Applied Network Security Analysis: Moving from Data to Information

Version 1.3

Released: November 21, 2011

Securosis, L.L.C. 515 E. Carefree Blvd. Suite #766 Phoenix, AZ 85085 **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, but has been enhanced, reviewed, and professionally edited.

Special thanks to Chris Pepper for editing and content support.

Licensed by Solera Networks

Award-winning Network Security Analytics solutions from Solera Networks record, classify, index and store network traffic data to provide comprehensive situational awareness of all network events. The technology provides complete, real-time visibility and accurate incident reconstruction, allowing customers to detect and identify the root cause of advanced threats, mitigate the loss of intellectual property and reputational damage, reduce the time to respond and remediate, and minimize exposure to ongoing breaches, protecting critical information assets.

Global 2000 enterprises, cloud service providers and government agencies use Network Security Analytics solutions from Solera Networks to combat today's increasingly sophisticated and targeted threats. Our clients see everything and know everything, allowing them to identify any and all events that existing toolsets fail to recognize, recover, or reconstruct. For more information on Solera Networks, visit www.soleranetworks.com.

Contributors

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

Anon, Betsy Nichols, Vivek Rajan

Copyright

Table of Contents

Introduction

Network security analysis is a rather misunderstood term, mostly due to its grand and nebulous connotations, but we consider it the next step on the path which starts with Incident Response Fundamentals and continues with React Faster and Better.

The issues highlighting the need for network security analysis are clear. We cannot assume we can stop the attackers, so we have to plan for compromise. The difference between success and failure breaks down to how quickly you can isolate the attack, contain the damage, and then remediate the issue. So Securosis has built our core security philosophy around monitoring critical networks and devices, developing our ability to find the root cause of any attack.

> We cannot assume we can stop the attackers, so we have to plan for compromise. The difference between success and failure breaks down to how quickly you can isolate the attack, contain the damage, and then remediate the issue.

Revisiting Monitor Everything

Back in early 2010, we published a set of Network Security Fundamentals, one of which was Monitor Everything. If you read the comments at the bottom of that post you will see some divergent opinions of what 'everything' means to different folks, but nobody really disagrees with broad monitoring as a core tenet of security nowadays. We can thank the compliance gods for that.

To understand the importance of monitoring everything, let's excerpt some research I published back in early 2008 (in my Security Incite days) that is still relevant today.

> *New attacks are happening at a fast and furious pace. It is a fool's errand to spend time trying to anticipate where the issues are. REACT FASTER first acknowledges that all attacks cannot be stopped. Thus, focus remains on understanding typical traffic and application usage trends and monitoring for anomalous behavior, which could indicate an attack. By focusing on detecting attacks earlier and minimizing damage, security professionals both streamline their activities and improve their effectiveness.*

There are many data sources you can (and should) monitor, including firewalls, IDS/IPS, vulnerability scans, network flows, device configurations, and content security devices. This data then becomes part of a profile of sorts, describing what has happened and using that as a baseline. Then the concept is to watch for variations beyond a defined tolerance and fire alerts when those conditions are met.

We still believe in this fundamental approach. Log Management has clearly become the place to start for most organizations, since any data is more than they have now. But let's examine why logs are only the start for maturing security organizations.

Logs Are Not Enough

Back when I worked for a SIEM vendor it was clear that event logs provided a great basis for compliance reporting, because they effectively substantiate implemented controls. As long as the logs are not tampered with, at least. But when you need to quickly isolate a security issue, the logs tell you what happened, but without sufficient depth for you to truly understand how it happened. Additionally, isolating a security attack using log data requires logs from all points in the path between attacker and target. If you aren't capturing information from the application servers, databases, and applications themselves, visibility is severely impaired and so is your ability to figure out what really happened.

Contrast that against the ability to literally replay an attack from a full network packet capture. You could follow along as the attacker broke your stuff. See the path they took to traverse your network, the exploits they used to compromise devices, the data they exfiltrated, and how they covered their tracks by tampering with the logs. Of course this assumes you are capturing the right network traffic along the attacker's path, and it might not be feasible to capture all traffic all the time. But if you implement a full network packet capture sandwich (as we described in the React Faster and Better paper), incident responders have much more information to work with. We will discuss how to deploy the technology to address some of these issues later in this paper. But given that you need additional data to do your job, where should you look?

The Network Doesn't Lie

For this discussion let's assume time starts at the moment an attacker gains a foothold in your network. That could involve compromising a device (through whatever means) already on the network, or by having a compromised device connect to the internal network. At that point the attacker is in the house, so the clock is ticking. What do they do next? An attacker will try to move through your environment to achieve their ultimate goal, whether may include compromising a specific data store or adding to their bot army, or whatever.

Attackers **need** the network, pure and simple. Which means they will leave tracks, but you will see them only if you are looking.

Attackers can do about a zillion different things to attack your network, and 99% of them depend on the network in some way. They can't find another target without using the network to locate it. They can't attack a target without connecting to it. Furthermore, even if they are able to compromise the ultimate target, the attackers must then exfiltrate the data. So they need the network to move the data.

They *need* the network, pure and simple. Which means they will leave tracks, but you will see them only if you are looking. This is why we favor capturing as much of the full network packet data as possible, as described in React Faster and Better. Attackers could compromise network devices and delete log records. They could generate all sorts of meaningless traffic to confuse network behavioral analysis. But they can't alter the packet stream as it's captured, which becomes the linchpin of the data you'll collect to perform this advanced network security analysis.

Data Is Not Information

Collecting data isn't enough. In order to figure out the root cause you need to use the data to draw conclusions about what's happening in your environment. That requires indexing the data, supplementing and enriching it with additional context, alerting on the data, and then searching through it to pursue an investigation. This requires significant technical horsepower. Just capturing the full network packet stream requires a purpose-built data store, with some black magic to digest and index network traffic at sufficient speed to provide usable, actionable information to shorten the exploit window.

To get an idea of the magnitude of this challenge, note that many SIEM platforms struggle to handle 10,000-15,000 events per second. We are talking here about capturing 10-100gbps of full honest-to-goodness network traffic – not 100kbyte log records. They should put a warning label on network packet capture devices: “Don't try this with your SIEM or log aggregation device.”

Leveraging the Information

Most folks only think about enhanced data collection in terms of forensics because that is the obvious use case. We won't neglect forensics, but full packet streams are invaluable in various other use cases, including:

- **Improving Alerts**: By properly building a specific threat model and enumerating it in a tool (as described in our Understanding and Selecting SIEM paper), alerts based on log data can point you in the right direction to determine whether you are under attack. But what if you could look for specific attack strings, correlate session data and/or parse database calls as they happen? You would have detect attacks faster and more precisely. It's not easy to filter this effectively, but it is possible.

- **Malware Analysis**: We have been talking for years about the folly of traditional anti-malware approaches, enforced on the endpoint. At that point it's generally too late to do anything useful aside from figuring out which machines you need to clean up. What if you could look into network traffic and see known malware coming into the network? This also requires knowing what to look for but works much better than simply capturing events from your AV console – which tells you what already happened, rather than what is about to happen.

- **Breach Confirmation**: Given the difficulty of isolating an attack preemptively, how do you know what really happened? By scrutinizing a copy of all traffic sent to a specific device, you can generally tell quickly whether you have a problem – as well as what's wrong and how serious it is. Again, this is not available from traditional event or configuration logs.

We do not claim that capturing full network traffic is as good as a time machine – it doesn't tell you what's going to happen before it occurs (how cool would that be?). Nor do we position full packet capture as an alternative to SIEM/Log Management. We believe in using both, because we need to focus on closing the exploit window as quickly as possible and containing any damage. Capturing network traffic is rapidly becoming a must-have capability for organizations which want to be more effective at detecting attacks and isolating their root causes.

We will dig into the specifics of capturing full network traffic and performing the analysis to leverage it; then dive into additional use cases to really illuminate how valuable the full packet capture stream can be – not just when you are dealing with an incident, but in everyday practice.

Collection + Analysis = A Fighting Chance

Now let's dig in a little deeper to understand what kind of data collection foundation makes sense, given the types of analysis we need to detect our adversaries.

First we define the critical data types for our analysis. We start with the foundational elements, which we covered ad nauseum in our Monitoring Up the Stack paper. These include event logs from the network, security, databases, and applications. We have already pointed out that *log data is not enough* but you still need the logs for a historical view of what happened, and the basis for the rule base for actionable alerts. Next we'll add additional data commonly used by SIEM devices – including network flows, configuration data, and some identity information. These additional data types provide more context to detect patterns of potential badness. But all this is still not enough – we need to look beyond these data types for more detail.

Full Packet Capture

As we wrote in the React Faster and Better paper:

> *One emerging advanced monitoring capability – the most interesting to us – is full packet capture. These devices basically capture all traffic on a given network segment. Why? The only way you can really piece together exactly what happened is to use the actual traffic. In a forensic investigation this is absolutely crucial, providing detail you cannot get from log records.*
>
> *Going back to a concept we call the Data Breach Triangle, you need three components for a real breach: an attack vector, something to steal, and a way to exfiltrate it. It's impossible to stop all potential attacks, and you can't simply delete all your data, so we advocate heavy perimeter egress filtering and monitoring, to (hopefully) prevent valuable data from escaping your network.*

So why is having the packet stream so important? It is a critical facet of *heavy perimeter monitoring*. The full stream can provide a smoking gun for an actual breach, showing whether data actually left the organization, and **what data** specifically. If you look at ingress traffic, the network capture enables you to pinpoint the specific attack vector(s) as well. We will discuss both these use cases, and more, in additional detail later in this paper, but for now it's enough to say that full network packet capture data is the cornerstone of Applied Network Security Analysis.

Intelligence and Context

Two additional data sources bear mentioning: reputation and malware. Both these data types provide extra context to understand what is happening on your networks and are invaluable for refining alerts.

- **Reputation**: Wouldn't it be great if you *knew* some devices and/or destinations were up to no good? If you could infer intent from just an IP address or other identifying characteristics? Well you can, at least a bit. By leveraging some of the services that aggregate data on command and control networks and on other known bad actors, you can refine your alerts and optimize your packet capture based on behavior, not just on luck. Reputation made a huge difference in the effectiveness of both email and web security, and we expect a similar impact on more general network security. This data helps focus monitoring and investigation on areas likely to cause problems.

- **Malware samples**: A log file won't tell you that a packet carried a payload with known malware. But samples of known malware are invaluable when scrutinizing traffic as it enters the network, before the attacker has a chance to do any damage. Of course nothing is foolproof, but we are trying to get smarter and optimize our efforts. Recognizing something that looks bad as it enters the network (and maybe blocking it) would provide a substantial advantage in blocking malware. Especially compared to traditional anti-malware controls, which don't do a very good job at blocking much of anything nowadays, and force you to clean up the compromised devices after the fact.

Digesting Massive Amounts of Data

The challenge of collecting and analyzing a multi-gigabit network stream is significant, and each vendor is likely to have its own *special sauce* to collect, index, and analyze the data stream in real time. We won't get into specific technologies or approaches – after all, beauty is in the eye of the beholder – but there are a couple things to look for:

- **Collection Integrity**: A network packet capture system that drops packets isn't very useful, so the first and foremost requirement is the ability to collect network traffic at wire speeds. Given that you are looking to use this data for investigation, it is also important to maintain traffic integrity to prove packets weren't dropped.

- **Purpose-built data store**: Unfortunately MySQL won't get it done as a data store for a full packet stream. The rate of insertions required to deal with 10gbps traffic demand something built specifically that purpose. Again, there will be lots of puffery about this data store or that one. Your objective is simply to ensure the platform or product you choose will scale to your needs.

- **High-speed indexing**: Once you get the data into the store you need to make sense of it. This is where indexing and deriving useful metadata become critical. Remember this happens at wire speeds, likely involves identifying applications (like an application-aware firewall or IDS/IPS), and requires enriching the data with geolocation and/or identity information. Yeah, it's pretty much rocket science.

- **Scalable storage**: Capturing high-speed network traffic consumes a lot of storage. And we mean a lot. So you need to calibrate onboard storage against archiving approaches, optimizing the amount of storage on the capture devices based on the number of days of traffic you want to keep. Keep in mind the metadata you extract from the traffic doesn't go away when you roll the network traffic, but you still want to size the system properly.

The collection foundation has to be a better SIEM than SIEM (since you are collecting many of the same data types), effectively an IDS/IPS, and a massive storage array. We don't see many of these products on the market because of the technical challenge of providing all these capabilities. So do your homework and make sure any technology you look at will scale to your needs.

The collection foundation has to be a better SIEM than SIEM, effectively an IDS/IPS, and a massive storage array. We don't see many of these products on the market because of the technical challenge of providing all these capabilities.

Nor do we expect full packet capture gear to supplant SIEMs or IDS/IPS gear any time soon. We are just pointing out that the analysis required is pretty similar, but must happen at wire speed with a much broader traffic stream.

Phases of Collection

We understand it's unlikely that you'll install this kind of collection infrastructure overnight. As valuable as full network capture data is, we are pragmatists. Nothing says you have to capture everything at the flip of a switch. Realistically you probably can't, so how do you start? We suggest starting with what we call the *Full Packet Capture Sandwich* (FPCS).

The FPCS starts by capturing traffic from the perimeter. Most perimeters run at far lower speeds than their core networks, so they serve as manageable starting points for traffic capture. And given the number of attacks originating from *out there* (yeah, the Internet) it's good to track what we can – both coming in and going out to the Internet. As you are ready, complement perimeter capture with traffic from the most critical internal segments as well. You know – those with the high-value assets (such as transaction systems, databases, intellectual property, etc.) – that stuff attackers go for.

If you capture data from key internal networks, as well as perimeter traffic (which is why we call this the sandwich), you have a better chance to piece together what happened during an attack. Over time you can capture more internal segments to get as broad a sampling of captured data as you can. That said, you can't collect everything – so you need the ability to capture on an *ad hoc* basis. Basically it's a SWAT capture capability, which may mean a portable packet capture rig or some network hocus-pocus to tap critical segments on demand. Either way, investigating using these tactics will likely involve packet capture in some way, shape or form.

The Forensics Use Case

Most organizations don't really learn about the limitations of event logs until the forensic investigators hold up their hands and explain they know *what* happened, but aren't really sure *how*. Huh? How could that happen? It's pretty simple: logs are a backward-looking indicator. They can help you piece together what happened, but you can only *infer* how.

In a forensic investigation inferring anything is suboptimal. You want to *know*, especially given the needs to isolate the root cause of the attack and to establish remediations to ensure it doesn't happen again. So we need to look at additional data sources to fill in gaps in what the logs tell you. Let's take a look at a simplified scenario to illuminate the issues. We'll look at the scenario both from the standpoint of a log-only analysis and then with a few other data sources (including network full packet capture) added. For a more detailed incident response scenario, check out our React Faster and Better paper.

The Forensic Limitations of Logs

It's the call you never want to get. The Special Agent on the other end of the line called to give you a heads-up: they found some of your customer data as part of another investigation into some cyber-crime activity that helps fund a domestic terrorist ring. Normally the Feds aren't interested in giving you a heads-up until their investigation is done, but you have a relationship with this agent from your work together in the local InfraGard chapter. So he did you a huge favor.

The first thing you need to do is figure out what was lost and how. To the logs! You aren't sure how it happened, but you see some strange log records indicating changes on a application server in the DMZ. Given the nature of the data your agent friend passed along, you check the logs on the database server where that data resides as well. Interestingly enough, you find a gap in the logs on the database server, where your system collected no log records for a five-minute period a few days ago. You aren't sure exactly what happened, but you know with reasonable certainty that *something* happened. And it probably wasn't good.

Your system collected no log records from the database for a five-minute period a few days ago. You aren't sure exactly what happened, but you know *something* happened. And it probably wasn't good.

Now you work backwards and isolate the additional systems compromised as the attackers made their way through the infrastructure to reach their target. It's pretty resource intensive, but by searching in the log manager you can isolate devices with gaps in their logs during the window you identified. The attackers were pretty effective, taking advantage of unpatched vulnerabilities (Damn, Ops!) and covering their tracks by turning off logging where necessary. At this point you know the attack path, and at least part of what was stolen, thanks to the FBI. Beyond that you are blind. So what can you do to make sure you aren't similarly surprised somewhere down the line? You can set the logging system to alert if you don't get any log records from critical assets in any 2-minute period. Again, this isn't perfect and will result in a bunch more alerts, but at least you'll know something is amiss *before* the FBI calls.

With only log data you can identify what was attacked but probably not how the attack happened.

Forensics Driven by Broader Data

Let's take a look at an alternative scenario with a few other data sources such as full network packet capture, network flow records, and configuration files. Of course it is still a bad day when you get the call from your pal the Special Agent. Unfortunately Applied Network Security Analysis cannot magically make you omniscient, but can change how you investigate breaches. You still start with the logs on the perimeter server and identify the device that served as the attacker's initial foothold. But you've implemented the Full Packet Capture Sandwich architecture described above, so you are capturing the network traffic in your DMZ. You proceed to the network analysis console (using the full packet capture stream) and search all the traffic to and from the compromised server. Most sessions to that server are typical – standard application traffic. But you find some reconnaissance, and then something pretty strange: an executable injected into the server via faulty field validation on the web app (Damn, Developers!). Okay, this confirms the first point of exploit.

You still have some blind spots, but with your additional data sources you are able to pinpoint the attack path.

Next we go to the target (keeping in mind what data was compromised) and do a similar analysis. Again, with our full packet capture sandwich in place, we captured traffic to/from the critical database server as well. As in the log-only scenario, we pinpoint the time period when logging was turned off, then perform a search in our analysis console to figure out what happened during that 5-minute period on that segment. Yep, a privileged account turned off logging on the database server and added an admin account to the database. Awesome. Using that account, the attacker dumped the database table and moved the data to a staging server elsewhere on your network. Now you know which data was taken, but how?

You aren't capturing all the traffic on your network (since that is infeasible), so you have some blind spots, but with your additional data sources you are able to pinpoint the attack path. The NetFlow records coming from the compromised database server show the path to the staging server. The configuration records from the staging server indicate what executables were installed, which enabled the attacker to package and encrypt the payload for exfiltration. The logs of the servers along the attack path also confirm the attack (as shown above).

Further analysis of the NetFlow data shows the exfiltration, presumably to yet another staging server on another compromised network elsewhere. It's not perfect, because you are figuring out what already happened. But now you can get back to your FBI buddy with a lot more information about what tactics the attacker used, and maybe even evidence that might be helpful in prosecution. And there is plenty of blame to go around, since the developers let the attackers through the door (with a faulty field validation) and the Ops team left unpatched vulnerabilities that allowed the attackers to work their way deeper into the infrastructure.

To be clear, it's not about having these tools to lay blame at the feet of the responsible parties. It's about making sure you learn exactly how the attack happened, and make sure you've got monitors and controls to make sure it doesn't happen again.

Can't Everyone Get Along?

Clearly this is a simplified scenario that perfectly demonstrates the need to collect additional data sources to isolate the root cause and attack path of any compromise. Imagine that! Yet, the real world is rarely so tidy. *Our point here is that no one data source stands alone.* We aren't claiming that logs are not important – they certainly are. As are the full packet capture stream, the configuration files, the NetFlow records, and a bunch of other stuff. By harnessing all this data you can more effectively figure out what happened, contain the damage, and figure out how to properly remediate.

Our point here is that no one data source stands alone. By harnessing all this data you can more effectively figure out what happened, contain the damage, and figure out how to properly remediate.

Security tools (and security people, for that matter) aren't very good at sharing. But the whole industry needs to get collectively better at it. Bidirectional integration between the SIEM/Log Manager, Full Packet Capture gear, Network Behavioral Analysis, and Configuration Tracking products makes all the tools more powerful, and enables us to take better advantage of our data. The security team needs to figure out which tool and repository will be primary and use the other tools as needed, but without the ability to share data and build alerts leveraging many of these data sources, the job of a security professional gets much harder.

The forensics capability is great, but as our contrived scenario showed, it's already too late and your data is gone. The next step is to figure out how to shorten that window between attack and detection.

The Advanced Security Use Case

The forensics use case we discussed previously is about taking a look at *something that already happened*. You presume your data is already lost, the horse is out of the barn, and Pandora's Box is open. But what if we used some of these additional data types to make security alerts better, with the clear goal of reducing the window between exploit and detection. You know, actually reacting faster?

Can we leverage a network full packet capture stream to learn sooner when something is amiss and to improve security? Yes, but this presents many of the same challenges as using log-based analysis to detect what is going on. You still need to know what you are looking for, and an analysis engine that can not only correlate behavior across multiple types of logs, but also analyze a massive amount of network traffic for signs of attack.

So when we made the point earlier in the paper that these Network Security Analysis platforms need to be better SIEMs than a SIEM, this is what we were talking about.

Pattern Matching and Correlation

Assuming that you are collecting some of these additional data sources, the next step is to turn said data into actionable information, which means some kind of alerting and correlation. We need to be careful when using the 'C' word (correlation), given the nightmare most organizations have when they try to correlate data using their SIEM platform. Unfortunately the job doesn't get any easier when extending the data types to include network traffic, network flow records, etc. Accordingly, we continue to advocate a realistic and incremental approach to analysis. Much of this approach was presented (in gory detail) in our Network Security Operations Quant project. Succinctly summarized, the process looks like this:

- **Identify high-value data**: This is key – you probably cannot collect from every network, nor should you. First you need to figure out the highest profile targets and start with them.
- **Build a realistic threat model**: Next put on your hacker hat and build a threat model for how you'd attack that high value data. It won't be comprehensive, but that's okay. You need to start somewhere, and modeling out how you'd attack the data is as good a place as any.
- **Enumerate those threats in the tool**: With the threat models, design rules to trigger alerts based on the specific attacks you modeled in the last step.
- **Refine the rules and thresholds**: The only thing we can know for certain is that your rules will be wrong. So you will go through a tuning process to hone in on the types of attacks you are looking for.

- **Wash, rinse, repeat**: Add another target or threat and build more rules as above.

With the additional data sources you can look for (and alert on) much more specific situations. Whether it's looking for known malware (which we will talk about next), traffic destined for a known command and control network, or tracking a buffer overflow targeted at an application residing in the DMZ, you get a lot more precision in refining rules to identify what you are looking for. Done correctly this reduces false positives and helps to zero in on specific attacks.

With these additional data sources, you get a lot more precision in refining rules to identify what you are looking for. Done correctly this reduces false positives and helps to zero in on specific attacks.

Of course the magic words are "done correctly". Having so much more data on which to build alerts is a double-edge sword. Thus, it is essential to build the rule base incrementally – test the rules and keep refining the alerting thresholds – especially given the more granular attacks you can look for.

Baselining

The other key aspect of leveraging this broader data collection capability is understanding how building a baseline changes. Using logs (or more likely NetFlow), you can get a feel for "normal" behavior and use it to kickstart your rule building. You may have already done that with your existing SIEM. But having the packet stream provides a lot more data, including things like HTTP content type, URL category, VLAN tag, or (perhaps) hundreds of others. Getting back to our double edged-sword, clearly having these extremely granular attributes allows you to set up far more precise rules, but you can both kill the system performance (with many rules, poorly structured) and miss obvious stuff because you are too focused on granularity. So the answer is balance, and you only get there over time.

Basically, you assume activity happening when you first implement the system is normal, and alert if something varies too far from that. Obviously can't assume that normal = good, but this does give you a place to start, which is more important than being perfect. As with correlation this process is incremental. Your baselines will be wrong when you start, so you adjust them over time using the operational experience you gain responding to alerts.

Revisiting the Scenario

Getting back to the scenario presented above, how would this more pseudo-real-time analysis help reduce the window between attack and detection? To recap that scenario briefly, a friend at the FBI informed you that some of your customer data showed up as part of a cybercrime investigation. The forensic analysis revealed an injection attack enabled by faulty field validation on a public-facing web app.

With the benefit of hindsight, let's see how you'd detect the attack using the network full packet capture stream. If you created a rule to look for executables entered into the form fields of POST transactions, it would fire when that transaction flew by. Since you are also capturing the traffic on the key database segment, you could establish a content rule looking for content strings you know are important (as a poor man's DLP), and alert when you see that type of data being sent anywhere, but the application servers that should have access to it. You could also, for instance, set up alerts on seeing an encrypted RAR file on an egress network path. There are multiple places you could detect the attack, if you know what to look for.

But a lot of this discipline is based on a basic concept: "Fool me once, shame on you. Fool me twice, shame on me." Once you have seen this kind of attack – especially if it succeeds – make sure it doesn't work again.

Of course that example is contrived and depends on your ability to predict the future, figuring out the vectors the attacker will use before the attack hits. But a lot of this discipline is based on a basic concept: "Fool me once, shame on you. Fool me twice, shame on me." Once you have seen this kind of attack – especially if it succeeds – make sure it doesn't work again. It's a bit of solving yesterday's problems tomorrow, but many security attacks use very similar tactics. So if you can enumerate a specific attack vector based on what you saw, there is an excellent chance that you will have another opportunity to recognize and block that attack again in the future. So it is worth looking, and using other controls to protect against attacks you haven't seen before, as much as you can.

The Malware Detection Use Case

As we continue our tour of advanced use cases for Network Security Analysis, it's time to consider malware analysis. Most successful attacks involve some kind of malware at some point during the attack. If only just to maintain a presence on the compromised device, malware will be injected. And once the bad stuff is on a device, it's very very hard to get rid of it – and even harder to be sure a device is clean. Most folks (including us) recommend you just re-image the device, as opposed to trying to clean the malware, which costs time and money.

This makes it even more important to detect malware as quickly as possible at the point of entry and (hopefully) block it before a user does something stupid to compromise their device. There are many ways to detect malware, depending on the attack vector, but a lot of what we see today sneaks in through port 80 as web traffic. Sure, dimwit users occasionally open a PDF or ZIP file from someone they don't know (and get owned), but more often it's a drive-by download.

Let's examine two situations to detect malware, one with a purpose-built device to protect against web malware, and another where we're analyzing malware directly on the network analysis platform.

Detecting Malware at the Perimeter

As we've been saying throughout this paper, extending data collection and capture beyond logs is essential to detecting modern attacks. One advantage of capturing the full network packet stream at the ingress point of your network (basically your perimeter) is you can check for known malware directly on the analysis platform and alert as it enters the network. This approach is better than nothing but it has two main issues:

1. **Malware sample accuracy**: This approach requires accurate and comprehensive malware samples already loaded into the network packet capture device to detect the attack. We all know that approach doesn't work well with endpoint anti-virus and is completely useless against zero-day attacks, and this approach has similar issues, since it's only as good as what it's looking for.

2. **No blocking**: Additionally, once you detect something on the analysis platform, your options to remediate are pretty limited. Alerting on malware entering the network is useful, but blocking it is much better.

Alternatively, a new class of network security device has emerged to deal with this kind of sneaky malware, by executing the malware (using cool virtualization technology) as it enters the network to understand the behavior of the inbound executables. Again, given the prevalence of unknown zero-day attacks, the ability to classify known bad behavior and see how an executable file actually behaves can be very helpful. In a lot of cases, you won't be able to detect it based on what the malware looks like, but you'll know it based on it's bad behavior. Of course no device or detection technique is foolproof, but these devices can provide earlier warning of impending problems than traditional perimeter network security controls like traditional firewalls and IPS devices.

Using these new web malware gateways you can block offending traffic at the perimeter, reducing the likelihood of device compromise. Obviously you can't guarantee you will catch all malware, so you must figure out the extent of the compromise. But that's not the only option for advanced malware detection. You should also analyze outbound traffic, looking for traffic to known command and control targets, which usually indicates a compromised device. At this point the device is already pwned, so you need to contain the damage.

Either way, you must figure out *exactly* what happened and whether you need to sound the alarm.

Containing the Damage

Based on the analysis on the perimeter, we know both the compromised device and the originating network address. With our trusty analysis platform we can then figure out the extent of the damage. Let's walk through the steps:

1. **Evaluate the device**: First you need to figure out if the device is truly compromised (or if the system fired off a false positive). Your endpoint protection suite might not be able to catch an advanced attack (imagine that!), so search your analysis platform (and logging system) to find any configuration changes made on the device, and look for any strange behavior – typically through network flow analysis. If the device comes up clean all the better. But let's assume it's not.

2. **Profile the malware**: Now you know the device is compromised, you need to figure out how. Sure you could just wipe it, but that eliminates the best opportunity to profile the attack to make sure it doesn't happen again. The network traffic and device information enable your analysts to piece together exactly what the malware does, replay the attack to confirm, and profile its behavior. This helps figure out how many other devices have been compromised, because you know what to look for.

3. **Determine the extent of the damage**: Next you track malware proliferation. You search the analysis platform to look for the malware profile you built in the last step. This might mean looking for communication with the external addresses you identified, identifying command and control patterns, or watching for the indicative configuration changes; but however you proceed, having all that data in one place facilitates identifying compromised devices.

4. **Watch for the same attack**: Shame on you if you let the same attack succeed on your network. Add rules to detect and block attacks you have seen for the future.

We have acknowledged repeatedly that security professionals get no credit for blocking attacks, but you certainly look like a fool if you get compromised repeatedly by the same attack. You are only as good as your handling of the latest attack. Always remember that.

The Breach Confirmation Use Case

Our last use case involves breach confirmation: confirming and investigating a breach that has already happened. There are clear similarities to the forensics use case, but breach confirmation takes forensic analysis to the next level: you need to learn the extent of the breach, determining *exactly* what was taken and from where. So let's revisit our scenario to look at how that can be extended to confirm a breach.

In that scenario, a friend at the FBI gave you a ring to let you know they found some of your organization's private data during a cybercrime investigation. In the initial analysis you found the compromised devices and the attack paths, as part of isolating the attack and containing the damage. You cleaned the devices and configured IPS rules to ensure those particular attacks will be blocked in the future. You also added some rules to the network security analysis platform to ensure you are watching for those attack traffic patterns moving forward – just in case the attackers evade the IPS.

But you still can't answer the question: "What was stolen?" with any precision. We know the attackers compromised a device and used it to pivot within your environment to get to the target: a database of sensitive information. We can't assume that was the only target, and if your attack was like any other involving a persistent attacker, they found multiple entry points and have a presence on multiple devices. So it's time to head back into your analysis console to figure out a few more things.

> Breach confirmation takes forensic analysis to the next level: you need to learn the extent of the breach, determining *exactly* what was taken and from where.

- **What other devices were impacted**: Start by figuring out how many other machines were compromised by searching all traffic originating from the compromised DMZ server. This analysis shows which devices were scanned and possibly owned. Then you confirm the attack using either the configuration data you've collected, or by analyzing the machine using an endpoint forensics tool.

- **What was taken**: Next you need to figure out what was taken. You already know at least one egress path, identified during the initial forensic analysis. Now you need to dig deeper into the egress traffic captures to see if there were any other connections or file transfers to unauthorized sites. The attackers continue to improve their exfiltration techniques, so it's likely they'll use both encrypted protocols and encrypted files to make it hard to figure out what was actually stolen. So having the full packet stream ins't a foolproof means to analyze the actual files (and see exactly what was taken). It may not be possible to break the crypto, so by default you likely assume the worst case scenario that all the data on the compromised machine is in the wild.

Remember the first wave of forensic investigation focuses on determining the attack paths and gathering enough information to do an initial damage assessment and remediation. That's all about reacting faster and containing the immediate damage as best you can. This next level of forensic analysis is more comprehensive, with a focus on determining the true extent of the compromise and inventorying what was taken as best you can. As you can imagine, without having the network packet capture it's impossible to do this level of analysis. You would be stuck with log files telling you what happened, but not what, how, or how much was taken.

That's why we keep harping on the need for more comprehensive data on which to base network security analysis. Clearly you can't capture all the data flowing around your networks, so it's likely you'll miss something. But you will have a lot more useful information for responding to attacks than organizations which do not capture traffic.

Summary

In today's environment, you can't assume you will be able to stop a targeted attacker. A much smarter and more realistic approach assumes your devices are already compromised, and acting accordingly. This assumption puts a premium on detecting successful attacks, preventing breaches, and containing damage. All those functions require a broader and more detailed level of data collection to understand what is happening in your environment.

- **Log Data Is Not Enough**: Most organizations start by collecting their logs, which is clearly necessary for compliance purposes. But log aggregation is not sufficient – not by a long shot. Additional data sources – including configuration, network flow, and even the full network packet stream, provide key information to understanding what happens in your environment with far more granularity.
- **Forensics**: We walked through how these additional data sources can be instrumental in a forensic investigation, ultimately resulting in a breach confirmation (or not). The key objective of forensics (and the more detailed breach confirmation analysis) is to figure out what happened. The ability of a network security analysis platform to replay attacks and monitor egress points (using the actual traffic) replaces forensic interpretation with tested fact. And forensics folks like facts much better.
- **Security**: These additional data sources are not just useful *after* an attack has happened, but also to recognize issues earlier. By capturing and analyzing the traffic streams within your perimeter and critical network segments, you can spot anomalous behavior and investigate preemptively. To be fair, the security use cases are predicated on knowledge of what to look for, which is never perfect. But in light of the number of less sophisticated attackers using known attacks (because they can), making sure you don't get hit by the same attack twice is very useful.

We all know there are always more projects than your finite resources and funding can support. So why is network security analysis something that should bubble up to the top of your list? We don't know much about future attacks, but we do know they will be hard to detect and will steal critical information. We built the Securosis security philosophy on Reacting Faster and Better, focusing on gathering as much data as possible — which is appropriate regardless of what techniques they come up with tomorrow, but requires an institutional commitment to data collection and analysis. **You cannot prevent all attacks, so you need to be good at detecting what has already happened.**

Never forget that network security analysis is not a shiny object you can bolt into a data center rack, then set and forget. As long as your organization remains in business, you will need to continuously collect, analyze and act on what you find. But you cannot get anywhere until you start. Once you have a core log aggregation capability in place (which you need for compliance), the next place to invest is in a capability to analyze network traffic – typically a full packet capture platform.

You don't have to capture everything at first, and in fact we suggest you start slowly, capturing from only your most important and exposed network segments: the perimeter and protected databases (usually for PCI purposes). That's what we call the Full Packet Capture Sandwich – then strategically capture more, based on need in other areas of your network. But the most important step is simply to get started – the bad guys certainly have already.

If you have any questions on this subject, or want to discuss your situation specifically, feel free to send us a note at info@securosis.com or ask us a question via The Securosis Nexus (http://nexus.securosis.com).

About the Analyst

Mike Rothman, Analyst/President

Mike's bold perspectives and irreverent style are invaluable as companies 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 executive roles at CipherTrust and TruSecure. 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 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.

About Securosis

Securosis, LLC 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:

- **The Securosis Nexus**: The Securosis Nexus is an online environment to help you get your job done better and faster. It provides pragmatic research on security topics that tells you exactly what you need to know, backed with industry-leading expert advice to answer your questions. The Nexus was designed to be fast and easy to use, and to get you the information you need as quickly as possible. Access it at https://nexus.securosis.com.

- **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 conform to our Totally Transparent Research policy.

- **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. More information on our retainer services (PDF) is available.

- **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. For more information about Securosis, visit our website: <http://securosis.com/>.