SANS

Analyst Program

Whitepaper

Packets or It Didn't Happen: Network-Driven Incident Investigations

Written by **Jake Williams**

May 2021

Symantec

In the context of security, it seems as if endpoint detections are currently all the rage. Determining whether endpoint or network monitoring and detection is more important, however, is not the purpose of this paper. Let's be clear: Endpoint detection and prevention solutions are extremely important and will likely always remain so. But in today's complicated threat landscape, endpoint detection and response (EDR) are almost table stakes.

As organizations seek to mature their threat detection and response capabilities, they need to include **network detection and response** (NDR). But those asking the question of whether they need EDR or NDR are missing the point. We are not in a situation where maturing to include NDR means that EDR isn't valuable. Rather, combining the power of NDR with EDR (sometimes referred to as *XDR* or *cross-layered detection and response*) provides visibility that simply isn't available with either solution alone. For instance, your EDR solution might help you discover a malicious binary, but did that binary exfiltrate data? The NDR system knows. Your NDR system might identify command and control (C2) traffic, but what privileges does the malware running on the endpoint have? The EDR knows. The capability to pivot seamlessly from network to endpoint and back again provides substantial benefits to analysts and enables them to answer critical stakeholder questions during an incident.

Threat Hunting: Endpoint or Network?

While it is true that the volume of data to analyze is larger on the network, packets never lie. With network data, the investigator is only dealing with relevant data. After all, everything that is sent to or from an endpoint is interesting to the user, an attacker, or potentially both. To put this another way, when looking at network data, there's very little noise. What little there is can be quickly reduced. For instance, it is easy to simply filter out all `NTP` or `ICMP` traffic if the analyst decides it's not relevant to the investigation.

Contrast this to hunting on an endpoint. Whereas with the network data the analyst is looking at is either content or metadata, the endpoint offers a virtual cornucopia of possible threat hunting avenues: the registry, the filesystem, running processes, open handles, loaded drives, and the list goes on. Even within the registry, there are hundreds of potentially interesting artifacts that must be considered during a threat hunt.

Endpoint logs may not show ground truth, but packets do.

Add to this the fact that the anti-forensics used by attackers are usually trivial on the endpoint and it's clear to see why so many threat hunters prefer to start on the network. Attackers can often hide on the endpoint using techniques such as rootkits. But if they want to communicate and/or exfiltrate data, they must talk on the network. For that reason, we're reminded that packets never lie. Endpoint logs may not show ground truth (information provided by direct observation), but packets do.

The Network Is Ideal

While it is possible to be stealthy on both the network and the endpoint, there is little doubt that the network offers an ideal position from which to collect data for monitoring and incident response. This is true for at least four reasons.

- Unlike endpoint operations, it is impossible for attackers to remove evidence of their activity from a network sensor after the fact.
- While a network may have thousands or tens of thousands of endpoints, it likely has far fewer network egress paths.
- In most incident response scenarios, it is more important to understand what data was exfiltrated from a compromised system than it is to determine how the attacker performed specific actions on the system in the first place.
- Network traffic gives the analyst empirical findings about when the attack began. Anti-forensics on the endpoint may obfuscate the attack timeline, but delivery of malware on the wire can't be hidden and the timestamps can't be changed.

Network data gets us closer to answering the ever-important question of what data was stolen than endpoint forensics do. Of course, the appropriate network traffic capture can often answer many of the questions regarding what was done on the endpoint. Conversely, endpoint forensics rarely answers the question of what data was stolen. While endpoint forensics can certainly highlight attacker intent by enumerating the data viewed, the network shows this, too. See Table 1 for an example of using endpoint forensics versus full packet capture on the network to conduct an incident investigation.

Scenario: An attacker accesses a compromised system via **RDP** *with stolen credentials and launches programs on the compromised machine. But what did the attacker do? Depending on the event logging configuration, there's probably very little available for the investigator to analyze.*

Table 1. A Tale of Two Investigations

Endpoint Forensics Method	Full Packet Capture on the Network
The investigator can perform endpoint forensics on the user profile to discover files that might have been accessed. However, assuming the attacker is utilizing a valid user account, the investigator will have to distinguish between which files were accessed by the attacker and which were accessed by the legitimate user. Timestamps are unlikely to be helpful here because only registry keys have timestamps. (Registry values do not.) After the attacker identifies data of interest (which may never have been opened on the compromised machine), they typically compress it into an archive and exfiltrate the archive via a portable **SFTP** application. Once the attacker has exfiltrated the archive, they securely delete the file. Although some filesystem artifacts might remain, these are ephemeral and are usually overwritten relatively quickly (especially compared to the time between compromise and identification).	If the analyst has access to the full **RDP** session packets, they can use the private key stored on the compromised endpoint to decrypt the traffic and identify exactly what the attacker saw during the **RDP** session. (This assumes Perfect Forward Secrecy [PFS] is not in use.) Even when PFS is used for the **RDP** session, the analyst can still identify the following from the **SFTP** session packets that would likely not be captured on the endpoint: • The IP address of the exfiltration server • Any other services contacted on the exfiltration server (may also serve some C2 functions) • The relative size of data transferred • The public key used to secure the **SFTP** server (may be used for threat hunting)

Anti-Forensics and the Endpoint

Although anti-forensics have always been a concern, they have taken on an especially important role in the past few years. For instance, on Windows (unlike Unix/Linux), event logs traditionally have been difficult to surgically modify. Cleaning an event log on a Windows machine was typically an all-or-nothing operation, meaning that if an attacker wanted to clean an individual log, they had to clear the entire log. Some attackers (apex predators) had tools that allowed them to surgically edit Windows event logs, but that tooling was not widely available to most threat actors.

All of this changed in 2017 with the release of offensive cyber tools by The Shadow Brokers.[1] Suddenly, everyone had access to tools that allowed them to surgically modify event logging on an endpoint. These tools also show how attackers are increasingly hiding on the endpoint. At approximately the same time, we began to hear more about the importance of stealth in cyber investigations (sometimes referred to as *operational security* or *OPSEC*) and how that could be used most effectively.

On the endpoint, your detection and investigation tools work at the same privilege level as the attacker. On the network, the attacker's only chance is to hope they can blend into the noise—a much more difficult proposition.

Sensor Placement Is Key

When considering network traffic capture, sensor placement is key. See the checklist in the "What to Consider with Network Traffic Capture" box.

Ultimately, you can't analyze what you don't capture in a usable form. Sensor placement can make or break any deployment, making careful consideration of what you expect to be able to monitor critical. The most common sensor placement issue, by far, is deploying the sensor before the VPN concentrator can decrypt the traffic. In such a configuration, the analyst is robbed of deep packet inspection and the rich metadata it provides.

What to Consider with Network Traffic Capture

Do you want to capture full content, metadata, enhanced metadata, simply NetFlow records, or some combination thereof?

How long will you retain captured data for use in an investigation? Note that retention times are often different for different types of capture.

Will you monitor north-south (ingress/egress) traffic, east-west (internal network or user to data center) traffic, or both?

If you are performing east-west monitoring, will you capture data at all inter-VLAN routing points, only at data center ingress and egress points, or at some combination?

If you are performing north-south monitoring, how many egress points will you cover?

Will the organization perform break and inspect for `TLS` traffic?[2]

Where will you position sensors relative to VPN concentrators, NAT gateways, or proxies?

How will you cover cloud deployments?

Ubiquitous Use of Network Encryption

Some analysts cite the increasing use of encryption as a reason why they don't need network traffic capture in their control frameworks. While it is certainly true that more network traffic is encrypted than ever before, that doesn't mean investigators can't identify useful findings from that traffic. For instance, Server Name Indication (SNI) provides the analyst with additional information not present in NetFlow. (Note: SNI may be encrypted in TLS[3] 1.3.) Likewise, many fields in the certificate are also interesting for investigators, though TLS 1.3 can add some challenges here as well. In some cases, certificate analysis alone can be used to positively identify a threat actor because many attackers use invariants in their certificate generation frameworks that are easily detected with the appropriate analysis.

---

[1] The Shadow Brokers, wikipedia.org/wiki/The_Shadow_Brokers

[2] "Managing Risk from Transport Layer Security Inspection," National Security Agency, https://media.defense.gov/2019/Dec/16/2002225460/-1/-1/0/INFO%20SHEET%20%20MANAGING%20RISK%20FROM%20TRANSPORT%20LAYER%20SECURITY%20INSPECTION.PDF

[3] Transport Layer Security, Wikipedia, https://en.wikipedia.org/wiki/Transport_Layer_Security

While we can certainly identify useful information while examining encrypted network traffic, there's truly no substitute for TLS break and inspect for many applications. The decision of whether to perform TLS break and inspect is guided by resources available. Additionally, the organization must consider local laws and regulations because TLS break and inspect may affect privacy.

Although most communications traversing the internet today are encrypted, many communications on internal networks are not. For those organizations performing east-west monitoring, TLS break and inspect does not pose the same challenges as it does in north-south monitoring. For those wishing to obtain maximum value from content capture in north-south monitoring, however, TLS break and inspect is critical.

Enriched Metadata vs. Packet Capture

Many organizations that deploy network monitoring feel that NetFlow falls short of providing the necessary data to investigate, especially when threat hunting. The reality is that while NetFlow is better than nothing, it is actually quite difficult for most analysts to investigate. This is particularly true for analysts who need to be able to see higher-level protocol information to meaningfully contextualize the data they see in their threat hunt.

Many of the organizations that realize NetFlow alone is insufficient for detailed analysis also realize that packet capture alone is not an acceptable alternative. Packet capture is large; and the storage requirements for weeks or months of packet capture data often make this an unattractive or infeasible option, particularly for east-west deployments where traffic volumes are extremely large.

The middle ground between extremely large (but insightful) packet capture and small (but difficult to use) NetFlow is to capture enriched metadata. This metadata is generated as the result of protocol decoding and generates fields useful for the analyst. For example, consider an `HTTP` request. Metadata might include the User-Agent, request method (`GET`, `POST`, or similar), and page requested. However, all `POST` variables or other `HTTP` headers that could be viewed with full packet capture would not be available unless configured for storage. Because the metadata does not contain the content, it is also easier to deploy in regions where privacy laws may restrict the use of full packet capture.

So, metadata is richer than NetFlow, but less rich than full packets. Can we have our cake and eat it too? This is one of the rare situations in life where the answer is a resounding, "Yes." By deploying an intelligent capture system, investigators can capture enriched metadata and have long-term retention but a shorter period of retention for full packet capture. Additionally, findings from the enriched metadata can trigger automatic retrieval of specific packets from the buffer for long-term storage. The investigator won't have all packets from a specific time period, but they will have the packet capture most likely to provide context in their investigation. Clearly, although there's no substitute for full PCAP, intelligent capture systems are the future of network monitoring for many environments.

Although there's no substitute for full PCAP, intelligent capture systems are the future of network monitoring for many environments.

Proactive Incident Response Case Studies

In this section, we consider case studies where packet capture and other network data is used in proactive incident response. In some cases, the network data drives the response, while in others the data is useful in the incident response activity.

Confirming Endpoint Anti-Forensics

As mentioned previously, attackers are increasingly using anti-forensics techniques in their operations. In this case study, we examine how analysts can confirm that the attacker has employed anti-forensics techniques by using network packet capture.

The organization employs a packet capture appliance to monitor activity into and out of the data center (a limited east-west deployment). After compromising an endpoint through a phishing email, the attacker discovers credentials for an account with administrative permissions on a file server in the data center, effectively elevating their privileges. The attacker authenticates to the file server over **SMB** and uses the authenticated connection to trigger execution of a malicious payload by using a scheduled task.

After the attacker gains access on the file server with elevated privilege, they scour the system for files of interest, discovering files that they did not have access to remotely. The attacker creates a staging directory from which to exfiltrate the collected data and places 920MB of sensitive files there. The attacker does not exfiltrate the data directly because large outbound flows from a file server would be extremely suspicious to anyone monitoring the network egress. Instead, the attacker copies these files by accessing them from the compromised endpoint over **SMB**. Even if someone is monitoring the network, this looks like normal activity: a user connecting to a file server and retrieving files. Finally, the attacker exfiltrates the files from the compromised endpoint using their OneDrive account. To remove evidence on the file server, the attacker uploads **`sdelete.exe`** over SMB and securely deletes the staging directory and all staged files.

Two days after the data was exfiltrated, one of the attacker's tools on the compromised endpoint is detected by antivirus. There are no other indications of malicious activity on the endpoint, so the investigator quarantined the malware and decides (thankfully) it's worth spending a few minutes investigating.

The attacker has taken several anti-forensics steps to obfuscate their activity and to make investigation much more difficult, including:

- Not exfiltrating data from the file server directly
- Using a file sync service (OneDrive) to exfiltrate data from the endpoint
- Securely deleting filesystem evidence on the file server

Let's examine how network packet capture data can provide investigators context into each of these actions performed by the attacker.

First, assuming that the systems do not use **SMB 3.0** for their communication, all **SMB** communication is in plaintext (rather than being encrypted). Even for systems that can use **SMB 3.0**, encryption is not enabled by default (to prevent unnecessary system overhead for LAN communications). As a result, our network sensor is able to capture the full context of the communications between the compromised endpoint and the file server. Second, we can assume that any network where east-west network monitoring is deployed also has north-south network monitoring.

Not Exfiltrating Data from the File Server Directly

The attacker does not wish to cause an alarm by sending the data from the file server directly to the internet (assuming this is even allowed). However, when the attack is detected, the investigator can view the specific filenames that were transferred from the file server to the compromised endpoint. Full session reconstruction can provide file content as well, but for our initial scoping purposes, the filenames provide an adequate understanding of the attacker's motivation as well as the damage caused by the incident. We should note that if full content recovery is never needed (or storage demands prevent full packet capture), a simpler intelligent capture deployment where both NetFlow and metadata (including file and directory names accessed and transferred) are used would still provide substantial value to the investigation. The investigator can quickly identify that filenames were transferred to the compromised endpoint that have nothing to the user's work role. Correlated with the antivirus alarm, it's pretty clear that even if the malware was quarantined, a deeper intrusion investigation is needed.

Using a File Sync Service

Armed with the knowledge that multiple noteworthy (and unusual, given the user context) files were transferred from the file server to the endpoint, the investigator turns their attention to possible avenues of exfiltration. The days of searching NetFlow for a single large file upload are largely in the past, thanks to file sync services such as OneDrive and Dropbox. Even when these tools share large files, they do so by breaking the file into many small chunks and synchronizing each chunk independently. Although this process adds some overhead, it makes synchronizing file changes much easier. Unfortunately, it makes threat hunting and incident investigations using network data much harder. This is further complicated by the fact that a single service (such as OneDrive) may use multiple servers to sync to, even over the same session. Reconstructing this activity with NetFlow alone is no fun, to say the least.

In this case though, the security team deployed an intelligent capture system with deep packet inspection that provides rich metadata with their NetFlow. The team is immediately able to identify the OneDrive traffic from the compromised endpoint without having to jump through hoops trying to align IP addresses with OneDrive domain names. The intelligent capture system saw the DNS response and correlated it appropriately. OneDrive was a natural choice for the attacker because it is already used within the organization. Again, we see the attacker taking steps to blend into the environment—and it probably would have worked if all the analyst had was vanilla NetFlow.

The analyst queries over the past 30 days to determine whether OneDrive was regularly used on the compromised endpoint. While it was, the average daily outbound traffic from OneDrive hovered around 30MB—until two days ago. At that point, the analyst sees that there was almost a gigabyte of outbound file sync to OneDrive, certainly an anomaly. While the analyst can't see the specific filenames exfiltrated in the network data (even with metadata captured, this data is not available from OneDrive due to encryption), they can deduce the amount of data transferred and conclude this was likely attacker activity. Correlating this with the rich metadata captured by the east-west sensor and the fact that the size of transferred data roughly matches the size of the exfiltrated data, Occam's Razor suggests that this is the data. This assumption will be confirmed with endpoint forensics later, but the analyst is jump-starting the investigation using network data. Also, due to resource constraints, very few endpoints receive forensic analysis after an antivirus alarm. The availability of the network data actually ensures that this incident isn't seen as just one more commodity infection and identifies the scope of the intrusion before the data shows up on the Dark Web.

Securely Deleting Filesystem Evidence

Now, armed with the knowledge that the attacker pivoted from the compromised endpoint to the file server, the analyst can direct a forensic examination of each machine. In any endpoint forensic investigation, the possibility of anti-forensics techniques such as file wiping must be considered. In this case, the analyst knows that **`sdelete.exe`** was transferred over **`SMB`** from the compromised endpoint to the file server. The endpoint forensic investigator now knows to be on the lookout for missing files and directories and that carving to discover deleted files will be minimally useful. The forensic investigator should also be on the lookout for other anti-forensics techniques used because these techniques are rarely used alone.

Case Study Conclusion

We note that without an east-west deployment of an intelligent capture system with enhanced metadata, the SOC analyst likely would have classified this incident as just another antivirus alarm (likely ending the analysis and closing the ticket). Even NetFlow would not have enabled the analyst to identify the importance of the alert quickly. When the investigation began, the data provided value that would have been difficult (and in many cases impossible) to discover from endpoint forensics—the specific filenames (and perhaps also file content) pilfered from the file server and exfiltrated from the compromised endpoint. Metadata and packets have fundamentally changed the game. Attackers can perform anti-forensics on the endpoint, but network traffic never lies.

Metadata vs. Packet Capture

Although packet data is the only authoritative source for ground truth analysis, as we saw in the previous case study, it's obviously not the only source that provides value in incident investigations. Many organizations today struggle with the volume of network data collected—a volume that is growing constantly. To alleviate this issue, some organizations choose to capture NetFlow with enhanced metadata for long-term storage and keep the corresponding packets for a shorter period of time. In this case study, we'll examine how the combination of these approaches provides the ammunition necessary for a cohesive incident investigation.

The organization has a number of web servers that are accessed through a load balancer. The load balancer handles **TLS** termination and sends requests to the three web servers that handle requests. A network traffic capture tool is deployed that combines long-term metadata storage with short-term full packet capture.

While hunting in the logs, the analyst performs some long-tail analysis on pages accessed and **HTTP** response codes. They find that an oddly named web page, **status1.php**, has both **200** (success) and **404** (page not found) response codes. The page always receives **POST** requests. The analyst digs into the data a little more and discovers that one of the servers always returns a **200 HTTP** status code for **status1.php**, but the other two servers return **404** responses. This activity seemingly indicates that the page exists on one server but not the other two. Could the analyst have stumbled onto a DevOps problem where changes were made to only one server? Working under this theory, the analyst checks the earliest date that the page exists and finds that it was accessed successfully 32 days prior. Unfortunately, due to storage limitations, the organization only keeps 14 days of packet capture, so the analyst can't observe the first requests to the new web page.

At this point, the analyst needs to understand more about the **status1.php** page, so they dig into the 14 days of packet capture data to view the **POST** variables and the data returned. Given a little analysis, the analyst quickly deduces that this is a web shell. Because of the packet capture, the analyst can view all activity from the web shell for the past 14 days, building a profile of the attacker's behavior and finding indicators that the attacker likely has used the web shell to pivot deeper into the environment.

Case Study Conclusion

In this case study, the analyst discovers a web shell through threat hunting network data. NetFlow alone would not have been useful because we expect large volumes of **HTTP** traffic to web servers. Without decryption at the load balancer, there would have been far less metadata to observe, so sensor placement here was a key enabler. Finally, the organization benefited from the availability of packet capture data, even though it did not provide a full historical record. By ensuring six months of metadata retention and 14 days of full packet capture, the organization balanced its long- and short-term needs beautifully. Had the hunting operation been performed earlier, the packet capture data would have revealed specifically how the web shell was uploaded. In this case, the analyst can likely only identify the web pages accessed immediately before the web shell was uploaded (this information is still probably enough to identify the vulnerable page). Also, while the web shell might have been discovered through analysis of log data on the web servers, it would not have revealed anything about the commands executed or the specific data returned to the attacker (**HTTP** logs don't contain **POST** data).

Network Traffic Capture and SOAR: A Perfect Match

Security orchestration, automation, and response (SOAR) is a technology widely deployed in security organizations today. The guiding principle behind SOAR is to automate security responses in cases where playbook actions are routine, freeing the analyst for the more complex decisions that don't have predictable workflows. Let's examine a case where SOAR can be used with network traffic capture.

The organization has an intelligent capture network monitor that is logging NetFlow with enhanced metadata at the network egress. Additionally, the organization employs a packet capture system that logs a rolling collection buffer that is usually retained for about 24 hours.

In an earlier investigation, the analyst noted an attacker whose malware C2 server utilizes a self-signed **TLS** certificate. One of the fields in the certificate indicates the "not valid before" date was April 1, 2015. This stood out to the analyst because the certificate expired in 2021. Nobody would be issuing **TLS** certificates with a six-year validity, would they? Later, the analyst found another C2 server and the **TLS** certificate used the same "not valid before" date.

Armed with this information, the analyst creates a detection and SOAR runbook. When the network traffic capture system sees a "not valid before" date of April 1, 2015, it generates an event in the SOAR platform to execute the following actions:

- Insert a firewall block to prevent access to the destination IP address.
- Save any packet capture from the source and destination IP addresses noted for additional analysis.
- Task the EDR to gather information from the impacted endpoint.
- Query threat intelligence sources for the destination IP and domain in the subject field of the **TLS** certificate.

Network traffic capture data from the endpoint paints a clear picture of a download that infected the endpoint after the user accessed a typosquatting website. The user intended to download an executable program (a portable app) but misspelled the domain name and received malware instead. The size of the packet capture from the affected system to the C2 server shows that there was not significant data exfiltrated and the attacker likely was only able to execute survey commands.

Case Study Conclusion

Given that the analyst generated the rules used to create the alert and the SOAR runbook did most of the heavy lifting, it should be clear how integral network monitoring is to starting the entire process. The SOAR runbook can only save the network packet capture data for additional analysis. Without the packet capture data, the analyst can't quickly tell how the malware appeared on the machine—it can only reveal that something is there. Furthermore, the fact that so many actions occurred automatically as a result of SOAR integration means that the analyst could devote more time to performing the deep analysis required to understand the scope of the intrusion.

Incident Response in the Cloud with Packet Capture

Just because your organization has adopted a cloud-first approach to deployments doesn't mean that network monitoring needs go away. If anything, network monitoring in the cloud is even more important than traditional on-premises deployments. Unlike on-premises servers, cloud assets are often not placed behind a dedicated perimeter firewall. Even when they are, understanding who is connecting to these servers (and how) is a key enabler for proactive incident response and threat hunting.

The organization is operating a database server in the cloud that unfortunately was provisioned with weak credentials. Although the organization knew this at the time of deployment, it accepted the risk because of filtering rules applied through the cloud service provider (CSP) environment as a compensating control. Unfortunately, this rule was inadvertently removed in response to a change control request that failed to account for the nature of the compensating control. As a result, the database is now a sitting duck.

Fortunately, the organization has deployed network monitoring, allowing it to view NetFlow and metadata on all servers in the CSP environment. An analyst reviewing network traffic notes an unusually high number of connections to the database server, something that is definitely worthy of additional investigation.

After a quick analysis of network metadata, the analyst deduces that the database server has been the victim of a dictionary attack. The analyst can see the specific usernames attempted for authentication, with the attacker finally discovering a valid username/password combination for the user **`dba`**. Once connected, the attacker runs a number of queries before disconnecting.

The organization now knows it has a problem: An attacker has had unauthorized access to the database containing troves of regulated data. Even without network data, perhaps this access would have eventually been discovered through a database review. But without query logs (something only logged with rarely deployed database activity monitoring solutions), the organization can't tell whether the attacker ran a few survey commands or whether they stole all the data from the database. And, if it doesn't know the extent of the intrusion, the organization will be on the hook for a lot of costly victim notifications (probably with free credit monitoring).

But network traffic capture changes the game. While metadata gave us the username used in the attack (**`dba`**), basic NetFlow tells the investigator whether significant data was exfiltrated to the attacker. Upon noting that only about 28k of data was returned to the attacker, the analyst might rightly conclude that only database survey commands were run. The analyst can confidently conclude that the attacker did not dump entire database tables full of sensitive information. Will that be enough to convince the auditor that this is not a reportable incident? Perhaps, but it definitely removes any argument about the scope of the compromise.

But what if full packet capture is in place? In this case, let's assume the use of a MySQL server. For performance reasons and because the **dba** account performs many large queries with substantial data returned, the account is set to not use encryption on transferred data. Normally this setting benefits system performance, but now it saves the day. The analyst can confidently assess the specific queries the attacker ran and the data returned, concluding that the organization dodged a bullet.

Case Study Conclusion

Endpoint analysis alone would likely not have even identified this intrusion, but it certainly would not have facilitated the identification of what was accessed in the way that packet capture can. Even without full packet capture, metadata provided a robust source of data for analysis, enabling the organization to use NetFlow to determine that only a limited amount of sensitive data (at most) was taken. Not only was the intrusion detected through threat hunting in network data, but network data was the most valuable investigative source.

Conclusion

In this paper, we examined use cases for network monitoring and how organizations can use it to drive incident investigations. While packet capture is still the best evidence when it comes to network data, we examined case studies showing how intelligent capture systems that combine enriched metadata and selective packet capture can be extremely useful. We also examined use cases for network monitoring in cloud environments and demonstrated how network capture can be used to combat the proliferation of endpoint anti-forensics techniques. If your organization doesn't have a network monitoring system today or is only capturing NetFlow, consider what an intelligent packet capture system can do for your security program.

About the Author

Jake Williams is a SANS analyst, senior SANS instructor, course author, and designer of several NetWars challenges for use in SANS' popular, "gamified" information security training suite. Jake spent more than a decade in information security roles at several government agencies, developing specialties in offensive forensics, malware development, and digital counterespionage. Jake is the founder of Rendition InfoSec, which provides penetration testing, digital forensics and incident response, expertise in cloud data exfiltration, and the tools and guidance to secure client data against sophisticated, persistent attacks on-premises and in the cloud.

Sponsor

SANS would like to thank this paper's sponsor: