where the integration is typically focused on resources like the URI and HTTP streams); or can be integrated closer to the code through logging in the application. The book *Enterprise Integration Patterns* by Gregor Hohpe and Bobby Woolf (and companion website: <http://www.enterpriseintegrationpatterns.com/>) contains plenty of useful real-world guidance on getting started with integration, including patterns for endpoints (where application monitors may be deployed), message construction and transformation (where and how Audit Log Observers collect and report events), message channels and routing (how publishers send data to a SIEM/Log Manager), and systems management (making sure it works!). Whether delivered as an off-the-shelf product such as a WAF or in custom code, the combination of these patterns makes for an end-to-end integrated system that can report context straight from the authoritative source: the application.

Identity Monitoring

As we continue up the monitoring stack, we next reach identity monitoring, which is a distinct set of concerns from user activity monitoring — which we'll cover next. By monitoring identity, the SIEM/Log Management systems gain visibility into the provisioning and identity management processes that enterprises use to identify, store and process user accounts to prepare a user to use the system. Contrast that with user activity monitoring, where SIEM/Log Management systems focus on monitoring how the user interacts with the system at runtime and monitors for bad behavior. As an example, do you remember when you got your driver's license? All the processes that you went through at the DMV — getting your picture taken, verifying your address, and taking the driving tests — are related to provisioning an account (license); actually getting credentials created is identity management. When you are asked to provide your driver's license when checking in at a hotel or by a police officer after driving too fast — that's user activity monitoring. identity monitoring is an important first step because we need to associate a user's identity with network events and system usage to enable user activity monitoring. Each requires a different type of monitoring and reporting. Now let's discuss identity management (and no, we won't make you wait in line like the DMV).

To enable identity monitoring, the SIEM/Log Management project inventories the relevant identity management processes (such as Provisioning), data stores (such as Active Directory and LDAP) and technologies (such as identity management suites). The inventory should include the identity repositories that store accounts used for access to critical assets. In the old days it was as simple as going to RACF and examining the user accounts and rules for who was allowed to access what. Nowadays there can be many repositories that store and manage account credentials, so inventorying the critical account stores is the first step.

Nowadays there can be many repositories that store and manage account credentials, so inventorying the critical account stores is the first step.

Process

The next step is to identify the identity management processes that govern the identity repositories. How did the accounts get into LDAP or Active Directory? Who signs off on them? Who updates them? There are many facets to consider in the identity management lifecycle. The basic identity management process includes the following steps:

- **Provisioning:** Account creation and registration
- **Propagating:** Synchronizing or replicating the account to the account directory or database
- **Access:** Accessing the account at runtime
- **Maintenance:** Changing account data

- **End of Life:** Deleting and disabling accounts

The identity monitoring system should verify events at each process step, record them, and log audit events that can be correlated for security incident response and compliance. This links the event to the account(s) that initiated and authorized them. For example: who authorized provisioning this account? What manager(s) authorized these account updates? As we saw in the Societe Generale case from 2008, Jerome Kerviel (the trader who lost billions of the bank's money) was originally an IT employee who moved over to the trading desk. When he made the move from IT to trading, his account retained IT privileges and gained new trading privileges. Snowball entitlements enabled him to execute trades, and then remove logs and hide evidence. It appears there was a process mishap in the account update and maintenance rules that allowed this to happen, which shows how important the identity management processes are to access control.

In complex systems, the identity management process is often automated using an identity management suite. These suites generate reports for compliance and security purposes, which can be published to the SIEM/Log Management system for analysis. Whether automated with a big name suite or not, it is important to understand your account lifecycle for your critical systems before you begin work on identity monitoring. To fully close the loop, some processes also reconcile changes against change requests (and authorizations) to ensure every change was properly requested and authorized.

Data

In addition to identifying the identity repositories and the management processes around them, the data itself is useful to inform the audited messages published to SIEM/Log Management systems. The data points for collection typically include the following:

- **Subject:** User or entity, which could be a person, organization, host, or application.

- **Resource Object:** Typically a database, URL, component, queue, or Web Service,

- **Attributes:** Roles, groups, and other information used to make authorization decisions.

The identity data should be monitored to record all lifecycle events such as Create, Read, Update, Delete, and Usage. This is important for giving the SIEM/Log Management system an end-to-end view of both the account lifecycle and the account data.

Challenges

One challenge in identity monitoring is that the systems to be monitored, such as authentication systems, sport byzantine protocols and are not easy to extract data and reports from. You may need to do some extra spelunking to find the optimal protocol to communicate with the identity repository. The good news is that this is a one-time effort, during implementation — these protocols do not change frequently.

Another challenge accurately associating user identity with activity collected by SIEM. Simply matching user ID to IP or MAC address is quite limited, so heuristic and deterministic algorithms are used to help associate users with events. The association can be performed by the collector, but more commonly this feature is integrated within the SIEM engine as an log/event enrichment activity. User identification occurs as data is normalized and results are stored with the events.

Federated identity systems that separate the authentication, authorization and attribution create additional challenges, because it is difficult to synthesize the end-to-end view of the account across the identity provider and the relying party. Granted, the point of federation is to resolve the relationship at runtime, but it's important to recognize the difficulty this presents for end-to-end monitoring.

Finally, naming and hierarchies can create challenges for reporting on subjects, objects, and attributes because differing namespaces and management techniques can create collisions and redundancies.

Bottom Line

Monitoring identity systems benefits both security and compliance. Monitoring identity process and data events gives them a view into one of the most critical parts of the security architecture: identity repositories. The identity repositories are the source of many access control decisions and this visibility into how they are populated and managed is fundamental to monitoring the overall security architecture. Identity monitoring is also a prerequisite for user activity monitoring, which is used to provide the linkage between how accounts are provisioned and how they are actually used.

User Activity Monitoring

We are fans of identity monitoring, but it is typically blind to one very important aspect of accounts: how those accounts are used in practice. So you know who the user is, but not what they are doing. User activity monitoring bridges this gap by tracking user actions on systems and applications, and linking actions and users to assigned roles to make sure what you intend to happen is actually happening.

Implementing User Activity Monitoring

User activity monitors can be deployed to monitor access patterns and system usage. The collected data regarding how the system is being used and by who is then sent to the SIEM/Log Management system. This data is particularly useful for attribution. Implementing user activity monitoring requires answers to four key questions. First, what constitutes a user? Next, which activities are worth monitoring? Third, what does typical activity look like, and how can we define policies to scope acceptable use? And finally, where and how should the monitor be deployed?

Implementing user activity monitoring requires answers to four key questions. First, what constitutes a user? Next, which activities are worth monitoring? Third, what does typical activity look like, and how can we define policies to scope acceptable use? And finally, where and how should the monitor be deployed?

The question of what constitutes a user seems simple. Mostly, a user is an account in the corporate or customer directory, such as Active Directory or LDAP — and a human being hired through HR. But there are also accounts for various non-human system users, such as service accounts and machine accounts. In many systems service accounts, machine accounts, and other automated batch processes can do just as much damage as any other account/function. After all, these features were programmed and configured by humans, and are subject to misuse like any other accounts, so likely are worth monitoring as well.

Drilling down further into users, how are they identified? To start with, there is probably a username. But remember the data that the user activity monitor sends to the SIEM/Log Management system is for use after the fact. What user data will help a security analyst understand the user's actions and whether they were malicious or harmful? Several data elements are useful for building a meaningful user record:

- **Username:** The basic identifier for a user in the system, including the namespace or other protocol-specific data.
- **Identity Provider:** The name of the directory or database that authenticated the user.

- **Group/Role Membership:** Any group or role information assigned to the user account, or other data used for authorization purposes.

- **Attributes:** Was the user account assigned any privileges or capabilities? Are there time of day or location attributes useful for verifying user authenticity?

- **Authentication Information:** If available, information regarding how the user was authenticated can be helpful. Was the user dialed in from a remote location? Did they log in from the office? When did they log in? And so on.

A log entry that reads `user=rajpatel`; is far less useful than one that contains "`user=rajpatel; identityprovider=ExternalCORPLDAP; Group=Admin; Authenticated=OTP`". The more detailed information around the user and their credential, the more the analyst has to work with. Usually this data is easy to get at runtime — it is available in security tokens such as SAML and Kerberos — but the monitor must be configured to collect it.

Now that we see how to identify a user, what activities are of interest for the SIEM/Log Management system? The additional data types described previously — database activity, application, etc. — can all be enriched with the user data model described above; in addition user-specific events worth tracking include:

- **User Session Activities:** Events that create, use, and terminate sessions; such as login and logout events.

- **Security Token Activities:** Events that issue, validate, exchange, and terminate security tokens.

- **System Activities:** Events corresponding to system exceptions, startups, shutdowns, and availability issues.

- **Platform Activities:** Events from specific ports or interfaces, such as USB drive access.

- **Inter-Application Activities:** Events performed by more than one application on behalf of the user, all linked to the same business function.

Now that we know what kind of events we are looking for, what do we want to do with them? For monitoring we need to specify policies to define appropriate use, and what should be done when an event — or in some cases a **series** of events — occurs. Policy setup and administration is a giant hurdle with SIEM systems today, and adding user activity monitoring — or any other type — will require the same kind of setup and adjustment over time. Based on an event type listed above, you select the behavior type you want to monitor and define what users can and cannot do. User monitoring systems offer attribute-based analysis at minimum. More advanced systems offer heuristics and behavioral analysis — these provide flexibility in how users are monitored, and reduce false positives as the analysis adapts to user actions over time.

The final step is deployment of the user activity monitor; and the logical place to start is the identity repository because repositories can write auditable log events when they issue, validate, and terminate sessions and security tokens. This way the identity repository can report to the SIEM/Log Management system on what users were issued what sessions and tokens. This location can be leveraged further by adding user activity monitors closer to the monitored resources, such as Web Application Firewalls and Web Access Managers. These systems can enhance visibility beyond simply what tokens and sessions were issued from the identity repository, adding information on how were they used and what users accesses.

Correlation: Putting the Data to Work

With monitors situated to report on User Activity, the next step is to use the data. The data and event models described above provide an enriched model that enables the analyst to trace events back upstream. For example, the analyst can set up rules that identify known good and bad behavior patterns to reflect authorized usage and potentially malicious patterns.

Authorized usage patterns generally reflect the use case flows that users follow. In most cases these do not trigger alarms — for example a failed authentication is not necessarily suspicious because many users trigger these multiple times each week. But the stream of events is worth recording because it may be useful later. Consider a case of stock fraud like insider trading. There is usually nothing inherently suspicious about a series of trades at the time, but once turned onto a potential fraud, the evidence can be used to prove a history of bad behavior.

Potentially malicious usage escalates priority because they contain suspicious data, commands, or sequences. The data is likely not enough to interrupt the application's processing, but sufficiently noteworthy for the analyst to review and perhaps investigate further. These signatures are generally not based on use cases, but rather on threat models and attack patterns. The CAPEC community is one source to consider tapping for attack pattern events and signatures.

The collected data can be analyzed using these models to find activity trends. Authorized user activities are kept primarily for evidence purposes, while suspicious usage is retained as evidence and also flagged for more immediate attention. Rules are typically built into the SIEM/Log Management platform and can correlate the audit records with other sources to provide a more complete picture.

1+1 > 2

The combination of identity monitoring and user activity monitoring provides a powerful mechanism for a SIEM/Log Management system to attribute activities to specific user accounts. This enables analysts to tie back to their sessions and tokens, and how they were issued in the first place. When analyzing an incident this evidence can be quite valuable.

The combination of identity monitoring and user activity monitoring provides a powerful mechanism for a SIEM/Log Management system to attribute activities to specific user accounts.

Platform Considerations

So far we have focused on a number of additional data types and analysis techniques that extend security monitoring to gain a deeper and better perspective on what's happening. We have been looking at added value, but we all know there is no free lunch. So now let's consider some of the problems, challenges, and extra work that come with deeper monitoring. We know most of you who have labored with scalability and configuration challenges with your SIEM products have been waiting for the other shoe to drop. Each new data type and its associated analysis impact the platform. So let's discuss some of these considerations and how to work around the issues.

To be fair, it's not all bad news. Some additional data sources are already integrated with the SIEM, such as identity and database activity monitoring, minimizing deployment concerns. But most options for application, database, user activity, and file monitoring are not offered as fully integrated features. Monitoring products sometimes need to be set up in parallel — yes, another product to deploy, configure, and manage. You'll configure the separate monitor to feed some combination of events, configuration details, and/or alerts to the SIEM platform — but the integration likely stops there. And each type of monitoring we have discussed has its own idiosyncrasies and/or special deployment requirements, so the blade cuts both ways. Adding hard-to-get data and real-time analysis for these additional data sources comes at a cost. But what fun would it be if everything was standardized and worked out of the box? So you know what you're getting yourself into, the following is a checklist of platform issues to consider when adding additional data types to your monitoring capabilities.

- **Scalability**: When adding monitoring capabilities — integrated or standalone — you need additional processing power. SIEM solutions offer distributed models to leverage multi-tier or multi-platform deployments which may provide the horsepower to process additional data types. You may need to reconfigure your collection and/or analysis architecture to redistribute compute power for these added capabilities. Alternatively, many application and/or database monitoring approaches utilize software agents *on the target platform*. In some cases this is to access data otherwise not available, or to remove network latency from analysis response times, as well as to distribute processing load across the organization. Of course there is a downside to agents: overhead and memory consumption can impact the target platform, as well as the normal installation & management headaches. The point is that you need to be aware of the extra work being performed and where it's occurring and you will need to absorb that requirement on the target platforms or add horsepower to the SIEM system. Regardless of the deployment model you choose, you will need additional storage to accommodate the additional data. You may already be monitoring some application events through `syslog`, but transaction history can increase event volume per application by an order of magnitude. All monitoring platforms can be set to filter out events by policy, but filtering too much defeats the purpose of monitoring these other sources in the first place.

- **Integration:** There are three principal integration points to consider. The first is getting data into the SIEM and integrated with other event types, and second is how to configure the monitors regarding what to look for. Fully integrated SIEM systems handle both policy management and normalization / correlation of events. While you may need to alter some of your correlation rules and reports to take advantage of new data types, it can all be performed from a single management console. Standalone monitoring systems can easily be configured to send events, configuration settings, and alerts directly to a SIEM, or drop the data into files for batch processing. SIEM platforms are adept at handling data from heterogenous sources; so you need only change the correlation, event filtering, and data retention rules to accommodate the additional data. The second — and most challenging — part of integration is sharing policies and reports between SIEM and standalone monitors. Keep in mind that things like configuration analysis, behavioral monitoring, and file integrity monitoring all work by comparing current results against reference values. Unlike hard-coded attribute comparisons in most SIEM platforms, these reference values change over time — this flexibility is what makes them so useful. Policies need to be flexible enough to handle these dynamic values, so if your SIEM platform can't you'll need to use the monitoring platform's interface for policies, reporting, and data management. We see that with most of the Database Activity Monitoring platforms, where the SIEM is not flexible enough to alert properly. This means customers need to maintain separate rule bases in the two products. Whenever a rule changes on either side, this disconnection requires manually verifying that settings remain in sync between the two platforms. Some monitoring tools have import and export features so you can create a master policy set for all servers, along with policy reports that detail which rules are active for auditing. The third point to consider is that most monitoring systems leverage smart agents, with agent deployment and maintenance managed from the console. Most SIEM platforms leverage a web-based management platform which facilitates centralized management, or even the merging of consoles in a single "mother of all consoles." Many standalone monitoring systems for content, file integrity, and web application monitoring are Windows-specific applications which can't easily be merged and must be managed as standalone applications.

- **Analysis:** Each new data type needs its own set of analysis policies, alerting rules, dashboards, and reports. This is really where the bulk of the effort is spent — on making these broader data sources available and effective. It's not just that we have new types of data being collected — the flexible flat-file event stores used by most SIEM products adapt readily enough — but that monitoring tools should leverage more than merely attribute analysis. To detect SQL injection attacks, data exfiltration, or even something as simple as spam, we need to do more with the data we have. Content analysis, behavioral analysis, and contextual analysis — three of the most common options — look at the same events differently. The SIEM platform must have the flexibility to incorporate these analysis techniques, either as part of the remote data collectors, or as add-on functions within the platform. Lower-end platforms won't offer this and probably don't need to, but leveraging these additional monitoring capabilities within SIEM requires an architecture flexible enough to incorporate different analytics engines. When we refer to the SIEM *platform* this is what we are talking about. It's basically an analysis engine, and must be flexible enough to take lots of data and provide multi-variate correlation and alerting.

- **Other considerations:** Application monitors are more likely to intercept sensitive data, as they dig around in places built-in SIEM collectors don't look. You may not care about the privacy of `syslog` data over the network but that won't fly for application, database, or identity traffic. You need to secure application requests and database queries because some of this information is private and therefore protected by any number of regulatory hierarchies. SIEM securely stores data once collected, and offers encryption of stored data (with a performance cost, of course). If you need to encrypt the event stream as it is routed to the SIEM platform, you'll need to set up the platform — or the collector software itself — to secure data transmissions. Collection architecture also needs to account for the intended use case — for instance application and database monitors used to block activity or perform virtual patching must be deployed "in front of" the platforms they monitor. And to monitor web applications in the DMZ the collectors must support different network addressing and tunnel data between the collector and SIEM; you might even need to alter your network topology.

There are plenty of reasons to extend monitoring beyond the traditional security and network devices. With the growing popularity of application and database attacks you cannot afford not to monitor these additional data sources. So at least go into the project with your eyes open as to how these additional data types will impact your monitoring infrastructure.

Climbing the Stack

As we have discussed through this report, monitoring additional data types can positively extend the capabilities of SIEM in a number of different ways. But you have plenty of options for which way to go. So the real question is: where do you start? Clearly you will not start monitoring all of these data types at once, especially considering most forms require some (or possibly major) integration work. Honestly, there are no hard and fast answers on where to start, or what type of monitoring is most important. Those decisions must be made based on your specific requirements and objectives. But we can describe a couple common approaches to climbing the monitoring stack.

Get More from SIEM

The first path we'll describe involves organizations simply looking to do more with what they have, squeezing additional value from a SIEM system they already own. They start by collecting data on the existing monitoring systems already in place, where they already have the data or the ability to easily get it. Then they add capabilities, from easiest to hardest. Usually that means file integrity monitoring first. Of the additional monitoring capabilities, file integrity is a bit of a standalone, but critical because most attacks have some kind of impact on critical system files and can be detected that way. Next comes identity monitoring — most SIEM platforms coordinate with server/desktop operations management systems, so it's relatively straightforward to add. Why bother? Because identity monitoring systems' audit capabilities enable SIEM to audit access control activity and map domain identities to events.

From there the logical progression is to add user activity monitoring. You can leverage the combination of SIEM functions and identity monitoring data with a bunch of new rules and dashboards to track user activity. As sophistication increases third party web security, endpoint agents, and content analysis tools can provide additional data for a comprehensive view of user activity.

Once those activities are mastered, organizations tackle database and application monitoring. These two data types have less overlap in analysis and data collection techniques than the others, provide more specialized analysis and detect different classes of attacks. They also tend to be the most resource-intensive to implement, so fall to the bottom of the list if there is no specific catalyst to drive implementation.

Responding to Threats

Recall the variety of threats we outlined earlier in the report which prompt IT organizations to consider monitoring: malware, SQL injection, and other types of system misuse. If managing these threats is the catalyst for extending your monitoring infrastructure, the progression of what data types to add will depend entirely on what attacks you need to address. If you're interested in stopping web attacks, you'll likely start with application monitoring, followed by database activity and identity monitoring. Malware detection will drive you toward file integrity monitoring at first, and then probably identity and user activity monitoring, since bad "user behavior" can indicate a malware outbreak. If you want to detect botnets, user activity and identity monitoring is a good place to start since you want to detect anomalous behavior from the compromised machines.

Basically your mix of new data types will be driven by what you want to detect, based on what you believe presents the greatest risk to your organization. Though it's a bit beyond the scope of this project, we are big fans of threat modeling because it puts structure around what you need to worry about and how to defend against it. With a threat model (even on the back of an envelope), you can then map the threats to information your SIEM already provides, and then decide which supplementary add-on functions are necessary to detect the attacks.

Privileged Users

One area we tend to forget is the folks who hold the keys to the kingdom. Administrators and other folks with privileged access to the resources that drive your organization. This is also a favorite for the auditors out there (perhaps something to do with low hanging fruit), but we see a lot of folks look to advanced monitoring to address an audit deficiency. To monitor activity on the part of your privileged users, you would move toward identity and user activity monitoring first. These data types allow you to identify who is doing what, and where, to detect malfeasance.

From there, you would probably add file integrity monitoring because changing system files is an easy way for someone with access to make sure they can retain it, and also to hide their trail. Database monitoring would then come next, as users changing database access roles can indicate something amiss. The point here is you've probably been doing security far too long to trust anyone, and enhanced monitoring can provide the data you need to understand what those insiders are *really* doing on your key systems.

Political Land Mines

Any time new technologies are introduced, someone has to do the work. Monitoring up the stack is not different, and perhaps a bit harder since it crosses multiple organizations and requires consensus — which translates roughly to politics. And politics means that you can't get something done without cooperation from your co-workers. We can't stress this enough: many good projects die not because of need, budget, or technology — but due to lack of inter-departmental cooperation. And why not? Most of the time the people who need the data, or even fund the project, are not the people who have to manage things on a day to day basis.

As an example, DAM installation and maintenance falls upon the shoulders of database administrators. From their perspective, all they see is more work (and possibly a disturbing lack of faith). Not only do they have to install the product, but they get blamed for any performance and reliability issues that result. Pouring more salt into the wound, the DAM system is designed to *monitor database administrators*! Not only is the DBA's job now harder because they can't use their typical shortcuts, but now someone's looking over their shoulder and asking annoying questions. Very quickly, the DBA looks for ways to scuttle the project as technically infeasible. This is just one example; application and user activity monitors are usually subverted by being labelled as *destabilizing the business* or *being big brother*.

How do you address this reality? We recommend you pave the way for enhanced monitoring internally before you start. Not just who gets to pay for it, but also lay groundwork on how automation will make jobs easier in the long run. Paint these new capabilities as making everyone's job easier and increasing security. Explain that if rules and reports are automated, the audit staff won't be knocking on your cubicle asking for a whole bunch of stuff every week. Regulatory requirements don't just go away, and making it a team effort and giving each stakeholder a say in the process goes a long way. Ultimately you need to sell it as a win/win. Or watch your monitoring project get pulled under by the weight of organizational politics.

Conclusion

Many organizations have embraced Security Information and Event Monitoring (SIEM) as a way to understand what is happening in their environments and be alerted to potential attacks, *before* catastrophic loss. Whether the catalyst is compliance or a successful attack, those organizations making the investment in not only software, but also resources and process, can clearly improve their ability to deal with the myriad of attacks happening each day.

SIEM historically has focused on analyzing traditional infrastructure devices, such as network and security devices. But as the technology platforms mature and the attack space evolves, many organizations are now looking to extend the impact of their SIEM beyond infrastructure and focus on where most of the attacks happen: Up the Stack.

In this report we have focused on identifying the threats we now face and how advanced monitoring techniques such as file integrity monitoring, database activity monitoring, application monitoring, identity monitoring, and user activity monitoring can improve an organization's security posture. We have also discussed how these additional data types impact the SIEM platform. But all these issues hint at our main point: **the need for context**. You get alerts all day, every day, from pretty much all your devices. But you don't have the time or the resources to really understand which attacks are real, which are imagined, and which present a clear and present danger to critical data. "Monitor everything" is far from a panacea, but does provide you with the data necessary to find the answer. And with a decent amount of elbow grease to configure alerting rules to detect anomalous behavior in your environment, you can get better utilization of your scarce resources and positively impact your security posture.

Best of all, if you have already embraced SIEM for network and security monitoring, you are more than halfway there. Now it's just a matter of taking that next step, and we aren't religious about which advanced monitoring direction you take. Only that you keep moving. You know the bad guys are, for what that's worth.

About the Analysts

Adrian Lane, Analyst/CTO

Adrian is a Senior Security Strategist and brings over 22 years of industry experience to the Securosis team, much of it at the executive level. Adrian specializes in database security, data security, and software development. With experience at Ingres, Oracle, and Unisys, he has extensive experience in the vendor community, but brings a pragmatic perspective to selecting and deploying technologies — having worked on “the other side” as CIO in the finance vertical. Prior to joining Securosis, Adrian served as the CTO/VP at companies such as IPLocks, Touchpoint, CPMi and Transactor/Brodia. He has been an invited presenter at dozens of security conferences, contributes articles to many major publications, and is easily recognizable by his “network hair” and propensity to wearing loud colors. Once you get past his windy rants on data security and incessant coffee consumption, he is quite entertaining.

Adrian is a Computer Science graduate of the University of California at Berkeley with post-graduate work in operating systems at Stanford University. He can be reached at alane (at) securosis (dot) com.

Mike Rothman, Analyst/President

Mike’s bold perspectives and irreverent style are invaluable to companies as they determine effective strategies to grapple with the dynamic security threatscape. Mike specializes in the sexy aspects of security — such as protecting networks and endpoints, security management, and compliance. Mike is one of the most sought-after speakers and commentators in the security business, and brings a deep background in information security. After 20 years in and around security, he’s one of the guys who “knows where the bodies are buried” in the space.

Starting his career as a programmer and networking consultant, Mike joined META Group in 1993 and spearheaded META’s initial foray into information security research. Mike left META in 1998 to found SHYM Technology, a pioneer in the PKI software market, and then held senior roles at CipherTrust and TruSecure — providing experience in marketing, business development, and channel operations for both product and services companies.

After getting fed up with vendor life, Mike started Security Incite in 2006 to provide a voice of reason in an over-hyped yet underwhelming security industry. After taking a short detour as Senior VP, Strategy and CMO at eIQnetworks to chase shiny objects in security and compliance management, Mike joined Securosis with a rejuvenated cynicism about the state of security and what it takes to survive as a security professional.

Mike published *The Pragmatic CSO* <http://www.pragmaticcso.com/> in 2007 to introduce technically oriented security professionals to the nuances of what is required to be a senior security professional. He also possesses a very expensive engineering degree in Operations Research and Industrial Engineering from Cornell University. His folks are overjoyed that he uses literally zero percent of his education on a daily basis. He can be reached at mrothman (at) securosis (dot) com.

Gunnar Peterson, Contributing Analyst

Gunnar Peterson is a Managing Principal at Arctec Group. He is focused on distributed systems security for large mission critical financial, financial exchanges, healthcare, manufacturer, and insurance systems, as well as emerging startups. Mr. Peterson is an internationally recognized software security expert, frequently published, an Associate Editor for IEEE Security & Privacy Journal on Building Security In, a contributor to the SEI and DHS Build Security In portal on software security, a Visiting Scientist at Carnegie Mellon Software Engineering Institute, and an in-demand speaker at security conferences. He maintains a popular information security blog at http://1raindrop.typepad.com.

Gunnar resides in Minnesota, even in winter.

About Securosis

Securosis, L.L.C. <http://securosis.com> is an independent research and analysis firm dedicated to thought leadership, objectivity, and transparency. Our analysts have all held executive level positions and are dedicated to providing high-value, pragmatic advisory services.

Our services include:

- *Primary research publishing*: We currently release the vast majority of our research for free through our blog, and archive it in our Research Library. Most of these research documents can be sponsored for distribution on an annual basis. All published materials and presentations meet our strict objectivity requirements and follow our Totally Transparent Research policy <http://securosis.com/about/totally-transparent-research>.
- *Research products and strategic advisory services for end users*: Securosis will be introducing a line of research products and inquiry-based subscription services designed to assist end user organizations in accelerating project and program success. Additional advisory projects are also available, including product selection assistance, technology and architecture strategy, education, security management evaluations, and risk assessment.
- *Retainer services for vendors*: Although we will accept briefings from anyone, some vendors opt for a tighter ongoing relationship. We offer a number of flexible retainer packages. Services available as part of a retainer package include market and product analysis and strategy, technology guidance, product evaluation, and merger and acquisition assessment. Even with paid clients, we maintain our strict objectivity and confidentiality requirements.
- *External speaking and editorial*: Securosis analysts frequently speak at industry events, give online presentations, and write and/or speak for a variety of publications and media.
- *Other expert services*: Securosis analysts are available for other services as well, including Strategic Advisory Days, Strategy Consulting engagements, and Investor Services. These tend to be customized to meet a client's particular requirements.

Our clients range from stealth startups to some of the best known technology vendors and end users. Clients include large financial institutions, institutional investors, mid-sized enterprises, and major security vendors.

Additionally, Securosis partners with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis.