COVERT COMMAND AND CONTROL USING THE SERVER MESSAGE BLOCK PROTOCOL

LE COMMANDEMENT ET LE CONTRÔLE DÉROBÉ UTILISANT LE PROTOCOLE SERVER MESSAGE BLOCK

A Thesis Submitted to the Division of Graduate Studies
of the Royal Military College of Canada
by

Robin Edward Moll, CD, BSc(Eng)
Lieutenant-Commander

In Partial Fulfillment of the Requirements for the Degree of
Master of Applied Science in Electrical and Computer Engineering

April, 2018

Abstract

In order to successfully exploit a target network, an attacker must have the ability to exercise command and control (C2) within the network without detection by its defenders. The aim of this research was to investigate how an attacker might exploit existing network support for the Server Message Block (SMB) protocol in order to implement a C2 communications channel that would be difficult to detect through network traffic analysis.

This research demonstrates how a stealthy SMB C2 communications channel can be designed by first characterizing typical SMB network traffic. Once normal SMB traffic parameters are identified, suitable mechanisms for stealthy communication channels are identified, implemented and evaluated as a proof of the concept.

By characterizing the network traffic and by investigating how an attacker might implement a C2 channel, this research demonstrates the threat SMB covert channels may pose to our networks.

Résumé

Afin d'exploiter un réseau complexe, un attaquant doit pouvoir effectuer le commandement et le contrôle (C2) sur ce réseau sans être détecté par ses défendeurs. Le but de cette recherche était d'enquêter comme un attaqueur pourrait exploiter le soutien réseau natif pour le protocole *Server Message Block* (SMB) afin d'implémenter un canal de communication pour le C2 qui serait difficile à détecter par l'analyse du trafic sur le réseau.

La recherche démontre qu'un canal de C2 SMB furtif peut être conçu en caractérisant tout d'abord le trafic SMB typique du réseau. Une fois que les paramètres du trafic SMB sont identifiés, des mécanismes de communications furtives sont choisis, implémentés et évalués pour démonstrer la faisabilité.

En caractérisant le trafic du réseau, et en enquêtant comme un attaquant pourrait implémenter un canal C2, cette recherche démontre la menace que les canaux dérobés SMB portent à nos réseaux.

This thesis is dedicated to my beloved wife and children. I am truly blessed to have had your love and support throughout this endeavour.

Acknowledgements

I would first like to thank my thesis advisor Dr. Sylvain Leblanc of the Department of Electrical and Computer Engineering at the Royal Military College of Canada. The door to Dr. Leblanc's office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right direction whenever he thought I needed it.

I would also like to thank Mike Shuck of the United States Department of Defense who inspired the development of my research topic. Initially suggesting what I considered to be an impossible set of requirements, he forced me to think outside of the box which ultimately led to the impossible becoming possible.

I would also like to thank the technical support staff from both the Department of Electrical and Computer Engineering at the Royal Military College and Defence Research and Development Canada Shirley's Bay Campus who provided outstanding support throughout my research. In particular, a special thanks to General Kenobi, who very politely tolerated my constant dropping by to discuss the latest anomalous behavior observed on the network.

Finally, I must express my very profound gratitude to Dr. Kathy Perrett and the Building 6 lunch crew at Shirley's Bay for making room for me and providing me with a supportive environment throughout this endeavour. In particular, I want to thank Grant Vandenberghe who provided mentorship, support and inspiration for my approach to the research problem.

Thank you.

Contents

List of Tables

List of Figures

Abbreviations

C2	Command And Control
DC	Domain Controller
DFS	Distributed File System
FOSS	Free Open Source Software
GPO	Group Policy Object
IPC	Inter-process Communication
NCP	Network Control Program
NFS	Network File System
POC	Proof Of Concept
SMB	Server Message Block
SRM	Shared Resource Matrix
WDS	Windows Deployment Server
WMI	Windows Management Instrumentation
WSUS	Windows Server Update Services

1 Introduction

In order to successfully exploit a target network, an attacker must have the ability to exercise command and control (C2) within the network without detection by its defenders. Advances in network security architecture and best practices significantly limit authorized communications within an internal enterprise network leaving an attacker with a limited number of viable options for intra-network communication. The SMB protocol remains a tenable contender for transporting covert C2 messages since it is deeply integrated into modern enterprise infrastructure, enabling enterprise wide file and printer sharing. This thesis investigates how an attacker might implement a C2 channel using SMB that would be difficult to detect through network traffic analysis.

1.1 Problem Space

The SMB protocol is complex, consisting of nineteen different request types, as well as a multitude of possible requests and responses associated with each dependent upon the desired functionality. Generally speaking however, the SMB protocol enables authenticated remote file and printer sharing as well as inter-process communication (IPC) over the network using named pipes [1].

Having identified a commonly used protocol for potential C2 transport, the question then becomes one of how to ensure that malicious SMB traffic appears consistent with legitimate SMB traffic. Lateral movement attack techniques have already been devised which make use of SMB named pipes for IPC between compromised hosts within a target network [2]. While these techniques remain largely effective today, using anomaly based intrusion detection methods, network defenders with advanced skill sets are learning to distinguish this malicious C2 traffic from legitimate SMB network traffic [3, 4]. As network defender skills and tools improve, attackers will seek to develop innovative C2 mechanisms to maintain covert persistence on target networks. By investigating how an attacker might implement an improved C2 channel

over SMB, we will be able to better understand the extent of the threat and how to better defend against such an eventuality.

1.2 Research Scope

Previous C2 channel over SMB development work has focused on using the functionality inherent to the protocol to implement a communication system without attempting to characterize normal network traffic patterns. This leaves the previously established SMB C2 channels potentially susceptible to detection. Formal attempts to characterize malicious SMB C2 traffic have been made [3, 4], but the existing work is limited in scope, assuming to a large extent that an attacker will not attempt to conceal their actions or intent.

1.2.1 Aim

The aim of this research is to investigate how an attacker might exploit existing network support for SMB in order to implement a C2 channel that would be difficult to detect through network traffic analysis.

1.2.2 New approach and activities

In order to be be difficult to detect through network traffic analysis, a characterization of normal traffic is required. Understanding typical features of SMB network traffic serves to inform the design of communication channels that will appear consistent with naturally occurring traffic and provides the basis by which both channel capacity and ease of detection can be evaluated. Normal SMB traffic parameters are identified by:

1. applying flow attribute analysis to production network traffic to characterize normal SMB traffic flow attributes, and
2. analyzing production network traffic to characterize specific types of normal SMB conversations (i.e. file server access, print server access, IPC).

In addition, possible mechanisms to communicate C2 information between network hosts using SMB are identified and evaluated for suitability with consideration for generating network traffic that is consistent with the characterization of typical network traffic. Suitable mechanisms are then selected and implemented as proofs of the concept. Finally, the implemented proof of concept (POC) C2 channels are evaluated in terms of their capabilities and limitations in light of the performance trade-offs that are possible.

1.3 Potential Impact

There is currently limited research that seeks to characterize normal SMB traffic flows on enterprise networks. As will be discussed further in Chapter 3, by understanding typical SMB traffic patterns, network defenders can gain a better appreciation of how to better identify anomalous traffic on their networks. In addition, it is reasonable to expect that attackers will attempt to evolve their techniques to minimize the likelihood of detection by seeking to conceal their communications in otherwise normal traffic. By identifying and characterizing likely vectors for potential covert attack, more resources can be dedicated to understanding the extent of the existing vulnerability and how best to mitigate against it.

While this research seeks to identify how an attacker would implement a difficult to detect C2 channel using existing network support for SMB, the traffic characterization effort can also serve to provide a starting point to discuss likely vectors for implementing (and thus detecting) covert channels over SMB at the network layer using more traditional covert network storage and timing channel techniques discussed in [5, 6, 7].

Finally, given that the latest release of SMB (version 3.0) supports encryption [1], a characterization of the unencrypted interactions may lay the groundwork to support future characterization of encrypted SMB traffic.

1.4 Approach Sketch

The remainder of this thesis is presented as follows:

- Chapter 2 - Background Research
- Chapter 3 - Characterizing Live Network SMB Traffic
- Chapter 4 - SMB C2 Communications Channels
- Chapter 5 - Discussion and Conclusion

Chapter 2 includes an overview of the SMB protocol, its uses and origins. In addition, a brief introduction to covert channel development will be presented. This is followed by a review of the network analysis techniques that can be used in order to classify, categorize and characterize network traffic. Finally, a review of existing research relating to the detection of malicious SMB C2 is presented. This background information is important for establishing the context in which the research question has been investigated and answered.

Chapter 3 discusses the characterization of naturally occurring (or normal) SMB traffic on a network using heuristic network traffic analysis techniques.

Once typical traffic parameters are established, viable options for communication channels using SMB are identified and evaluated in Chapter 4.

Chapter 4 further discusses the validation of the work through the design and implementation of POC SMB C2 communications channels. These channels are evaluated in terms of effectiveness and detection difficulty based on their network footprints as compared to the normal SMB traffic patterns identified in Chapter 3.

Chapter 5 discusses the overall results, the implications and recommendations for future work.

2 Background Research

2.1 The SMB Protocol as an Attack Vector

As detailed at [1, 8], the original purpose of the SMB protocol when it was conceived in the 1980s was to implement a networked file system. Since then, the protocol has evolved over many iterations, integrating new capabilities and security features, but always maintaining a focus on preserving backward compatibility. In 2007, a significantly redesigned protocol was released as SMB version 2.0 (SMB2) with the Windows Vista operating system. This new version represented a significant change in overall design, eventually leading to the deprecation of the original SMB protocol (SMB1) with the release of Windows Server 2012 R2. Today, with few notable exceptions (such as Windows XP\Server 2003 networks) SMB1 is no longer recommended for use [9]. As stated by security blogger and principal program manager Ned Pyle at Microsoft: "The original SMB1 protocol is nearly 30 years old, and like much of the software made in the 80's, it was designed for a world that no longer exists."

Notwithstanding recent news about malware such as the Wannacry ransomware [10] and the Notpetya wiper [11] which made use of a SMB1 vulnerability, this older version of the SMB protocol is not particularly relevant to research seeking to characterize modern enterprise traffic and as such it will not be considered further. Any subsequent reference to SMB in this document will necessarily be referring to the SMB2 specification unless otherwise noted.

The SMB protocol remains in use extensively in modern enterprise networks as it enables remote file, directory and share access in an authenticated context. As a general overview of the types of network traffic associated with the protocol, we note the following from the Microsoft website [12]:

> The Microsoft SMB protocol is a client-server implementation and consists of a set of data packets, each containing a request sent by the client or a response sent by the server. These packets can be

broadly classified as follows:

1. Session control packets: Establishes and discontinues a connection to shared server resources.
2. File access packets: Accesses and manipulates files and directories on the remote server.
3. General message packets: Sends data to print queues, mailslots, and named pipes, and provides data about the status of print queues.

Contemporary best practices suggest that user files should be accessed, stored and maintained on network storage devices. This approach can increase durability and availability of user files in an economic manner as network storage solutions can employ redundancy and backup systems not otherwise practical for use in individual hosts. In addition, it remains common practice that shared files are often made available on network storage devices for access by many, often for information sharing or collaboration. With multiple network hosts having the ability to read and/or write to the same location on the network, the possibility of an exploitable covert channel exists [13].

2.1.1 Use of SMB Named Pipes For Exercising Command and Control

Named pipes provide interprocess communication between pipe servers and pipe clients. A pipe server is the process that creates the named pipe and the pipe client is process that communicates with it. The pipe client can be on the same host or running as a process on a remote machine across the network [14]. When a pipe client establishes a connection to a pipe server over the network, this is accomplished by accessing a `named pipe` share using the SMB protocol.

Attack techniques have been devised that maintain C2 through the use SMB named pipes for interprocess communication. This is seen in both the threat emulation software Cobalt Strike [15] and the Duqu malware [16]. In both of these cases, newly infected systems can be configured to connect back to a previously infected host on the local network through a predefined named pipe.

2.2 Covert Channels

A covert channel is defined as “any communication channel that can be exploited by a process to transfer information in a manner that violates the

system's security policy" [17]. Traditionally, there are two types of covert channels: storage and timing. They are defined in [17] as follows:

> **covert storage channel**: includes all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another.
>
> **covert timing channel**: includes all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by the second process would provide information.

Smith and Knight further introduce the following related terms [6]:

> **covert exploit**: the specific technique used to inject hidden data.
>
> **covert communication system**: the collective set of the covert exploit(s) and the channel, the coding scheme, and any other modules necessary to effectively use the channel for communications.

Wendzel et al. perform a detailed survey of existing network covert channel techniques at [7]. A significant contribution they make is the development of general characterizations for the 109 covert channels reviewed, enabling the various techniques analyzed to be reduced to a subset of only 11 covert exploits. Furthermore, the authors found that nearly 70% of the channels characterized could be categorized as only one of four different types of exploits, significantly narrowing the field for analysis of previously implemented covert channels.

In addition, the notion of using different exploits for different bandwidth requirements is introduced, suggesting that covert benefits can be achieved by selecting and tuning exploits as required by the context of the situation. Wendzel et al. also suggest new ideas in terms of how exploits can be varied and combined. They suggest that bandwidth can be increased by combining multiple exploits in parallel and that covertness can be increased by sequentially "hopping" through different covert channels [7].

Zander et al. suggest that many of the covert channels that they analyzed provide security by obscurity [5]. That is to say, detection is often possible, once the exploit technique is known. They add however that there are two situations where a covert channel will be very difficult to detect. Namely:

1. when a covert channel looks identical to normal use; and
2. if there is a lot of variation in the normal behaviour of a protocol.

Owens at [18] introduces the topic of covert channels in terms of different forms of digital steganography. In addition to traditional injection and substitution steganography, he coins the term "propagation steganography" which he describes as a new file which can be generated using the intended payload

to be concealed and some sort of encoding engine. Typically this would be a graphic, music file or verbose text which can then be reprocessed to reveal the original payload. Furthermore, according to Gardiner [19], "although in the wild it is currently extremely rare to find examples, there is a high probability that techniques involving steganography will become more widespread. This will allow the malware to use legitimate services to transmit information, i.e. by hiding in plain sight." Ultimately, in the context presented, steganography roughly translates to information encoding and it is interesting to consider implementing SMB communication channels in relation to the three forms of steganography discussed (injection, substitution and propagation). Elements of steganography are incorporated into the SMB communication channel designs presented later in Chapter 4.

A methodology for identifying covert channels from specifications, requirements and even raw code is proposed by Kemmerer at [20]. He proposes the Shared Resource Matrix (SRM) methodology which is intended to be used as a tool to aid in the identification of possible covert channels. The proposed SRM methodology involves identifying what resources are shared and which operation primitives are possible on the given system. Each possible operation is then considered in terms of whether or not it can modify or read a resource attribute. In the example matrix shown at Table 2.1, the system objects are comprised of processes and files. These objects are further subdivided into attributes which are listed in the second column. Processes have the ability to perform several operations relating to the identified attributes and these functions are listed in the first row of the table. Identifying which operations have the ability to read or modify attribute information provides the foundation for identifying whether a storage channel could exist between processes.

Kemmerer establishes criteria that must be met to allow for the existence of storage channels [20]:

1. The sending and receiving processes must have access to the same attribute of a shared resource.
2. There must be some means by which the sending process can force the shared attribute to change.
3. There must be some means by which the receiving process can detect the attribute change.
4. There must be some mechanism for initiating the communication between the sending and receiving processes and for sequencing the events correctly. This mechanism could be another channel with a smaller bandwidth.

RESOURCE ATTRIBUTE \ PRIMITIVE		WRITE FILE	READ FILE	LOCK FILE	UNLOCK FILE	OPEN FILE	CLOSE FILE	FILE LOCKED	FILE OPENED
PROCESS	ID								
	ACCESS RIGHTS			R		R		R	R
	BUFFER	R	M						
FILES	ID								
	SECURITY CLASSES			R		R		R	R
	LOCKED BY	R		M	R				
	LOCKED	R		R,M	R,M	R		R	
	IN-USE SET		R	R		R,M	R,M		R
	VALUE	M	R						
CURRENT PROCESS		R	R	R	R	R	R		

R = Read M = Modify

Table 2.1: Example shared resource matrix, reproduced from [20]

Applying the above criteria to the example matrix at Table 2.1 reveals that a covert channel may be possible using the `Locked` and `In-Use Set` attributes.

In [13], Donaldson et al. discuss the applicability of the wiretap threat in trusted systems where physical security can provide adequate protection. They argue that an overlooked covert channel threat comes from the application layer and emphasize that "covert channel analyses must provide for identifying channels between individual host applications running on top of the distributed network trusted computing base." While not perfectly analogous to the enterprise network scenario, there is a common thread that the application layer must not be overlooked when evaluating the potential for covert channels.

2.3 Network Traffic Analysis

The work at [21] presents a very interesting (though dated) view of enterprise network traffic. This characterization includes analysis of over 100 hours of activity on the Lawrence Berkeley National Laboratory academic network captured in 2005. SMB traffic captured was sorted into the following categories:

1. SMB basic commands;
2. RPC pipes;
3. file sharing;
4. LANMAN; and
5. other

It is interesting to note that in the study [21], named pipe commands represented the largest number of packets and total bytes sent across the network. The vast majority of the traffic was specifically attributed to printing. This can somewhat be explained by the fact that at the time of the study, Windows file sharing was less prevalent and the Network File System (NFS) and Network Control Program (NCP) protocols were in heavy use for network file access at Berkeley.

Due to their prevalence, named pipe commands were further sorted into the following categories:

1. NetLogon;
2. LsaRPC;
3. spoolss/WritePrinter;
4. spoolss/other; and
5. other.

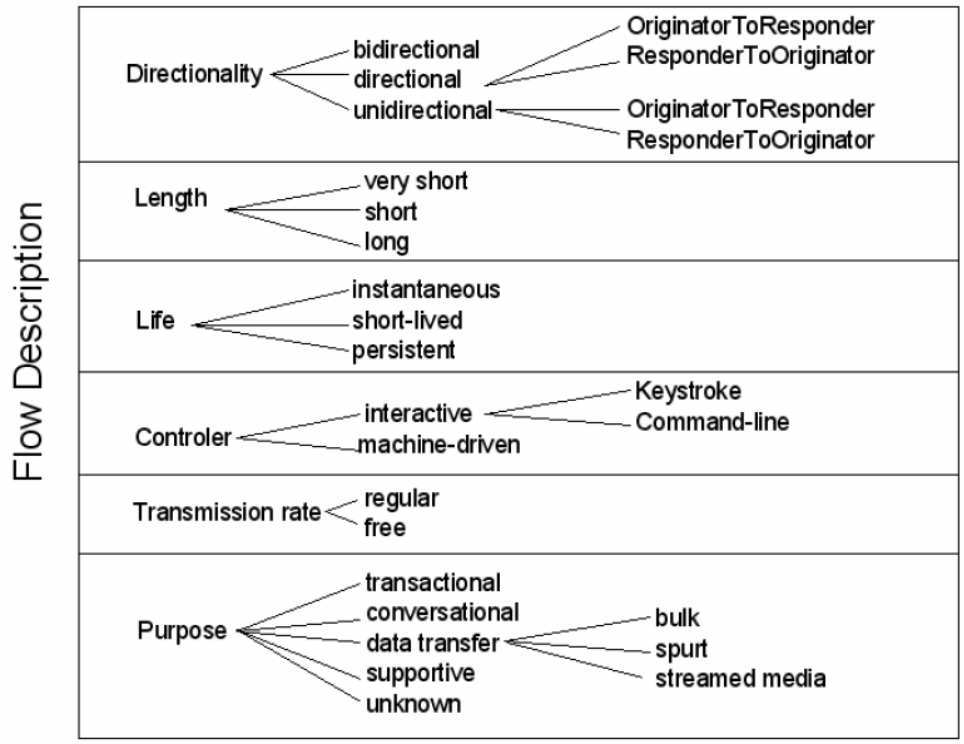

Figure 2.1: Flow description, reproduced from [23]

A more recent characterization of enterprise network data is found at [22], however the level of detail is not nearly as in depth as what was found at [21]. Of note, the NCP protocol was in extensive use and SMB traffic only made up 5% of all traffic on average.

With very little existing work characterizing modern SMB traffic, an investigation into generic heuristic traffic analysis and characterization techniques was undertaken.

The focus of [23] is to identify "relevant and discriminative flow attributes for use in traffic characterization." The work identifies 40 flow features that can be used to differentiate traffic. To characterize traffic, the methodology involves grouping all packets into respective flows and then measuring the 40 discriminative flow features to determine a flow description. Elements of de Montigny-Leboeuf's flow descriptions are presented in Figure 2.1.

Several common tactics used by network analysts to detect anomalies in traffic are outlined at [24]. These are illustrated in Figure 2.2 and summarized

as:

1. outlier detection;
2. pattern matching;
3. comparative analysis;
4. feature summarization;
5. entropy analysis; and
6. cross correlation.

In addition to these tactics, the author details ninety-one distinct traffic features that can be extracted for review in the course of anomaly detection.

2.4 Detecting Malicious SMB Traffic

The thesis at [3] seeks to answer the question: "How can we detect lateral movement attacks over SMB using an anomaly based approach?" Ultimately, the author chooses to implement five rules in order to identify malicious SMB traffic. Specifically rules will trigger if traffic:

1. contains random hostnames;
2. contains empty session keys during authentication;
3. has a high prevalence of WRITE command towards IPC shares;
4. has a high prevalence of error messages; or
5. has a high prevalence of services created or started.

These rules serve to detect traditional network exploitation techniques over SMB such as brute force attacks, calls to `psexec` or direct writes to IPC shares. The ability to detect these types of attacks over the network reinforces the idea that attackers will necessarily seek to build new covert exploits that are not so easily detectable by network defenders who are hunting for them.

Interestingly, another paper was published seemingly independently around the same time which attempts to address the same problem. In [4], Cyrus demonstrates how to implement rules using the Bro network monitoring framework in order to detect typical SMB lateral movement attacks. These rules include screening for traffic that:

1. contains random hostnames;
2. attempts to write to hidden shares (`IPC$`, `ADMIN$`, `C$`); and
3. contains malicious files.

As suggested by Zander [5], the types of exploits detected here have persisted up until now due to obscurity. Since these channels do not resemble normal network use, they are much easier to detect. If nothing else, additional work at [4] serves to inform that traditional network exploitation techniques over SMB are becoming well known and mechanisms to detect them are becoming more widely publicized. This further reinforces the need to examine what new vectors are likely to be implemented by those seeking to subvert detection.

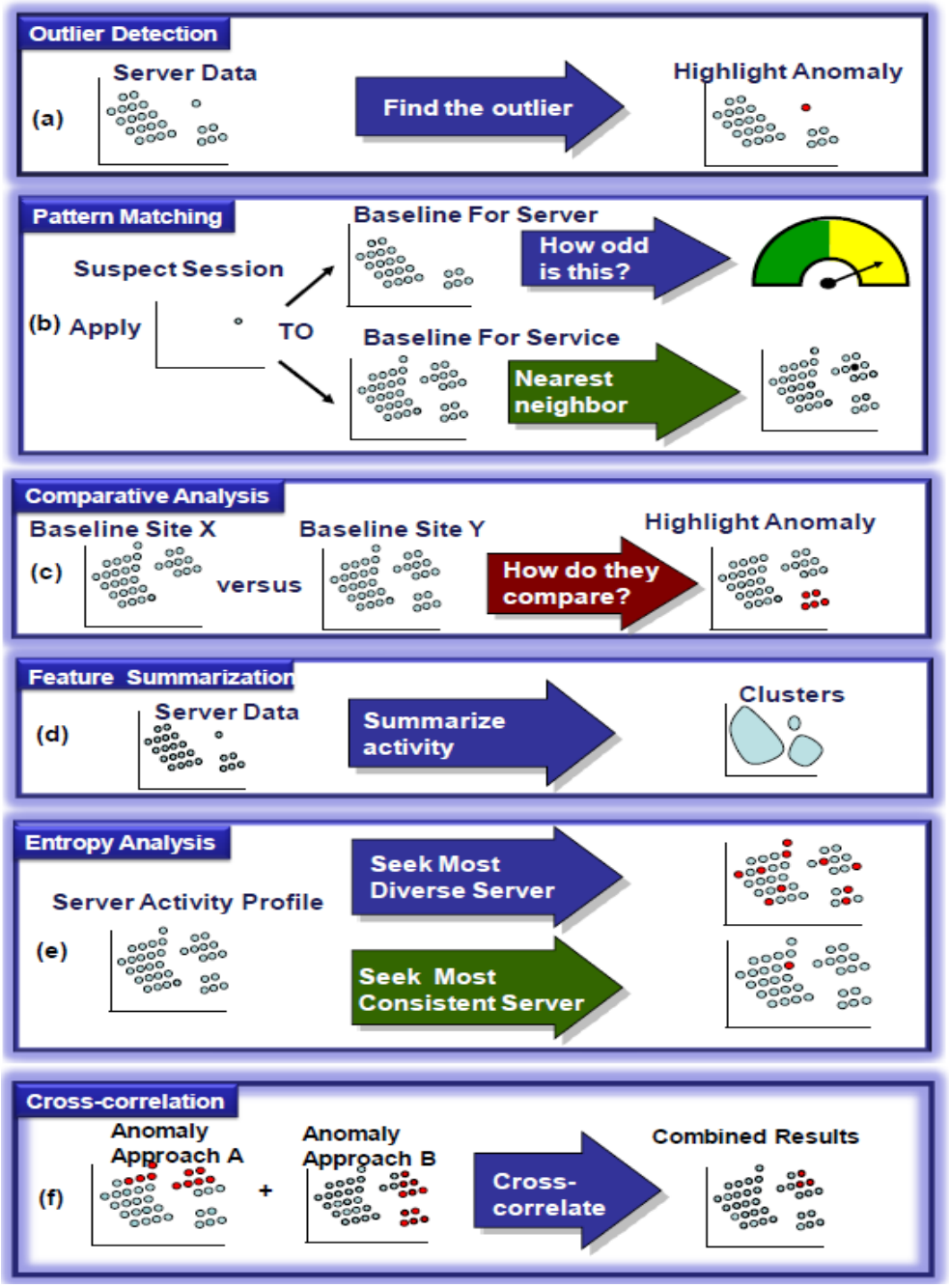

Figure 2.2: Common network anomaly detection tactics, reproduced from [24]

2.5 Summary

This chapter provides background information on the SMB protocol and details regarding its suitability for use as a C2 channel. Background information was also presented on the topic of covert channels, their development and their detection. In addition, previous work relating to the characterization of SMB network traffic was presented, as well as techniques for the detection of anomalies within network traffic. Finally, relevant research in the field of detecting malicious use of existing SMB C2 channels was discussed; however this research suggests that existing techniques are relatively straightforward to detect.

3 Characterizing Live Network SMB Traffic

3.1 Introduction

The previous chapter provides sufficient background material to understand the purpose of the SMB protocol, how covert channels can be implemented, how network traffic can be analyzed for anomalies and the state of existing research relating to detecting malicious use of the SMB protocol on networks. This chapter provides detailed analysis of typical SMB traffic on a research network, which serves to inform the design of SMB C2 communications channels that are intended to be difficult to detect.

3.2 Characterization of SMB Network Traffic

In search of unknown malicious traffic, network defenders can characterize network flow attributes in an attempt to identify outliers and exceptional behaviour. A flow attribute analysis of SMB traffic serves to define a baseline of normal flow characteristics which should be considered when implementing a stealthy communications channel. This includes characterizing features such as the frequency and time of use, the nodes that typically communicate, flow packet counts, direction of the flows, as well as the packet sizes.

While staying below the radar in terms of detectability by network flow analysis may provide some level of stealth, a network defender can also characterize the data found in the SMB headers and messages in an attempt to identify outliers and exceptional behaviour and thus develop a baseline of normal behaviour for future comparative analysis. As a result, a characterization of the data found in SMB headers and messages in normal traffic must also be considered when implementing a SMB C2 channel in order to minimize the likelihood of detection.

At all levels of characterization, an important component of this effort is to determine which features have a high degree of variability and which ones remain relatively constant. The intent is to ensure that features that are determined to exhibit little variance, or variability within specific ranges will be of foremost consideration when designing a covert communications system.

While some of the features are found to have a wide range of normal values, others vary minimally. The product of the characterization is metrics and data that can be used to establish normal operating parameters for SMB traffic on the network, recognizing that in some cases, what is considered normal is in fact a high degree of variability.

3.3 Data Collection, Research Network Description and Data Analysis

For the purpose of this research, initial SMB traffic was captured for nearly eight days on a live network at an academic institution (herein referred to as the "research network"). Users included faculty, administrative staff, IT support staff, researchers and students. The rationale for using this network rather than a simulation in a lab environment is that it presents live, typical network traffic that is neither too large to analyze nor too small to be trivial. The research network included multiple domain controllers, file servers and print servers.

The capture activity spanned more than a one-week period enabling traffic to be characterized during different periods of utilization such as weekdays, weekends, holidays and overnight. Multiple days of capture also enabled a better characterization of the normal variance found within the SMB traffic.

Network traffic flow data was generated from packet captures using Argus [25] and SMB specific data was extracted using Bro Network Security Monitor [26] with SMB scripts that were customized by the author.

3.4 Flow Analysis

3.4.1 Normal Node Communication Patterns

In order to be difficult to detect by network traffic analysis, it is important to respect the normal node communication patterns of the network. Specifically, the direction of communication and endpoints must be consistent. Analysis of the Argus flow data demonstrated that when limiting the scope to SMB traffic on this network, client hosts typically communicated with known servers and

did not communicate with other client hosts. In addition, servers did not initiate connections to client hosts.

From a network analysts perspective, the direction of connections and the SMB client/server node pairs are straightforward to extract from Argus flow data. As such, it is very important to respect these typical node communication patterns if a C2 channel is to remain difficult to detect.

As an example, Figure 3.1 was generated directly from the entire week of Argus flow data using free open source software (FOSS) command line tools Racluster [25], Afterglow [27] and Graphiz [28]. The node graph shows only clients from the various client workstation networks initiating communications with known SMB servers specified in the Afterglow configuration file.[1] In this case an anomaly was detected (depicted in red) because one of hosts serving SMB was not previously identified as a recognized server. While this did not necessarily reflect malicious client behaviour, it was anomalous and resulted in further inspection and investigation by the network administrators.

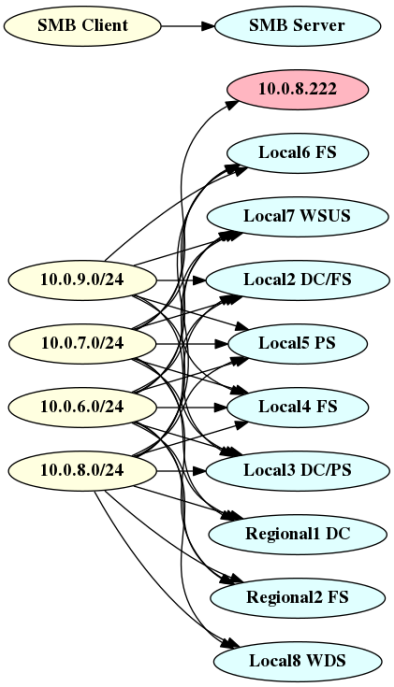

Figure 3.1: SMB client/server node graph

[1] IP addresses and server names have been anonymized

It should also be noted that generally speaking, dependant on network policy, there is nothing preventing host clients from acting as SMB servers. In practice this is not a behaviour that is typical or natural on this network and it could be argued that from a security standpoint, client workstations should not be permitted to interact directly with one another without justification.

3.4.2 Flow Attribute Analysis of SMB Traffic

Understanding the baseline level of activity of each server provides some of the insight necessary to determine whether a communication channel would be difficult to detect. For the purpose of this research, the concept of a channel being "difficult to detect" is strongly tied to whether or not the communication channel traffic is likely to be inspected by an analyst. The author assumes that one of the most common discriminators to trigger inspection of traffic is increases in activity beyond an acceptable threshold, creating what is often referred to as outliers. In order to better quantify the magnitude of normal activity, flow, packet and byte count attributes were characterized.

The plot at Figure 3.2 represents the total number of packets and total number of bytes of SMB traffic observed on the research network during data

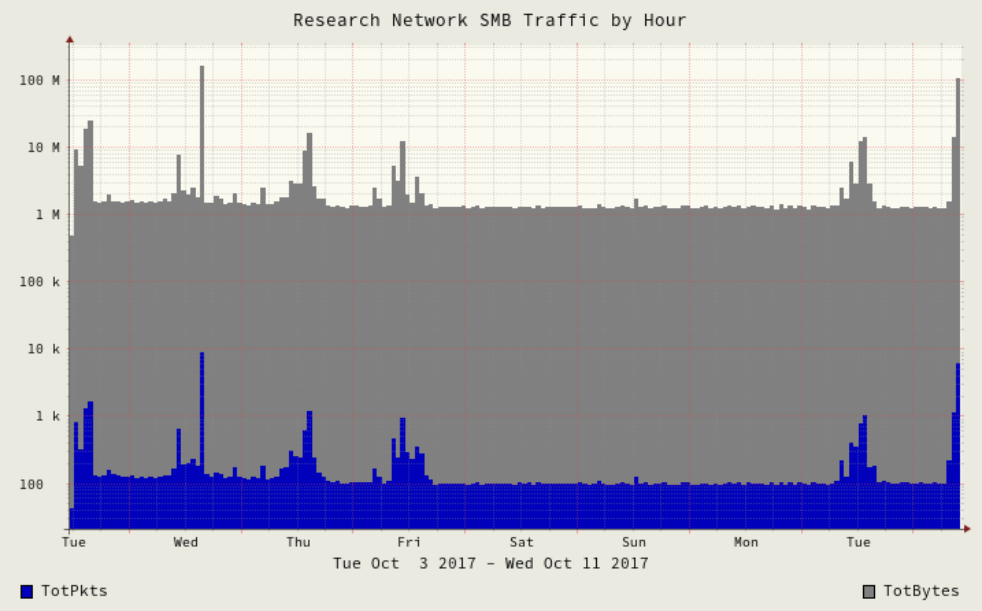

Figure 3.2: All SMB traffic by hour

collection. It establishes that there are clear and distinctive observable patterns of SMB traffic associated with working hours and non-working hours.[2] This graph shows that SMB activity spikes during working hours but that the peaks are not necessarily uniformly distributed across the workday. In some instances SMB traffic peaked in the afternoons, in others it peaked in the morning. Hour to hour, traffic activity could dip or increase with no predictable regularity other than generally increased traffic during working hours. During non-working hours, traffic was relatively predictable with few notable spikes.

SMB Clients (End User Workstations)

Given the distinctive difference in usage patterns observed for workdays and non-work days, basic client attribute flow data from the research network have been tabulated separately and presented in Tables 3.1 and 3.2 respectively. Source packets and bytes refers to data sent from the clients to the servers whereas destination packets and bytes refers to data sent from the servers to the clients.

Table 3.1: Workday client flow attribute data summary

Workday (24 hour period)				
Attribute	Mean	Median	Min	Max
Flow Count	148	154	1	2933
Source Packets	54,643	16,817	4	7,227,120
Destination Packets	103,864	47,185	0	18,314,445
Source Bytes	56,920,867	3,174,581	816	27,319,256,510
Destination Bytes	230,269,925	119,414, 199	0	44,317,249,916

Table 3.2: Non-workday client flow attribute data summary

Non-Workday (24 hour period)				
Attribute	Mean	Median	Min	Max
Flow Count	131	148	1	509
Source Packets	15,649	16,041	4	225,574
Destination Packets	33,120	45,764	2	203,613
Source Bytes	3,154,197	3,047,613	816	44,394,715
Destination Bytes	74,794,391	116,643,482	1116	248,974,094

[2]It should be noted that for this traffic set, the Monday was a holiday

This basic flow attribute summary provides some insight into the network and the typical activity of each client. It can be seen that on this network the range of possible values defined by the Max and Min are much larger during work days, showing increased variability. It can also be seen in the summary tables that during the workdays the means are generally more skewed from the medians, suggesting more outliers which also indicates increased volatility in the data.

Given the presence of outliers within the data set, the median is a much better measure of central tendency of the data but the extent to which an attacker may consider deviating from the median should be influenced by the typical range, the variability and the volatility (the magnitude of differences of typical values) of the supporting data.

SMB Servers

To avoid being detected, an attacker might wish to consider specifically targeting servers with higher volumes of normal traffic.

Initially SMB server flow attributes were characterized using a similar process that was put in place for the clients on the network and they were characterized as a group. This made sense with the SMB clients because the majority of the clients exhibited very similar behaviour over the observation period. It later became apparent that each SMB server has a significantly different level of utilization and a general flow attribute characterization of all SMB servers would not add significant value.

The plot at Figure 3.3 shows the notable difference in the distribution of the activity across the various servers on the network. The figure shows that Local2 Domain Controller and File Server (DC/FS) as well as Local3 Domain Controller and Print Server (DC/PS) dominate the traffic in terms of flow counts and this was also subsequently verified to hold true for packet and byte counts.

Figure 3.4 illustrates that the baseline of traffic observed both during working and non-working hours in Figure 3.2 is predominantly a result of Local2 and Local3. Local4 and Local5 have orders of magnitude less traffic however follow a similar pattern of unpredictable variable peak activity during work hours and low volatility and variability outside of working hours.

3.5 SMB Command Analysis

As discussed in Chapter 2 the SMB protocol supports nineteen different commands for session control, file access, and general messaging functions.

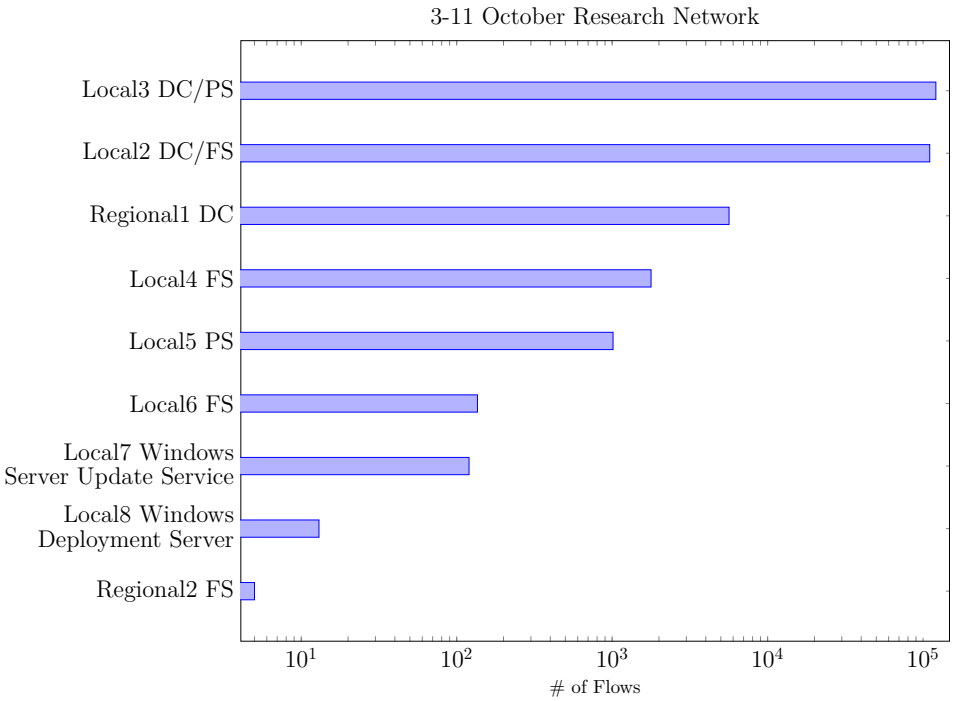

Figure 3.3: Flow counts by SMB server

In [6], Smith identifies that in order to avoid being trivial to detect, there must be a non-zero natural occurrence of any given symbol used in a covert communication channel. This means that in order to avoid trivial detection, any communication channel implemented over SMB must not use SMB commands that have a zero natural occurrence. To this end an attacker would need to identify the naturally occurring commands between clients and servers.

Furthermore, beyond simply identifying which commands are present, an attacker can characterize the extent to which commands are used, identify any proportional command relationships that are observed and further characterize relevant SMB protocol attributes that are present to ensure a communication channel is difficult to detect

3.5.1 Typical SMB Commands in Use and Share Types

A summary of the SMB commands extracted from the SMB headers for select servers is found tabulated in Table 3.3. In the case of Local2 and Local3, the two domain controllers (DCs) were dual purposed as file and print servers

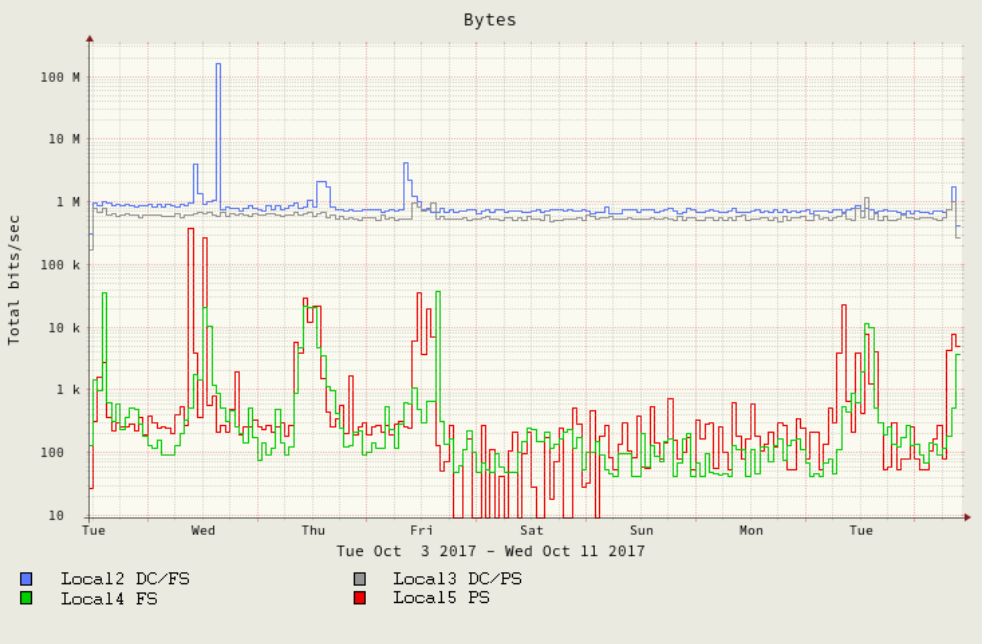

Figure 3.4: Select SMB server traffic by hour

respectively. In order to provide more homogeneous data sets for characterization (as might be done by a network defender), DC-specific traffic was segregated from the file and print server traffic for further analysis. This had the added benefit of being conducive to supporting comparative analysis with the other file and print servers on the network.

Separating the traffic consisted of analyzing the transactions within the data set and looking for patterns in the DC-specific traffic. In general, the DC SMB transactions consisted of clients downloading group policy object (GPO) updates at regular intervals. This traffic is relatively consistent and as such, a script was written to identify GPO update downloads within the logs, isolate the associated TCP streams and then export those streams to new logs for separate analysis.

Based on the data extracted, it is possible to draw some general observations about the characteristics of SMB commands used with different types of servers that an attacker designing a communication channel over SMB should consider.

One particularly noteworthy observation from this data set is that not a single SMB `Echo` command was observed exchanged with any of the servers

Table 3.3: Instances of SMB commands by server (over data collection period)

	Print Servers		**File Servers**		**Domain Controllers**	
	Local5	**Local3**	**Local4**	**Local2**	**Local2**	**Local3**
Negotiate	1112	21498	1781	8899	103809	105086
Session_Setup	828	16586	1750	7399	103853	105141
Logoff	631	12338	450	5699	103827	105100
TreeConnect	874	16121	2648	7122	120969	123200
TreeDisconnect	599	12409	1926	5924	120896	123117
Create	4376	81194	36327	290504	310718	345970
::File_Open	1984	3259	35884	287537	287383	328429
::Pipe_Open	2090	77681	361	2631	3003	371
::Print_Open	2	-	-	-	-	-
Close	3122	81445	31242	278768	340313	346735
::File	819	3254	30786	278303	339455	345703
::Pipe	2084	77698	362	14	691	957
::Print	2	-	-	-	-	-
Flush	-	-	-	916	-	-
Read	9191	160049	379	631990	712547	647007
::File	5507	1448	15	630392	711752	645395
::Pipe	3218	158251	364	14	646	1472
Write	3699	158901	365	282648	612	1446
::File	-	11	1	58207	-	-
::Pipe	3370	158533	363	14	612	1446
::Print	14	-	-	-	-	-
Lock	-	-	3	95766	75227	83977
FS/IOCTL	13126	91211	5911	37746	105959	115181
Cancel	-	-	41	254	-	-
Echo	-	-	-	-	-	-
Query_Directory	2	122	9875	2239	19878	35255
Change_Notify	-	-	14	1167	-	-
Query_Info	2151	56155	1093	129644	286646	313508
Set_Info	-	-	4	1290	-	-
::File_Rename	-	-	2	2	-	-
Oplock_Break	-	3	-	213	620	549
Flow Count	911	12645	2020	6936	102556	107556

on the network throughout the observation period. The SMB `Echo` request is sent by a client to determine whether a server is processing requests, but given that this command is not observed naturally on the network, it would be a poor choice for establishing a communication channel that is intended to be difficult to detect.

Every server used the SMB `Create`, `Close`, `Read`, `Write`, `FS/IOCTL`, `Query`

`Directory` and `Query Information` commands, however it should be noted that each server type only used a subset of the remaining eleven SMB commands.

Domain Controllers

The DCs used very few additional commands; in addition to those previously named, SMB `Lock` and `Oplock Break` commands were found. This is consistent with the expected behaviour since network policies were constantly being read from the DCs by the various network clients. It should also be noted that as shown in Table 3.3, only named pipe type shares were written to the DCs, never files.

In order to be difficult to detect, any communication channel targeting a DC should respect the subset of SMB commands used and the type of SMB shares written to.

Print Servers

The only additional SMB command that appeared on either of the print servers was `Oplock Break`. This command only appeared on the Local3 server and as there is only 3 of these commands observed over the data collection period, it could be said that this behaviour is anomalous, resulting in a higher likelihood of inspection. While naturally occurring, the frequency of appearance is low to the extent that an attacker building a covert communication channel should avoid using this command.

Also worth noting is how few SMB `Write` commands were executed on the print share. After much investigation by the author, it was found that printing on the network typically does not utilize SMB print shares. Instead, print jobs are coordinated with the print servers over SMB through the `spoolss` named pipe. Printer and print job data is then managed on new direct TCP connections that do not use the SMB protocol.

The very few direct writes to print shares observed on this network were a result of system administrators executing older scripts that called `print.exe` from the command line, which essentially bypasses the normal network printing and queuing processes. Since this activity is such a rare occurrence, it is effectively anomalous on this network. As a result, any covert communication system design should avoid establishing connections to shares of the type `Print` on print servers (or any other servers for that matter).

File Servers

The file servers were the only servers where SMB `Change Notify` and `Cancel` commands could be found. `Set Info` commands were also present, but observed in relatively small numbers on Local4, which suggests a different usage pattern than that of the Local2 file server.

`Set Info` commands are used to change file attributes such as owner and security settings. This can expected to be a normal command to be present on a file server, however if the typical usage is extremely infrequent, this command should be avoided to ensure a communication channel that is difficult to detect.

Local2 also had naturally occurring SMB `Lock`, `Oplock Break` and `Flush` commands. This was explained by the fact that Local2 hosts a large database that is accessed by multiple users for reading and writing.

Since file servers can be used in different ways depending on their intended purpose, an understanding of the typical uses can be important when designing a communications channel. For example, considerations such as whether or not files have the potential to be accessed simultaneously will provide insight as to whether or not SMB `Lock`, `Oplock Break` and `Flush` commands should be permitted for use in a communication channel that is intended to be difficult to detect.

3.5.2 Frequency of SMB Command Occurrence and Relative Proportional Relationships

From Table 3.3, all servers on the network had a natural presence of the five basic session control commands, namely: SMB `Negotiate`, `Session Setup`, `Logoff`, `Tree Connect` and `Tree Disconnect`. In terms of relative frequency, on any given server these commands appeared to be in the same order of magnitude. That is to say, it would not be unusual to find twice the number of `Negotiate` commands as you would find `Logoff` commands for example, but it would be anomalous to find ten times more `Negotiate` commands than `Logoff`.

Other proportional relationships between the session control commands noted were:

1. The number of `Session Setup` commands was less than or approximately the same as the number of `Negotiate` commands.
2. The number of `Tree Connect` commands was greater than or equal to the number of `Tree Disconnect` commands.
3. The number of `Logoff` commands was less than (or equal to) the number of `Session Setup` commands.

4. The number of `Logoff` commands was less than (or equal to) the number of `Tree Connect` commands

Domain Controllers

Some types of servers behave more similarly than others. Each domain controller for instance had very similar relative proportions of commands, as depicted in Figure 3.5. The session control commands (SMB `Negotiate`, `Logoff`, `Session Setup`, `Tree Disconnect`, `Tree Connect`) on each DC have a very tight proportional relationship of 1:1 which is also maintained with the `FS/IO Control` commands. It can also be seen that SMB `Create` and `Close` commands maintain a near 1:1 relationship (+/- 10%) with one another for both DCs.

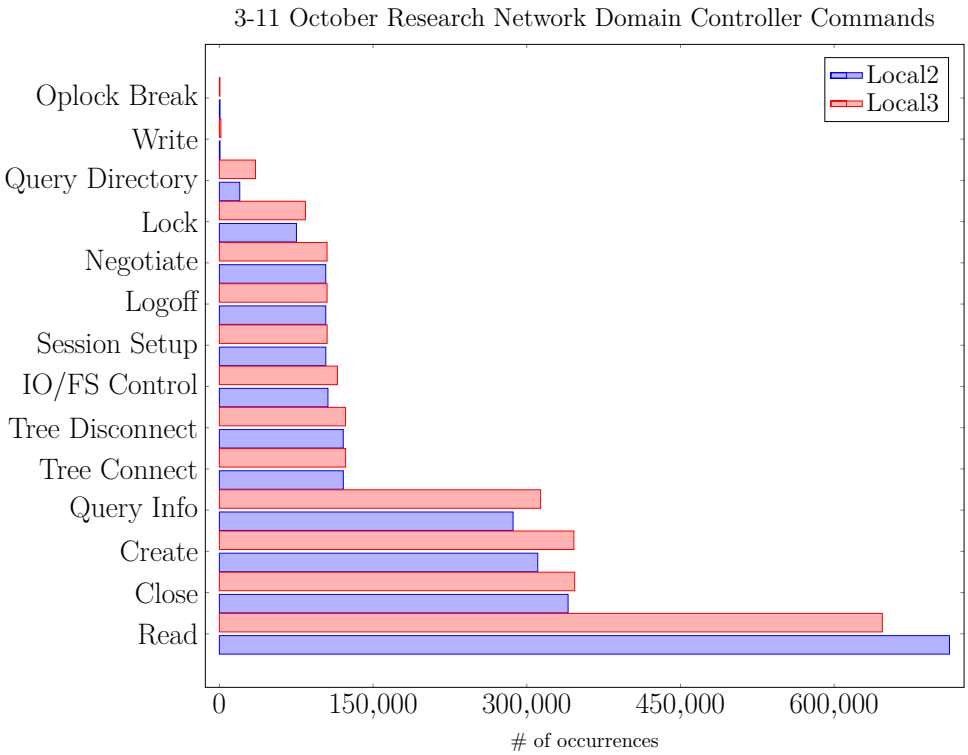

Figure 3.5: Domain controller command counts

Print Servers

Print server commands are illustrated in Figure 3.6. For a given server, the volume of session control commands is similarly proportional in magnitude, however the relationship does not appear as rigid as that seen with the DCs. The exception is the near one-to-one relationship observed between the `Logoff` and `Tree Disconnect` commands. `Create` and `Close` commands also maintained a near one-to-one relationship for a given server. In addition, in contrast to the DCs, the `FS/IO Control` commands do not appear to maintain a one-to-one relationship with the session control commands for a given server and instead appear to generally have a value greater than the number of `Create`/`Close` commands, but less than the value of `Read`/`Write` commands. `Query Info` commands are found similar in magnitude, but less than the number of `Create`/`Close` commands observed. Finally, `Read`/`Write` commands are also found to have a strong near one-to-one relationship for a given print server.

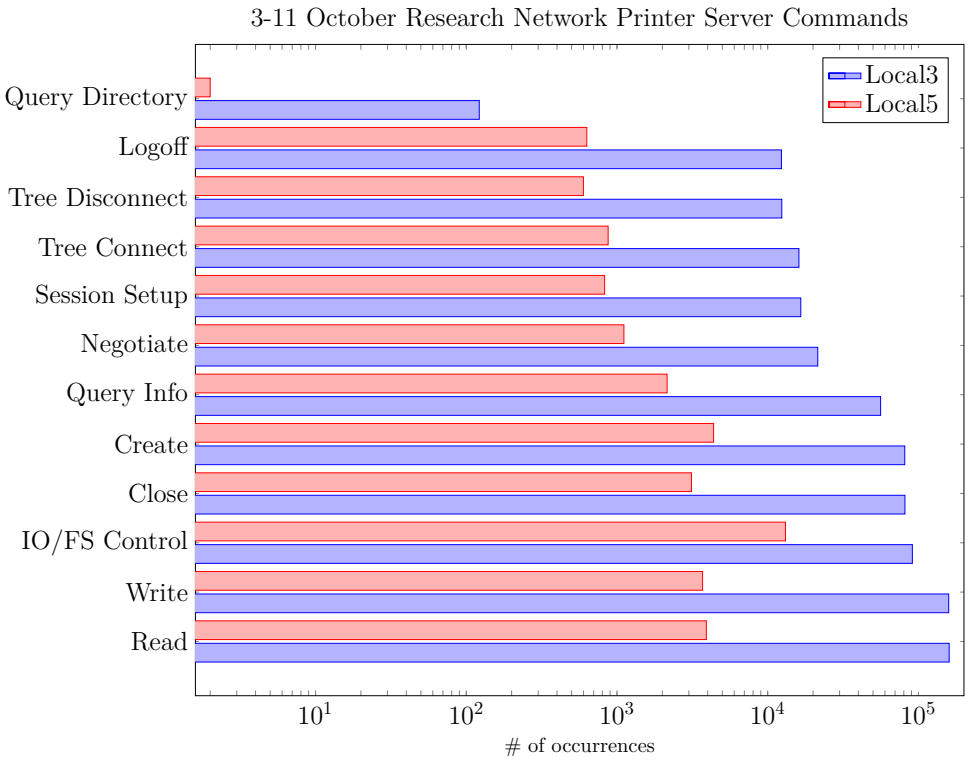

Figure 3.6: Print server command counts

It is interesting to note that when the data was initially compiled, the `Read/Write` commands on Local5 were not as tightly correlated (as seen in Table 3.3). As this was believed to be an anomaly, further investigation was performed which revealed that Local5 also hosts some infrequently accessed packages for download. When these associated file server type activities were eliminated from the characterization, the `Read/Write` proportional relationship was restored. This reinforces the idea that maintaining proportional relationships is important to help prevent anomaly based detection for any covert communication channel. Further, it suggests that from a security perspective, network defenders may find it easier to defend a network with more homogeneous traffic. Combining file server, domain controller and print server functionality together on one server makes it much more difficult to characterize typical network behaviour, potentially providing a would-be attacker with more flexibility when implementing a communications channel that is difficult to detect. For this research the author assumes that a network defender can and will separate traffic for detailed characterization but it should be noted that the author does not believe this level of effort to be common practice.

File Servers

As for the file servers depicted in Figure 3.7, the session control packets show similar proportions as those observed for the print servers (similar in magnitude but not showing a tight one-to-one relationship). None of the remaining commands present showed consistent proportional relationships between the file servers except `Create/Close` which remained within the same order of magnitude (+/- 25%).

Of the three server types characterized, the file servers are by far the most variable and inconsistent in terms of establishing fixed proportional relationships between various commands. The more difficult the traffic is to characterize on a server, the better potential candidate it will be for hosting a C2 communications channel that is difficult to detect [5].

3.5.3 Time of Use

Figures 3.2 and 3.4 show that periods of high utilization generally occurred during working hours, however this analysis provides no insight into whether there was any relationship between working hours/non-working hours and the presence of specific SMB commands. In order to build a C2 communications channel using SMB that is difficult to detect, an attacker would need to ensure that that there is a non-zero occurrence [6] of any of the SMB commands

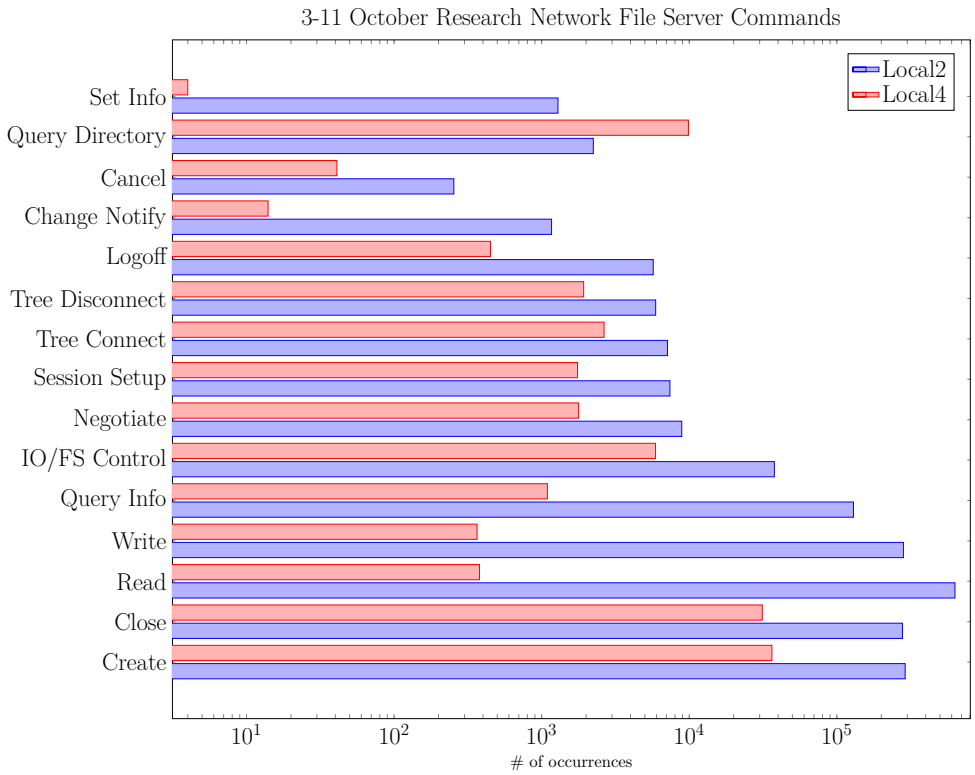

Figure 3.7: File server command counts

used in the communicating channel during the normal periods in which C2 communications are taking place. If there are periods when specific SMB commands are not found naturally (or where frew are found), then those periods should be avoided in order to minimize the likelihood of detection.

Domain Controllers

In addition to authenticating and authorizing users and computers on the network, DCs also assign and enforce security policy. Security policy updates are transmitted via the SMB protocol and were found to be sent continuously to client workstations, regardless of working hours/non working hours. This resulted in a constant use of most of the SMB commands in the DC profile highlighted in Table 3.3. The exception was SMB `Write` commands which were typically only found in quantity during working hours as illustrated in Figure 3.8. In the case of the research network data, there were some SMB

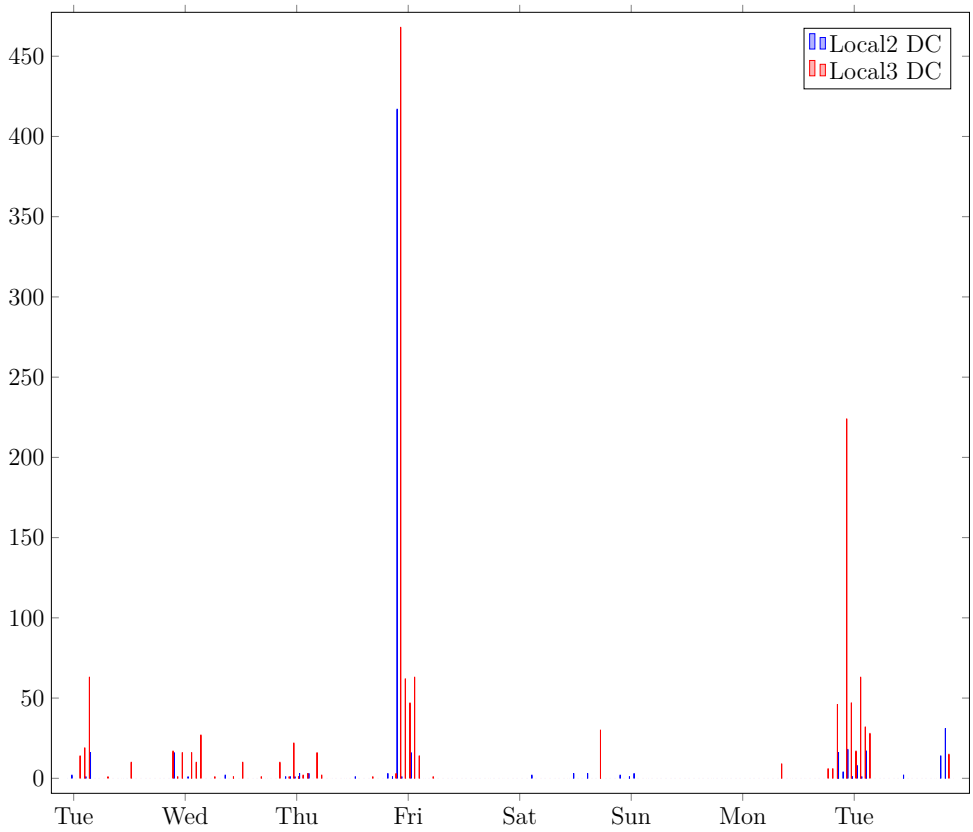

Figure 3.8: Domain controller SMB Write commands by hour

`Write` commands observed outside of working hours, however this was the exception rather than the norm and only observed in small quantities. As a result, when designing a communication system involving the DCs, use of the SMB `Write` command should be avoided outside of working hours.

Conversely, when examined in a similar manner, the other SMB commands found naturally on the DCs occur regularly throughout working and non-working hours. Figure 3.9 shows the distribution of SMB `Read` commands, which is somewhat representative of the typical distribution of the other file access and general messaging SMB commands observed. While peak usage tended to form toward the middle of workdays, there was sufficient variability and volatility to suggest that any of these commands could be well suited for a covert communications channel, regardless of time of use.

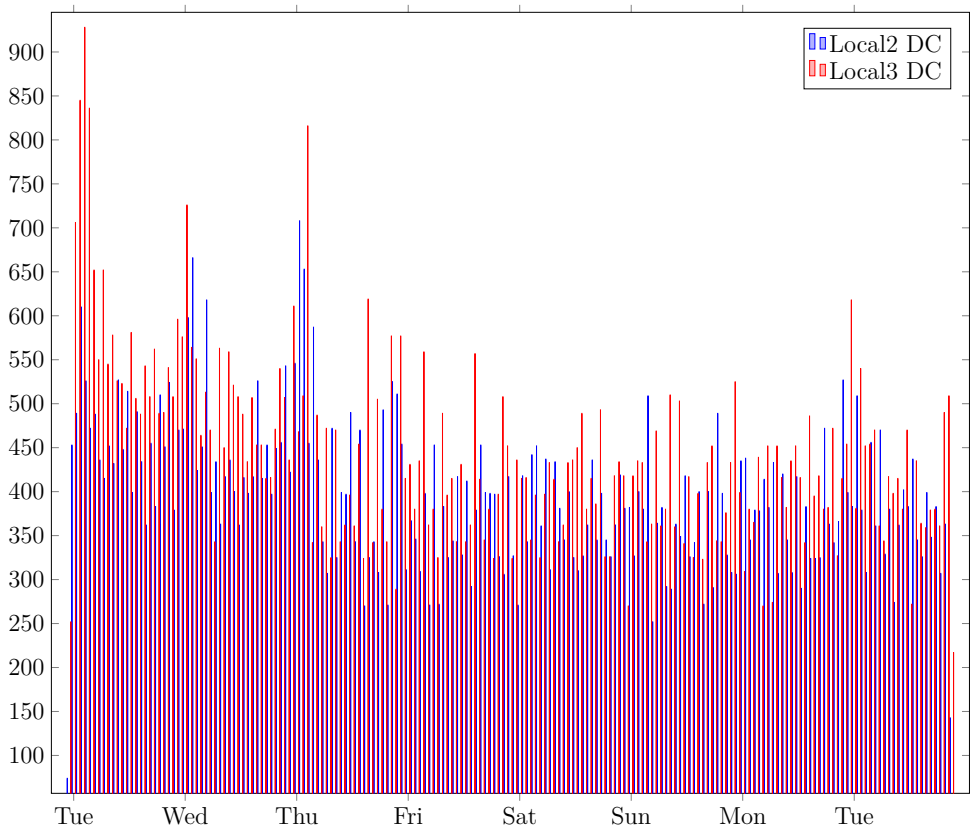

Figure 3.9: Domain controller SMB Read commands by hour

Printer Servers

For the printer servers, the SMB commands identified in Table 3.3 generally occurred throughout the week at all hours of the day, though there was a much higher density of commands during working hours. Figure 3.10 shows typical examples of this behaviour. In all cases, peak usage time changed day to day, with a high degree of volatility.

Generally speaking there was a consistent flow of the various SMB commands present and this behaviour was consistent across both print servers. With this knowledge, an attacker could consider designing a SMB covert communication channel capable of operating at any time of day with the understanding that during non-work hours, the usage may be limited in order to remain below thresholds that would increase the likelihood of detection.

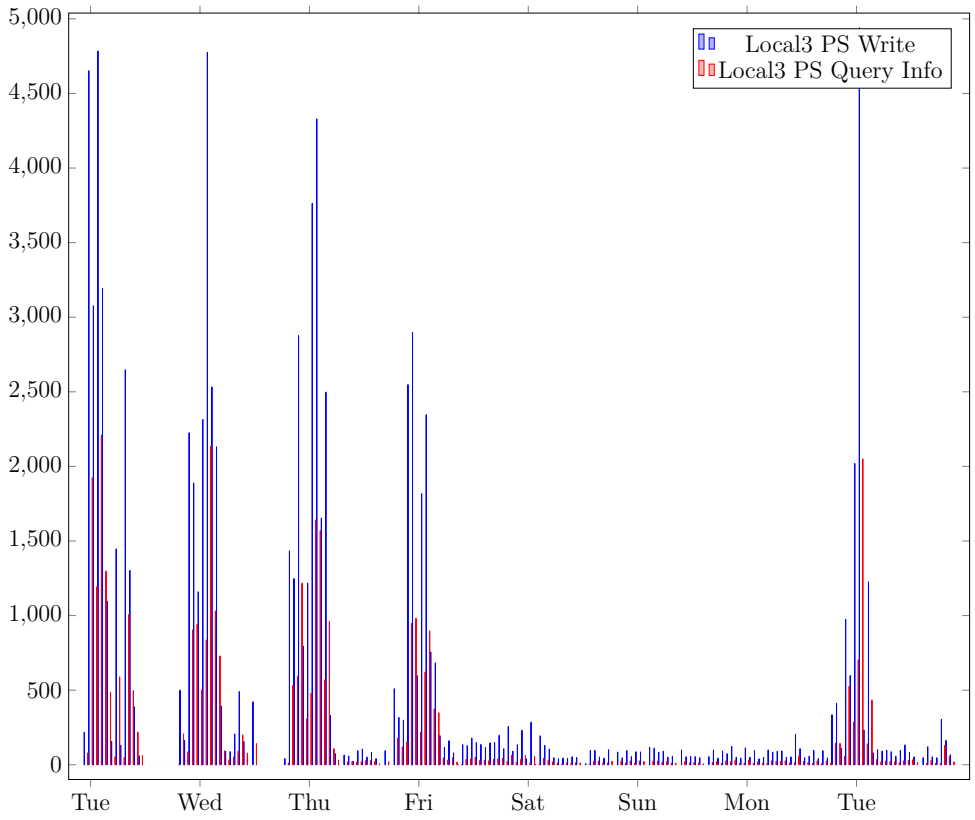

Figure 3.10: Local3 print server SMB Write and Query Info commands by hour

File Servers

The file servers exhibited the most profound differences between working and non-working hours on this network. Specifically, while the session control commands were relatively continuous with peaks during working hours, the majority of file access and general messaging commands occurred during working hours. At those times, the frequency of command usage showed high variability and volatility. This behaviour is illustrated in Figure 3.11 where SMB `Read`, `Write` and `Query Info` commands are plotted for Local4.

It is also interesting to note that there are differences in the working hours range between the two file servers. Specifically, the Local4 file server shows a wider range of busy periods with some random intermittent usage, suggestive of the working hours students and researchers are more likely to maintain.

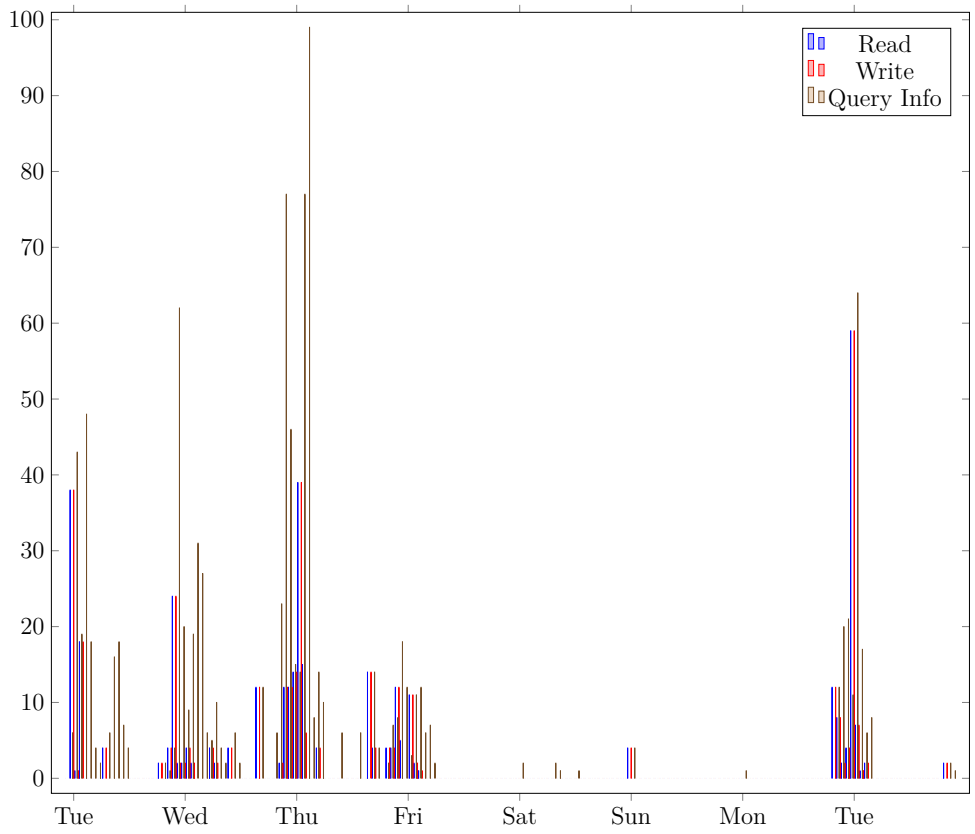

Figure 3.11: Local4 SMB Read, Write and Query Info commands by hour

Local2 file server has a more narrow band of peak utilization, indicative of a traditional employee workday with less variation. Figure 3.12 illustrates this phenomenon and depicts the SMB `Oplock Break`, `Set Info` and `Query Directory` command usage by hour for the Local2 file server.

The time of use characteristics of the commands in Figures 3.11 and 3.12 are representative of the other respective SMB file access and messaging commands present unless noted otherwise. One notable exception was `FS/IO Control` commands on both file servers which tended to be distributed throughout the collection period, similar to the session control packets, with peaks during working hours (as was the case generally with the other server types).

The other notable exception was observed on Local2, where SMB `Read` commands appeared continuously throughout the period with higher density

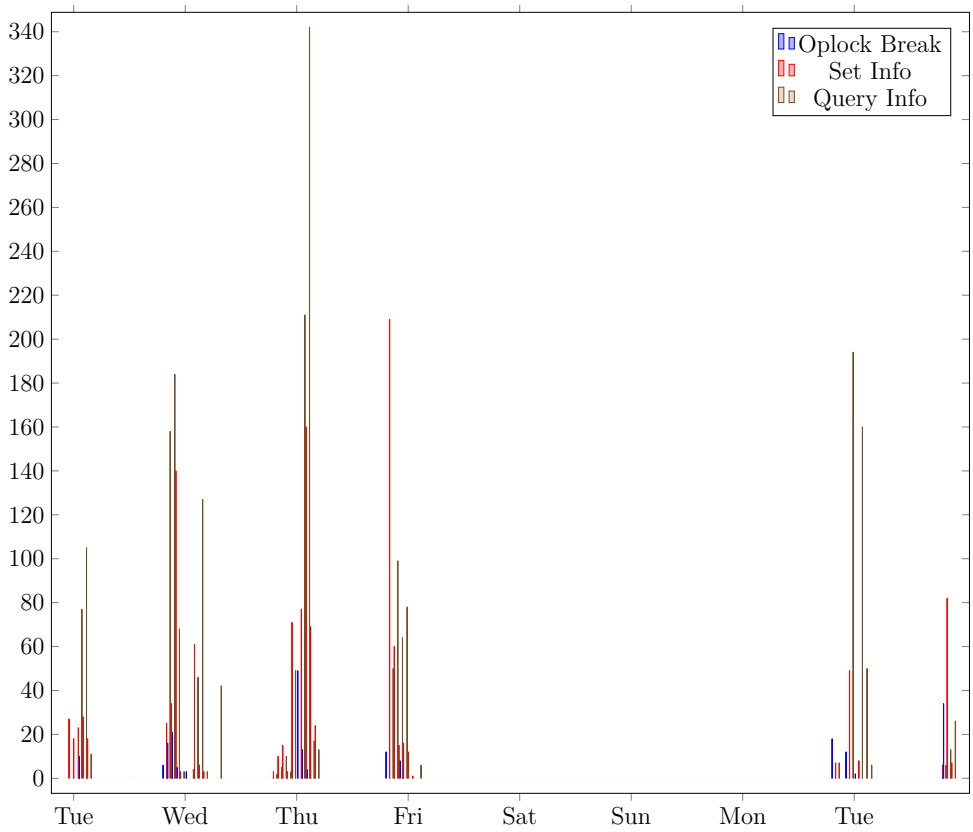

Figure 3.12: Local2 SMB Oplock Break, Set Info and Query Info commands by hour

and peaks during the work day. In this case of SMB `Read` commands, upon further investigation, the author found that some users had left GitHub applications running that were reading `gitconfig` files from the Local2 file server continuously at predefined intervals. At first blush an attacker might think this creates an opportunity for a covert communications channel to hide among otherwise legitimate traffic however the GitHub application traffic is rigid in timing and size attributes to the extent that it is very easy to remove from the traffic analysis once verified. With GitHub application traffic removed, the `Read` commands are only otherwise present during working hours.

In order to implement a communication channel that is difficult to detect on these file servers, an attacker would need to limit communication to the working hours observed for each or conservatively limit covert communications

to occur during core business hours to be safe. Given that Local4 exhibits more random behaviour than Local2, an attacker might also wish to consider establishing more permissive thresholds for communications limitations, particularly if considering time of use, as aberrant behaviour is less likely to be detected on Local4. Conversely, Local2 has a much greater volume of traffic and as such, a higher bandwidth channel might be possible while remaining difficult to detect.

3.5.4 Characterizing Additional SMB Packet Data

In addition to characterizing frequency of SMB command occurrence and relative SMB command proportional relationships, the data within the SMB packets can also be further characterized. The SMB headers are fixed length and the fields remain constant regardless of the SMB command being executed. Conversely, each SMB command's requests and responses have different corresponding SMB message formats with different types of information available to be characterized.

For example, it is within the SMB `Tree Connect` response message that the share type is identified as being either a file, a printer or a pipe. Additionally, SMB `Lock` request messages contain a byte offset and length field whereas SMB `Create`, `Read` and `Write` request commands all have a similar message format that contains information such as resource name, path and size (if applicable). The remainder of this section presents characterizations of some of the relevant fields in the network traffic SMB headers as well as messages not previously discussed.

SMB Header Information

Of particular interest within the SMB header are the `status` and `flag` fields which can be used to further characterize typical network traffic. The `flag` field contains a variety of information such as whether the message is a request or a response, synchronous or asynchronous, signed or unsigned. The `flag` field is also used to indicate support for distributed file systems (DFSs) and to indicate the presence of chained commands (multiple SMB commands within one single SMB header), whereas the `status` field is used in SMB command responses to indicate the status code of SMB command requests.

If a C2 communication channel resulted in status codes that did not otherwise naturally exist or existed in small quantities, it would be more likely to be detected. A summary of the frequency of naturally occurring command status code returns for select servers during the observation period is shown in

Table 3.4. Generally speaking, in order to be difficult to detect, a C2 communication channel will need to avoid SMB command requests that will generate response statuses other than those found in the table. In general, the `Success` status code is by far the most prevalent and should not raise any concerns if returned. Other status codes returned within a SMB communications channel should be considered on case by case basis for impact on the likelihood of detection.

Table 3.4: SMB command response status code occurrences summary

	Print Servers		**File Servers**		**Domain Controllers**	
Status Code	**Local5**	**Local3**	**Local4**	**Local2**	**Local2**	**Local3**
Access Denied	-	2403	144	1998	1015	1274
Bad Network Name	72	-	-	-	-	-
Buffer Overflow	-	14	-	-	8	26
Buffer Too Small	-	-	-	20	-	-
Cancelled	-	-	41	256	-	-
Driver Blocked Critical	-	24	-	14	31	280
Failed Driver Entry	-	-	-	5	210	-
File Closed	3	1594	597	1216	1703	2858
File Is A Directory	778	-	2425	4359	4	-
Invalid Device Request	-	-	-	-	2	3
Invalid Parameter	-	-	6	12	-	-
Lock Not Granted	-	-	-	63	-	-
Logon Failure	-	501	-	539	-	-
Network Name Deleted	-	13	28	65	2498	179
Network Session Expired	-	-	-	12	-	-
No EAS On File	-	-	-	-	2	3
No More Files	-	18	2708	292	252	330
No Such Device	-	-	-	-	-	-
No Such File	-	1	4	25	4	13
Not A Reparse Point	7405	-	2122	22955	-	-
Not Found	8	71	-	211	3	6
Notify Enum Dir	-	-	3	973	-	-
Not Supported	-	-	1	-	-	-
Object Name Collision	-	-	-	30	-	-
Object Name Not Found	328	116	1581	7670	9285	12192
Object Path Not Found	45	-	156	315	18	25
Path Not Covered	1	-	238	-	-	-
Pending	66	524	48	1029	25065	27963
Pipe Not Available	-	10	-	-	-	-
Success	31005	702414	83559	1744911	2319731	2402636
User Session Deleted	-	329	183	2146	46043	3481

The `flag` field provides for additional interesting ways to characterize the natural network traffic. Typical characterizations would include identifying

which SMB command requests and responses normally have which flags set for both the general case and for a given server. Table 3.5 summarizes the frequency of the different `flag` field combinations possible for select servers on the network. From the table many general characteristics of the SMB traffic on this network can be extracted. Specifically:

- Only a small subset of flag combinations appear naturally.
- The majority of SMB requests and responses are signed.
- Only SMB responses are asynchronous.
- Chained Commands are only found in asynchronous responses.
- Local4 file server supports DFS, but Local2 file server does not.

Table 3.5: Occurrences of header flag combinations

		Print Servers		**File Servers**		**Domain Controllers**	
	Flags	**Local5**	**Local3**	**Local4**	**Local2**	**Local2**	**Local3**
Requests	None	1573	25777	3529	19616	209630	211638
	Signed	74161	2036947	31962	1539346	2156318	2215416
	Signed, DFS	98	-	58572	-	-	-
Responses	Response Only	454	8813	52	2856	633	565
	Signed	2359	13633	277	230618	20811	17541
	Signed, DFS	-	-	51	-	-	-
	Async	-	-	-	-	3	-
	Async Signed	336	2696	3	856	25069	28008
	Async DFS	-	-	18	-	-	-
	Chained, Async, Signed	-	-	10	393	-	-
	Chained, Async, Signed	-	-	54	-	-	-

In addition, the flags expected for specific SMB command requests and responses can also be characterized. A study of the traffic data showed that SMB `Negotiate` command requests and responses (across all servers on the network) never had any additional flags set. In addition, SMB `chained` command flags were only ever observed in SMB `Change Notify` responses. Table 3.6 shows the distribution of SMB header flags as observed on the Local2 file server.

If a C2 communications channel is to be difficult to detect, the traffic generated would need to respect characterizations of the SMB header `flag` field. More specifically, when a C2 channel is in operation, only header flag combinations that appear naturally in relation to the SMB commands should be present. Flag combinations that appear infrequently should also be avoided if the communication channel is to remain difficult to detect.

Table 3.6: Summary of SMB request and response flags observed for Local2 file server

	No Flag (Request)	Signed Request	Response Only	Signed, Response	Signed, Async, Response	Signed, Async, Chained, Response
Negotiate	6693	-	-	2730	-	-
Session Setup	6697	12	1	1212	-	-
Tree Connect	2075	4404	-	1188	-	-
Tree Disconnect	2075	4308	-	73	-	-
Logoff	2076	4066	-	74	-	-
Cancel	-	53	-	-	-	-
Change Notify	-	1190	-	9	748	393
Create	-	290828	267		69	-
Flush	-	918	-	-	-	-
FS/IO Control	-	36538	-	1220	39	-
Lock	-	95782	6	-	-	-
Oplock Break	-	91	125	1	-	-
Query Directory	-	2241	-	49	-	-
Query Info	-	129891	-	50	-	-
Read	-	630603	-	1583	-	-
Set Info	-	1301	-	7	-	-
Write	-	58236	-	224428	-	-

3.5.5 SMB Message Information

Every message type has different data that could potentially be useful to extract and characterize, however the number of combinations and permutations possible for evaluating the data makes it impractical to characterize all SMB message information without additional context. From a practical perspective, in order to ensure a C2 communication channel is difficult to detect, an attacker can ensure the communication channel does not cause SMB message information to be populated with data that is unnatural on the target network. As an example, accessing SMB shares of the type `print` on this network was shown in Table 3.3 to be rather exceptional and as such any access to `print` shares should be avoided. Additionally, the frequency of occurrence and time of use characterizations previously discussed for SMB `Create`, `Close`, `Read` and `Write` commands can be further refined by share type being accessed if appropriate (shown in Table 3.3).

From a file server perspective, another example might include characterizing the typical format of names of files, paths to files and the sizes of files accessed. If a C2 channel is implemented that manipulates the file system, an attacker can characterize the names or sizes of files typically accessed in order to ensure the communication channel remains difficult to detect.

The same principle can also extend to the names of pipes accessed if named pipe shares are used for C2 communication. As depicted in Table 3.7, the names of the pipes read from and written to during the network observation period were relatively fixed by server type with little variation. Any addition of a new named pipe would be relatively easy to detect. This was particularly evident within the research network data where there was a large number of reads to the `browser` named pipe on Local2 which did not appear consistent with the other file server. Upon investigation by the network administrators, a misconfigured application was found to be responsible for the attempts to access the `browser` named pipe on the server.

Table 3.7: Named pipe access counts by server

	Print Servers		**File Servers**		**Domain Controllers**	
Pipe Names	**Local5**	**Local3**	**Local4**	**Local2**	**Local2**	**Local3**
srvsvc	8	34	1512	78	72	100
netlogon	-	-	-	-	69	143
protected_storage	-	-	-	-	56	46
lsarpc	-	-	-	-	658	779
samr	-	-	-	-	758	1954
spoolss	24517	1289305	-	-	-	-
browser	-	-	-	6419	-	-
wkssvc	8	12		-	-	-

3.6 Summary

This chapter identified several features of normal SMB traffic that must be respected in the design of any SMB communications channel if it is to be difficult to detect. In summary:

- Workstations do not communicate with other workstations directly.
- TCP flow data shows a clear usage pattern where a much higher volume of SMB traffic is observed during working hours.
- TCP flow data shows that there is a large baseline of consistent natural traffic that is present 24 hours per day from clients interacting with the DCs, however the other server types show minimal, highly variable traffic outside of working hours.
- TCP flow data shows that there are vast differences in utilization of each server.
- Not all SMB commands are seen on every server. SMB commands seen tend to be consistent among different server types (file, print, DC).

- Proportional frequency relationships between SMB commands can be observed. Some of these relationships are fixed regardless of server type and some are server type specific. File servers show the least evidence of strict proportional relationships.
- Time of use is predictable. Different servers supporting different users show evidence of different core work hours.
- SMB header `flag` and `status` fields can be used to further characterize typical traffic, as can SMB message information.

The general characterizations of SMB network traffic features presented should apply to similarly configured enterprise networks. It is the author's position however that in the absence of further validation, the characterizations should be confirmed for any given target network before they are assumed to be correct.

4 SMB C2 Communications Channels

4.1 Introduction

The previous chapter characterized typical SMB traffic on a production network in order to establish a baseline for normal SMB traffic. This characterization effort serves as the foundation by which suitable SMB C2 channel development is informed and ultimately measured in terms of its difficulty to detect.

This chapter discusses the validation of this network traffic characterization approach for covert communication channel development through the selection, design and implementation of POC SMB C2 communication channels. The channels are then validated for being difficult to detect through the measurement of the channel network footprints which are then evaluated for likelihood of detection in consideration of the characteristics of the typical traffic previously observed.

4.2 SMB Communication Channel Development - Overview

The SMB protocol is unique among the TCP/IP application protocols in that it exists to allow networked resources to be accessed nearly transparently by a client host. Hosts can access files, printers, devices and processes over the network as if they were local.

As discussed in Chapter 2, traditional network based covert timing and storage channels often involve piggybacking communications on existing user traffic using some form of encoding, with the expectation that an eavesdropper (either on the network or a process on the target system) would decode

messages, without impacting otherwise normal traffic or generating new traffic.

For this C2 communications channel, the intent is to exercise C2 between compromised clients, without assuming the presence of a compromised server. Given the modern implementation of network switches which will generally prevent eavesdropping by clients and given the analysis in Chapter 3 that showed workstation clients do not normally communicate with one another directly over SMB, an approach other than direct peer to peer communication is required if the channel is to be difficult to detect.

Since it would be straight forward to detect clients communicating directly, it is potentially more useful to consider how the protocol can be used as a C2 communications platform that was never intended, without relying on the presence of a wiretap or eavesdropper. To this end, the intent is to use an unwitting SMB server as an intermediary for C2 communications between compromised clients (Figure 4.1).

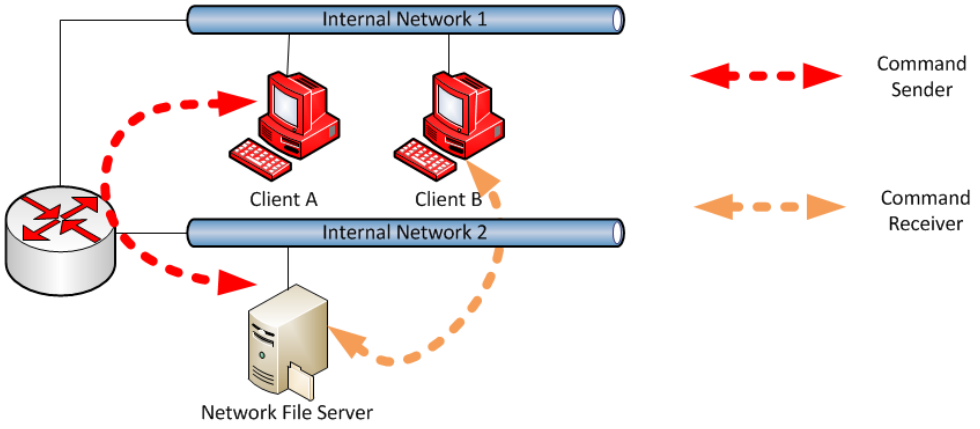

Figure 4.1: Intermediary for communications between exploited clients

There are several legitimate SMB servers on the research network including DCs as well as file and print servers. The analysis of the typical SMB network traffic revealed that client workstations do not typically modify data found on DCs or access SMB print shares to the extent that would be required to support a stealthy communication channel. Conversely, file servers showed better potential for use as there was a variety of different read and modify operations present within the SMB traffic with multiple network clients. As a result, the remaining focus for implementing and designing a SMB C2 channel

is through using a file server as an intermediary for communications between clients.

4.3 Identify and Evaluate Mechanisms for Intra-network C2 Using SMB

4.3.1 Shared Resource Matrix (SRM) Methodology

Kemmerer proposed the SRM as a formal mechanism for identifying possible covert channels between processes on monolithic systems[20]. This methodology was adapted for use here to identify resources and attributes that could be read and written using the SMB protocol from clients on the network that can access common locations on file servers.

Following the methodology established by Kemmerer, shared resources and attributes were identified. Primitives (SMB commands) that could be used by clients to access these resources were labeled as either having the ability to read or modify a given resource or attribute. The results for a typical shared file server are summarized in Table 4.1 where resource attributes are shown in row headings and command primitives are shown in column headings. Primitives that modify or reference an attribute are indicated with `M` or `R` respectively. As Kemmerer pointed out, any resource that can be modified by one process and read by another suggests the possibility of a covert channel.

Table 4.1: Network file server shared resource matrix

	Create	**Close**	**Write**	**Read**	**Lock**	**Query Dir**	**Query Info**	**Set Info**	**Change Notify**	**FS/IO Control**
File/Stream Exists?	M,R					R	R	R	R	M,R
Filename/ Streamname	M					R	R	M	R	M,R
File/Stream Contents			M	R						M,R
File/Stream Size	M,R	R					R		R	M,R
Timestamps	M,R	R	M	M			R	M	R	M,R
File Locks	M,R				M,R					
File Access Attributes	M,R	R				R	R	M	R	
OS File Attributes	M,R	R				R	R	M	R	

4.3.2 Identifying Possible Channels

The SRM at Table 4.1 reveals that there are many possible commands that can be used in support of communicating C2 messages across the network. While not exhaustive, the following provides example communication channels that can be implemented based on analysis of the SRM:

1. **Creating and deleting files, streams and directories**. Communications could be achieved simply by specifying an encoding relating to the existence of files or data streams in a specified folder. The encoding could be based on the number of files present or the change in the number of files since the last read of the directory.

2. **Modifying file names or stream names**. Data can be encoded through the text used in file and/or stream names.

3. **Contents of files and data streams**. Communications can be encoded in the content of files or data streams, either replacing legitimate content, or appending to existing legitimate content (to an image or pdf file for example), or by modifying existing legitimate content to encode a message (steganographic techniques for example).

4. **Changing the size of files or streams, security and operating system attributes or time stamp information**. While the file content could remain benign, an encoding scheme could be used to pass C2 messages through the many file attributes that exist. For example, the file size of various files within a directory could decode to a message, or the combination of attributes such as file share permissions, read-only or hidden flags could be similarly encoded to provide C2. Even the various time stamps present on a file (created, last accessed, last written) can be used to encode information or signal between clients.

5. **Locking and unlocking bytes in a file or data stream**. Locking and unlocking bytes in shared files or data streams can be used to pass signals between SMB clients. Communications can be established by encoding messages in the pattern of locks put in place on a particular file or groups of files.

For all the channels discussed, timing could be asynchronous, making use of the SMB `Change Notify` command, or could be synchronous with planned reads at established intervals (or even polling using one or more of the attributes as a signalling semaphore). In addition, the channel concepts can

be combined, either to achieve increased bandwidth or to achieve increased stealth.

To illustrate the concept of combining communication channel exploits, consider the following example scenario: An infected client checks once a day at a random time for a specific byte to be locked in a common shared file. When the byte is found locked, it will execute a SMB `Query Directory` command to get a directory listing of a previously established common directory and look for the presence of a 240 byte file with read-only attribute set and a 17 character filename. If the file exists, the client combines the 2nd, 5th, 8th, 12th and 17th characters from the file name and executes a previously agreed upon command from its own lookup table. To confirm execution of the command, the client creates a new arbitrary file in an established directory location which signals the original sender who is watching the directory with a SMB `Change Notify` command that the task is complete and the original byte that was locked can now be unlocked.

4.4 Implement SMB Communication Channels as a Proof of Concept

Given the characterization of the research network traffic from Chapter 3, possible SMB communication channels can now be vetted for suitability against typical network use. The SRM at Table 4.1 shows that the following SMB commands support modifying data: `Create`, `Write`, `Read`, `Lock`, `Set Info` and `FS/IO Control`. Reviewing the network characterization, it can be seen that `Set Info` is used infrequently to the point that it would not be suitable for communications that are difficult to detect. In addition, `FS/IO Control` functions that provide for modifying data (i.e. the server side copy function `FSCTL_SRV_COPYCHUNK`) are not supported by Windows 7 File Explorer, so it is unlikely to exist naturally on the observed network made up of primarily Windows 7 clients. This leaves SMB `Create`, `Write`, `Read` and `Lock` Commands for further consideration for implementing the writing portion of the communications channel.

In the research network data set, files are observed being transferred to and from the file server and in some cases accessed by multiple clients. Given this behaviour was normal, we believe it is reasonable to use this type of transaction as a template for implementing a communication channel. This channel would use the SMB `Write` command for signalling data, and the SMB `Read` command for receiving the signalled data.

In addition, given the existence of a database that showed multiple users locking and unlocking bytes of files on Local2, the SMB `Lock` command also showed good potential for use in a communication channel that would be difficult to detect on this network.

4.4.1 The Lock Suite

The Lock Suite was created as the first POC tool set. It is comprised of three applications based on the same principle, namely locking and unlocking bytes in shared files in order to signal messages. The three applications are:

1. `LockIt/UnlockIt`: the most basic tool, enables a supplied message to be encoded into and decoded from any file of sufficient size using byte locking.
2. `LockChat`: provides for continuous half-duplex communication between two clients by using byte locking and unlocking on a single shared remote file in order to communicate.
3. `LockCommand`: encodes and decodes Windows and PowerShell[29] commands via file byte locking, executing the commands on the remote system and then encoding/decoding and displaying the return value.

These programs were written in C++[30] and utilize the Windows API[31] functions `LockFileEX` and `UnlockFileEX` to lock and unlock bytes in specified shared network files. For this POC the encoding was a simple mapping to base 2, such that every character is encoded by the lock state of a sequence of 8 bytes (i.e. 'A' = 0x41 = 01000001, therefore 'A' is encoded by locking bytes 1 and 7). For `LockChat` and `LockCommand`, semaphore bytes are implemented for signalling readiness between the two running instances of the applications. Polling intervals for the semaphores can be adjusted in the source code to reduce the network footprint or increase responsiveness.

4.4.2 File Monitoring

This second POC communication channel was implemented as a PowerShell script executed on the remote client to be controlled. The script registers a file system watcher event on a specified shared file. When notified of a change, the contents of the specified file (or data stream) are retrieved, executed as either a Windows or PowerShell command and then the result is stored in a file at a specified location. The specified shared file or stream can be written to by any host that has access to the shared file on the network.

4.5 Ease of Detection Evaluation Methodology

Throughout Chapter 3, SMB traffic was characterized in such a way as one might expect a network defender to develop a baseline of SMB traffic with a goal of identifying anomalies within future comparative sets. From this work, several traffic features can be extracted from the normal traffic to define typical behaviour that should not be expected to trigger additional packet inspection.

For the purpose of this research, the author assumes that if a communication channel is implemented using no features that are expected to trigger additional inspection by a network defender, either employing comparative analysis techniques or looking for outliers, then the channel is considered to be difficult to detect. To facilitate assessment of the communication channel with respect to typical network traffic, the following seven elements are proposed as a framework for evaluating if a communication channel would be difficult to detect on a given network:

1. Respect for Normal Node Communication Patterns. If a communication system is to be difficult to detect, it must respect the normal node communication patterns of the network. SMB flow direction and node pairs should follow typical network traffic patterns.

2. SMB Flow Characteristics. If a communication system is to be difficult to detect, it must not change typical flow characteristics of the traffic (such as source/destination bytes or number of packets, or connection durations) to an extent that would be considered anomalous.

3. Presence of SMB Commands. If a communication system is to be difficult to detect, it must only use commands that have a natural occurrence between the nodes communicating.

4. Frequency of SMB Command Occurrence. If a communication system is to be difficult to detect, the frequency of occurrence of legitimate SMB commands must be respected in the context of normal network operation so as not to be observed as anomalous.

5. Respect for SMB Command Proportional Relationships. If frequency relationships can be established among the commands observed during normal network operation, then those relationships must be respected in order for the communication system to remain difficult to detect.

6. Time of Use. SMB network activity can be highly influenced by the presence of users and activities during working hours. If a communication system is to be difficult to detect, then both flow characteristics and command usage characteristics must not deviate substantially from normal characterization of the behaviour observed during the hours the channel is to be in use.

7. SMB Header and Message Data. If a communication system is to be difficult to detect, the SMB header and message data must be consistent with normal SMB traffic on the network. This could include considering file naming conventions, typical file sizes, files/directories accessed, flags set, statuses returned, print job names, named pipe names or any other data found within the SMB message.

It should be noted that existing techniques utilizing SMB named pipes to exercise C2 between infected hosts would not fare well against this evaluation framework. As discussed in Chapter 2, the Cobalt Strike threat emulation software and Duqu malware establish named pipe servers on infected clients for peer-to-peer communications over SMB with other infected clients and servers. As seen in Chapter 3, client workstations and network servers do not typically initiate SMB connections to other client workstations and as such, detecting these connections would be trivial using basic flow analysis techniques. In addition, as seen in Table 3.7, the list of named pipes accessed over this network is relatively short and is limited to legitimate Windows services. By inspecting SMB message information, additions to the list of named pipes accessed over the network would similarly be trivial to detect.

4.5.1 Determining Detection Threshold Values

In Chapter 3, normal SMB network traffic was analyzed to identify typical network usage, volume and patterns. It was assumed that SMB traffic on the research network would be relatively similar week to week, such that characterizations made from the observed data captured would hold true in the future. This assumption provides an advantage to the network defender as anomaly detection can only be successful in an environment that is relatively static. In the event that network behaviour changes significantly week to week, the likelihood of detecting a SMB channel through network analysis would be expected to decrease as a network analyst would not have a strong basis for comparative analysis, or outlier detection.

In order to detect anomalous activity, a network defender would need to identify which features are to be analyzed and what range of values for the

selected features should be accepted without triggering the need for additional inspection. Generally speaking, we believe that the thresholds for triggering additional inspections are largely subjective and as a result, an attacker must estimate the thresholds that are likely to be used by network defenders. These estimates are required in order for us to assess a SMB C2 channel's likelihood of detection and calculate its potential bandwidth.

Practical features relating to the observed normal network traffic will serve to increase or decrease the thresholds for anomalous activity detection. Specifically, consider the range, the variability and the volatility of accepted values for a given feature being analyzed.

In this research, the **range** refers to the spread of possible values for a given feature (i.e. during a typical workday, the range of SMB `Read` commands on a given server X is between 50 and 10,000).

Variability speaks to the distribution by which values are taken within the range (i.e. If packet sizes are being measured and they have a wide range of values, but only take on a small number of discrete values within the range, the variability would be considered low). Another example would be if the frequency of a particular command used always peaked at the same time around the same value everyday, this would demonstrate low variability. If however the peak usage was random within the range throughout the day, this would be seen as high variability.

Finally, **volatility** speaks to the tendency of a value for a given feature to change significantly between observation periods. An example would be if the number of SMB `Write` commands varied significantly from hour to hour or from day to day, this would show high volatility.

Generally speaking, the larger the range of values and the higher the variability and volatility, the more permissive the threshold for anomaly detection must be set by the defender in order to avoid an unmanageable number of false positives. Conversely, the smaller the range and the lower the variability and volatility, the more restrictive the threshold for anomalous activity can be set.

4.6 Validating Ease of Detection

In order to determine the network footprint of any SMB communications channel, it must be tested and measured in the most basic, trivial use cases. The characteristics of the channel in operation can then be analyzed by considering the seven elements in the evaluation framework in order to demonstrate whether or not it it is easy or difficult to detect. Once the trivial use case is defined, appropriate thresholds for anomaly detection can be estimated in

order to extrapolate the capabilities and limitations of the communication channel while remaining difficult to detect through network traffic analysis.

It should generally be expected that frequency of occurrence of SMB command analysis will yield greater restrictions on the channel capacity than analysis of the SMB traffic flow characteristics. Notwithstanding, the channel capacity is determined in consideration for both as the results of each analysis could lead to different channel usage restrictions. In addition, considering each separately serves to illustrate that channel capacity can be greatly increased if an attacker is able to assume that only flow analysis is being conducted by the network defenders instead of also using SMB protocol level analysis.

The following subsections illustrate how the two previously presented POC SMB communications channels can be shown to be be difficult to detect. The evaluations below are with respect to the live operation of the POC channels developed and implemented by the author (as discussed in Sections 4.4.1 and 4.4.2).

4.6.1 Evaluating LockIt/UnlockIt

The Trivial Use Case

Trivial use case activity to be evaluated:

1. During the work day, Client A encodes message "A" into a shared file on the remote file server (using `LockIt`).
2. Client B decodes the message encoded into the shared file on the remote file server (using `UnlockIt`).
3. Client A removes encoding from shared file on the remote file server.

Evaluation with respect to Difficult to Detect Elements

1. Node Communication Patterns - Not likely to be detected. All clients on the research network normally initiate SMB requests with the file server.

2. SMB Flow Characteristics - Not likely to be detected. No new flows are created since the server is automatically mapped and connected through normal login scripts. Based on the implementation of the Lock Suite tools, communication will be initiated over the pre-existing flow established at login. The duration of the flow will not change since this connection remains in place naturally throughout a client session. During this communication:

- Client A transmits 15 packets (2652 bytes) to the server, and receives 13 packets (2358 bytes) from the server.

- Client B transmits 52 packets (9937 bytes) to the server, and receives 50 packets (8079 bytes) from the server.

For both clients, the number of packets and bytes is negligible as compared to typical observed workday traffic. From Table 3.1, the median number of packets transmitted and received daily is in the order of tens of thousands and the number of bytes received and transmitted daily is in the order of millions.

3. Presence of SMB Commands - Not likely to be detected. The communication channel utilizes the following SMB commands which are found naturally in typical workday traffic: `Create`, `Close`, `Lock`, `Oplock Break`, `Query Directory`, `Query Info`.

4. Frequency of SMB Command Occurrence - Not likely to be detected. During this communication the following SMB commands are transmitted from Client A:
   - 1x `Create`
   - 1x `Close`,
   - 7x `Lock`,
   - 4x `Oplock Break`.

   From Client B:
   - 2x `Create`,
   - 2x `Close`,
   - 2x `Query Directory`,
   - 2x `Query Info`,
   - 42x `Lock`.

   For both clients, the number of additional SMB commands is small relative to typical SMB traffic observed as shown in Table 3.3

5. Respect for Proportional Relationships - Not likely to be detected. Previously characterized SMB command proportional relationships on the file server are respected. More specifically, during this communication, the SMB `Create` and `Close` proportional relationship of approximately 1:1 is respected. In addition, the relationship between SMB `Lock` and

`Oplock Break` is respected (that is, more `Lock` than `Oplock Break` commands typically occur).

6. Time of Use - Not likely to be detected. Since the Local2 server was acting as both a DC and a file server, it had regular traffic all day and all night throughout the week, including file locking, unlocking and opportunistic lock breaking. When the DC traffic was segregated from the file server traffic however, `Oplock Break` commands were only observed during working hours within the file server traffic. As a result, this channel can only be used during working hours if it is to be difficult to detect.

7. SMB Header and Message Data - Not likely to be detected. The most commonly locked files observed in the captured traffic for the file server were part of a large database in a specific directory. In addition, portions of excel spreadsheets were observed being locked and unlocked at different times by different clients. In order to remain difficult to detect on this server, the communication channel should be locking bytes in files of these types.

   All combinations of SMB `Lock` commands were observed in the normal network traffic based on the different flags possible within the SMB `Lock` message. In particular, the type of `Lock` command recorded the most was 1 byte in range, of the type `Exclusive, Fail Immediately`, which is consistent with the size and flags that are set in the SMB commands issued over the network using the Lock Suite tools developed.

   Similarly, the SMB `Oplock Break` commands found in the normal network traffic had the same `lease flags` and `lease state` attributes as the `Oplock Break` commands issued over the network using the Lock Suite tools. In addition, SMB command `status` responses for all commands were also consistent with those observed in natural traffic.

   A variety of lock byte offsets were also observed in normal network traffic, some of which were frequently re-used (as is the case by default with the Lock Suite tools). Of note however is that byte offsets were not typically at the very start of the file, which is also the case by default with Lock Suite tools. Additional characterization work could involve determining appropriate offsets to start encoding messages within a file, and determining if it would be advantageous to have the offset used by the Lock Suite tools change dynamically with every message sent.

   While the author believes it to be relatively unlikely that a network defender would attempt to characterize typical values for the offset, this would be possible given enough resources, a sensitive enough network

security posture and a high enough tolerance for false positives. While further characterizations can improve the consistency of the communication channel with that of typical traffic, they can also overly constrain its application. It is left to the attacker to assess at what point further characterizations of SMB messages provides diminishing returns. Ultimately, the byte offset to encode and decode messages using the Lock Suite is easy to change as necessary.

Based on the above analysis, the `LockIt/UnlockIt` tool set can be used to effectively send and receive a single byte message between hosts during working hours using the SMB protocol in a manner that is difficult to detect.

Extending the Analysis to Determine Channel Capacity

In order to determine the channel capacity, we ask:

- How many bytes can be encoded/decoded in one instance without raising the likelihood of detection beyond an acceptable threshold for the research network? and
- At what rate could commands be sent without raising the likelihood of detection beyond an acceptable threshold?

For the `LockIt/UnlockIt` tool set, these questions can be answered by revisiting the elements of the analysis that are likely to be impacted when increasing throughput, namely the SMB flow characteristics and frequency of SMB command occurrence.

While highly subjective, reasonable detection thresholds must be estimated where applicable in order to identify the extent to which the tools can be used while maintaining a communications channel that is difficult to detect.

Calculating Channel Capacity from Analysis of SMB Flows

With respect to SMB flow characteristics, no additional flows are created as the `LockIt` channel usage increases. As message length and number of messages increases however, the number of packets and bytes sent by the clients and the server will change in a commensurate manner.

While flow attributes can be considered from a variety of view points, many can be discounted if the worst case perspective is adopted. Specifically, for this channel, Client B (the receiving client) will necessarily generate the largest network footprint. As a result, setting acceptable usage limitations on the communication channel from Client B's perspective will indirectly ensure acceptable limitations are established on the sending client (Client A) as well as the file server.

Given the channel implementation, the network footprint for each message processed by `LockIt/UnlockIt` can be calculated if the message is known in advance. For a conservative estimate of the general case, the worst case message must be assumed which requires 8 bytes to be locked and then unlocked by the message receiver per symbol in the message.

The worst case number of new packets generated by receiving a message can thus be calculated by the equation:

$$NumberofPackets = (MessageLength) \times 16 + Overhead \tag{4.1}$$

Where 16 represents the worst case number of additional SMB `Lock` packets per message symbol and *Overhead* represents the fixed number of packets that are required as overhead for channel management (which is 36 for `LockIt/UnlockIt`).

In order to determine the maximum permissible message length, we must estimate the number of additional packets a client can have per day without being observed as an anomaly by a network defender. Analysis of the typical source packet counts during workdays for individual clients and summarized for the week in Table 3.1 shows that there is a very large normal variation in the number of packets sent by each client (between 4 and 7,227,120 packets was observed). Furthermore, when evaluated day to day, the mean number of packets sent from each client varies greatly, as does the range for maximum number of packets sent. Based on a review of the typical variation of the volume of source packets, we estimate that up to 30,000 additional packets per work day on any given client would not appreciably increase the likelihood of detection. Therefore:

$$MaxMessageLength = \frac{30,000 - 36}{16} \approx 1872\ bytes \tag{4.2}$$

Calculating Channel Capacity from Analysis of Frequency of SMB Command Occurrence

In the case of `LockIt/UnlockIt`, one channel limitation affecting possible detection is the use of the `Oplock Break` command, which has a relatively low natural occurrence on the network. Fortunately, the number of `Oplock Break` commands used in the `LockIt` communication channel remains static, regardless of message length, however a fixed number of these commands are associated with every message sent.

For Local2, `Oplock Break` commands were observed between 0-54 times per day per client and the usage was highly variable and volatile within that

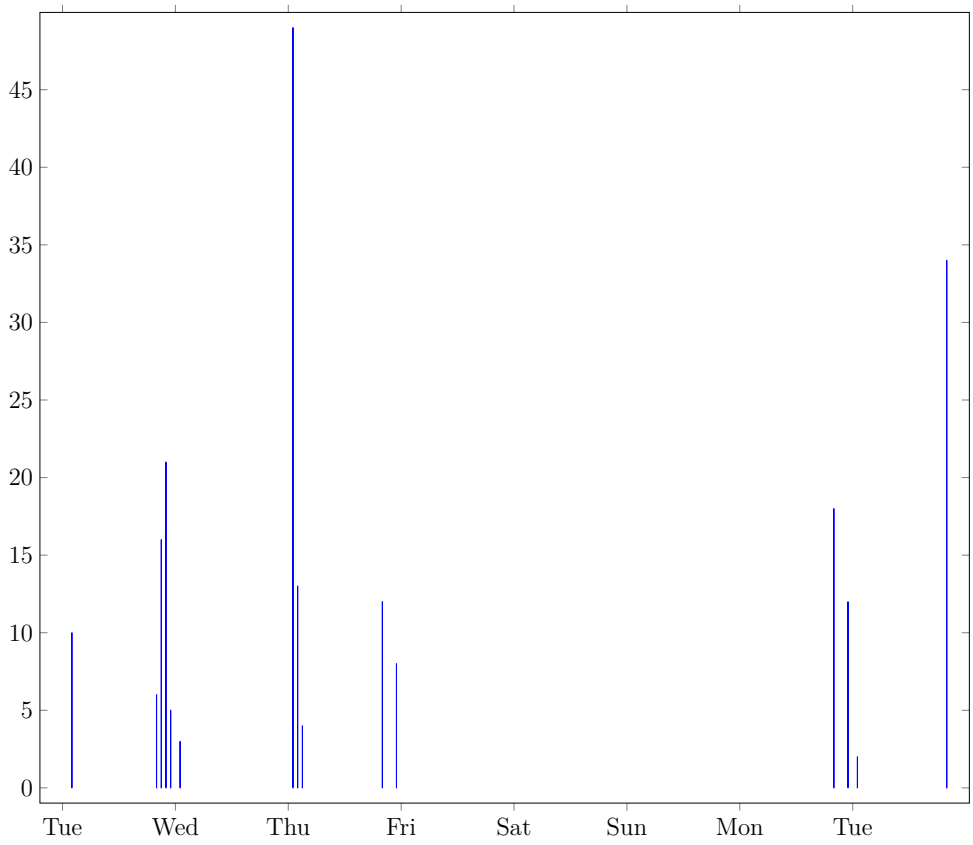

Figure 4.2: Local2 SMB Oplock Break commands by hour

range as can be seen in Figure 4.2. Based on this analysis, we estimate that up to 20 additional SMB `Oplock Break` commands per work day would not appreciably increase the likelihood of detection. This estimate imposes a limitation of no more than 5 messages per day if the channel is to remain difficult to detect.

Additionally, limitations based on the use of the SMB `Lock` command must also be established. Individual clients were observed sending between 0 and 47,920 SMB `Lock` commands per day, however for the most part, the number of SMB locks was below 1400 per day (per client). The client that was observed sending 47,920 SMB `Lock` commands in one day stood out as anomalous. As a result, despite this behaviour being normal, we removed it from the characterization since the goal is to avoid standing out as anomalous. Plotted by hour, SMB `Lock` commands sent to Local2 shown in Figure 4.3

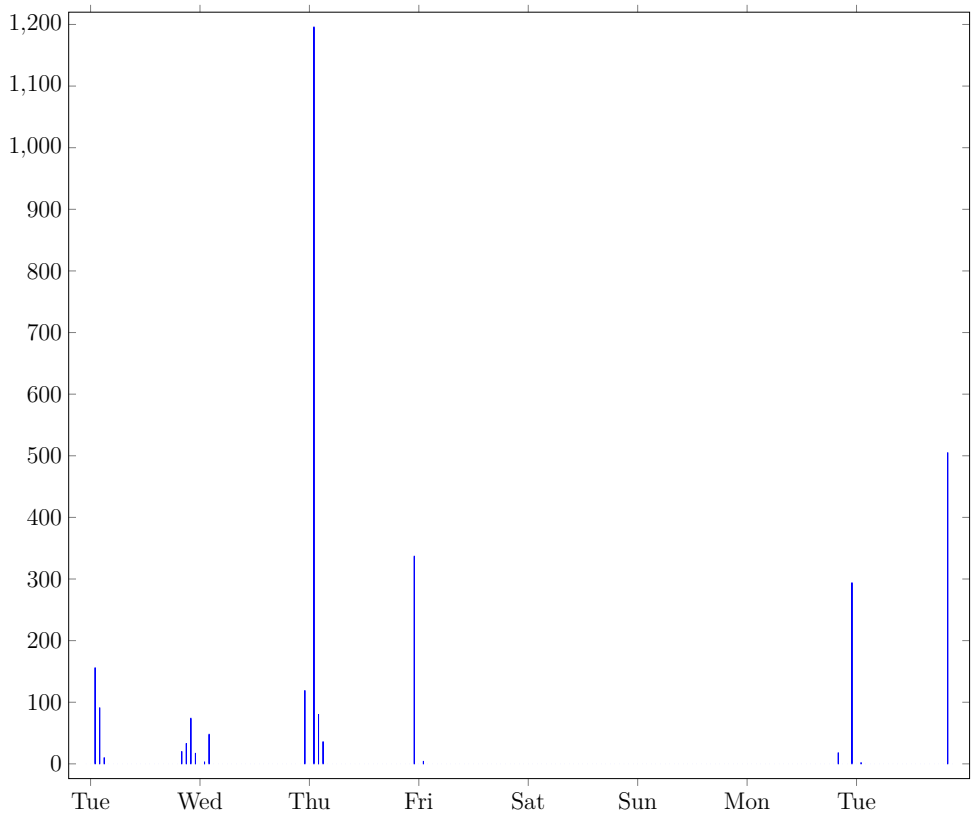

Figure 4.3: Local2 SMB Lock commands by hour

also show a high degree of variability and volatility. Based on this analysis, we estimate that up to 400 additional SMB `Lock` commands per work day on any given client would not appreciably increase the likelihood of detection.

There are 28 SMB `Lock` commands issued by the message receiver as part of the overhead for the implemented communication channel resulting in the worst case number of additional SMB `Lock` commands being calculated by equation:

$$SMBLockCommands = MessageLength \times 16 + Overhead \tag{4.3}$$

Therefore:

$$MaxMessageLength = \frac{400 - 28}{16} \approx 23\ bytes \tag{4.4}$$

The maximum number of SMB `Lock` commands sent from a single client in one hour was observed as approximately 1200, with the next highest maximum

usage around 500 and 350 commands in one hour. In general, there was a high degree of variability and volatility in the usage of this command as observed in Figure 4.3. Based on this analysis, we estimate that up to 150 SMB `Lock` commands per hour on any given client would not appreciably increase the likelihood of detection. Substituting this value into the previous equation:

$$MaxMessageLength = \frac{150 - 28}{16} \approx 7\ bytes \tag{4.5}$$

A similar analysis of the SMB `Oplock Break` commands in further consultation with Figure 4.2 (which shows high volatility of command occurrence ranging from 0-49/hour) leads us to estimate that up to and additional 8 SMB `Oplock Break` commands per hour would not appreciably increase the likelihood of detection. This translates into a a maximum of 2 messages sent per hour.

LockIt/UnlockIt SMB C2 Communication Channel Summary

Combining the limitations discussed, the capabilities of the `LockIt/UnlockIt` tool set can be articulated as follows:

To maintain a low likelihood of detection, no more than two 7-byte messages can be sent per day during working hours and their transmission should be separated by at least one hour in order to respect the esimated threshold for permitted number of additional SMB `Lock` commands per hour.

Alternatively, up to four 4-byte messages can be sent during the work day (separated by an hour). The message size is reduced to ensure no more than 400 SMB `Lock` commands are sent in total during a single day. Finally, if the message size is reduced to 2 bytes, up to 2 messages can be sent per hour (limited by the number of `Oplock Break` commands permitted), up to a total of 5 messages per day.

Note that the limitations on message length provided are conservative estimates since the worst case message symbol is assumed (all zeros) for the calculations. If an attacker wishes to have a more accurate assessment of the channel footprint as they intend to use it, the number of SMB `Lock` commands required for any message to be decoded using the Lock Suite tools can be determined as follows:

- Translate the message to base 2.
- Let x be the number of zeros.
- Let y be the number of ones.

$$ReceiverSMBLockCommands = 2x + y + 28 \tag{4.6}$$

4.6.2 Evaluating the File Monitor Channel

The Trivial Use Case

Trivial use case activity to be evaluated:

1. Client B launches file monitoring script, watching `thumbs.db:AFP_AfpInfo` on remote file server.
2. Client A sends command `whoami` by writing to `thumbs.db:AFP_AfpInfo`
3. Client B writes response to `thumbs.db:AFP_AfpInfo` in a different path on remote file server.

Evaluation with respect to difficult to detect elements

1. Node Communication Patterns - Not likely to be detected. All clients on the research network normally initiate SMB requests with file servers.

2. SMB Flow Characteristics - Not likely to be detected. No new flows are created since the server is automatically mapped and connected through normal login scripts. Based on the implementation of the file monitoring script, communication will be initiated over the pre-existing flow established at login. The duration of the flow will not change since this connection remains in place naturally throughout a client's normal session. During this communication:

   - Client A (command sender) transmits 6 packets (1245 bytes) to the server, and receives 8 packets (1574 bytes) from the server.

   - Client B (command receiver) transmits 28 packets (6000 bytes) to the server, and receives 28 packets (6059 bytes) from the server.

   For both clients, the number of packets and bytes is negligible as compared to typical observed workday traffic. From Table 3.1, the median number of packets transmitted and received daily is in the order of tens of thousands and the number of bytes received and transmitted daily is in the order of millions.

3. Presence of SMB Commands - Not likely to be detected. The communication channel utilizes the following SMB commands which are found naturally in typical work day traffic: `Create`, `Close`, `Change Notify`, `Get Info`, `Read`, `Write` and `Cancel`.

4. Frequency of SMB Command Occurrence - Not likely to be detected. During this communication the following SMB commands are transmitted from Client A:
   - 2x `Create`,
   - 2x `Close`,
   - 4x `Get Info`,
   - 1x `Write`,
   - 1x `Read`.

   From Client B:
   - 6x `Create`,
   - 6x `Close`,
   - 3x `Change Notify`,
   - 8x `Get Info`,
   - 2x `Read`,
   - 4x `Write`,
   - 1x `Cancel`.

   For both clients, the number of additional SMB commands is small relative to typical observed SMB traffic summarized in Table 3.3.

5. Respect for Proportional Relationships - Not likely to be detected. Previously characterized SMB command proportional relationships for the file servers are respected when the channel is in use. Specifically, SMB `Create` and `Close` commands maintain a 1:1 relationship and there are more SMB `Query Info` commands than `Change Notify` commands. In addition, there are more `Change Notify` commands present than `Cancel` commands, which is also consistent with what was observed in normal traffic.

6. Time of Use - Not likely to be detected. When only considering file server traffic, SMB `Get Info` and `Cancel` commands were only observed during working hours. As a result, this channel can only be used during working hours if it is to be considered difficult to detect.

7. SMB Header and Message Data - Not likely to be detected. SMB `Change Notify` requests in observed traffic were always synchronous and the SMB `Change Notify` responses were always asynchronous. These flags were consistent with the SMB traffic generated by the file monitoring

script. In addition, only the `Success` status code was returned for every SMB command sent, which is consistent with the observed natural traffic status responses from Table 3.4.

The file `thumbs.db` was a filename that existed in several directory paths and was accessed by multiple clients throughout the observation period. Some of the `thumbs.db` files observed had alternate data streams named `AFP_AfpInfo`. As a result, accessing and writing these files should not increase the likelihood of inspection. Interestingly, while users did not typically have access to write to `thumbs.db` files in a shared directory, users could alter the attached `AFP_AfpInfo` alternate data stream (or `Create` it themselves).

Observed `AFP_AfpInfo` streams had a fixed size of either 0 or 60 bytes. As a result, if using this stream to send a command, it should be padded and limited in size to 60 bytes to minimize the likelihood of additional inspection.

Based on the analysis, the file monitor channel can be used to effectively send simple commands for processing and retrieve responses across the network in a manner that is difficult to detect.

Extending the Analysis to Determine Channel Capacity

In order to determine the channel capacity, the rate at which subsequent commands can be sent while maintaining a channel that is difficult to detect must be determined. For the file monitoring script, SMB flow characteristics and SMB command frequency of occurrence elements from the evaluation framework should be revisited as they are impacted by increasing the channel usage.

Calculating Channel Limitations from Analysis of SMB Flows

For the file monitoring script, no additional flows are created using this method as channel usage increases, however as the number of messages changes, the number of packets and bytes sent by the clients and server will change in a commensurate manner.

Client B (the command receiver) will be used to determine maximum usage thresholds since it will necessarily generate more traffic for every command received than Client A generates and will necessarily be less tolerant to additional traffic than the file server in terms of anomaly detection.

When the file system monitoring script is initially executed on Client B, four packets consisting of 984 bytes total are sent to the file server. When

a command is received, Client B will write the response to the file server by sending 24 additional packets consisting of 5016 bytes total.

As noted earlier, it is estimated that up to 30,000 additional packets per work day on any given client would not appreciably increase the likelihood of detection on this network from flow analysis. Based on this estimate, it is possible to calculate the maximum number of commands per day that can sent to a client using this channel while avoiding increasing the likelihood of detection by flow analysis:

$$NumberOfPackets = (NumberOfCommandsSent) \times 24 + 4 \tag{4.7}$$

$$MaxNumberOfCommands = \frac{NumberOfPackets - 4}{24} \approx 1250 \tag{4.8}$$

Calculating Channel Limitations from Analysis of Frequency of SMB Command Occurrence

Client B as the command receiver can be used to determine the maximum usage threshold since it will be impacted the most by an increase in channel use.

In the case of the file monitoring channel, the occurrence of SMB `Change Notify` commands shows the most sensitivity to impacting the likelihood of detection as the natural range of occurrences is on the low end (between 210-610 per work day) as compared to the other SMB commands present in the channel. Taking into account the variability and volatility observed, we estimate that up to 200 additional SMB `Change Notify` commands per work day on any given client would not appreciably increase the likelihood of detection.

Since three SMB `Change Notify` commands are associated with every message sent using this channel, the new upper limit becomes $200 \div 3 \approx 66$ commands per day.

Furthermore, if hourly behaviour is considered (shown in Figure 4.4), peak usage of the SMB `Change Notify` command during working hours varied daily from 20-160 commands per hour, showing once again a high degree of both variability and volatility. Based on the analysis, we estimate that during working hours, up to 30 additional SMB `Change Notify` commands commands per hour on any given client would not appreciably increase the likelihood of detection. From this analysis, a limitation of no more than $30 \div 3 = 10$ commands per hour can be imposed.

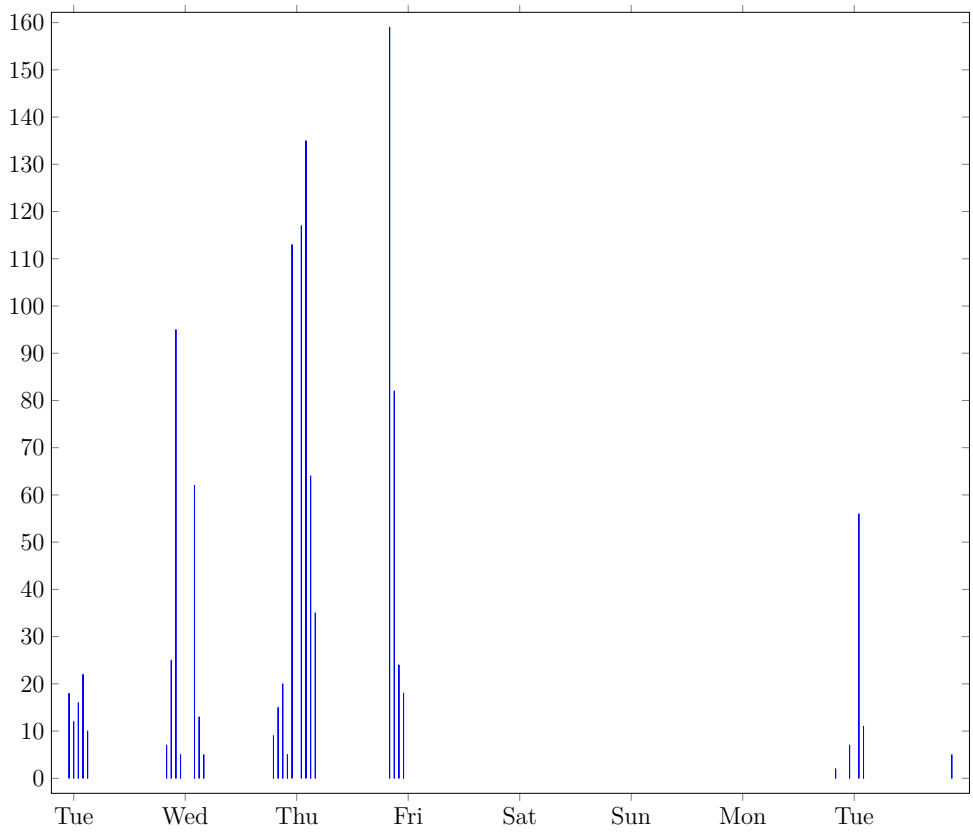

Figure 4.4: Local2 SMB Change Notify commands by hour

File Monitor SMB C2 Communication Channel Summary

Combining the limitations discussed above, no more than 10 commands per hour could be sent using this channel during working hours, with a maximum number of 66 commands per day.

The measurements on this channel assume that commands that are issued have relatively small returns that will fit within the normal 60 bytes of the AFP_AfpInfo alternate data stream.

4.6.3 Sensitivity of Detection Threshold Estimates

As discussed in Section 4.5.1, in order to complete channel capacity calculations for both POC SMB channels, the thresholds that might be used to trigger additional packet inspections by network defenders must be estimated. To il-

lustrate a channel's sensitivity to these largely subjective estimates, Tables 4.2 and 4.3 show the impact to a channel's capacity if daily detection thresholds are estimated to be more permissive or more restrictive. Estimates used in previous calculations are highlighted for comparison and the bottom row in each table shows the impact of setting the highest observed daily value for the given attribute as the estimated threshold for detection. As seen in the tables and given the linear relationships established at Equations 4.1, 4.3 and 4.7, an increase in a detection threshold's permissivity will result in a proportional increase in channel capacity.

Table 4.2: Channel capacity sensitivity to Additional SMB Flows permitted

	LockIt/UnlockIt	File Monitoring
Flows	Message bytes/day	Commands/Day
15,000	935	625
30,000	1873	1250
60,000	3748	2500
300,000	18,748	12,500
7,228,270*	451,765	301,178

* This value reflects setting the highest observed daily flow count within the data set as the estimated threshold.

Table 4.3: Channel capacity sensitivity to additional SMB commands permitted

LockIt/UnlockIt				File Monitoring Channel	
Lock Commands	Message bytes/day	Oplock Break Commands	Messages per Hour	Change Notify Commands	Commands per Day
200	10	4	1	100	33
400	23	8	2	200	66
800	48	16	4	400	133
4000	248	80	20	2000	666
47920*	2993	54*	13	610*	203

* These values illustrate the impact of using the highest observed daily value for a given attribute within the data set as the estimated threshold.

4.7 Persistence of Traffic Characterizations

As discussed earlier, in order to to support stealthy SMB channel development, it is necessary to assume that the traffic features characterized would remain consistent week to week. This assumption provides an advantage to the network defender as static network patterns necessarily facilitate anomaly detection.

To better support this assumption, a second data set was obtained for the same network spanning a different period of approximately seven days. This second data set was characterized with a view to appreciating the extent to which SMB traffic on this network might be similar during different observation periods. Table 4.4 shows a comparison between the workday flow attribute summaries for the first and second data sets.

Table 4.4: Workday client flow attribute summaries for both data sets

	Mean		Median		Min		Max	
Data Set	1	2	1	2	1	2	1	2
Flow Count	148	171	154	154	1	1	2,933	5,109
Source Packets	54,643	50,338	16,817	39,069	4	0	7,227,120	3,242,578
Destination Packets	103,864	111,659	47,185	87,277	0	0	18,314,445	11,692,004
Source MBytes	56.9	7.1	3.2	5.1	0	0	27,319.3	332.9
Destination MBytes	230.3	274.4	119.4	230.0	0	0	44,317.2	29,823.9

The general characterizations summarized at the end of Chapter 3 were found to remain valid. In addition, the trivial use cases of the implemented POC SMB C2 channels remained difficult to detect if put into the context of this subsequent traffic capture. With respect to the thresholds that were estimated for maximizing use of the communications channels, they remained fairly consistent with what would have been estimated had the second data set been considered in isolation. To illustrate, Figures 4.5 and 4.6 show the relative distribution of commands by hour for comparison of SMB `Change Notify` and `Lock` commands respectively[1].

Ultimately, the second traffic capture shows that typical usage is less predictable than what was revealed through characterizing only one week of traffic. More permissive anomaly detection thresholds are likely to arise from characterizing larger data sets, as the range of typical values for observed characteristics is likely to increase. As noted in [5], the less predictable the traffic, the more difficult channel detection will be.

[1]Note that the Monday was a holiday during the Week 1 capture and the Week 2 capture did not extend into the following Tuesday.

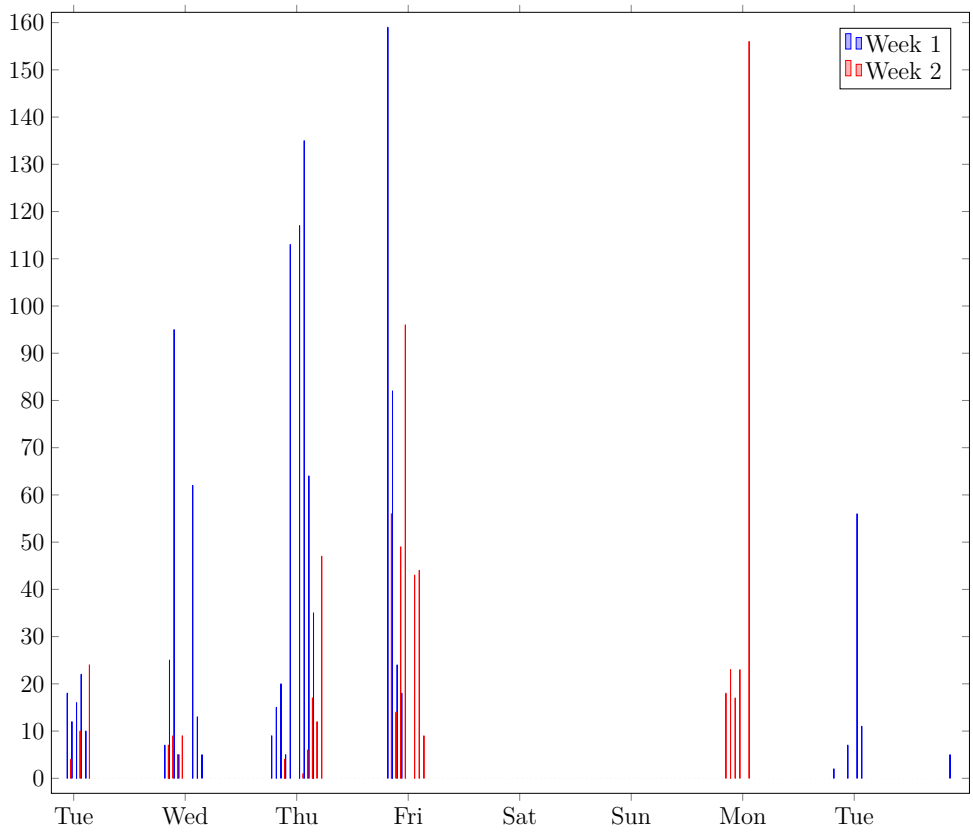

Figure 4.5: Two week comparison of Local2 SMB Change Notify commands by hour

4.8 Summary

In this chapter, the SRM methodology was employed to help identify communications channels that would be possible when interacting with a remote file server using the SMB protocol. Based on the network traffic characterization in Chapter 3, two possible channels were selected and implemented as proofs of concept with the goal of creating a C2 communications channel over SMB that would be difficult to detect through network traffic analysis.

Once the POC communications channels were built, a methodology was proposed for evaluating whether or not an SMB communication channel would be difficult to detect. This methodology was then applied in order to evaluate the POC channels, which were ultimately validated as being difficult to detect.

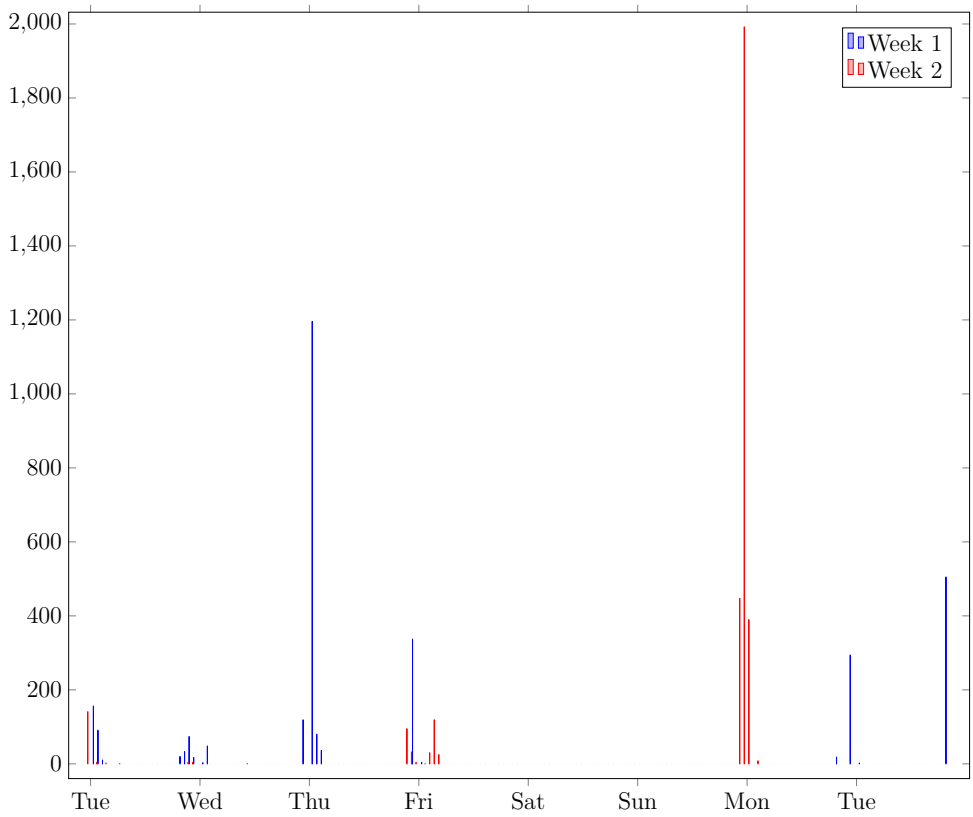

Figure 4.6: Two week comparison of Local2 SMB Lock commands by hour

Channel capacities were then identified by estimating the detection thresholds that a network defender would apply given the characterization of normal SMB traffic on the network discussed in Chapter 3.

Validation for the research question *how an attacker might exploit existing network support for SMB in order to implement a C2 channel that would be difficult to detect through network traffic analysis* was achieved through the design and implementation of POC SMB C2 channels that are demonstrably difficult to detect.

5 Discussion and Conclusion

5.1 Discussion

This chapter concludes the thesis by discussing the overall results, threats to validity and importance of the work performed. Recommendations for future work in SMB communication channel development and detection are also made.

5.1.1 Results

This thesis investigated the process an adversary might follow to design and implement a SMB C2 communications channel that would be difficult to detect in a networked environment.

The research presents a suitable framework by which potential SMB communications channels can be identified (the SRM methodology) and proposes a framework by which an SMB communication channel can be assessed for detection difficulty based on knowledge of the typical network traffic.

Typical SMB network traffic was captured and characterized on a live network and potential SMB communications channels were identified using the SRM methodology. Based on the traffic characterization effort, two distinct communications channels were created as proofs of concept and these channels were characterized through experiment and analysis. The characteristics of these POC channels were then compared with the baseline characterization of the typical SMB network traffic captured in order to validate the assertion that the channels could be used in a way that would be difficult to detect.

Overall, each of the two channels implemented as a POC have advantages and disadvantages for use. In the case of the `LockIt/UnlockIt` tool set, one of the advantages is that no file system artifacts are created on the file server. One of the disadvantages however is that the channel is very limited in terms of its capacity to communicate large amounts of information without being detected. In practical terms, based on the characterization of traffic on the

research network, this channel would be best suited for sending signals versus actually transferring data between two compromised hosts.

For the file monitoring channel, the main disadvantage is that file system artifacts are necessarily created as part of the channel's use. Though these artifacts can be deleted, file server logging and forensic file system analysis could present additional vectors for channel detection. On the plus side however, the file monitoring channel has a much higher capacity than the `LockIt/UnlockIt` channel. It also provides much more flexibility in terms of transferring large amounts of data between the infected clients through the file server. The behaviour of the channel very closely mimics the typical user behaviour of browsing to a common file server, saving files and reading files. As a result, it has much potential for use in many typical networked environments.

Ultimately, the research aim of *investigating how an attacker might exploit existing network support for SMB in order to implement a C2 channel that would be difficult to detect through network traffic analysis* has been accomplished through the work presented herein.

5.1.2 Threats to Validity

While conducting research, threats to validity are unavoidable and this research is by no means exempt from the potential for bias. For this research, the process for designing and evaluating SMB C2 channels was applied to a single medium sized network at an academic institution over a relatively short period. Though we hypothesize the process would be similarly effective in other enterprise networked environments, this proposition has not been tested further within the scope of this thesis.

In addition, for our research we defined the communication channel as being "difficult to detect" if it possesses no features that are expected to trigger additional inspection by a network defender. We also established which subset of features would be characterized and then constructed communications channels in consideration for those features. Moreover, we estimated the detection thresholds network defenders might use when calculating the potential bandwidth of the communications channels. While the potential for bias clearly exists, we strove to be impartial, applying conservative estimates and definitions where possible.

5.2 Implications

While the SMB specification provides the necessary detail to understand how the protocol can be used, it does not provide any insight with respect to

what features are actively in use on enterprise networks today. In order to both exploit SMB and avoid detection, it is important to understand the characteristics of the SMB activity found on networks under normal operation.

We found no other existing published research which provides for a general characterization of SMB2 traffic in a live network environment. This research establishes a baseline characterization of SMB2 traffic that can be used as a jumping off point for future research in SMB C2 channel detection and development.

In terms of network defence, the SMB network traffic characterization can serve to help identify what sort of network interactions are particularly suspect in order to help network defenders in their detection practices. For example, despite being permitted by the SMB specification, many command, argument and flag combinations may not exist naturally under normal operating conditions and as such, might always be treated as anomalous. The research presented herein could form the basis for the development of signature based intrusion detection algorithms that would report unexpected communications such as servers initiating TCP connections with clients directly or SMB commands being used that are not normally present such as the SMB `Echo` command.

The POC channels demonstrate that there is a real and present threat from this vector on existing enterprise networks. This work is important in highlighting to cyber security professionals that supporting SMB on an internal network necessarily creates the potential for unintended C2 communications channels and as a result, security practices need to evolve in order to better detect the threat. In addition to increased monitoring, one practice that should be considered for change is the dual purposing of servers for multiple functions (such as file, print and DC). Dual purposed servers made the characterization effort exceedingly difficult, which suggests anomalies would be similarly much more difficult to detect by a network analyst.

Finally, given that the latest version of SMB (3.0) supports encryption, a characterization of the unencrypted activity also provides a sample of some of the background data necessary to support future characterization of SMB encrypted traffic. This could prove to be particularly important moving forward as encrypted SMB traffic becomes the new normal on enterprise networks.

5.3 Future Work

The work presented in this thesis teases the imagination with future research possibilities. The following areas are identified for future work:

5.3.1 SMB Traffic Characterization of Other Enterprise Networks

Many features of typical SMB traffic are identified herein based on a characterization of the traffic observed on a single network. Performing a similar characterization on other networks would provide a better understanding of what features tend to be unique and what features tend to be universal which could then serve as a basis for developing improved general purpose SMB communications channels that would be difficult to detect.

5.3.2 Conventional Eavesdropper Covert Messaging

While this research effort sought to develop SMB communications channels by creating new SMB traffic, the idea of a traditional network protocol covert channel that piggybacks on existing legitimate traffic remains worthy of exploration. Given SMB session control commands were shown to have a consistent presence on all SMB servers, it stands to reason that they might serve as a good starting point for looking for opportunities to piggyback covert messages. The SMB header field `Credits Requested` for example was observed as having good potential for this purpose.

5.3.3 SMB FS/IO Control Command

SMB `FS/IO Control` commands are present among all the different SMB server types and have a relatively large natural presence at all hours of the day. In addition, the Bro protocol analyzer does not have a module that decodes known `FS/IO control` codes, making characterization more difficult. Moreover, the SMB `FS/IO Control` command does not have a rigid format because it supports proprietary communications between different SMB server applications. Since a decoding module does not currently exist in Bro and due to the support for proprietary communications, the SMB `FS/IO Control` command is worthy of additional investigation to determine options for implementing C2 communications channels that would be difficult to detect.

A very promising existing `FS Control` command function is the server side copy (`FSCTL_SRV_COPYCHUNK`), which allows a client to specify data to be copied from one location on the file server to another, without transferring the data back and forth on the network. In consideration of the file monitoring POC channel, it would be worth investigating if this command has the potential to significantly reduce the size of the network footprint created by the command sender when this channel is in use.

5.3.4 Suite of SMB Communication Techniques

Through the development of the SRM, multiple possible communication channels were identified, but only two were implemented to demonstrate the POC. Building upon the techniques demonstrated, a suite of more mature SMB communication tools could be built that would utilize the full range of file system attributes that could be manipulated to encode C2 information. These tools could then be integrated into an exploitation framework such as Metasploit, which would provide a much more robust environment for control and execution.

5.3.5 Printer C2 Communications Channel

While not related directly to the SMB protocol, it was observed that there is good potential to implement a similarly straightforward C2 channel using print servers as an intermediary. Specifically, Windows Management Instrumentation (WMI) commands [32] can be used to easily script sending print jobs, pausing print jobs, deleting print jobs and retrieving print job attributes from print servers. All these activities are generally observed as typical user behaviour on the network and as such, implementing a C2 communications channel based on two or more clients monitoring, writing and deleting jobs to a common print queue should yield interesting results.

5.3.6 Persistent SMB C2 Using the Domain Controllers

One of the limitations observed with the POC SMB C2 channel implementations is that outside of working hours, the channels are largely unusable if there is a threat of network surveillance. The exception noted is that DCs can be expected to maintain constant communications with all hosts on the network 24 hours a day, 7 days a week. In the case of a compromised DC, it would be interesting to investigate establishing a SMB C2 channel that replaces the existing policy files regularly downloaded by network clients with files that contain C2 instructions. While this this type of channel would necessarily require the DC to be compromised, once established it would likely be extremely difficult to detect and exceptionally flexible as it could be used continuously both during and outside of working hours without increasing the likelihood of detection.

5.4 Conclusion

The SMB protocol remains a highly integrated component of our enterprise networking environment. Given its ubiquity and powerful built-in capabilities, so long as it persists on our networks, would-be attackers will seek to exploit its presence. Previous SMB C2 channel development efforts have not considered evading detection through network traffic analysis. This thesis shows that SMB C2 channels can be designed and implemented in a manner that would make detection through network traffic analysis difficult.

By characterizing the network traffic and demonstrating how an attacker might implement a C2 channel that is difficult to detect, this research confirms the threat SMB communications channels pose to enterprise networks and shows potential avenues for investigating how best to defend against them.

References

[1] "[MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3," Jun 2017. [Online]. Available: https://msdn.microsoft.com/en-us/library/cc246482.aspx (Accessed 2017-09-13).

[2] R. Mudge, "Stealthy Peer-to-peer C&C over SMB pipes," Dec. 2013. [Online]. Available: https://blog.cobaltstrike.com/2013/12/06/stealthy-peer-to-peer-cc-over-smb-pipes/ (Accessed 2017-09-08).

[3] I. Ullah, "Detecting Lateral Movement Attacks through SMB using BRO," Ph.D. dissertation, University of Twente, 2016. [Online]. Available: http://essay.utwente.nl/71415/ (Accessed 2017-08-29).

[4] R. Cyrus, "Detecting malicious smb activity using bro," *SANS institute*, 2016. [Online]. Available: https://www.sans.org/reading-room/whitepapers/detection/detecting-malicious-smb-activity-bro-37472 (Accessed 2017-09-12).

[5] S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 44–57, 2007. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/4317620/ (Accessed 2017-08-29).

[6] R. W. Smith and G. S. Knight, "Predictable Design of Network-Based Covert Communication Systems." IEEE, May 2008, pp. 311–321. [Online]. Available: http://ieeexplore.ieee.org/document/4531161/ (Accessed 2017-08-29).

[7] S. Wendzel, S. Zander, B. Fechner, and C. Herdin, "Pattern-Based Survey and Categorization of Network Covert Channel Techniques," *ACM Computing Surveys*, vol. 47, no. 3, pp. 1–26, Apr. 2015. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2737799.2684195 (Accessed 2017-08-29).

[8] J. Barreto, "SMB2, a complete redesign of the main remote file protocol for Windows," Dec. 2008. [Online]. Available: https://blogs.technet.microsoft.com/josebda/2008/12/09/smb2-a-complete-redesign-of-the-main-remote-file-protocol-for-windows/ (Accessed 2017-09-14).

[9] N. Pyle, "Stop using SMB1," Sep. 2016. [Online]. Available: https://blogs.technet.microsoft.com/filecab/2016/09/16/stop-using-smb1/ (Accessed 2017-09-20).

[10] "The global ransomware attack: How to respond and what else you should know," May 2017. [Online]. Available: http://www.cbc.ca/news/technology/wannacry-ransonware-attack-1.4115239 (Accessed 2017-09-14).

[11] J. Křoustek, "Petya-based ransomware using EternalBlue to infect computers around the world," Jun. 2017. [Online]. Available: https://blog.avast.com/petya-based-ransomware-using-eternalblue-to-infect-computers-around-the-world (Accessed 2017-09-14).

[12] "Microsoft SMB Protocol and CIFS Protocol Overview (Windows)." [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa365233(v=vs.85).aspx (Accessed 2017-09-14).

[13] A. L. Donaldson, J. McHugh, and K. A. Nyberg, "Covert channels in trusted lans," in *Proceedings of the 11th National Computer Security Conference*, 1988, pp. 17–20.

[14] "[MS-WPO]: Windows Protocols Overview." [Online]. Available: https://msdn.microsoft.com/en-us/library/jj709787.aspx (Accessed 2018-03-26).

[15] Raphael Mudge, "Lateral Movement with Cobalt Strike," Sep. 2015. [Online]. Available: https://www.youtube.com/watch?v=mpsdLuJ-J_U (Accessed 2017-09-14).

[16] "w32_duqu_the_precursor_to_the_next_stuxnet_research.pdf." [Online]. Available: https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_duqu_the_precursor_to_the_next_stuxnet_research.pdf (Accessed 2018-03-26).

[17] S. L. Brand, "Dod 5200.28-std department of defense trusted computer system evaluation criteria (orange book)," *National Computer Security Center*, pp. 1–94, 1985.

[18] M. Owens, "A discussion of covert channels and steganography," *SANS institute*, vol. 1, pp. 1–18, 2002. [Online]. Available: http://www.gray-world.net/cn/papers/adiscussionofcc.pdf (Accessed 2017-08-29).

[19] J. Gardiner, M. Cova, and S. Nagaraja, "Command & Control: Understanding, Denying and Detecting-A review of malware C2 techniques, detection and defences," *arXiv preprint arXiv:1408.1136*, 2014. [Online]. Available: https://arxiv.org/abs/1408.1136 (Accessed 2017-08-29).

[20] R. A. Kemmerer, "Shared resource matrix methodology: An approach to identifying storage and timing channels," *ACM Transactions on Computer Systems (TOCS)*, vol. 1, no. 3, pp. 256–277, 1983. [Online]. Available: http://dl.acm.org/citation.cfm?id=357374 (Accessed 2017-08-29).

[21] R. Pang, M. Allman, M. Bennett, J. Lee, V. Paxson, and B. Tierney, "A first look at modern enterprise traffic," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. USENIX Association, 2005, pp. 2–2. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251088 (Accessed 2017-08-29).

[22] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010, pp. 267–280. [Online]. Available: http://dl.acm.org/citation.cfm?id=1879175 (Accessed 2017-08-29).

[23] A. De Montigny-Leboeuf, *Flow attributes for use in traffic characterization*. Communications Research Centre, 2005. [Online]. Available: https://centauri.stat.purdue.edu:98/netsecure/Papers/flowattributes_ademontigny.pdf (Accessed 2017-08-29).

[24] G. Vandenberghe, "Visual presentation of network anomalies using the network traffic explorer," Defence Research and Development Canada Ottawa (Ontario), Tech. Rep., May 2012.

[25] "ARGUS- Auditing Network Activity." [Online]. Available: http://qosient.com/argus/index.shtml (Accessed 2018-03-22).

[26] "The Bro Network Security Monitor." [Online]. Available: https://www.bro.org/ (Accessed 2018-03-22).

[27] "AfterGlow | Link Graph Visualization | Project Home." [Online]. Available: http://afterglow.sourceforge.net/ (Accessed 2018-03-22).

[28] “Graphviz - Graph Visualization Software.” [Online]. Available: https://www.graphviz.org/ (Accessed 2018-03-22).

[29] Sankethka, “PowerShell Documentation.” [Online]. Available: https://docs.microsoft.com/en-us/powershell/ (Accessed 2018-03-22).

[30] “ISO/IEC JTC1/SC22/WG21 - The C++ Standards Committee - ISOCPP.” [Online]. Available: http://www.open-std.org/jtc1/sc22/wg21/ (Accessed 2018-03-22).

[31] “API Index (Windows).” [Online]. Available: https://msdn.microsoft.com/library/windows/desktop/hh920508.aspx (Accessed 2018-03-22).

[32] “Windows Management Instrumentation (Windows).” [Online]. Available: https://msdn.microsoft.com/en-us/library/aa394582(v=vs.85).aspx (Accessed 2018-03-21).