SUDAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

COLLEGE OF GRADUATE STUDIES

كلية الدراسات العليا

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Computer and Network Engineering.

Detection of Attacks in Storage Area Networks

اكتشاف تحمل الهجمات في شبكات نطاق التخزين

Prepared By

Manhal Mohammed Mokhtar Hamed

Supervised by

Dr. Ahmed Abdallah

June-2017

الآيــــة

بسم الله الرحمن الرحيم

قال تعالى

(قَالُوا سُبْحَانَكَ لَا عِلْمَ لَنَا إِلَّا مَا عَلَّمْتَنَا ۖ إِنَّكَ أَنْتَ الْعَلِيمُ الْحَكِيمُ)

صدق الله العظيم

سورة البقرة الاية 32

DEDICATION

To my beloved parents and all of the fellows who participated

in this Inquiry - formally and informally

I dedicate this modest effort.

ACKNOWLEDGMENT

Thanks be to God, before and after, characterized the human mind and distinguished it.

This journey would not have been possible without the support of my family, professors and mentors, and friends. To my family, thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams. I am especially grateful to my parents, who supported me emotionally and financially. I always knew that you believed in me and wanted the best for me.

ABSTRACT

Storage-area networks are a popular and efficient way of building large storage systems, both in an enterprise environment and for multi-domain storage service providers, in which requires high availability, confidentiality, integrity and performance. In such frameworks, all hosts connect to storage through a network. There is more security risk than traditional storage system. Available intrusion detection systems do not apply efficiently to SANs environments due to the use of static rules and the lack of cooperation between detection modules. Furthermore, detection components may be compromised if the intruder gains access to the system. Moreover, detection is performed for the most proposed solutions at the system and network levels. The purpose of this theses is to develop a Storage based intrusion detection technique to detect lateral movement attack in using Storage area network providing a shared folder as a first test using BRO network analyzer which is an open source network security platform. Lateral movement attack is one of the phases of Advance Persistent Threat attack during which the attacker progressively moves from one system to another in the network, exploit credentials to perform pass the hash attack, escalate privileges, and finally reaching his final targets which are critical systems where key data and assets resides. Although there are many methods of performing lateral movement attack, we have evaluated our detection mechanism against two of the most common lateral movement methods: PSEXEC Windows Management Instrumentation. One of the consequences of a successful lateral movement attack can be the unauthorized access to personal and financial information of a corporate or organization.

المستخلص

شبكات منطقة التخزين هي طريقة شائعة وفعالة لبناء أنظمة تخزين كبيرة ، سواء في بيئة المؤسسات أو لمزودي خدمات التخزين متعددة النطاقات ، والتي تتطلب توفرًا عاليًا وسرية وسلامة وأداءً. في مثل هذه الأطر ، يتصل كل المضيفين بالتخزين عبر شبكة. ولكن هناك مخاطر أمنية أكثر من نظام التخزين التقليدي. لا تنطبق أنظمة كشف التطفل المتوفر بكفاءة على بيئات شبكة التخزين بسبب استخدام القواعد الثابتة وعدم التعاون بين وحدات الكشف. علاوة على ذلك ، قد يتم اختراق مكونات الكشف إذا تمكن الدخيل من الوصول إلى النظام. علاوة على ذلك ، يتم إجراء الكشف عن الحلول الأكثر شيوعًا على مستوى انظمة التشغيل أوالشبكة. الغرض من هذه الأطروحة هو تطوير تقنية الكشف عن الاقتحام الموجهه إلى نظام التخزين للكشف عن هجوم الحركة الجانبية في استخدام شبكة منطقة التخزين التي توفر مجلدًا مشتركًا كنموذج إختبار باستخدام محلل شبكة "برو" وهو نظام أساسي لأمن الشبكات مفتوح المصدر. الهجوم الجانبي هو أحد مراحل هجوم التهديد المستمر المتقدم الذي ينتقل خلاله المهاجم بشكل تدريجي من نظام إلى آخر في الشبكة ، ويستغل أوراق اعتماده لتنفيذ هجوم البعثرة ، وتصعيد الامتيازات ، والوصول أخيراً إلى أهدافه النهائية وهي أنظمة حرجة حيث توجد البيانات الرئيسية. على الرغم من وجود العديد من الطرق لتنفيذ هجوم الحركة الجانبية ، فقد قمنا بتقييم آلية الكشف لدينا ضد اثنين من أكثر أساليب الحركة الجانبية شيوعا ، يمكن أن تكون إحدى النتائج المترتبة على هجوم حركة جانبيةناجح هو الوصول غير المصرح به إلى المعلومات الشخصيةوالماليةللشركات أو المنظمةالمعنية

TABLE OF CONTENTS

TABLE OF FIGURES:

LIST OF TABLES:

LIST OF ABBREVIATIONS

ANSI	American National Standards Institute
BRO	Bro Network Analyzer
CIFS	Common Internet File System
COM	Communication port
DAS	Direct Attached Storage
ESS	Enterprise Storage Servers
FC	Fibre Channel
FCIP	Fiber Channel Internet Protocol
HBA	Host Bus Adapters
HIDS	Host-Based Intrusion Detection Systems
HMAC	message authentication code
ICT	information and communication, technology
IDPS	Intrusion detection and prevention systems
IDS	Intrusion Detection Systems
iFCP	Internet Fibre Channel Protocol
INCITS	International Committee for Information Technology Standards
IPC	Inter process communication
ISCSI	internet Small Computer Interface
iSNS	Internet Storage Name Services
JBOD	Just a Bunch of Disks
LAN	Local Area Network
MAN	metropolitan area network
MBps	megabytes per second
NAS	Network Attached Storage

NFS	Network File Systems
NIC	network interface card
NIDS	Network-Based Intrusion Detection Systems
NTLM	NT LAN Manager
RAID	of Redundant Array of Independent Disks
SAN	Storage Area Network
SCSI	Small Computer Systems Interface
SIDS	Storage-Based Intrusion Detection Systems
SMB	Server Message Block protocol
SNIA	The Storage Network Industry Association
SSA	Serial Storage Architecture
VM	Virtual Machine
WAN	Wide Area Network

CHAPTER ONE

INTRODUCTION

Chapter One

Introduction

1.1 Preface

Due to the explosion of internet and the e-commerce, a tremendous amount of data has been created and made available to users. In addition to this, new type of data such as images, audio and video have been stored and integrated with applications and databases, further accelerating the demand for storage capacity.[1] This data must also be capable of being accurately accessed and processed to generate more information. The demand for storage devices and their economical worth has reached new heights. To address this huge requirement, constant advances in storage technology are the need of the hour.

Many technologies have been developed to manage and handle this traffic of data for use in different scales of networks such as LAN, MAN and WAN. Some examples of these technologies include Network Attach Storage (NAS), Direct Attach Storage (DAS) and Storage Area Network (SAN).

Storage Area Networks (SAN) is one such technology that emerged to quench this need to store, manage, process and access data efficiently and securely. A Storage area network, or SAN, maybe defined as a high-speed network of storage devices. Also, these storage devices are connected to the servers. Applications running on any of the networked servers can access and use this storage. Due to its versatile functionality and their apparent property of being able to relieve overburdened LANs from high volumes of data, SANs have become overwhelmingly popular in the global market. They reduce administrative and equipment costs, provide high data availability and ensure regular backups. However, companies and its customers need to ensure the safety and security of the information that is being routed through the storage area network. Today, most of the data handled by large computing companies are managed on SANs. [5] Hence, the Security has always been highest priority in such networks for network administrators, working with information and sensitive data of their companies.

With the development of computer and network utilization, intrusion detection became the hot topic in the field of security study of computer. Intrusion detection systems (IDS) have been developed over the years, mainly two types: host-based and network-based. Network-based IDS (NIDS) are usually embedded in network devices such as sniffers or firewall to monitor network traffic. Through analyzing the content of the text and feature of the traffic, they watch for signs of attack activities. Host-based IDS (HIDS) are usually embedded in the host's OS. Through scanning and analyzing the local audit logs, system logs, and features of processes' activity and users' activity, they watch for signs of attack activities. In the network storage system, the storage system is independent of the host, and has high processing power and memory space. And these features make it possible that the IDS can be embedded in the storage system, many attack activities would lead to read or write the storage, such as manipulating system utilities (e.g. to add backdoors), tampering with audit log contents (e.g. to eliminate evidence), and resetting attributes (e.g. to hide changes). If there is an IDS embedded in the storage system, when such an intruder comes, it will be found. Storage-based IDS have some individual vantage than other-based IDS. The first, the storage system being independent of the host OS, leads to the independence of the SIDS, which means that the IDS and the storage system cannot be disabled by an intruder even though the intruder has successfully compromised the host' OS. It can be said that the SIDS is the vantage point for intrusion detection. The second, because there are very few people who master storage technology, the chance of breaking through the storage system is rare. This makes the SIDS very strong. [2]

1.2 Problem Statement

Storage-area networks are a popular and efficient way of building large storage systems both in an enterprise environment and for multi-domain storage service providers in which requires high availability, confidentiality, integrity and performance.

In such frameworks, all hosts connect to storage through a network. There is more security risk than traditional storage system. Available intrusion detection systems do not apply efficiently to SANs environments due to the use of static rules and the lack of cooperation between detection modules. Furthermore, detection components may be compromised if the intruder gains access to the system. Moreover, detection is performed for the most proposed solutions at the system and network levels

1.3 Proposed Solution

The proposed solution to detect attacks is based on a set of intrusion detection and tolerance components that are added to SAN system by:

A. the management of two areas (virtual area and protected area) at each storage node;

B. the cooperation of detection modules running on each SAN component; and

C. The use of distributed set of rules and scripts that are updated and managed in a secure manner.

1.4 Aim and Objectives

- To make the detection capabilities protected against intruder activities since they are performed by compromise independent components.
- To protect data held by this system by dividing the disk into two areas and granting the management of the protected area to only the disk side components.
- To enhance detection by cooperating three levels of collected data (network, host and storage levels) and dynamically updating detection rules in all the SAN system.
- To tolerates attacks in order to collect information about malicious activity for postmortem investigation.

1.5 Methodology

Storage-based intrusion detection consists of storage systems watching for and identifying data access patterns characteristic of system intrusions to detect lateral movement attack through SMB by Implementation of the open source BRO analyzer in storage area network environment.

The goal is to find out intrusion activity and attacks like ISCSI attacks or detect lateral movement attack through SMB presents in packet with the help of BRO analyzer. BRO analyzer also summarizes the intensive Storage IDS alerts by sending summary reports to the administrator of the System. In which we will use the virtualization environment to implement the storage-based IDS which is connected to each virtual network as well as Open-filer software to provide storage for VMs and by using Kali system to perform different attacks through the storage network.

1.6 Research Outlines

In general, the thesis will be divided into five chapters. Each chapter will discuss on different issues related to the project. The following are the issues discussed.

Chapter Two will describe the background required to understand the functionality of SAN and other technologies and protocols as well as intrusion detection and prevention systems, as well as the related work in intrusion detection field especially for SANs is introduced. It also introduces the SAN concept and the architecture of SAN systems. Also, a formal model for transactions in SANs is given. And chapter Three; introduces the proposed detection and tolerance system to detect lateral movement attack through SMB in the Storage area network. Chapter Four; Describes and discusses the results obtained from the system. And last chapter Chapter Five will outlines the conclusions drawn from this research and identifies possible extensions of this project.

CHAPTER TWO

LITERATURE REVIEW

CHAPTER TWO

Literature Review

2.1 Introduction

With advancement of information and communication, technology (ICT) the amount of data that needs to be transferred and stored on disks has grown enormously in a computer network environment and growth from Gigabyte in early 1990 to Exabyte in 2010. Many technologies have been developed to manage and handle this traffic of data for use in different scales of networks such as LAN, MAN and WAN. Some examples of these technologies include Network Attach Storage (NAS), Direct Attach Storage (DAS) and Storage Area Network (SAN). [3]

Storage devices were up to recently locked into a glass room and hence was the data stored on them enjoying privileges of the physical data center security and protection mechanisms. With a development of a Storage Area Network (SAN) technology, hard drives and tape drives are not necessarily directly attached to a host any more but could be rather physically distant up to several hundred kilometers or even around a globe. Such a flexibility of logically instead of physically attached storage devices to a host made them remotely accessible and highly available, however it brought into a consideration all security elements of the modern network environment like privacy, integrity of the data in transit and authentication of the remotely connected devices. From the data perspective, we could distinguish the storage network security, which refers to protection of the data while it is in transit versus storage data security to which we refer when the data is stored on the tapes or the hard drives.[4]

2.2 SAN technology overview

2.2.1 DAS vs. NAS vs. SAN

The Storage Network Industry Association (SNIA) defines SAN as a network in which the main purpose is to transfer data between servers and storages. The network consists of several computers, servers and devices that are interconnected with each other; this infrastructure allows different computers to communicate with each other. The operation of each SAN consists of basic elements for communication, which manages the physical connections, management layers for organizing the available connections, computer system and storage devices for reliable and secure handling of data. SAN manage the data at the block level and thus not at the file level for keeping track of and allocating free space on disk to the data. SANs are used to make a high speed connection between storages and servers.[3]

Historically, storage devices, such as disk drives and backup tapes, were directly attached to a host, hence the name Direct Attached Storage or DAS. This was performed via SCSI (Small Computer Systems Interface) parallel bus interface with a speed of up to 320 MBps. This approach of attaching storage devices is coming from internal computer architecture, which has obviously got to its limits in several ways. Number of devices, which could be attached to one bus, is limited even in latest version of SCSI protocol to only 16 devices while the distances are not bigger than 15 meters. Sharing disk or tapes drives amongst multiple hosts were due to architecture of DAS impossible or required specialized and typically expensive software or controllers for device sharing. On the other side, utilization of the storage spread across the multiple servers was typically lower than on one single pool. Often necessary expansions of storage volumes and replacement of the failed hard drives have in DAS architecture frequently generated system downtimes. DAS Architecture is illustrated in Figure 2.1 .[4]

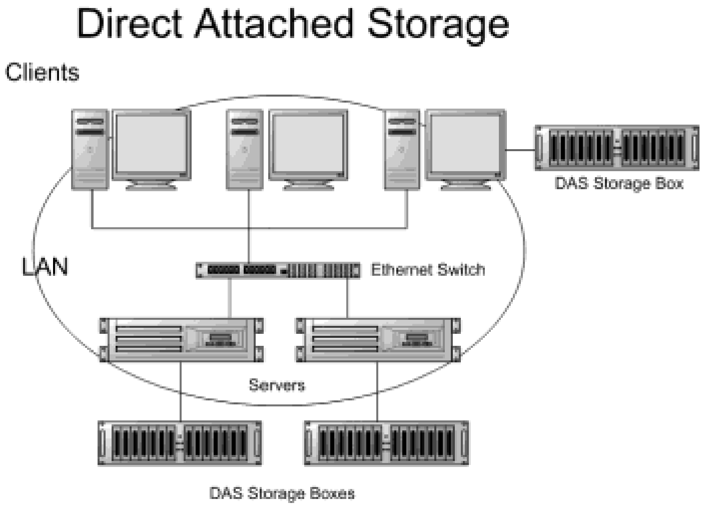


Figure 1.1: DAS Architecture.

The effort to get a better usage of storage devices by the multiple hosts has generated specialized devices for shared storage access on the file level. This architecture is commonly referred as Network Attached SAN Storage or shortly NAS. NAS architecture consist of a dedicated device named Filer which is actually a stripped down and optimized host for very fast network file sharing. Two most typically supported file systems on Filers are NFS (Network File Systems) for a Unix world and CIFS (Common Internet File System) for the Microsoft world. While NAS solution has its main advantage in simplicity in maintenance and installation, its main drawback is limited file and operating system support or support of future new file systems. Architecture of a NAS illustrated in Figure 2.2.

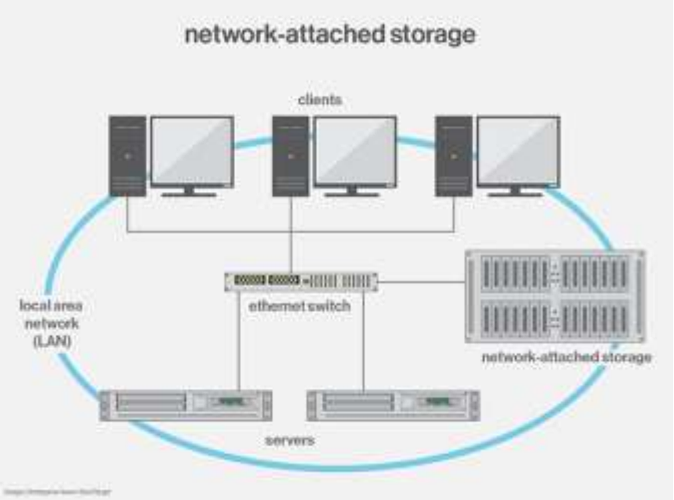


Figure 2. 2: NAS Architecture

The latest mechanism of attaching storage remotely with a block level access is commonly referred as Storage Area Network or SAN. SAN consist of hosts, switches and storage devices. Hosts equipped with Host Bus Adapters (HBA) are attached via optical cable to a storage switches which act as a fabric between the hosts and the storage devices. SAN architecture is illustrated in Figure 2. 3.[4]

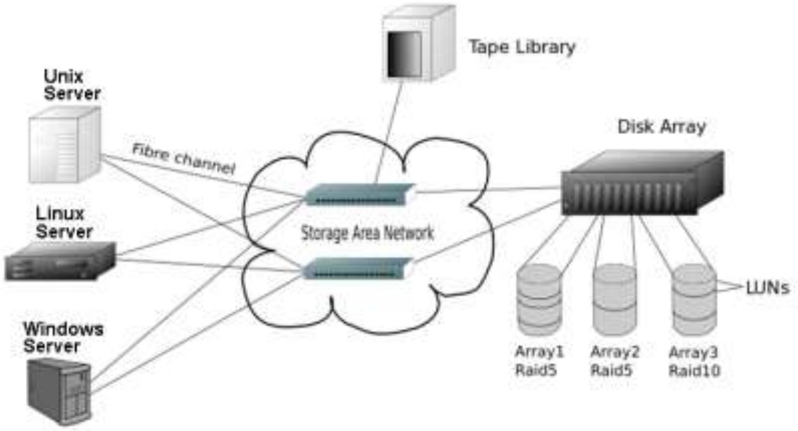


Figure 2. 3: SAN architecture

2.2.2 Storage Area Network objectives

The main objectives that make SAN a popular solution for storage networks are: disk utilization, disaster recovery methods, availability of data and fast backup data ability. SAN help users to use disk resources in a more efficient way, since all the disks in SAN are kept together as one resource so the management of disks become easier and disks can work better and more utilized, resulting in less waste of free space. One can therefor save power and increase the performance of the system. SANs are capable of adding or removing new disks for expanding the free space to servers and applications, whenever an application need more space, it is thus easier to make free space available without turning servers down or power them off to allocate free space to applications. SAN has good disaster recovery method; by mirroring the data to another disk that located in another place and used different types of Redundant Array of Independent Disks (RAID) to provide mirroring and data duplication, SAN improve the communication I/ O by using fiber optic cables and gigabit Ethernet LAN also reduce the physical space that need for keeping storage devices and servers, because SAN handle the data management with lower number of servers and higher number of disks. SAN components consist of basic elements such as connectivity part that typically is fibre optic in FC and fast or gigabit Ethernet for iSCSI, hubs, switches, directors, connectors and routers are themain components of SAN. Components can be from different storage devices e.g. tape, Just a Bunch of Disks (JBOD), Enterprise Storage Servers (ESS), Serial Storage Architecture (SSA) and IBM DS family storages. Different servers in SAN can use different operating systems such as Windows, UNIX and LINUX. By help of different communication techniques and communication protocols such as iSCSI and Fiber Channel Internet Protocol (FCIP) SAN allows the storage management over long distances with high speeds in centralized and efficient way. Traditional storage devices work with SCSI connectors for making communication with host, that makes the connection length limited to 25 meters but SAN with using fiber optic technology overcame to this limitation and extended it up to 10 kilometers and increased the number of connections that was 16 in SCSI to unlimited for FC.[3]

2.3 Storage Area Network protocols

A protocol is defined by a set of rules that enables the communication between two computers in any networks, communication between two devices from different vendors become capable by using protocol, because the protocol acts as the translator that all the devices talk to each other with the same language. There are several protocols for implementation of SAN, the common are internet Small Computer Interface (iSCSI) and Fibre Channel (FC).[3]

2.3.1 Small Computer Systems Interface known as SCSI

In the long history of adaptations and improvements, the line sometimes blurs between where one Small Computer System Interface (SCSI) ends and another begins. The original SCSI standard approved in 1986 by the American National Standards Institute (ANSI), supported transfer rates of up to 5 MBps (megabytes per second) which is, measured by today's standards, slow. Worse yet, it supported a very short bus length. When original SCSI was introduced, however, it represented a significant improvement over what was available at that time, but the problem was the compatibility - since many vendors offered their own unique SCSI options. The next generation of SCSI standard SCSI-2, incorporated SCSI-1 as its subset. In development since 1986, SCSI-2 gained its final approval in 1994 and resolved many of the compatibility issues original SCSI-1 faced. With SCSI-2, it was possible to construct more complex configurations using a mix of peripherals. The most noticeable benefit of SCSI-2 over SCSI-1 was its speed. Also called Fast SCSI,

SCSI-2 typically supported bus speeds up to 10 MBps but could go up to 20 MBps when combined with fast and wide SCSI connectors. Fast SCSI enabled faster timing on the bus (from 5 to 10 MHz), thereby providing for higher speed. Wide SCSI used an extra cable to send data that's 16 or 32 bits wide, which allowed for double or quadruple the speed over the bus versus standard, narrow SCSI interfaces that were only 8 bits wide. The latest specification of SCSI protocol, SCSI-3 was among other improvements the first one that did a separation of the higher-level SCSI protocol from the physical layer. This was the prerequisite of giving alternatives to run SCSI commands on top of different physical layers than the parallel bus. Hence the SCSI-3 specification was the basis of porting the SCSI protocol to different media carriers such as Fibre Channel or even other transport protocols as TCP/IP[3]

2.3.2 Internet SCSI

The SCSI-3 protocol has been mapped over various transports such as parallel SCSI, IEEE-1394 (firewire) and Fibre Channel. All these transports have their specifics but also all have limited distance capabilities. The Internet SCSI or shortly iSCSI protocol is the IETF draft standard protocol that describes means of transporting SCSI packets over TCP/IP. The iSCSI interoperable solution can take advantage of existing IP network infrastructure which have virtually no distance limitations. Encapsulation of the SCSI frames in the TCP/IP protocol is illustrated in Figure 2. 4.[4]

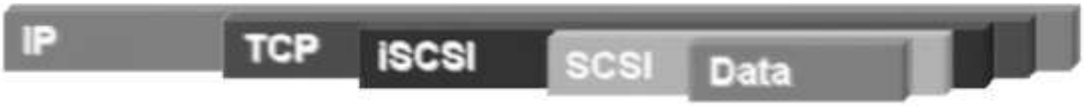


Figure 2. 4: iSCSI Encapsulation.

The primary market driver for the development of the iSCSI protocol was to enable broader access of the large installed base of DAS over IP network infrastructures. By allowing greater access to DAS devices over IP networks, storage resources can be maximized by any number of users or utilized by a variety of applications such as remote backup, disaster recovery, and storage virtualization. A secondary driver of iSCSI is to allow other SAN architectures such as Fibre Channel to be accessed from a wide variety of hosts across IP networks. iSCSI enables block-level storage to be accessed from Fibre Channel SANs using IP storage routers or switches, furthering its applicability as an IP-based storage transport protocol. iSCSI defines the rules and processes to transmit and receive block storage applications over TCP/IP networks. Although iSCSI can be supported over any physical media that supports TCP/IP as a transport, most iSCSI implementations runs on Gigabit Ethernet. iSCSI protocol can run in software over a standard Gigabit Ethernet network interface card (NIC) or can be optimized in hardware for better performance on an iSCSI host bus adapter (HBA). iSCSI enables SCSI-3 commands to be encapsulated in TCP/IP packets and delivered reliably over IP networks. As it sits above the physical and data-link layers, iSCSI interfaces to the operating system's standard SCSI access method command set to enable the access of block-level storage that resides on Fibre Channel SANs over an IP network via iSCSI-to-Fibre Channel gateways such as storage routers and switches. iSCSI protocol stack building blocks are illustrated in Figure 2. 5.

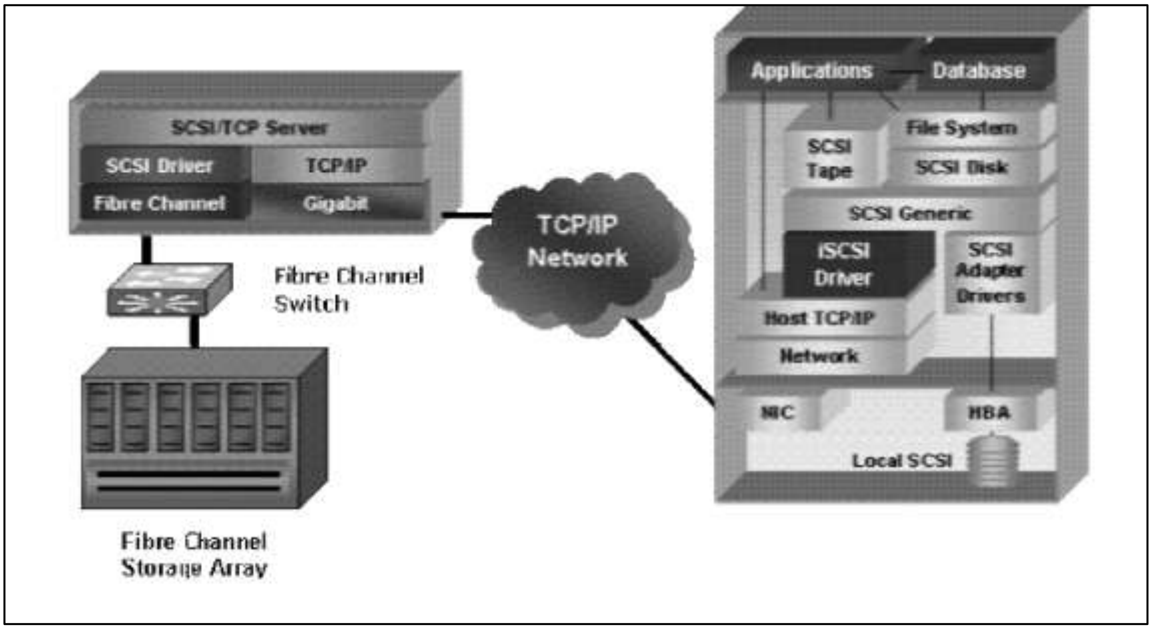


Figure 2. 5: iSCSI Solution Architecture

Initial iSCSI deployments were targeted at small to medium-sized businesses and departments or branch offices of larger enterprises that have not deployed Fibre Channel SANs yet, however iSCSI is also an affordable way to create IP SANs from a number of local or remote DAS devices. If there is Fibre Channel present, as it is typically in a data center, it could be also accessed by the iSCSI SANs via an iSCSI-to-Fibre Channel storage routers and switches.

2.3.3 Fiber Channel protocol (FC)

Fibre Channel (FC) is an open industry standard serial interface for high-speed systems. FC is a protocol for transferring the data over fibber cable that consists of multiple layers covering different functions. As a protocol between the host and a storage device, FC was out of a scope of an average information technology professional for a simple reason that it was point to point connection between the host with a HBA and storage device of typically same vendor which did not require any knowledge or understanding except maybe during the installation process. From the speed perspective, FC is available already in flavors of 1 Gbps and 2 Gbps while specifications for 4Gbps as well as 10Gbps are being worked on and are not that far away.

FC becomes a popular model of SAN because of the limitations of the previous technologies. FC overcame the limitations of the I/ O speed, flexibility and Distance limit of traditional protocols, in SAN all hosts can see the storage like local attach disks to the system, multi-protocol support is another advantage of SAN. FC has two types of cables for using shorter and longer distances, fibre optic cable can manage the connectivity for the longer distances and copper cable is used for shorter distances and the characteristics of FC make it compatible with a wide variety of devices that support FC.

FC protocol stack is defined in a standard specification of a Technical Committee T11.3 of an INCITS (InterNational Committee for Information Technology Standards) and is illustrated in Figure 2. 6.[3, 4]

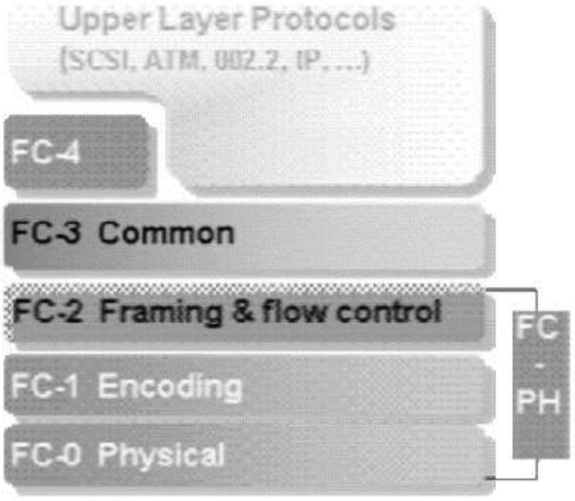


Figure 2. 6: Fibre Channel Protocol Stack

The lowest level (FC-0) defines the physical link in the system, including the fibre, connectors, optical and electrical parameters for a variety of data rates. FC-1 defines the transmission protocol including serial encoding and decoding rules, special characters and error control.

The Signaling Protocol (FC-2) level serves as the transport mechanism of Fibre Channel. It defines the framing rules of the data to be transferred between ports, mechanisms for controlling the different service classes and the means of managing the sequence of a data transfer.

The FC-3 level of the FC standard is intended to provide the common services required for advanced features such as:

- Striping -To multiply bandwidth using multiple ports in parallel to transmit a single information unit across multiple links.

- Hunt groups - The ability for more than one port to respond to the same alias address. This improves efficiency by decreasing the chance of reaching a busy port.
- Multicast

FC-3 Layer is the one initially thought to be also used for encryption or compression services, however latest development have put these services to the Layer 2 of a FC architecture as it will be described later.

FC-4, the highest level in the FC structure defines the application interfaces that can execute over Fibre Channel. It specifies the mapping rules of upper layer protocols such as SCSI, ATM, 802.2 or IP using the FC levels below.

2.3.4 Fiber Channel over TCP/IP

Fiber Channel Over TCP/IP (FCIP) protocol is described in the IETF draft standard as the mechanisms that allow the interconnection of islands of Fibre Channel storage area networks over IP-based networks to form a unified storage area network in a single Fibre Channel fabric. Encapsulation of the FC frames which are carrying SCSI frames on top of the TCP is illustrated in Figure 2. 7.[1, 4]

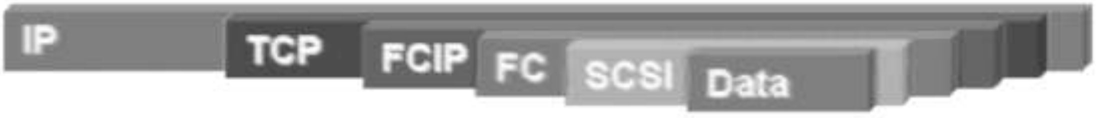


Figure 2. 7: FCIP Encapsulation.

2.3.5 Other SAN Protocols

There are several other SAN protocols which are in IETF draft proposal or development like Internet Fibre Channel Protocol (iFCP) or Internet Storage Name Services (iSNS). iFCP is also a gateway-togateway approach in which FC frames are encapsulated directly into IP packets and IP addresses are mapped to a FC device. This is more iP-oriented scheme than the IP tunneled SCSI frames FCIP, but is a more complex protocol that was designed to overcome the potential vulnerabilities of stretched fabrics, enable multi-point deployments and provide native IP addressing to individual Fibre Channel transactions. [4]

iSNS protocol is used for interaction between iSNS servers and iSNS clients in order to facilitate automated discovery, management, and configuration of iSCSI and FC devices on a TCP/IP network. iSNS provides intelligent storage discovery and management services comparable to those found in FC networks, allowing a commodity IP network to function in a similar capacity as a storage area network. iSNS also facilitates a seamless integration of IP and FC networks, due to its ability to emulate FC fabric services, and manage both iSCSI and Fibre Channel devices. iSNS thereby provides value in any storage network comprised of iSCSI devices, Fibre Channel devices (using iFCP gateways), or any combination thereof. iFCP requires iSNS for discovery and management, while iSCSI may use iSNS for discovery, and FCIP does not use iSNS.[3, 4]

2.4 SAN Security Threats Analysis

Security is a key source of a wide acceptance when it comes to SAN technologies. According to numerous market surveys, the main reason why most enterprises have not yet deployed SANs been due to security concern. When SAN technology was introduced, security was routinely ignored. This was partly because the largely unknown Fibre Channel protocol used for communication was not a big target for attackers and mainly because security simply wasn't a priority. Today, when SANs are starting to reach across the country or even around the globe, storing and transferring terabytes of sensitive and confidential data, may quickly draw the attention of potential attackers. When the underlying protocol carrying the data over long distance and out of the glass room does not provide the essential data protecting mechanism, data in transit is exposed to a threat of being stolen, seen by the unintended party, modified or simple being not available when it is needed. Logical instead of physical attachment of the storage devices also opens issues of the access control and an authentication of the remote nodes exchanging the data. Moving SAN communications to IP-based networks makes it even more exposed and vulnerable to many of the attacks made on corporate networks[4]

2.5 Intrusion detection and prevention systems IDPS

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. Intrusion prevention is the process of performing intrusion detection and attempting to stop detected possible incidents. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, attempting to stop them, and reporting them to security administrators. In addition, organizations use IDPSs for other purposes, such as identifying problems with security policies, documenting existing threats, and deterring individuals from violating security policies. IDPSs have become a necessary addition to the security infrastructure of nearly every organization.

There are many types of IDPS technologies, which are differentiated primarily by the types of events that they can recognize and the methodologies that they use to identify possible incidents. This publication discusses the following four types of IDPS technologies:

- **Network-Based**, which monitors network traffic for particular network segments or devices and analyzes the network and application protocol activity to identify suspicious activity.

- **Wireless**, which monitors wireless network traffic and analyzes it to identify suspicious activity involving the wireless networking protocols themselves.

- **Network Behavior Analysis (NBA)**, which examines network traffic to identify threats that generate unusual traffic flows, such as DDoS attacks, scanning, and certain forms of malware.

- **Host-Based**, which monitors the characteristics of a single host and the events occurring within that host for suspicious activity.

IDPSs typically record information related to observed events, notify security administrators of important observed events, and produce reports. Many IDPSs can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g., reconfiguring a firewall), or changing the attack's content.

IDPSs cannot provide completely accurate detection; they all generate false positives (incorrectly identifying benign activity as malicious) and false negatives (failing to identify malicious activity). Many organizations choose to tune IDPSs so that false negatives are decreased, and false positives increased, which necessitates additional analysis resources to differentiate false positives from true malicious events. Most IDPSs also offer features that compensate for the use of common evasion techniques, which modify the format or timing of malicious activity to alter its appearance but not its effect, to attempt to avoid detection by IDPSs.

Most IDPSs use multiple detection methodologies, either separately or integrated, to provide more broad and accurate detection. The primary classes of detection methodologies are as follows:

> **Signature-based**, which compares known threat signatures to observed events to identify incidents. This is very effective at detecting known threats but largely ineffective at detecting unknown threats and many variants on known threats. Signature-based detection cannot track and understand the state of complex communications, so it cannot detect most attacks that comprise multiple events.

Anomaly-based detection, which compares definitions of what activity is considered normal against observed events to identify significant deviations. This method uses profiles that are developed by monitoring the characteristics of typical activity over a period of time. The IDPS then compares the characteristics of current activity to thresholds related to the profile. Anomaly-based detection methods can be very effective at detecting previously unknown threats. Common problems with anomaly-based detection are inadvertently including malicious activity within a profile, establishing profiles that are not sufficiently complex to reflect real-world computing activity, and generating many false positives.

Stateful protocol analysis, which compares predetermined profiles of generally accepted definitions of benign protocol activity for each protocol state against observed events to identify deviations. Unlike anomaly-based detection, which uses host or network-specific profiles, stateful protocol analysis relies on vendor-developed universal profiles that specify how particular protocols should and should not be used. It is capable of understanding and tracking the state of protocols that have a notion of state, which allows it to detect many attacks that other methods cannot. Problems with stateful protocol analysis include that it is often very difficult or impossible to develop completely accurate models of protocols, it is very resource-intensive, and it cannot detect attacks that do not violate the characteristics of generally acceptable protocol behavior.[5]

2.5.1 Intrusion Detection and prevention systems “IDPS” Components and Architecture

The typical components in an IDPS solution are as follows:

Sensor or Agent. Sensors and agents monitor and analyze activity. The term sensor is typically used for IDPSs that monitor networks, including network-based, wireless, and network behavior analysis technologies. The term agent is typically used for host-based IDPS technologies.

Management Server. A management server is a centralized device that receives information from the sensors or agents and manages them. Some management servers perform analysis on the event information that the sensors or agents provide and can identify events that the individual sensors or agents cannot. Matching event information from multiple sensors or agents, such as finding events triggered by the same IP address, is known as correlation. Management servers are available as both appliance and software-only products. Some small IDPS deployments do not use any management servers, but most IDPS deployments do. In larger IDPS deployments, there are often multiple management servers, and in some cases, there are two tiers of management servers.

Database Server. A database server is a repository for event information recorded by sensors, agents, and/or management servers. Many IDPSs provide support for database servers.

Console. A console is a program that provides an interface for the IDPS's users and administrators. Console software is typically installed onto standard desktop or laptop computers. Some consoles are used for IDPS administration only, such as configuring sensors or agents and applying software updates, while other consoles are used strictly for monitoring and analysis. Some IDPS consoles provide both administration and monitoring capabilities.

2.6 Preliminary Knowledge

2.7.1 Server Message Block Protocol

Server Message Block protocol (SMB) is a client-server and request-response protocol in a computer network. SMB protocol can be used on the top of protocols like TCP/IP, IPX/SPX or other network protocols. It is installed in almost all Microsoft Windows machines. Through SMB protocol clients can access files that are present on the server. Based on the file access control, the client can create, read and update the files on the server. Including file system support, SMB protocol also specializes in Inter process communication (IPC). IPC share is useful because it facilitates data exchange between computers over SMB protocol. SMB protocol is evolving and continuously updating protocol. It evolved from CIFS to SMBv1 to SMBv2 to SMBv3. SMB protocol is remote sharing/file protocol for accessing files and printers across the LAN and WAN. Many Operating systems vendors like Apple, EMC, Microsoft, and Linux have implemented SMB protocol. With the passage of time Windows has made enormous improvements to the protocols such as adding Kerberos authentication, Signing using HMAC MD5 and many other improvements. SMBv2 is the first upgraded version of SMB. Compare to SMB, SMBv2 has increased file-sharing scalability, security, number of round trips of request is reduced, asynchronous operations, and the larger reads and writes (more data in each packet. SMB protocol provides two level of security that is user level and share level. Share is a file or printer that can be accessed by client. In user level authentication the client provides username and password to access a share. In share level authentication each share is protected individually. So, the client has to provide password for each share. The password is encrypted in both the cases. Some of SMB supported authentication protocols are Windows.NET Challenge/Response NTLM, NTLMv2, KERBEROS, W2K and NT Domain Authentication. NTLMv2 authentication is based on challenge response, which contains nonce from a client and nonce from server[6]

Figure 2.8 shows how SMB client and server initiates communication. Client sends SMB C0M Negotiate to server to request negotiation of SMB protocol dialect. In this message the client includes his dialects (max buffer size, canonical file names, etc). In response to the client request the Server identify SMB protocol dialects for the session and also includes 8-byte random string used in the next to authenticate client. Client then sends SMB COM Session Setup ANDX message to identify his

capabilities. The client also sends username, domain name, or password hash; the server supports both plaintext password as well as password hash. In response to this message, if the server accepts challenge/response, Server will issue a valid UID to the client for the session else the server will deny the client access request. The issued UID is submitted with all subsequent SMBs on that connection to the server. If the client is granted access, the client sends SMB COM Tree Connect ANDX message to request access to the share (e.g. IPC$, Admin$, C$) and fully specifying the path of the share. Based on the client credentials if the client is allowed to access the requested share, the server returns 16-bit tree ID (TID) else the server responds with error message and deny access to the request share. With SMB COM Open ANDX message the client request the server to open a file on the accessed share. If access to the requested file is granted, in response the server returns file ID of the requested file. In SMB COM Read ANDX message the client includes the file ID issued to the client in the previous message to request the server to read data from the previously opened file and return its data to the client. In response to this request the server return the requests file data.

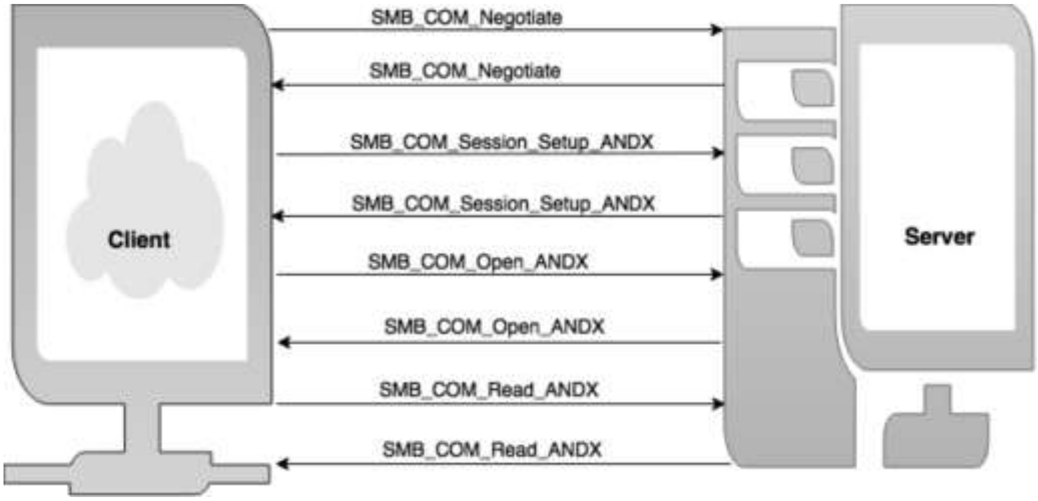


Figure 2.8 Microsoft SMB Protocol Packet Exchange Scenario

2.7.2 BRO Network Analyzer

BRO is a stand-alone open source network traffic analyzing system . This stand-alone system has the capability to monitor traffic directly and passively using packet filters. It monitors incoming, outgoing as well as internal traffic to trace suspicious traffic or traffic that violates policies. To avoid any intrusion; BRO monitors the traffic and raises alarm as soon as it sees unusual (malicious) traffic. BRO performs the detection of malicious activities in real time. BRO transforms traffic into high-level events, based on the policies and configurations alarm is raises for malicious traffic. BRO can be used to analyze real-time flow or pre-recorded flow and packets (pcaps), to extract files from network traffic streams, as an intrusion detection system by enforcing policy, and to generate statistics about network traffic patterns and usage [5]. BRO supports both signature-based and anomaly-based detection. In this work we research a solution for lateral movement that can be used by BRO. An advantage of anomaly-based intrusion detection system is to detect new attacks. We have chosen BRO because it is well-known open-source tool, that is widely used by the entire security community.

Bro architecture as shown in Figure 2.9 comprises of: the libpcap libraries, an event engine and a policy script interpreter. Below is the brief description of BRO architecture.

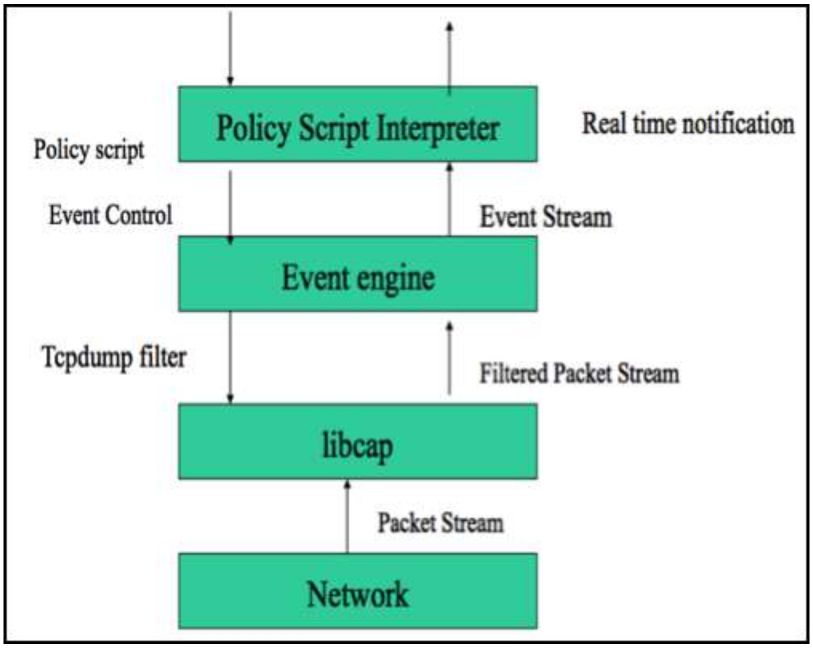


Figure 2.9 BRO Architecture

2.7 Related Works

Many intrusion detection systems (IDSs) have been developed over the years, with most falling into one of two categories: network-based or host-based. Network IDSs (NIDS) are usually embedded in sniffers or firewalls, scanning traffic to, from, and within a network environment for attack signatures and suspicious traffic. Host-based IDSs (HIDS) are fully or partially embedded within each host's OS. They examine local information for signs of intrusion or suspicious behavior. Many environments employ multiple IDSs, each watching activity from its own vantage point.

The storage system is another interesting vantage point for intrusion detection. Several common intruder actions are quite visible at the storage interface. Examples include manipulating system utilities (e.g., to add backdoors or Trojan horses), tampering with audit log contents (e.g., to eliminate evidence), and resetting attributes (e.g., to hide changes). By design, a storage server sees all changes to persistent data, allowing it to transparently watch for suspicious changes and issue alerts about the corresponding client systems. Also, like a NIDS, a storage IDS must be compromise-independent of the host.[7]

For emphasis, we note that there have been many intrusion detection systems focused on host OS activity and network communication. Axelsson [1998] surveyed and laid out classifications for many of these IDS types. Also, the most closely related tool, Tripwire [Kim and Spafford 1994], was used as an initial template for our prototype’s file modification detection rule set. Our work is part of a line of research exploiting physical [Ganger and Nagle 2001; Zhang et al. 2002] and virtual [Chen and Noble 2001; Payne et al. 2007] protection boundaries to detect intrusions into system software. Notably, Garfinkel and Rosenblum [2003] explore the utility of an IDS embedded in a Virtual Machine Monitor (VMM), which can inspect machine state while being compromise-independent of most host software. Storage-based intrusion detection rules could be embedded in a VMM’s storage module, rather than in a physical storage device, to identify suspicious storage activity. After our initial explorations of the field of storage-based intrusion detection [Pennington et al. 2003; Griffin 2004], other researchers pursued complementary projects that advanced the field and demonstrated the versatility of active monitoring behind the storage interface. Banikazemi et al. at IBM Research explored the commercial viability of storage-based intrusion detection [Banikazemi et al. 2005]. First, they extended our IDD concept beyond a single disk and into a managed storage area network, demonstrating a concrete realization of a real-time block-based storage device inside a commercially viable storage platform. Second, they observed the utility of using delayed execution an IDS rule-set over file system snapshots as an efficient complement to real-time IDS activity. Paul et al. at the University of Virginia explored the architectural issues that will be faced in stand-alone semantically smart disk systems [Paul 2008; Paul et al. 2005]. They identified observable disk-level behaviors that are characteristic of malware and explored disk detection algorithms tuned to operate in the limited-resource embedded computational environments that will likely be characteristic of programmable storage devices. Butler et al. at the Pennsylvania State University introduced the idea of using a removable administrative token to perform safe programming and administration of a storage-based IDS [Butler et al. 2008]. They identified an elegant empirical alternative to selecting which blocks should be treated as immutable by the IDS rule-set: with some exceptions, all blocks written to storage whenever the administrative token is present are considered immutable. The results from these three independent projects collectively underscore our claims of the feasibility and efficacy of locating independent security monitoring and response utilities behind the storage interface. Somewhere between block-based storage and file-based storage lies the emerging concept of object-based storage [Gibson et al. 1998; Weber 2004]. Storage-based intrusion detection is easier for storage objects than for blocks, since files often map directly to one or more objects. One such system has been demonstrated by Zhang and Wang [2006] who created a storage-based IDS running on an early object-based storage implementation. Perhaps the most closely related work is the original proposal for self-securing storage [Strunk et al. 2000], which argued for storage-embedded support for intrusion survival. Self-securing storage retains every version of all data and a log of all requests for a period of time called the detection window. For intrusions detected within this window, security administrators have a wealth of information for post intrusion diagnosis and recovery. Such versioning and auditing complements storage-based intrusion detection in several additional ways.

First, when creating rules about storage activity for use in detection, administrators can use the latest audit log and version history to test new rules for false alarms.

Second, the audit log could simplify implementation of rules looking for patterns of requests.

Third, administrator scan uses the history to investigate alerts of suspicious behavior (i.e., to check for supporting evidence within the history).

Fourth, since the history is retained, a storage IDS can delay checks until the device is idle, allowing the device to avoid performance penalties for expensive checks by accepting a potentially longer detection latency.[1, 7]

According to the best of our knowledge so far there is no standard evaluation technique or model to implement BRO to detect lateral movement attack through SMB in Storage area networks

CHAPTER THREE

METHODOLOGY

Chapter Three

Methodology

As shown in Figure 3.1, this chapter is consisting of four major topics which describes how the lab and testing environment are made to reach the final results:

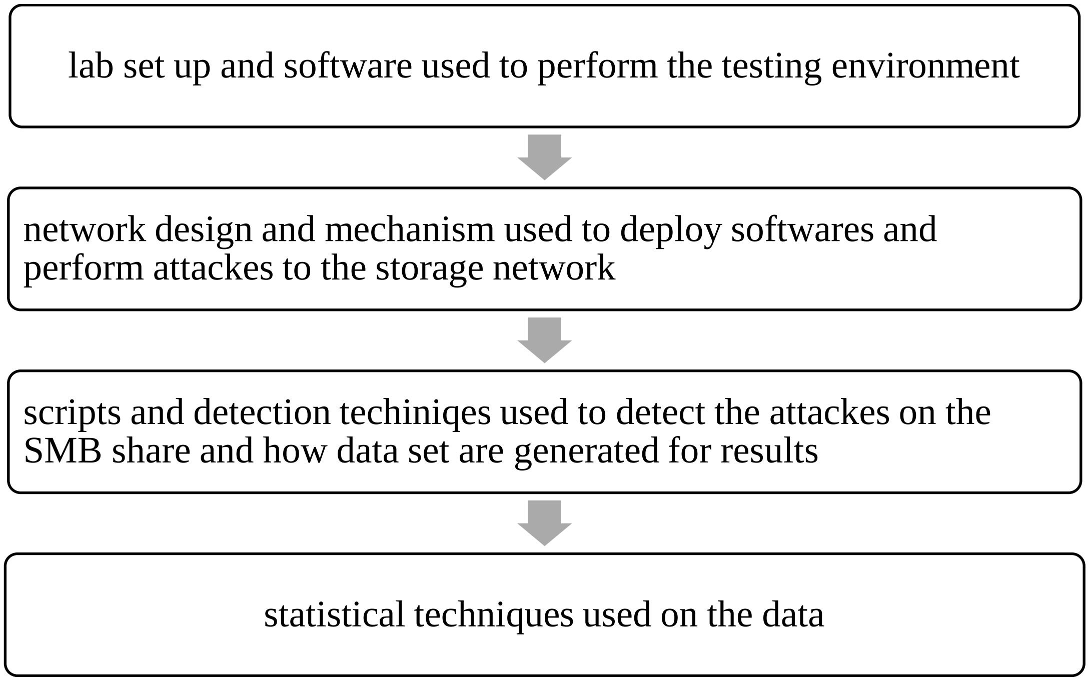

lab set up and software used to perform the testing environment
network design and mechanism used to deploy softwares and perform attackes to the storage network
scripts and detection techiniqes used to detect the attackes on the SMB share and how data set are generated for results
statistical techniques used on the data

Figure 3.1 lab and testing environment description

3.1 Network Design

The examples, data, and traffic discussed in this research were generated within the test environment shown in Figure 3.2 and table 3.1 below. VMWare workstation hosts the following virtual machines, which replicates a corporate environment: SBID proposed system (Centos 7 host with installed BRO IDS integrated with kitana log for GUI), Windows Sever 2012, Window 7 (Staff machine), Windows 7 (Staff machine) and Open-Filer storage to represent the SAN storage network.

The server running Windows Server 2012 is the domain controller for both Windows 7 systems, which are part of a domain named “WORKGROUP” and connected to the shared storage provided by OpenFiler storage through the network.

The SBIDS system is monitoring all ingress and egress traffic from the systems on the WORKGROUP domain through port mirroring on the switch. A GNS3 core switch “Cisco 26125S” is connecting all the VMs through a cloud interfaces, the attacker running Kali Linux is assumed to be on the network post-compromise, within the testing environment.

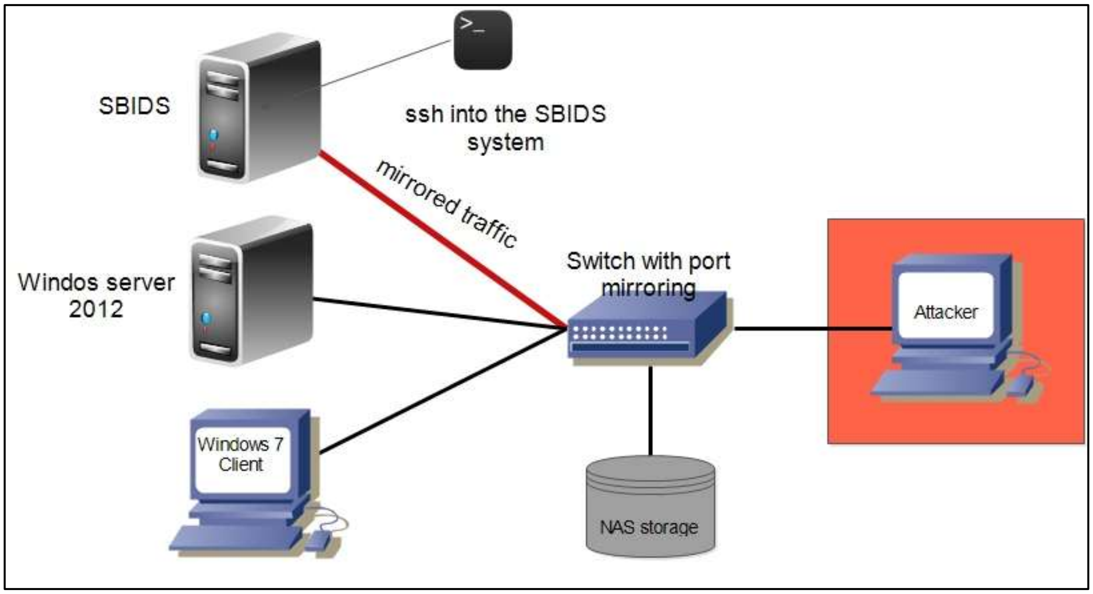


Figure 3. 2 Block Diagram of the Proposed Model

Device	SAN network IP	Shared SMB folder	VMnet-If
Centos 7 server with BRO	192.168.20.254/24	-	VMnet11
OpenFiler SAN storage	192.168.20.100/24	\\192.168.20.100\sharenas.smb.windo	Vmnet12
Windows 7 Client	192.168.20.30/24		VMnet15
Windwos server 2012	192.168.20.2/24		VMnet13
Kali linux	192.168.20.14/24	-	VMnet14

Table 3.1 SAN network details

3.2 Detection Using Bro

Bro presents an alternative method of detection through network security monitoring. Bro version 2.5 provides new detection capabilities by way of an SMB protocol analyzer. The analyzer provides insight into files transferred over SMB, SMB commands, SMB trees, NTLM activity. Below are the log files introduced in Bro version 2. in Table 3.2.

Log File Description	**Log File Description**
dce_rpc.log	Distributed Computing Environment/RPC
ntlm.log	NT LAN Manager (NTLM)
smb_cmd.log	SMB commands
smb_files.log	SMB files
smb_mapping.log	SMB trees

Table 3.2 log files introduced in Bro version 2.5

With Bro, we can build additional detections, and generate alerts on instances of suspicious SMB activity. Bro offers the advantage of detecting activity in real-time by-passing traffic to the analysis engine. It can also detect malicious activity in packet captures, which can be applied retrospectively to past events. [6]

3.3 Bro Scripting for Detection

The ability to create custom Bro scripts to fit an organization’s environment makes Bro a flexible network security monitoring solution. The protocols analyzed by Bro extract metadata used for scripting. The Bro Scripting Framework is based on C++. Also, it reduces incoming packet streams into higher-level events and applies customizable scripts to determine the necessary course of action. This simple design allows us to configure an array of real-time alerts, execute arbitrary programs on demand, and log data for later use.

After reviewing the Bro logs generated by network traffic, script building occurs using string pattern or event-based indicators. Scripts can be configured to generate a “notice” which alerts an analyst to an adverse event on a network. [1]

3.4 Bro Scripts

Bro loads the scripts in the $PREFIX/bro/share/policy/ directory by default, where $PREFIX corresponds the root directory used in the Bro installation. Loading custom scripts requires adding an entry to the local.bro file with the directory path of the script. The local.bro file is located in the $PREFIX/bro/share/policy/site/ directory. Running the command “broctl deploy” from the $PREFIX/bro/bin directory deploys a custom script. making it active. All enabled scripts can be displayed by running the “broctl scripts” command.

Scripts loaded by Bro generate ASCII-based logs, which are populated with network metadata related to connections, files, and protocols. Viewing the various log files can assist analysts in finding malicious traffic on a network, which requires analysis. During analysis, if correlations are made that identify malicious activity, they can be used to create scripts. These scripts, in turn, can notify an analyst to malicious activity as it occurs on a Storage network. Using the Bro Notice framework, alerts may be sent via email or as a log entry in the notice.log file.

3.5 Proposed model Detection Scripts

The scripts in this section were created to detect potentially malicious SMB activity on a NAS network. Tracking malicious files sent via SMB assists analysts in identifying a potential incident requiring response. The script shown in appendix A, uses the files analysis (base/frameworks/files) and hash (base/frameworks/hash-all-files) frameworks in Bro to identify files transferred via SMB, and checks their cryptographic hashes against Virus Total’s anti-virus database. If the submitted hash of a file is identified as malicious, a log entry is added to notice.log, alerting an analyst to a potential incident on a NAS network. Searching Virus Total for the hashes of files, rather than submitting the files themselves, does not disclose the sensitive analysis work that is currently in progress. Submitting a file that is part of a targeted attack, may inform attackers that they are discovered on a network, giving them a chance to alter their tactics or cause further damage. Trusted systems should be added to the global trustedIPs variable, which would prevent them from being misidentified as malicious. Appendix B and C displays another custom script, created to detect the use of the C$, ADMIN$, or IPC$ shares. While there are legitimate uses for these shares, activity involving these shares should be limited. Attackers use these shares to execute services and processes, upload/transport malware, and move laterally. Any systems seen in the notice log for this alert should be investigated to determine if they are infected or compromised. Trusted administrative systems can be tuned out by adding their static IP to the trustedIPs set.

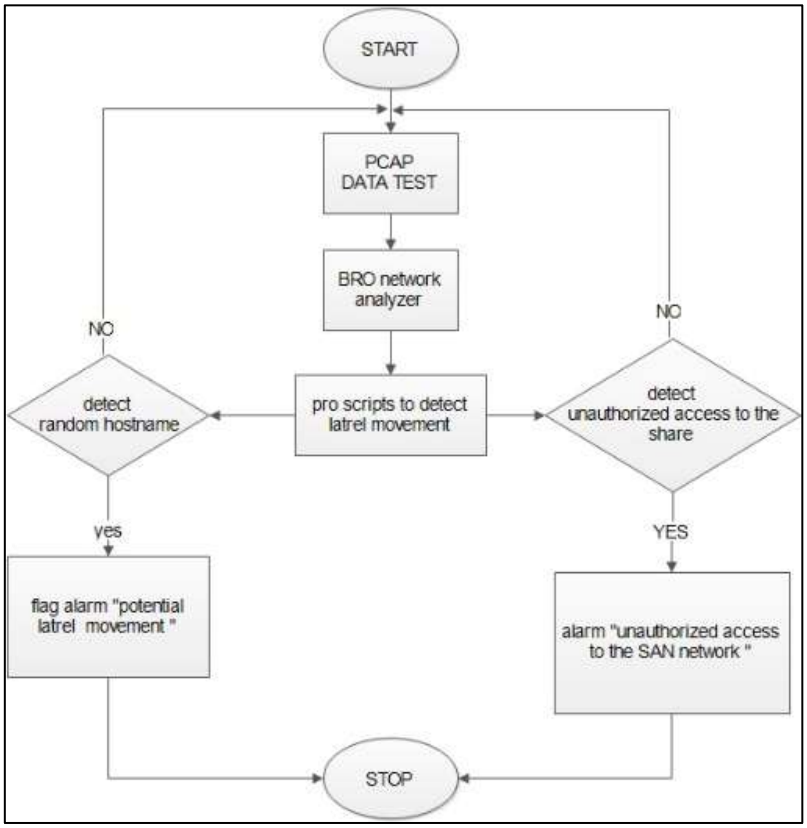

START
PCAP DATA TEST
BRO network analyzer
pro scripts to detect latrel movement
detect random hostname
NO
yes
flag alarm "potential latrel movement "
detect unauthorized access to the share
NO
YES
alarm "unauthorized access to the SAN network "
STOP

Figure 3.2. Testing phase of BRO policies.

3.6 Attack scenario:

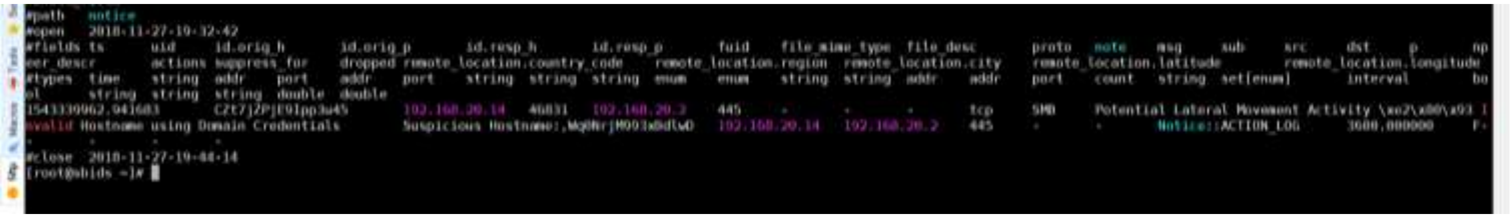
Table 3.3 below illustrates how to use the hypotheses laid out with the data and techniques enumerated.

(Hypothesis)	**Hypothesis:** Attackers may be attempting to move laterally in Windows environment by leveraging PsExec using the shared SAN. **Look for:** Anomalies in host to host traffic leveraging the PsExec binary, service, and/or network traffic. “C$\|ADMIN$|IPC$” shares being used in SAN network traffic.
Investigation (Data)	**Datasets:** For identifying use of PsExec, focus primarily on application protocol metadata, including: • Netflow (“flow” data in general) • Windows or Linux logs
Uncover Patterns and IOCs (Techniques)	1. Use a search to identify “Potentially Malicious Use of an Administrative Share” messages in bro_notice log. 2. remove hosts as you confirm they are legitimately connecting to a destination over SMB. This should leave only unexplained SMB connections that need further analysis. 3. Take the results of step 2 and stack the data for what is useful to investigating your hypothesis 4. For example: destination IP, port used, connection duration/length, etc.
Inform and Enrich Analytics (Takeaways)	The destination IP addresses, path, and ports involved in the Lateral Movement activity you have discovered can be taken as IOCs and added to an indicator database in order to expand automated detection systems. You can also create packet-level signatures to trigger alerts for cases where the admin share connections you have discovered may appear again.

Table 3.3 below illustrates how to use the hypotheses

3.7 Random Host Name:

We claim that any traffic containing random hostnames clearly signifies the possibility of unwanted activities in the network. The major reason we claim this is that we haven’t seen this behavior in normal traffic, only malicious traffic contained this behavior. With administrative credentials, attackers can use PsExec to execute processes on a remote computer. PsExec was used by the attacker to gain command line access to another computer on the lab network (WIN7PROD3) by executing cmd.exe. The “exploit” command in Kali runs the module which results in a meterpreter session, giving the attacker access to the remote system at IP address 192.168.20.2 as seen in Figure 3.4 displays the Bro notice log, verifying detection of Metasploit PsExec module use of ADMIN$ and IPC$ shares, as well as use of the random hostname “Wq0NrjM993xBdlwD”.



```
#path   notice
#open   2018-11-27-19-32-42
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       fuid    file_mime_type  file_desc       proto   note    msg     sub     src     dst     p       np
eer_descr       actions suppress_for    dropped remote_location.country_code    remote_location.region  remote_location.city    remote_location.latitude        remote_location.longitude
#types  time    string  addr    port    addr    port    string  string  string  enum    enum    string  string  addr    addr    port    count   string  set[enum]       interval        bo
ol      string  string  string  double  double
1543339962.941683       CZt7j2PjE91pp3u45       192.168.20.14   46831   192.168.20.2    445     -       -       -       tcp     SMB     Potential Lateral Movement Activity \xe2\x80\x93 I
nvalid Hostname using Domain Credentials        Suspicious Hostname:,Wq0NrjM993xBdlwD   192.168.20.14   192.168.20.2    445     -       -       Notice::ACTION_LOG      3600.000000     F-
-       -       -       -
#close  2018-11-27-19-44-14
[root@sbids ~]#
```

Figure 3.4. Bro Logs following The Metasploit PsExec Activity

CHAPTER FOUR
RESULTS AND DISCUSSION

Results and Discussion

4.1 Examples of Detection using the proposed model

The main challenge in detecting lateral movement attack is that the attacker is using legit tools: PsExec, WMI as this attack is targeted so it is conducted very precisely and throughout the attack, the attacker tries to stay under the radar being undetected. Also, the attacker is using credentials of authorized users which makes it further hard to detect. Initially we analysed the datasets we generated at the testing lab. In which protocols like SMB, NT LAN Manager (NTLM), NT LAN Manager Security Support Provider (NTLMSSP), Remote Procedure Call (RPC) are exploited to generate malicious datasets and these protocols are used normally to generate normal datasets. Tools like WIRESHARK and proposed system with BRO-network analyzer are used to analyze the traffic. During our analysis we analyzed the behavior of these protocols in normal traffic datasets and as well as malicious traffic.

Figure 4.1 shows the architecture of the testing lab. To collect the datasets, Wireshark is running on Windows 12 OS and SAN storage to detect the share traffic. Two types of datasets are generated at the lab, normal datasets and datasets containing lateral movement attacks in the proposed Storage Network. In normal datasets, real administrator behavior is simulated, while malicious datasets contain attacker behavior.

The metaexploit tool, used to generate normal and malicious datasets. The total size of the datasets generated at this lab is 10Mb. The total time duration of these datasets is approximately 11 minutes.

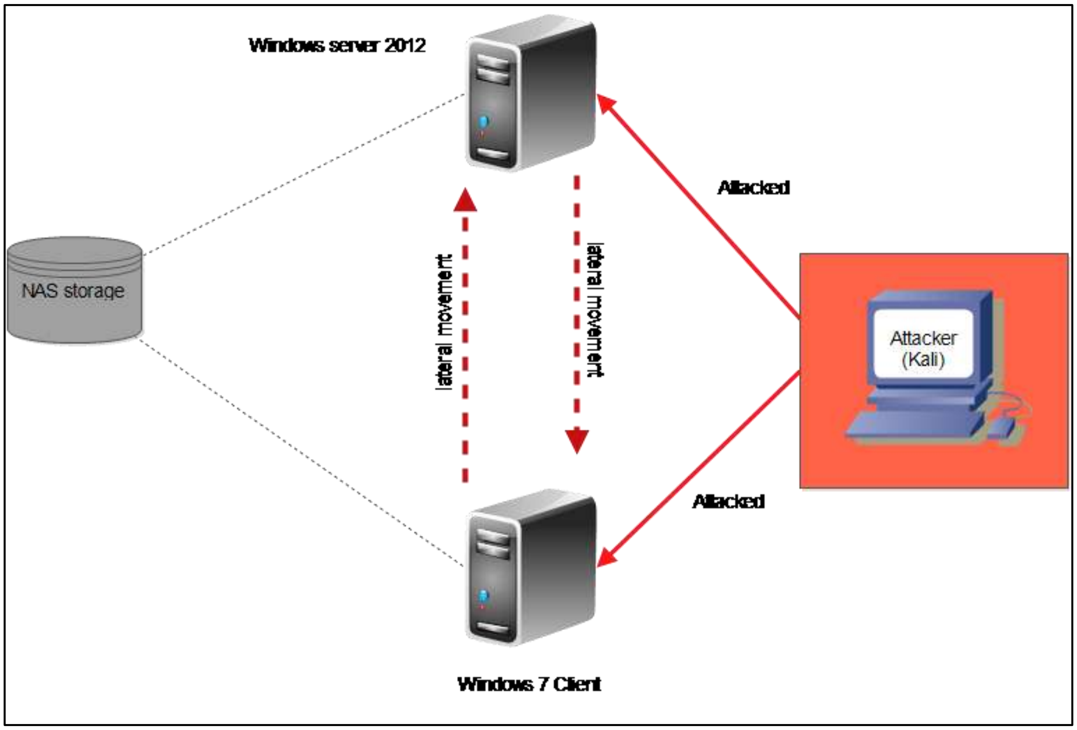


Figure 4.2 Architecture of the testing-lab

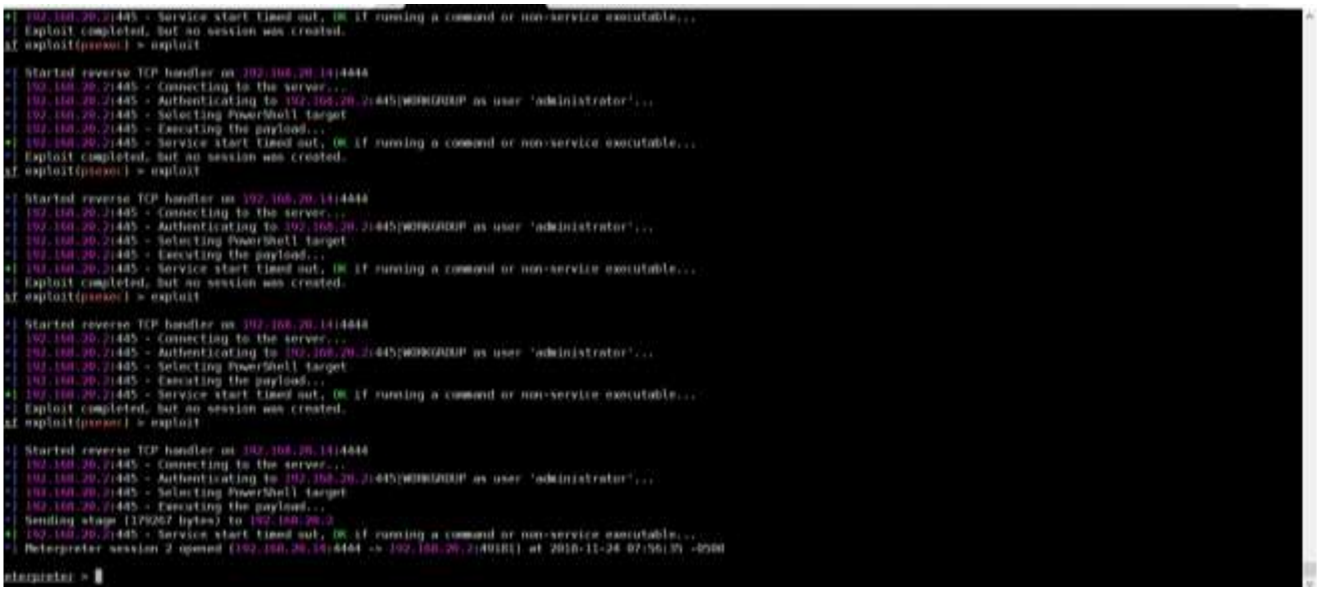
Below are some examples of the capabilities that the scripts shown in Appendix A and B provide to detect attacks utilizing SMB in Storage Area Network.

All cases of the received values have been illustrated in more details below.

4.2 Metasploit psexec

Metasploit is a popular framework used for penetration testing, and contains a modified version of PsExec.exe. Figure 4.2 shows the attacker's configured options for the PsExec Metasploit module within Metasploit's msfconsole. With the following details:

- db_status
- use exploit/windows/smb/psexec
- set RHOST 192.168.20.2
- set SMBDomain WORKGROUP
- set SMBPASS test@123
- set SMBUSER administrator
- set payload windows/meterpreter/reverse_tcp
- set LHOST 192.168.20.14
- exploit



```
+] 192.168.20.2:445 - Service start timed out, OK if running a command or non-service executable...
*] Exploit completed, but no session was created.
sf exploit(psexec) > exploit

*] Started reverse TCP handler on 192.168.20.14:4444
*] 192.168.20.2:445 - Connecting to the server...
*] 192.168.20.2:445 - Authenticating to 192.168.20.2:445|WORKGROUP as user 'administrator'...
*] 192.168.20.2:445 - Selecting PowerShell target
*] 192.168.20.2:445 - Executing the payload...
+] 192.168.20.2:445 - Service start timed out, OK if running a command or non-service executable...
*] Exploit completed, but no session was created.
sf exploit(psexec) > exploit

*] Started reverse TCP handler on 192.168.20.14:4444
*] 192.168.20.2:445 - Connecting to the server...
*] 192.168.20.2:445 - Authenticating to 192.168.20.2:445|WORKGROUP as user 'administrator'...
*] 192.168.20.2:445 - Selecting PowerShell target
*] 192.168.20.2:445 - Executing the payload...
+] 192.168.20.2:445 - Service start timed out, OK if running a command or non-service executable...
*] Exploit completed, but no session was created.
sf exploit(psexec) > exploit

*] Started reverse TCP handler on 192.168.20.14:4444
*] 192.168.20.2:445 - Connecting to the server...
*] 192.168.20.2:445 - Authenticating to 192.168.20.2:445|WORKGROUP as user 'administrator'...
*] 192.168.20.2:445 - Selecting PowerShell target
*] 192.168.20.2:445 - Executing the payload...
+] 192.168.20.2:445 - Service start timed out, OK if running a command or non-service executable...
*] Exploit completed, but no session was created.
sf exploit(psexec) > exploit

*] Started reverse TCP handler on 192.168.20.14:4444
*] 192.168.20.2:445 - Connecting to the server...
*] 192.168.20.2:445 - Authenticating to 192.168.20.2:445|WORKGROUP as user 'administrator'...
*] 192.168.20.2:445 - Selecting PowerShell target
*] 192.168.20.2:445 - Executing the payload...
*] Sending stage (179267 bytes) to 192.168.20.2
+] 192.168.20.2:445 - Service start timed out, OK if running a command or non-service executable...
*] Meterpreter session 2 opened (192.168.20.14:4444 -> 192.168.20.2:49181) at 2018-11-24 07:56:35 -0500

eterpreter >
```

Figure 4.2. Metasploit PsExec Module to Connect to 192.168.20.2

The “exploit” command runs the module which results in a meterpreter session, giving the attacker access to the remote system at IP address 192.168.20.2 as seen in Figure 4.2. Figure 4.3 displays the Bro notice log, verifying detection of Metasploit PsExec module use of ADMIN$ and IPC$ shares, as well as use of the random hostname “WIN-V34558051QU”.

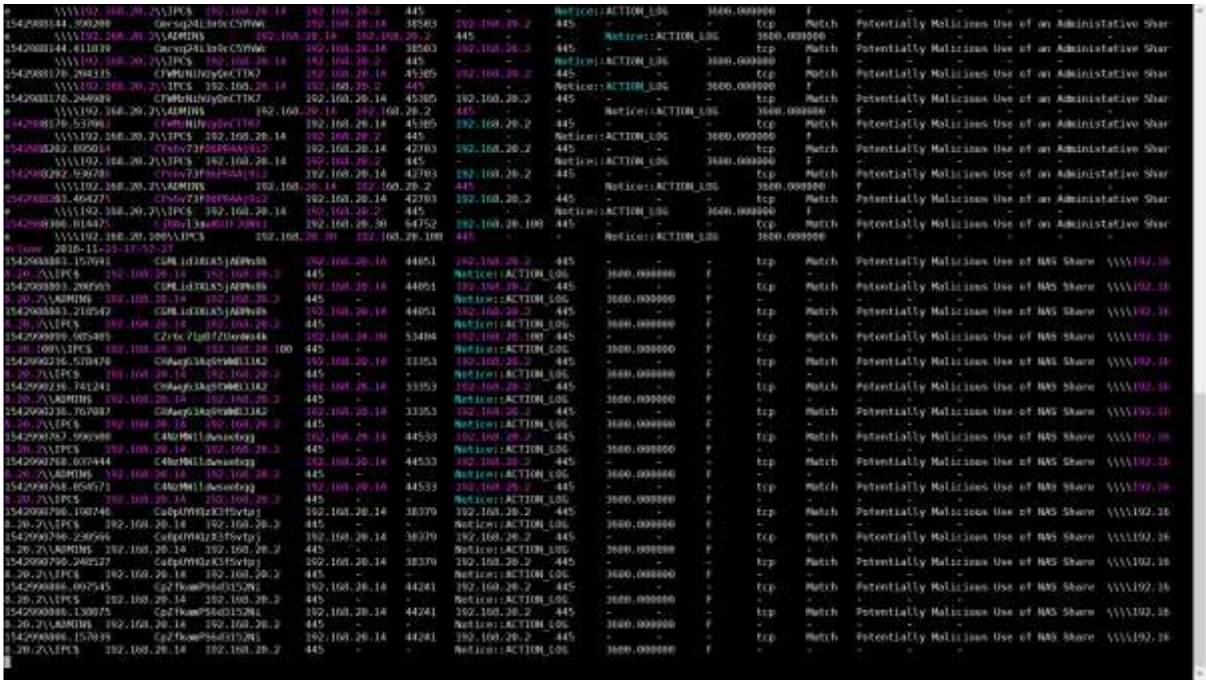
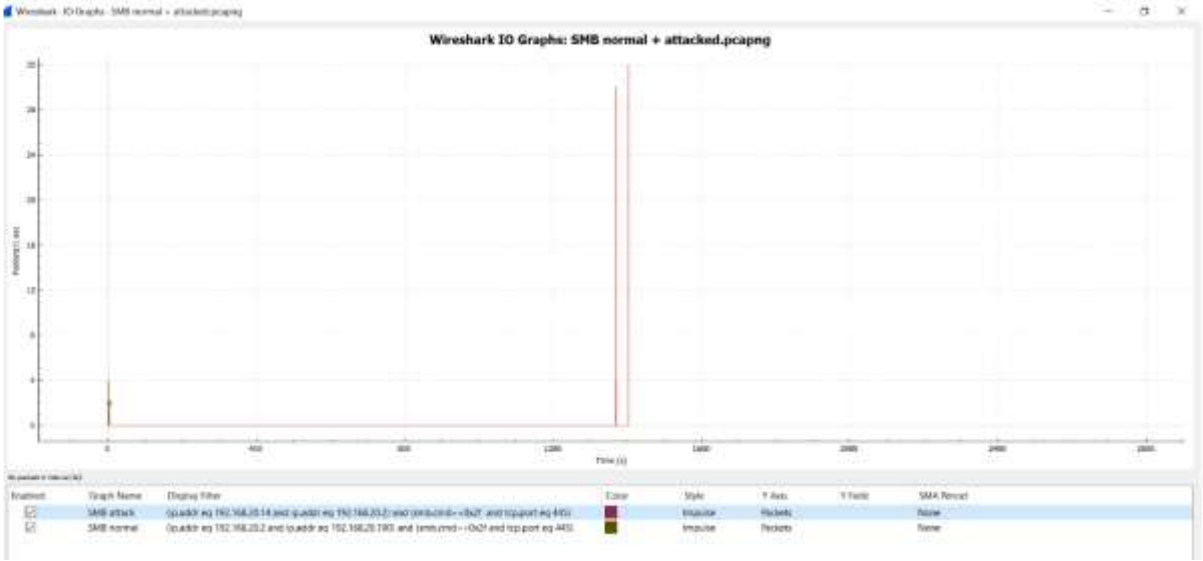


Figure 4.3. Bro Logs following The Metasploit PsExec Activity

Attackers are known to transfer or upload additional malware to file shares during attacks.

4.2.1 Datasets from lab setup

Prevalence of WRITE command towards IPC share the reason we consider this technique of high significance because there is a clear difference between number of WRITE commands towards IPC$ in normal and malicious traffic. With the Figure 4.4 we support this claim. From Home lab datasets, we extracted number of Write commands towards IPC$ from normal and malicious traffic datasets. As we can see in Figure 4.6 in normal datasets the number of WRITE commands towards IPC$ is very low, while in malicious datasets they are huge for the reasons mentioned earlier. Each of the dataset is of time stamp T=10 minutes. Figure 4.4 shows the comparison between number of Write commands towards IPC$ in normal and malicious datasets generated in Testing lab.


Wireshark · IO Graphs · SMB normal + attacked.pcapng
Wireshark IO Graphs: SMB normal + attacked.pcapng
32
28
24
20
16
12
8
4
0
Packets/1 sec
0
400
800
1200
1600
2000
2400
2800
Time (s)

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period
☑	SMB attack	(ip.addr eq 192.168.20.14 and ip.addr eq 192.168.20.2) and (smb.cmd==0x2f and tcp.port eq 445)	■	Impulse	Packets		None
☑	SMB normal	(ip.addr eq 192.168.20.2 and ip.addr eq 192.168.20.193 and (smb.cmd==0x2f and tcp.port eq 445)	■	Impulse	Packets		None

Figure 4.6 comparison between number of Write commands towards IPC$ in normal and malicious behavior while using shared SMB via NAS storage.

4.2.2 Experimental Results

Attackers use the SMB protocol in ways that blend in with day-to-day network traffic. These malicious entities then move laterally within a network, post-compromise, and attempt to access systems looking for sensitive data. The SMB protocol allows their activity hard to detect. Collecting Windows or Linux event logs related to file share auditing is a method for detecting malicious SMB activity, however this is not ideal due to the large volume of logs generated logs.

Intrusion detection systems, such as Snort rely primarily on pattern-based indicators, which can be bypassed and may be difficult to tune. Bro Network Security Monitor can analyze the SMB protocol and provide metadata which can be used to identify potential indicators of compromise. These indicators are the basis of scripts that are used to detect malicious activity and alert analysts. The scripts introduced in this research generate alerts when potentially malicious files transferred via SMB, hidden file shares such as C$ are used, and when suspicious hostnames seen in SMB traffic. Bro proves to be an effective, open-sourced, and cost efficient, solution to detect and respond to malicious activity using SMB.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

Chapter Five
Conclusion and Recommendations

5.1 Conclusion

A storage IDS watches system activity from a new view point, which immediately exposes some common intruder actions. Running on separate hardware, this functionality remains in place even when client OSes or user accounts are compromised. Our prototype storage IDS demonstrate both feasibility and efficiency within a file server. Analysis of real intrusion tools indicates that most would be immediately detected by a storage IDS. After adjusting for storage IDS presence, intrusion tools will have to choose between exposing themselves to detection or being removed whenever the system reboots. We described and evaluate a prototype storage IDS, embedded in an NFS server, to demonstrate both feasibility and efficiency of storage-based intrusion detection to detect lateral movement attack using SMB protocol by using BRO network analyzer, Intrusion detection systems, such as Snort rely primarily on pattern-based indicators, which can be bypassed and may be difficult to tune. Bro Network Security Monitor can analyze the SMB protocol and provide metadata which can be used to identify potential indicators of compromise. These indicators are the basis of scripts that are used to detect malicious activity and alert analysts. The scripts introduced in this research generate alerts when potentially malicious files transferred via SMB, hidden file shares such as C$ are used, and when suspicious hostnames seen in SMB traffic. Bro proves to be an effective, open-sourced, and cost efficient, solution to detect and respond to malicious activity.

5.2 Recommendations

This research presents a novel idea to design storage-based intrusion detection system for file-based share environment by applying IDS technology to the shared files via smb protocol. The advantage of this approach is that it makes fully use of exist technology and does not require many changes to the storage system or the intrusion detection software. It is easy to realize intrusion detecting but has negligible effect on storage system performance. A recommendation of developing a prototype storage IDS embedded in a device exporting a block-based interface (SCSI or FC) with the same proposed rules and system is recommended for future work.

REFERENCES

[1] M. Banikazemi, D. Poff, and B. Abali, "Storage-based intrusion detection for storage area networks (SANs)," in Mass Storage Systems and Technologies, 2005. Proceedings. 22nd IEEE/13th NASA Goddard Conference on, 2005, pp. 118-127.

[2] D. Yao and D. Feng, "Intrusion Detection for Object-Based Storage System," in Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for, 2008, pp. 218-222.

[3] S. Hajirostam, "Evaluation of Storage Area Network (SAN) Security and Performance," 2013.

[4] F. Majstor, "Storage Area Networks Security Protocols and Mechanisms," 2004.

[5] R. A. Kemmerer and G. Vigna, "Intrusion detection: a brief history and overview," Computer, vol. 35, pp. supl27-supl30, 2002.

[6] R. V. Richie Cyrus "Detecting Malicious SMB Activity Using Bro," 2017.

[7] A. G. Pennington, J. L. Griffin, J. S. Bucy, J. D. Strunk, and G. R. Ganger, "Storage-based intrusion detection," ACM Transactions on Information and System Security (TISSEC), vol. 13, p. 30, 2010.

[8] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System , 2019

[9] R. Kumar and D. Sharma, "HyINT: Signature-Anomaly Intrusion Detection System," 2018

[10] L. Xing, M. Tannous, V. M. Vokkarane, H. Wang and J. Guo, "Reliability Modeling of Mesh Storage Area Networks for Internet of Things,", Dec. 2017

[11] Z. S. Malek, B. Trivedi and A. Shah, "User behavior Pattern -Signature based Intrusion Detection," 2020

[12] Paxson, Vern, Campbell, Scott, leres, Craig, and Lee, Jason. Bro Intrusion Detection System. Computer software. Vers. 00. DOE and NSF. 25 Jan. 2006. Web.

APPENDICES

Appendix A

Bro Script Detecting the Use of Malicious Files in SMB Traffic

```
@load base/frameworks/files
@load base/frameworks/notice
@load frameworks/files/hash-all-files
export {
redef enum Notice::Type += { SMB};
global trustedIPs: set[addr] = {192.168.1.22,192.168.1.20} &redef;
# url needed to use VirusTotal API
const vt_url = "https://www.virustotal.com/vtapi/v2/file/report" &redef;
# VirusTotal API key
const vt_apikey = "<---- Enter your Virus Total API key here ---->" &redef;
# threshold of Anti-Virus hits that must be met to trigger an alert
const notice_threshold = 2 &redef;
event file_hash(f: fa_file, kind: string, hash: string)
{
# If the file "f" for the event has a source type, and if the source type equals
SMB, check file hash against VirusTotal
if ( f?$source && f$source == "SMB")
{
local data = fmt("resource=%s", hash);
local key = fmt("-d apikey=%s",vt_apikey);
# HTTP request out to VirusTotal via API
local req: ActiveHTTP::Request = ActiveHTTP::Request($url=vt_url,
$method="POST",$client_data=data, $addl_curl_args=key);
when (local res = ActiveHTTP::request(req))
{
if ( |res| > 0)
{
if ( res?$body )
{
local body = res$body;
local tmp = split_string(res$body,/\}\},/);
if ( |tmp| != 0 )
{
local stuff = split_string( tmp[1], /\,/ );
# splitting the string that contains the amount of
positive anti-virus hits on ":" "positives:23"
local pos = split_string(stuff[9],/\:/);
# converting the string from variable pos into a
integer
local notic = to_int(pos[1]);
# If the number of positives (number stored in
variable notic) equals or exceeds the threshold, generate a notice
if (notic >= notice_threshold )
```

Appendix B

Bro Script Detecting the Use of Hidden SMB Shares

```
@load base/frameworks/files
@load base/frameworks/notice
#@load policy/protocols/smb
@load base/protocols/smb
export {
redef enum Notice::Type += {
Match
};
global isTrusted = T;
global trustedIPs: set[addr] = {192.168.20.100,192.168.20.2} &redef;
function hostAdminCheck(sourceip: addr): bool
{
if (sourceip !in trustedIPs)
{
return F;
}
else
{
return T;
}}
event smb2_tree_connect_request(c: connection, hdr: SMB2::Header, path: string)
{
isTrusted = hostAdminCheck(c$id$orig_h);
if (isTrusted == F){
if ("IPC$" in path || "ADMIN$" in path || "C$" in path)
{
NOTICE([$note=Match, $msg=fmt("Potentially Malicious Use of NAS  Share"),
$sub=fmt("%s",path), $conn=c]);
}}}
event smb1_tree_connect_andx_request(c: connection, hdr: SMB1::Header, path: string, service:
string)
{
isTrusted = hostAdminCheck(c$id$orig_h);
if (isTrusted ==F){
if ("IPC$" in path || "ADMIN$" in path || "C$" in path)
{
NOTICE([$note=Match, $msg=fmt("Potentially Malicious Use of NAS Share"),
$sub=fmt("%s",path), $conn=c]);
}}}}
```

Appendix C

Bro Script for random hostname access through the SMB share:

```
@load base/frameworks/notice
#@load policy/protocols/smb
@load base/protocols/smb
export {
redef enum Notice::Type += {
SMB
};
event ntlm_authenticate(c: connection, request: NTLM::Authenticate)
{
# strip out the first 5 characters of workstation value to be compared to naming convention
local strcheck = sub_bytes(request$workstation, 1, 8);
# value of the comparison of the two strings
local comp_str = strcmp(strcheck ,"WIN-V34558051Qu" );
# If the comparison of the strings stored in variable comp_str are not the same, generate a notice.
if (comp_str != 0 )
{
NOTICE([$note=SMB, $msg=fmt("Potential Lateral Movement Activity – Invalid Hostname
using Domain Credentials"), $sub=fmt("%s,%s","Suspicious Hostname:",request$workstation),
$conn=c]);
}
}
}
```