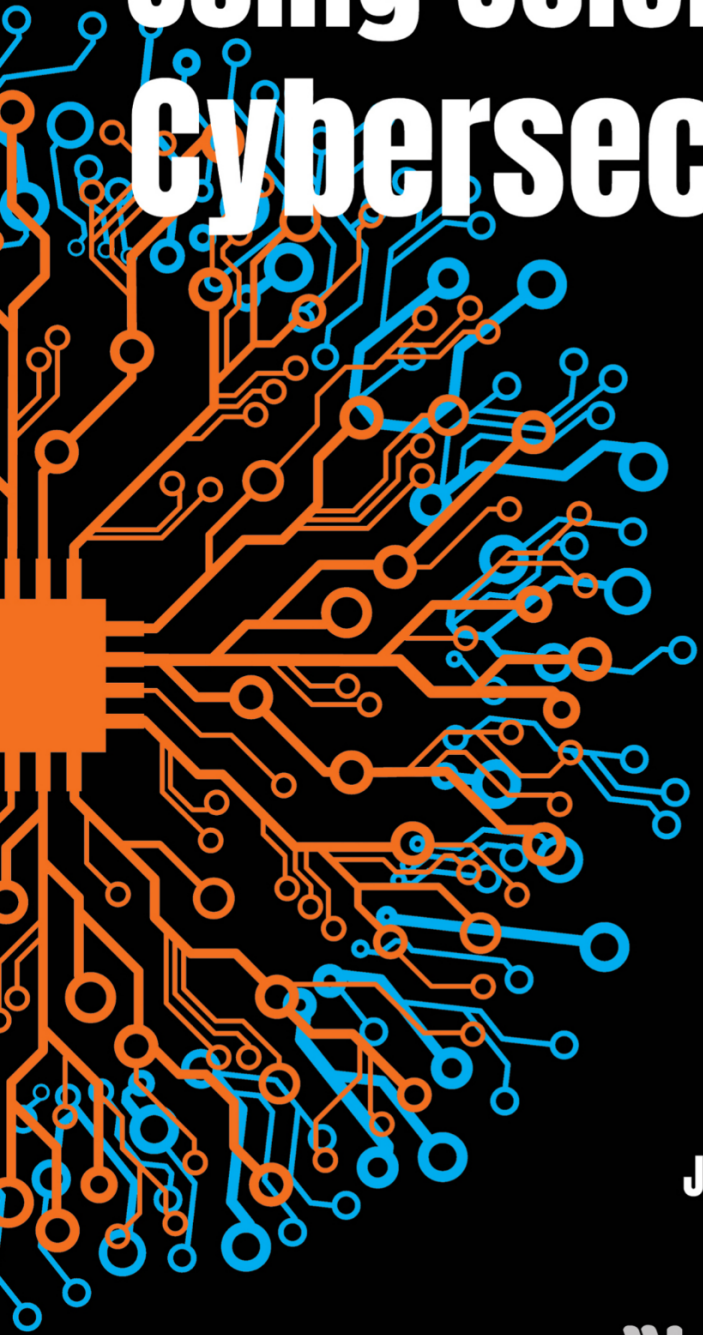
Using Science in Cybersecurity



Leigh Metcalf
Jonathan Spring

World Scientific

Contents

Chapter 1

Introduction

There have been prominent calls for improving cybersecurity through making professionals more academically rigorous as early as 2001 (National Science Foundation, 2001). These calls shifted to pleas for more "science" in security around 2008 in the upper levels of the US Department of Defense (DoD), as documented by MITRE Corporation (2010). Within just a few years, the Air Force, Army, National Security Agency (NSA), and US federal civilian government had joined this chorus, with some minor variation. The governments of the United Kingdom and Canada were using similar language by 2012. The first textbook aimed at giving security professionals a crash course in essential scientific methods was Dykstra (2015). By 2017, academic security researchers (Herley and van Oorschot, 2017) and top professional information security events (Evron, 2017) were discussing what it would take to make security more scientific.

We have written this book to provide an accessible, actionable path for anyone who wants to do cybersecurity work well. We say "well" and not "scientifically" because the only point of doing the work scientifically is that it is done well and that others can trust that it was done well. There are other ways to conduct good cybersecurity work; it's not our way or nothing. But the scientific methods, properly applied, have proven over the past few centuries to be the best way humans have for understanding and solving problems. Engineers' pride may be hurt by this. However, they should not fuss over whether science or engineering comes first; you cannot do science without tools and engineering, and you cannot do modern engineering without knowledge and methods from science (Dear, 2006; Vincenti, 1990).

We have been involved in bringing scientific and mathematical principles into our cybersecurity work for some time. We have been collaborating on using these mental tools to solve cybersecurity problems since 2010. The

first few years were focused on the problems, but a pattern emerged. We brought our backgrounds (philosophy of science and mathematics) to the cybersecurity work and our practice matured alongside our awareness of the broader science of security. We have unique perspectives to share. Of course, we think they are better perspectives. There are important gaps in teaching cybersecurity professionals how to reason about an incident or any problem they face (Spring and Illari, 2018b) and the science of security folks are not filling them (Spring *et al.*, 2017). Leigh wrote a book on applied mathematics for cybersecurity (Metcalf and Casey, 2016), and Jonathan has written almost as much on applying scientific reasoning and logic in cybersecurity. Along the way, we have applied and tested our thinking with results that have changed the way people use and think about blocklists, for example. But we have not laid out a how-to, with examples, explaining the mental tools and practical steps someone can take to practice cybersecurity ~~well~~ scientifically. Or *had* not. Until this book.

Most of the publications in the field are merely emphasizing the fact that scientific principles are necessary, but there are very few guides that aim to uncover these principles. The aim of this book is to begin developing the scientific method for cybersecurity, taking into account the vagaries of the data and the difficulty of the task. We will do this by using extensive examples and also take the time to point out the pitfalls and fallacious thinking that can arise.

We want the reader to learn the basics of how to perform a good study in the field of cybersecurity. We do this by discussing the various studies that are possible for a investigator and how to frame a question appropriately to gain a useful and applicable result.

Cybersecurity is an ever-changing field, which means the results of today may not be the correct results tomorrow. It is also a very broad field, encompassing computers, society, law, economics, and more. It is also a human-created field, unlike biology, for example. The artifacts and events that occur in cybersecurity were created by humans and are not naturally occurring.

But the fact that they're created does not mean they are any easier to understand or more accessible than those in the life sciences. There is no one person or group of people who are wholly responsible for how the Internet or computers work. Some have passed away, but more importantly, there are just too many people who have contributed. And new people are adding new technology and behaviors every day. Cybersecurity is like the life sciences in that there is no creator that we can ask how the systems

work. Practitioners have to study the situation and learn what they can through what tools and information is available. And at least in the life sciences, the viruses cannot read doctors' publications and directly learn what capabilities they need to subvert human defenses.

Cybersecurity is also an inherently practical field. Practitioners who use research want the results to be applicable to their very real and current problems. Usable results, that is. The results of a scientific study in cybersecurity should be usable beyond the study itself.

The chapters of this book are intertwined. We have arranged them in the order that we believe introduces the topic best, so the suggested reading order is Chapters 2 through 8. The last three chapters are examples of applying the principles discussed earlier in the book.

Chapter 2 is a catalog of data found in cybersecurity. We have often noticed that researchers are focused on a single area and aren't necessarily aware of other data sources that can help them. For example, knowing how the data were transported can be as important as the network flow. Route injection can mean that the data originated from a location other than what the Internet Protocol (IP) address suggests. The chapter is not a catalog of all the data available, but attempts to discuss the major data sets, how they work, and what useful information they may contain.

Chapter 3 is about setting goals. The goal in cybersecurity is usually knowing something about the data well enough to support or inform some action. When you know such a thing well enough, you know a truth. When someone goes in search of truths, they should be searching for adequate or satisfactory explanations that constrain and integrate with the other satisfactory explanations that people know about the topic. The chapter describes what this looks like, from both computing and practices in other scientific fields, to establish the goal for practicing a science of security.

Chapter 4 describes the desirable properties of studies and observations that are more likely to lead to this goal. Since cybersecurity crosses so many interrelated disciplines, it cannot simply take the desirably properties from just one other field. Parts of cybersecurity are like physics, parts are like psychology, and parts are like ecology. Chapter 4 works to respect and encourage this diversity of methodology while still usefully guiding how you can design studies in any part of cybersecurity.

The basics of exploratory data analysis are covered in Chapter 5. Statistics is a deep and extensive field; the chapter focuses on introducing the reader to the ability to take a data set and quickly analyze or visualize

it. We discuss what the statistics mean, what the visualizations can do for you, and how to create a good visualization depending on the data.

A common problem in cybersecurity is the amount of data that is available to analyze. It isn't always possible to analyze an entire data set, so sampling is often used. Chapter 6 discusses the basics of sampling and uses examples to illustrate the various kinds. Good and bad examples are given.

Chapter 7 ties the prior six chapters together into advice on types of structured observations to design in cybersecurity. Later chapters will demonstrate examples of designing studies of different types. There are always trade-offs among Chapter 4's study properties; no study can have all the desirable properties. Thus, the second part of Chapter 7 introduces designing research agendas composed of multiple studies whose strengths compensate for each other's weaknesses.

We discuss the goals and pitfalls of research in Chapter 8. The pitfalls can negate or reduce the impact of your research while the goals are what you wish to achieve in the research. This chapter covers these by looking at the data, the process, and the results. We also discuss common logical fallacies and how they can affect the research.

Chapters 9 through 12 use data drawn from open sources to put the principles discussed in the book into action. We look at Domain Name System (DNS) traffic, network traffic, malware, and humans.

The end goal of this work is to encourage research in the field as well as to discuss how to do it in a scientific manner. We want the reader to walk away with a greater understanding and practical help to ensure their research contributes to the field.

Chapter 2

Data in Cybersecurity

An arborist studies trees, so their catalog of available data to study includes a list of trees, the ecosystem a tree is found in, the soil, and other tree-related information. Similarly, cybersecurity research studies events and trends in the Internet, so the data catalog that a cybersecurity researcher would use includes security and Internet-related data. It also includes additional data sets that have been created by external sources. The problem with data created by external sources is that there is no way of knowing how good these data are nor what the provenance of them are. In general it's known that an event happened and data was collected.

This chapter covers common data in cybersecurity. Using the arborist analogy, it's the equivalent of a catalog of trees and their ecosystem that the arborist could use to start a research project. The catalog attempts to list common sources used in cybersecurity research, but it isn't exhaustive. It might seem disconnected as well, and that is mostly due to the nature of the work. DNS data are different and usually distinct from malware data, which is different (and distinct) from data used in Internet routing. Unlike trees, which have the basic connection of "tree," cybersecurity data runs the gamut from human-created to machine-created.

Again, this isn't comprehensive. It should be used to learn how to think about data and the pitfalls in using some of these data sets. Some people tend to focus on a single data set without being aware of the other possibilities available. Part of the goal of this chapter is to expand your knowledge of the available data sets.

It's possible to create a data set for research, but it's necessary to examine the potential problems in that set. No data set is perfect by any means; it's the imperfections that make the research interesting and sometimes difficult.

2.1 Domain Names

DNS is one of the core protocols that makes the Internet run. At its heart, it is the association of IP addresses with domain names. It allows users to type www.google.com rather than memorizing a series of numbers. DNS is the engine behind content distribution networks and allows the owner of a domain to change IP addresses without notifying users.

The protocol was designed to be a hierarchical directory (Liu, 2002). Instead of a single phone book with every domain to IP address listed, it's a telephone book that lists other telephone books that lists other ones. The resolution follow its way through the telephone books until the one that contains the information is found. This means that no one server, known as a *name server*, knows everything, they just know where to ask.

The process of finding the IP address of a domain is called domain resolution, and it works in reverse order by starting with `.`, moving to `com.` then to `google.com.` and finally, to `www.google.com.` In each step, the name servers associated with that step are asked for the answer, and they either give the answer or point the computer to the next server to ask.

DNS is used for more than just the domain name to IP mapping, it has almost forty different types of records. It can be used to determine what domain to send email to (`MX` records), storing information about the domains themselves (`TXT` records, `SOA` records), for security (both for DNS and mail), and more. DNS has been used to send signals as well, which means that the application looks up a domain and, based on the response, has some action.

DNS-based block lists (DNSBLs) (Levine, 2010) create domain names out of either IPs or domains by prepending them to the DNS blocklist domain. If the DNSBL is `example.com` and we're interested in `badguy.info`, then the look up would be `badguy.info.example.com`. The IP address is reversed, so that means `192.0.2.99` would have the look up `99.2.0.192.example.com`. The response from the query is a signal as to whether the IP address or domain has been tagged as bad by the blocklist owner. The responses should be within the 127.0.0.0/8 loopback network, and each application should have its own numbering specification for the results of the query.

In the original specification of the DNS protocol, there was no security built in. Instead, it is a network of trust. The computer trusts that the name server it queries will return the correct response. To resolve www.google.com, there were a minimum of three queries before a response

was returned that contained an IP address. Every step could give the wrong result, and the computer would never know.

Attempts have been made to add security through extensions, known as DNS security extensions (DNSSEC) (Kolkman and Gieben, 2006). As mentioned above, when www.google.com was resolved, it took a minimum of three name servers before any IPs that were associated with it were determined. At any point in the process, those answers could have been subverted and incorrect ones could have been given. To prevent this attack, DNSSEC was proposed. It uses cryptographic signatures to add a verification step to DNS resolutions. It is up to the owner of the domain whether or not to use it, so it isn't used everywhere.

2.2 Routing Data

Routing is the method that sends data through the Internet from the source to the destination. In the days when the Internet began, it wasn't large, so this process was relatively simple. Every router could know the location of every other router. As the Internet grew, the original protocols could no longer support it, leading to the development of two kinds of routing protocols, interior routing protocols and exterior routing protocols. Interior routing protocols are the protocols used inside of an organization; exterior routing protocols are the protocols used between organizations.

Border Gateway Protocol (BGP) (Caesar and Rexford, 2005) is an exterior routing protocol which is designed to route collections of networks between organizations. These collections are called Autonomous Systems (AS) and are denoted by an autonomous system number (ASN). A company is assigned an ASN by their regional Internet registry (RIR).

ASN is associated with a collection of networks; there doesn't have to be a physical location tied to the ASN. The networks associated with an ASN can span multiple countries as well, depending on the networks.

Each ASN has peers with which it shares information. They want their peers to route traffic to its networks, so they do this by telling the peers that they have the networks, known within the protocol as announcing the networks. Technically speaking, for each network the ASN has, it announces to its peers `ASN NETWORK`. In BGP speak this says that "I, ASN, have these networks." For example, a potential announcement could be:

```
64496 10.0.0.1/24
```

This tells the peers of 64496 that it has this network. Each of our ASN 64496's peers will tell its peers their ASN prepended to this announcement.

It looks like:
`PEER 64496 10.0.0.1/24`

This tells their peers that to access an IP address in 10.0.0.1/24, then they first must go to PEER which passes them to 64496, which owns that network. This does not mean that there is a device with that IP address, just that that combination of ASNs in that order will allow traffic to flow to that ASN that owns the IP address. This combination of ASNs in the given order with a network at the end is called a route.

Peering with multiple ASNs allows redundant routes to be present in the virtual ASN network. If the only route available is:
`ASN_A ASN_B ASN_C ASN_D NETWORK`

Then that is the only route that traffic can traverse. If there are multiple routes, then there must be a method by which the route is chosen. The Request for Comments (RFC) (Rekhter *et al.*, 1994) specifies the criteria for choosing a route and it includes:

- the shortest ASN path.
- the most specific network announcement. This means the network with the fewest number of IPs in it will win.
- the highest local preference. This is a value set by the router to determine which peer is preferred.

The other important part about multiple routes is the amount of control that the originator of the traffic has, which is to say very little. The source of the traffic chooses the peer to which it wants to send the traffic to. At that point, the source loses all control of the traffic. The peer chooses its next destination based on its own criteria, not on the origin's criteria. So while the router can say which path it wanted its traffic to take, it doesn't know what the actual path is. This can also be affected by filtering. An organization's peering agreement with another organization may include not announcing certain routes to its other peers, so the data may traverse a completely unknown route.

BGP also has no security built into the protocol (Murphy, 2006). This means that anyone can announce any network, and there's no inherent verification that this ASN is allowed to announce it. An Internet Routing Registry (IRR) is a mechanism (Bates *et al.*, 1995) where the owners of networks can register who announces those networks, but there is no requirement that autonomous aystem (AS) operators respect these. Another

method used to secure BGP is Resource Public Key Infrastructure (RPKI) (Cohen *et al.*, 2015). These certificates are used to authenticate announcements, but there is no requirement that the certificate is used.

2.3 Full Packet Capture

Full packet capture (Koch, 2018) is just what it sounds like. Every packet that traverses a network is captured, meaning copied, and saved for future study. A sensor is placed on the network that collects and stores this data.

Every action a user makes on the Internet is apparent, with some caveats. First, it completely depends on where the sensor is placed. If somehow the user is outside of the coverage of the sensor, that user's actions won't be recorded. The user can also encrypt their connection. If, for example, they visited a secure website, the website they visited would be recorded, when the visit was, how long the visit lasted, and the encrypted data.

On the other hand, if the website was unencrypted, everything would be recorded, from what they typed in to what they received. This means that if the website was used to deliver malware, then the malware can be extracted from the traffic. Every domain and IP address they access is recorded, every Uniform Resource Locator (URL) they click on, every email they receive, and every system they connect to.

The downside to full packet capture is that storing the data can take up a lot of disk space, depending on the size of the organization. Think about how much web surfing a typical user does in a day. Now imagine storing every bit of traffic sent to the Internet and received. Now, multiply that by the number of users in an organization. Add in traffic to the organization's webserver and mail server. In short, this means a lot of data to store.

In 2016, an estimate (Koch, 2018) was made of how much space would be required for 72 hours of full packet capture on a 1 gigabit (Gb) link. The computation determined it would take at least 24.3 terabytes(TB) of space. Not only is storing that amount of data difficult, but searching it becomes an untenable task.

2.4 Network Flow

If full packet capture is "catch everything as it goes by," then network flow is "take the trace of what went by." Think of full packet capture as capturing all the animals that visit a watering hole whereas network flow is examining the footprints left behind. Similar to full packet capture, a sensor is placed on the network and the data is collected.

Network flow captures (Gates *et al.*, 2004):

- Source IP address
- Source Port
- Destination IP address
- Destination Port
- Protocol
- Start time
- End time
- Number of Packets
- Number of Bytes
- Transmission Control Protocol (TCP) Flags

It's a trace of what the user did without storing what the user did. It clearly uses less space than a full packet capture, and so storing more data than full packet capture is possible, making historical analysis possible. However, context is lost. The fact that an IP visited a web server and downloaded 10M of data is recorded, but there's no clear context of what happened during the session.

2.5 GeoIP

GeoIP is the geographic location of an IP address (Holdener, 2011). There are many companies that sell this data, each claiming to be more accurate than the others. This is one of the problems with the data. Researchers must rely upon the company supplying data to tell them the right thing, but there is no way of double checking it short of going to the longitude and latitude given and trying to determine the current IP address. Companies will declare that their data is accurate, but they don't explain how they determine the location of an IP address, nor how they verify that they're right.

It's been known (Hill, 2016) to be very wrong, to the point of 600 million IP addresses pointing to a Kansas farm house due to a lack of precision.

Relying upon GeoIP to locate the origin of traffic has its own issues. Suppose the IP address a researcher is examining is malicious. It could because it is part of a botnet, so the true origin of the traffic is unknown. The owner of the system may not know that their system sent malicious

traffic. Using GeoIP to locate this system gets the researcher no closer to the actual location from which the traffic originated.

In summary, GeoIP data can be used to geographically locate systems, but take the result with a grain of salt, and be careful how it is used.

2.6 Blocklists

Blocklists are collections of any or all the following: IP addresses, domains, URLs, MD5s, and more. We discussed them briefly in Section 2.1, discussing those that were delivered via DNS.

The elements on a blocklist are generally called indicators, in that they indicate malicious behavior. The creators of these lists are looking for malicious behavior and provide them to the public either as open source or for purchase. Organizations then use blocklists to filter traffic, both inbound and outbound. They don't want spam (that's one blocklist to buy) nor do they they want their users to visit sites associated with malware (that's another one). Analysts use blocklists when investigating an incident as well.

In general, the creation of these lists is a black box. Someone decided that an indicator was associated with malicious behavior and added it to their list. If a list is bought from a company that tracks spam, then clearly anything on that list was associated with spam. Unfortunately, there's no direct knowledge on how the spam was created and by what process email was tagged as spam. The only knowledge we have about the list is that the company collected spam email in some way and pulled this information out of it.

Studies of the blocklist ecosystem have shown that there is very little overlap between the lists (Metcalf and Spring, 2013b). Even between lists that collect similar data, like spam lists, there is very little overlap. This could be related to the different methods each list owner uses to create their blocklist, but since the methods aren't disclosed, it's impossible to verify. The blocklist studies also looked at data related to the domains and IP addresses, like name servers or Autonomous Systems, to determine if there was agreement —very little was found.

Suppose a researcher's task is to find malicious domains, and after much research they've created a method to do this. They know that the domains found in their method are malicious because they found a blocklist that had every single one of the domains on it. Since there's very little overlap between lists, what the researcher has done is figured out how to recreate the list.

There can also be an issue with blocklist quality. Private IP addresses, that is, the addresses listed in RFC 1918, shouldn't be routed on the Internet. This means they also shouldn't show up on a well-tended blocklist; however, they often do. This can mean that the blocklist owner isn't performing due diligence before adding elements to their list. Well-known domains can also end up on blocklists, usually due to the ad network that they are using. If the ad network is known for serving up malware, then the well-known domain can be tagged as malicious. Any domain can be used maliciously.

Blocklists can be great sources for malicious behavior, but researchers must be careful when they use them. There's not an indication of why things are necessarily tagged as bad, just that they are on the list. Since the companies producing the blocklists don't share their methods, all researchers can say is "I found a domain that's on a blocklist, so it could be bad" not "it is definitely bad."

2.7 Log Files

Log files are records that applications and operating systems keep of their operation. For example, when an application starts, it can log the time at which it started and the steps it took. An operating system could log every time a user logs in or logs out. It is common for applications and operating systems to log errors, such as when a user attempts to log in, but fails to give the correct password.

Log files are local information. They are concerned only about the system from which they originated but nothing about any other system. If two webservers have similar configurations, it's expected that they have similar logs. On the other hand, if it is one webserver and one nameserver, then the logs would be different.

Logs are often subject to availability. Sometimes, the owner of a system configures logs to store everything, sometimes, they don't.

2.7.1 *Application Log Files*

Suppose a Linux® server is running an ssh daemon that is open to the world. One day the system administrator checks the log files and see a list of failed attempts to log in remotely via ssh. The attempts cycle through a list of user names, most of which are not on the system, and each of them fails. This is a direct attack on a system and the sysadmin is happy to see that they all failed. If they had noticed that one succeeded, then the system would have been compromised.

Application developers (DeLaRosa, 2018) are the ones that determine what an application will record in log files, so if the ssh daemon developers had not decided failed logins were important information, then the intrusion detect wouldn't have been found in the ssh log files. This means that researchers are dependent on what the application developers find worth logging and in general, researchers don't have any say in what gets logged.

Researchers are also dependent on the log retention schedule, which is a predetermined length of time that log files are kept. If log files are only kept for a week, then anything past those seven days is lost. If they are kept for much longer, then it becomes a disk space issue. This is usually determined by the administrator of the system, not necessarily with security in mind. Another issue is that the administrator may determine that some events should be logged and saved whereas other events are not. A mis-configured logging system can lose important security events, preventing later analysis.

Another factor in log files is that the log message format can change, depending on the operating system. The ssh failure on one system can have a completely different format than the message on another. In other words, there's no consistency of the data.

2.7.2 *Firewall Log Files*

Firewall log files (Winding *et al.*, 2006) are a specialized form of the log files discussed in the previous section. They generally have the same issues discussed in the previous section, but also have additional features and issues as well. Firewalls can be in multiple locations. A single host can be running a firewall or a network device may act as a firewall.

When a firewall sees a connection, it has two choices. It can `ALLOW` the connection through or it can `DENY` it. Depending on the configuration of the system, either or both messages can be logged. If a firewall `ALLOW`s a connection, then the connection is allowed to the destination. In the section on network flow, we discussed how the same data is collected. In other words, this is somewhat redundant information. Network flow knows about the connection, the firewall knows about the connection, and it has been recorded in both locations. If network flow is collected, it does seem extraneous to also be collecting the `ALLOW`ed connections from the firewall.

On the other hand, the DENY connections are where it gets interesting. Remember that to collect network flow, a sensor is needed. If the sensor placement looks like Fig. 2.1, then network flow will record that a connection occurred, even though the firewall denied the connection. By reversing the placement of the sensor and the firewall, network flow will no longer

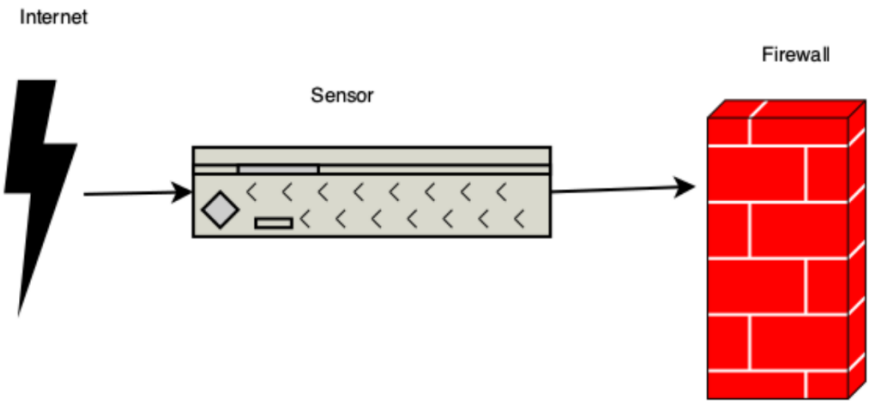


Fig. 2.1 Sensor Placement

record the connection that didn't occur. This means that when analyzing either firewall log files or network flow, sensor placement is key and knowing where the sensors are is important because it can affect results.

2.7.3 *Proxy Log Files*

A proxy is an interface between the users of an organization and the outside Internet. A common proxy is the web proxy, which allows the organization to enact content filtering. Proxies can also cache results and share them between multiple users, which can save bandwidth.

The log files of a proxy server (Fei *et al.*, 2006) are records of what web servers the users of an organization are contacting. Like all log files, they're dependent on the configuration of the appliance as well as the retention policy. If a new form of malware is discovered that's originating from `http://www.example.com/` and the proxy logs only go back 24 hours, then the research is limited. Similarly, if there is a mis-configuration and not every user is required to use the proxy, then if those users visited the site, no one would know.

2.7.4 *Certificate Transparency Logs*

Not every log is on a per-system basis. The *Certificate Transparency Log* is a public log of Transport Layer Security (TLS) certificates as they are issued. (Dowling *et al.*, 2016). No one can modify the logs after they've been written; they can only have log messages appended to them. This is

ensured using a cryptographic mechanism. Anyone can download and use these logs as well.

The logs contain the domain, the issuing authority, the certificate, the issuing date, and more. Any domain can get a TLS certificate, even names generated by a domain generation algorithm (DGA) (Metcalf, 2018b). The difficult question is why.

These logs are freely available and a valuable research tool.

2.8 Windows Registry

The Windows registry is a collection of low-level settings for Windows (Carvey, 2005). It is a hierarchical text file that controls various aspects of the system, such as device drivers. Malicious software often modifies the registry so a savvy investigator will analyze it for clues. There's also no such thing as a standard Windows registry; it's generally a per-system configuration.

These registries are very useful in research, but researchers need to be aware of the details of the system from which they collect them as the details can affect the entries in the registry. One registry may have an entry that another one doesn't because of system configuration. At this time, there is no way to centrally collect windows registries from group of systems; they must be collected one at a time.

2.9 Domain Registry

Suppose a researcher found a domain associated with malicious behavior, and they want to find the owner. Maybe they want to know what other domains they owns to see if they're malicious, or they want to create a report on the malicious domain, or they want to send this information to law enforcement. There are any number of reasons to look for the owner of a domain.

Luckily, this information is available. It's possible to search the registry operator for the top-level domain (TLD) of the domain, or use the command `whois` to query the owner. This should return the name of the organization or person that registered the domain, their address, and contact information as well as the name servers that serve that domain. That's the good news. The bad news is that many domains are protected by private domain registration. This means that rather than getting the name of the organization or person that registered the domain, a corporation that specializes in private registrations is returned as the response. This allows domain owners to maintain their privacy, but at the same time, it hides


image
not
available


image
not
available


image
not
available

Index