Università degli Studi di Pisa

DIPARTIMENTO DI INFORMATICA
Corso di Laurea Triennale in Informatica

Tesi di laurea

Tecniche per il riconoscimento di esfiltrazione di dati da sistemi informatici

Candidato:
Marco Piangatello

Relatore:
Luca Deri

Anno Accademico 2018-2019

Sommario

L'esfiltrazione dei dati è un processo di esportazione non autorizzata di dati confidenziali da un sistema. A differenza delle violazioni che hanno come obiettivo quello di bloccare o danneggiare il sistema, il fenomeno dell'efiltrazione è particolarmente difficile da rilevare. Non è semplice determinare quali informazioni stanno lasciando la rete in maniera legittima e quali comunicazioni invece sono parte di un processo malevolo. In alcuni casi non è nemmeno possibile stabilire se la trasmissione sta avvenendo.

L'insieme dei possibili metodi di esfiltrazione è molto grande, comprende tutti i canali di comunicazione standard e molti altri appositamente creati per questo scopo.

A causa della natura complessa del problema, lo sviluppo dei sistemi di sicurezza delle reti rappresenta una sfida ardua.

In questo elaborato, dopo aver presentato il problema e discusso di alcuni tra i possibili metodi di attuazione per l'esfiltrazione, si analizzano un insieme di tecniche utilizzate per rilevare un tale processo all'interno di una rete. Oltre ad uno studio teorico verrà presentato una possibile soluzione da me sviluppata per il riconoscimento di efiltrazione.

Indice

1 Introduzione

Un degli obiettivi principali dei recenti attacchi informatici è l'esfiltrazione dei dati, ovvero il furto di informazioni sensibili dall'interno di una rete privata.

Con il termine esfiltrazione si identifica il processo di trasmissione non autorizzata di dati da una rete, tipicamente una rete privata di una azienda, verso l'esterno. Questo processo può essere effettuato da uno o più individui dall'interno o dall'esterno della rete stessa.

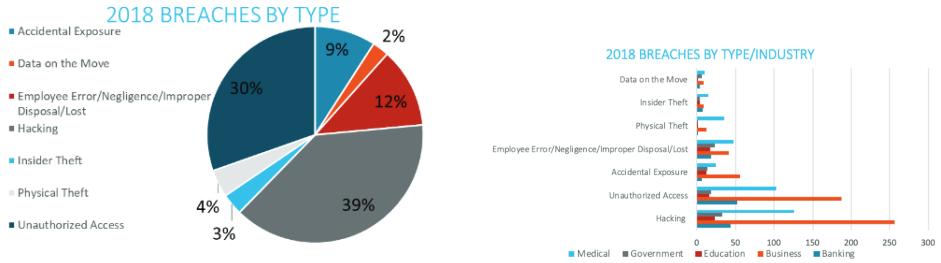
Prevenire questo fenomeno è diventata una necessità e una sfida sempre più complessa per diverse ragioni. In primo luogo, nel corso degli ultimi anni, i crimini informatici si sono trasformati da atti individuali perseguiti da singoli individui, ad attività compiute da vere e proprie organizzazioni[1]. Questa trasformazione ha aumentato il budget e le risorse a disposizione degli attaccanti[1] rendendo le tecniche di esfiltrazione molto sofisticate e professionali. Inoltre le reti delle organizzazioni sono progettate per un legittimo scambio interno di dati. Questo tipo di infrastruttura può essere sfruttata per l'esfiltrazione dei dati.

Secondo le rilevazioni dell'organizzazione ITRC[2], dal 2017 al 2018, sono avvenute 1244 violazioni documentate di sistemi informatici negli USA[2]. La figura1.1(a) mostra quale forme di violazioni sono state più comuni tra quelle documentate. Inoltre sono state studiati anche quali settori economici e sociali sono stati colpiti da violazioni informatici figura1.1(b). Il settore delle aziende private risulta essere il più colpito.

Il costo medio globale di una violazione di sicurezza nel 2018 è stato calcolato essere 3.86 milioni di dollari[3]. Sfortunatamente, l'impatto provocato da una perdita di dati da parte di una azienda va oltre questo dato. Ci sono implicazioni che riguardano l'impatto negativo sulla reputazione,costi di azioni legali e altri effetti negativi che comportano un danno molto maggiore di quello calcolato.

[1] spesso definiti con il termine hackers

[2] Identity Theft Resource Center



(a) **(b)**

Figura 1

Come evidenziano tali statistiche, una proliferazione di questi avvenimenti ha reso il problema di sicurezza dei dati molto importante per le aziende e per altre organizzazioni governative e non. Guidati da tale necessità, gli esperti di sicurezza hanno sviluppato vari sistemi che includono tra gli altri: Intrusion Detection System (IDS), Intrusion Prevention System (IPS) e firewall[4][5]. Questi sistemi non si adattano molto bene al problema dell'esfiltrazione di dati a causa della natura non strutturata e in costante evoluzione di questo fenomeno. Per superare queste limitazioni, gli esperti di sicurezza e i ricercatori hanno formalizzato il concetto di contromisura per l'esfiltrazione di dati. Questo termine si riferisce in particolare alla prevenzione, rilevamento e investigazione delle perdite di dati. I sistemi tradizionali (e.g, IDS, IPS, and firewall) sono stati creati con il fine di agire quando un attaccante cerca di accedere ad un sistema interno alla rete, le contromisure invece sono tecniche che si applicano nei casi in cui un attaccante è già all'interno di una rete privata e il suo obiettivo è quello di trasmettere dati sensibili verso l'esterno.

1.1 Il processo di Esfiltrazione dei dati

L'esfiltrazione è un processo di trasmissione di dati attraverso reti differenti, tipicamente i dati escono da una rete privata verso internet. Sono conosciute relativamente poche informazioni su come avviene in dettaglio questo processo. Ci sono però dei passaggi obbligati che devono essere effettuati per avere successo in questo tipo di attacco.

Prima di presentare un tipico scenario di esfiltrazione sono necessarie alcune considerazioni di carattere generale. Risulta di fondamentale importanza sottolineare che il concetto di sicurezza di una rete racchiude molti aspetti differenti. Ognuno di questi va approfondito singolarmente e richiede specifiche forme di protezione. Nel seguito della mia tesi mi concentrerò sul processo di esfiltrazione dei dati che è una delle molteplici fasi che rientrano nella definizione di attacco informatico. Descriverò brevemente le azioni che sono necessarie per arrivare a questo processo senza però approfondire le metodologie utilizzate e le forme di difesa adottate.

Per prima cosa è necessario avere accesso all'interno della rete. In alcuni casi la minaccia di sicurezza proviene direttamente dall'interno della rete. Una prima distinzione da fare è quella tra gli individui male intenzionati e quelli che invece commettono errori o in generale non sono consapevoli del potenziale danno delle loro azioni. La fuoriuscita di dati sensibili da una rete può avvenire a causa di un errore umano involontario. In particolare è comune che un dipendente o più in generale un individuo interno ad un azienda/organizzazione adotti involontariamente dei comportamenti che espongano ad un alto rischio di furto i dati a cui ha accesso. Un'altro aspetto importante da considerare è se l'attaccante ha accesso fisico alla rete interna. Sono molto frequenti i casi in cui le minacce alla sicurezza informatica di una rete derivano, non da un individuo esterno bensì da uno che si trova all'interno della rete stessa. Per esempio può trattarsi di un dipendente o più in generale da una qualsiasi persona con degli elevati permessi di accesso nella rete stessa.

Lo scenario più comune di attacco informatico è invece quello in cui, dall'esterno della rete, un attaccante cerca di prendere il controllo di un sistema obiettivo all'interno di essa. I vettori utilizzati per compiere questa azione sono molteplici e ampiamente descritti dalla letteratura in ambito di sicurezza informatica[6]. Esistono inoltre diversi meccanismi di difesa utilizzati nelle reti di aziende e organizzazioni private che hanno come obiettivo primario la prevenzione di questo tipo di attacchi. Questi meccanismi, già citati nel capitolo introduttivo, forniscono una primo livello di sicurezza di una rete. Se l'attaccante riesce a superare i meccanismi di difesa senza essere indivi-

duato, ha la possibilità di raccogliere dati di valore dalla rete. Alcuni tipi di attacchi molto sofisticati possono rimanere non rilevati per molto tempo rappresentando una seria minaccia di sicurezza[7].

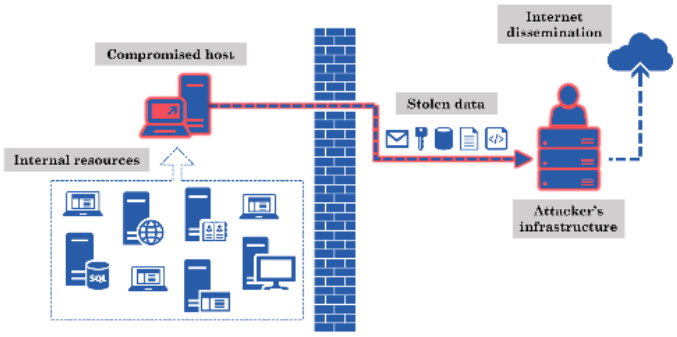


Figura 2: Percorso dei dati esfiltrati da una rete privata

L'attaccante può esfiltrare i dati automaticamente, mediante l'utilizzo di un software(malware), o manualmente, mediante l'esecuzione di un determinato codice. I metodi automatici devono essere molto 'robusti', poiché la maggior parte di questi rischia di fallire a causa della forte dipendenza rispetto alle caratteristiche della rete ospite. L'esfiltrazione manuale è molto più flessibile, offre la possibilità di esplorare diverse soluzione per poter scegliere la più conveniente. In ogni caso è necessario l'utilizzo di un 'canale', una connessione astratta tra il sistema obiettivo e l'esterno della rete(figura1.2). La scelta del canale ricadrà solitamente su quelli più comuni e che hanno un utilizzo legittimo da parte degli utenti all'interno della rete.

Per effettuare l'esfiltrazione verrà quindi eseguito un codice all'interno del sistema obiettivo. Può essere un programma eseguito automaticamente da un malware o manualmente dall'attacante. Ci sono alcuni meccanismi già presenti nella maggior parte dei sistemi di utilizzo comune che mettono a disposizione dell'attaccante un insieme di servizi già preinstallati. Per metodi più sofisticati l'attaccante avrà la necessità di caricare o creare manualmente librerie esterne sul sistema.

L'attaccante si connetterà ad un server appositamente configurato che utilizza per ricevere i dati. Il client obiettivo e il server devono necessariamente utilizzare lo stesso protocollo di comunicazione.

La fase che comporta la trasmissione dei dati in uscita dalla rete sarà quella approfondita nel seguito di questa tesi. In particolare presenterò alcuni dei metodi utilizzati per esportare i dati e le contromisure adottate per prevenire e rilevare questo tipo di comunicazione.

1.2 Motivazioni e obiettivi del lavoro

A causa della natura in costante evoluzione del problema dell'esfiltrazione dei dati da una rete, sono presenti in letteratura e nell'utilizzo pratico una grande quantità di tecniche utilizzate come contromisure riguardanti questo fenomeno. Il primo obiettivo del mio lavoro è quello di presentare e analizzare queste tecniche per poter meglio comprendere dove si colloca la soluzione da me approfondita nello sviluppo del software. Inoltre è di vitale importanza, in una analisi critica delle soluzioni proposte, comprendere quali tipi di dato e tecniche di esfiltrazione sono obiettivo delle contromisure e quali invece non sono investigate.

Successivamente presenterò un breve progetto da me sviluppato che si inserisce tra le possibili tecniche di rilevamento per l'esfiltrazione di dati, utilizzabili nell'ambito della sicurezza di una rete privata. Questo software si concentra sull'investigazione del traffico in uscita dalla rete alla ricerca di testo in formato leggibile da un umano.L'approccio scelto è uno tra i molti possibili e ha come obiettivo il monitoraggio di un sottoinsieme delle tecniche utilizzate per la trasmissione non autorizzata di dati all'esterno di una rete privata.

Il codice da me sviluppato è pensato per essere inserito all'interno più ampio meccanismo di monitoraggio del traffico di una rete.

2 Tassonomia dei metodi di esfiltrazione

Per capire meglio come difendersi dall'esfiltrazione dei dati esporrò una tassonomia dei più comuni metodi utilizzati. Ci sono molte altre tecniche rispetto a quelle presentate nel seguito del capitolo, sono infatti potenzialmente infinite i modi esfiltrare dati. Lo scopo di questa lista è quello di chiarire in dettaglio alcuni dei metodi utilizzati per l'esfiltrazione. Questi saranno i principali, ma non i soli, obiettivi della tecnica di rilevazione da me presentato.

Per ogni metodo descriverò le seguenti caratteristiche:

- Una breve descrizione generale del protocollo utilizzato
- Come viene sfruttato per l'esfiltrazione
- Come è possibile rilevare l'utilizzo di tali tecniche

2.1 File Transfer Protocol (FTP)

Protocollo

File Transfer Protocol è un protocollo di rete utilizzato per il trasferimento di dati tra un client e un server. La specifica ufficiale di questo protocollo si trova nell'RFC 959[8]. Come descritto nell'introduzione del documento di specifica, gli obiettivi principali di questo protocollo sono: promuovere la condivisione dei file (programmi o dati), Incoraggiare l'uso indiretto o implicito di computer remoti, risolvere incompatibilità tra differenti sistemi e trasferire dati in maniera affidabile ed efficiente. FTP è utilizzato per il trasferimento di file di grandi dimensioni, tuttavia è in crescita l'utilizzo di altri protocolli, come HTTP, in alternativa per il medesimo scopo.

Il protocollo FTP comprende una prima fase di autenticazione, seguita da una fase di scambio dei dati per poi terminare chiudendo la connessione. Sono utilizzati due canali di comunicazione differenti. Uno per i comandi e l'altro per il trasferimento dei dati.

In entrambe i canali non è previsto nessun tipo di riservatezza per i dati in transito. L'invio di comandi e dati in chiaro rappresenta un serio problema

di sicurezza in quanto, intercettando i pacchetti, è possibile conoscere le informazioni utilizzate dal client per autenticarsi con il server.

Utilizzo per L'esfiltrazione

Una caratteristica di questo protocollo, utilizzabile per rendere meno rilevabile l'esfiltrazione, è la possibilità di decidere quale tra i due host connessi si comporterà da client e inizierà la connessione. Questa flessibilità avvantaggia un eventuale attaccante permettendogli di by-passare alcune restrizioni sulle connessioni in uscita dovute alla presenza di firewall.

Per una macchina infettata che utilizza questo metodo di esfiltrazione, i dati sono solitamente incapsulati in un qualche formato binario(con la possibilità di utilizzare compressioni o criptazioni dei dati). Inoltre il client deve autenticarsi con il server. L'indirizzo e i dettagli di autenticazione possono essere determinati sia staticamente che dinamicamente[9]. In alternativa può essere utilizzato un login anonimo che espone però il server a problemi di sicurezza in quanto chiunque può connettersi ad esso.

Il protocollo FTP può essere implementato 'a mano' in C utilizzando i socket oppure si può utilizzare una libreria preesistente.

Altre tecniche per ridurre la possibilità di rilevamento di questo processo possono essere adottate, dall'attaccante, nell'impostazione del server. Si può predisporre che il server accetti connessioni durante un determinato periodo, nel quale è previsto di ricevere i dati esfiltrati.

Rilevazione

Essendo un protocollo basato sul testo, se vengono esfiltrati dati sensibili, molte soluzioni di monitoraggio delle reti possono rilevare questo processo. In caso di utilizzo di tecniche di oscuramento o criptazione dei dati la rilevazione può essere più difficoltosa.

L'utilizzo di una whitelist[3] è uno degli approcci più convenienti per prevenire l'esfiltrazione dei dati tramite FTP. Questo approccio necessita di un

[3]Una lista di entità accertate, nel caso specifico di indirizzi

ispezione del traffico in uscita dalla rete per verificare se i pacchetti FTP sono destinati a server approvati dalla whitelist. Se un utente ha bisogno di utilizzare questo protocollo verso un server non presente nella lista può contattare un amministratore di sistema per far approvare la comunicazione. In alternativa è possibile utilizzare metodi come RegEx[4] o generare un allerta in caso di trasferimento di dati criptati all'interno di una connessione FTP.

2.2 Hypertext Transfer Protocol (HTTP)

Protocollo

Hypertext Transfer Protocol[10] è un protocollo a livello applicativo usato come principale sistema di trasmissione d'informazioni tra un client e un server. HTTP è utilizzato dai web browsers per accedere ai siti internet e comunicare con i web server. Essendo uno dei protocolli più comunemente utilizzato all'interno delle reti è spesso scelto come metodo per l'esfiltrazione dei dati. Il grande volume di traffico HTTP presente nella rete permette di nascondere più facilmente la trasmissione di dati. La struttura di comunicazione di questo protocollo presenta diversi vantaggi per un attaccante. Può essere utilizzato per facilitare il controllo di un host compromesso per mezzo di traffico non facilmente rilevabile. Inoltre permette il trasferimento diretto di dati tra due host con facili meccanismi di controllo dell'integrità.

Utilizzo per L'esfiltrazione

Il metodo standard per inviare file, eventualmente frammentati, è l'utilizzo di una richiesta POST. Il dato è posto nel corpo della richiesta e inviato ad uno specifico URL[5] verso un web server configurato per gestire la richiesta. Questo metodo è non criptato e originariamente non pensato per l'invio di grandi dati. Più tardi sono state effettuate delle modifiche per soddisfare le recenti necessità di uploads di file su un server[11]. Adesso i dati possono essere codificati specificatamente per questo tipo di

[4]Regular expression, ricerca di pattern all'interno del traffico[12]
[5]Uniform Resource Locator

trasferimenti(multipart/form-data). Il server scriverà i dati ricevuti seguendo il corrispettivo attributo nel campo ACTION URL.

Il pacchetto necessario per inviare una richiesta POST può essere creato facilmente utilizzando un socket TCP. Nel caso in cui i dati siano di grandi dimensioni è necessario inviare POST multiple e il server deve ricostruire i dati frammentati che ha ricevuto.

Un attaccante deve considerare che la maggior parte delle applicazioni web che accettano l'upload dei file hanno un limite di grandezza, dell'ordine dei megabyte.

Per ricevere i dati è necessario configurare un server HTTP per gestire le richieste POST. Ci sono diversi accorgimenti, nella configurazione del server, che possono essere utilizzati dall'attaccante per diminuire le possibilità di rilevamento del processo di efiltrazione. Per esempio è possibile predisporre il server in modo che simuli una corretta risposta alle richieste POST. Inoltre, in questo metodo, non è presente nessun tipo di autenticazione tra client e server. Questa può essere implementata in diversi modi, ad esempio impostando il server in modo che risponda soltanto alle richieste che contengono una determinata stringa utilizzata come password.

Rilevazione

Gli attaccanti che utilizzano questo metodo tipicamente comprimono o codificano i dati prima di esfiltrarli poiché HTTP è un protocollo testuale e quindi trasmette i dati in chiaro.

Il processo di esfiltrazione dei dati utilizza solitamente la richiesta POST senza nessun altro tipo di richieste precedenti. Una normale comunicazione che utilizza il comando POST comprende anche una precedente richiesta di un oggetto HTML effettuata dal client. Monitorando le comunicazione si può individuare l'invio di richieste POST verso server con il quale non ci sono state comunicazioni precedenti. Un attaccante cerca solitamente di limitare l'esposizione delle infrastrutture da lui utilizzate per minimizzare la probabilità di essere individuato. Minore è l'interazione tra host compromesso e server minore è la possibilità di individuare la comunicazione. Rispetto

a queste considerazioni, per un sistema di sicurezza di una rete, un utilizzo di richieste POST senza precedenti comunicazioni può essere considerato un attività sospetta. Questo approccio presenta un elevato rischio di falsi positivi in quanto ci sono diversi scenari di comunicazioni legittimi in cui questo tipo di interazione avviene. L'utilizzo di una whitelist di domini e indirizzi IP può essere molto utile per rilevare esfiltrazioni che utilizzano HTTP.

Un altro approccio utilizzabile è quello di monitorare la durata delle connessioni TCP tra client e server, e anche la quantità di dati scambiati durante queste sessioni. Calcolando dei valori comuni per queste misurazioni, è possibile generare degli allarmi in caso di un utilizzo non standard rispetto a queste metriche. Anche in questo scenario il rischio di falsi positivi è alto.

2.3 Secure Socket Layer (SSL)

Protocollo

Secure Socket Layer[13] è un protocollo che fornisce sicurezza a livello trasposto tra il TCP e le applicazioni che lo utilizzano. Fra i sevizi forniti da SSL vi sono l'autenticazione di client e server, la riservatezza e l'integrità dei dati. I programmi client/server di livello applicazione, come HTTP, che utilizzano i servizi di TCP possono incapsulare i propri dati in pacchetti SSL (HTTPS)[14]. Una coppia di chiavi pubblica/privata sono utilizzare per la sicurezza nello scambio delle chiavi simmetriche che saranno utilizzate per la criptazione in entrambi i lati della comunicazione. Il modo più praticabile per esfiltrare i dati con questo metodo è l'utilizzo della richiesta POST HTTP. Così facendo si aumentano i vantaggi dell'utilizzo dell'HTTP in termini di sicurezza e capacità di rimanere non individuati.

Utilizzo per L'esfiltrazione

Il tipo di esfiltrazione è praticamente identico a quello che utilizza l'invio di una HTTP POST, dove è richiesto l'upload di un file verso un determinato servr. Inizialmente deve essere svolta la fase di SSL handshake, che serve per determinare gli algoritmi di crittografia utilizzati e il tipo di socket. I

dati esfiltrati dovranno essere incapsulati utilizzando una qualche forma di compressione. Come per il metodo precedente, per ricevere i dati è necessario configurare un srver HTTP o un servizio che lo emula. Inoltre è necessario l'utilizzo di un certificato valido per la connessione. Ci sono diverse soluzioni per ottenere un tale certificato. Per esempio è possibile acquistarne uno validato da una terza parte. Questo renderà meno sospetta la connessione.

Rilevazione

Questo tipo di esfiltrazione è più difficile da rilevare rispetto ad altre per diverse ragioni. Il tipo di traffico generato è uno dei più comuni all'interno di una rete e quindi i dati esfiltrati risultano ben nascosti tra il traffico legittimo. Il client compromesso può connettersi ad diversi HTTPS server rendendo l'utilizzo di una withlist più difficoltoso. Inoltre la criptazione dei dati rende l'ispezione del contenuto dei pacchetti impraticabile. Per questa ragione solo alcuni metodi di rilevazione, basati sul comportamento tipico dei sistemi della rete, possono riscontrare delle anomalie nel traffico dovute all'esfiltrazione con HTTPS.

2.4 Email

Protocollo

L'utilizzo di email per trasmettere file sfrutta il protocollo Simple Mail Transfer Protocol (SMTP)[15]. Questo metodo consiste nell'allegare un file ad un email e provare ad inviarla verso un server.

Utilizzo per L'esfiltrazione

Esistono diversi metodi di utilizzo di questo protocollo. Uno di questi è quello che prevede l'uso di un mail server e indirizzo email di un utente interno alla rete per trasmettere i dati. Altri metodi sono possibili, ma significativamente più complessi da implementare. Per esempio è possibile

utilizzare un programma che effettua il login in un servizio di webmail e lo utilizza per esfiltare i dati.

I metodi che richiedono la conoscenza delle credenziali di un utente sono di difficile realizzazione. Risulta più comune l'utilizzo di un mail server arbitrario a cui l'attaccante tenta di connettersi per inviare i dati esfiltrati. In questo caso non è necessario avere un indirizzo email a cui destinare i dati dell'esfiltrazione.

Rilevazione

Normalmente un utente si connette ad un piccolo numero di mail server. Questo rende molto semplice rilevare connessioni verso mail server non utilizzati in precedenza. Inoltre l'ispezione dei dati in uscita dalla rete può rilevare parole chiavi o espressioni regolari all'interno del traffico provocando un allarme.

2.5 Domain Name System (DNS)

Protocollo

Domain Name System[17] è un componente di internet che associa i nomi di dominio con i corrispondenti indirizzi IP necessari per la corretta consegna dei pacchetti. In una richiesta DNS viene inviata una stringa, che rappresenta un host, ad un server DNS per conoscere l'indirizzo IP corrispondente. Questo risponderà con l'indirizzo IP oppure informando il mittente di non conoscere l'associazione richiesta. In caso di sotto-domini[6] di host, il DNS server o il client[7] invierà la richiesta ad un Top Level Domain(TLD)[8] generico conosciuto. Questo aspetto rende possibile l'invio di dati attraverso l'utilizzo del protocollo DNS. Un host compromesso può richiedere la risoluzione per un stringa che rappresenta un sotto-dominio che fa parte di un TLD dell'attacante. Il server DNS risponderà che il sotto-dominio non esiste e registrerà

[6]Lo spazio dei nomi di dominio ha una struttura gerarchica.
[7]Dipende dal tipo di risoluzione utilizzata:iterativa o ricorsiva
[8]Uno dei domini con il più alto livello nella gerarchia.

la stringa di dati contenuta nella richiesta. All'interno di questa si possono inserire i dati da esfiltrare.

Utilizzo per L'esfiltrazione

Il primo passo per esfiltrare dati tramite questa tecnica è creare un DNS record contenente uno specifico nome di dominio che, per la risoluzione, deve essere inviato ad un DNS server sotto il controllo dell'attaccante. Dopo aver settato e testato il record, l'attaccante imposterà il server per agire come un DNS server autoritativo. A questo punto è in grado di esportare i dati inviando un arbitrario numero di interrogazioni dall'host compromesso. Ogni richiesta conterrà un frammento dei dati rubati nella parte contenente il sotto-dominio della richiesta.

Per esempio, per esfiltrare numeri di carte di credito con i rispettivi codici di sicurezza e data di scadenza, una stringa della interrogazioni che include i sotto-domini potrebbe risultare la seguente:
$1538495484571264x6884x978.esempio.com$
$4860395685471256x4775x243.esempio.com$
$3754009576124958x2231x265.esempio.com$
$3754956743956471x9651x466.esempio.com$

Il dominio *esempio.com* deve essere sotto il controllo dell'attaccante. Il sotto-dominio invece contiene le informazioni da esfiltrare. Esistono numerosi software gratuiti che rendano questo processo semplice per un attaccante, molti dei quali implementano tecniche, Lato server, di analisi che ricostruiscono o formattazione dei dati ricevuti.

Rilevazione

Uno dei più comuni indicatori di una esfiltrazione che utilizza questa tecnica è l'elevato numero di interrogazioni DNS complesse, verso uno stesso dominio e in un ristretto periodo di tempo. Ci sono però alcune situazioni legittime in cui questo scenario può presentarsi.

Se l'attaccante sta mirando a dati che contengono caratteri speciali avrà la necessità di codificarli prima di poterli inserire in una interrogazione DNS.

Questo perché i nomi di dominio possono contenere solo certi tipi di caratteri. Una codifica utilizzata è Base32[18] perché l'intero insieme di simboli utilizzati da questa codifica può essere parte di un legittimo nome di dominio.

2.6 Internet Control Message Protocol (ICMP)

Protocollo

Internet Control Message Protocol (ICMP)[16] è un protocollo di supporto che si occupa di trasmettere informazioni riguardanti malfunzionamenti, controllo o messaggi vari all'interno di una rete.

Un pacchetto richiesta eco[9] di ICMP è solitamente inviato da un utente per determinare se un hoste o un router è raggiungibile. Questo tipo pacchetto contiene normalmente poche informazioni. E' però presente un campo dati utilizzato nei messaggi di segnalazione degli errori. Questa parte può essere il contenitore per inviare i dati in un processo di esfiltrazione.

Utilizzo per L'esfiltrazione

I dati da trasmettere vengono inseriti come corpo di un pacchetto richiesta eco ed inviati ad un host. Normalmente è prevista la ricezione di un pacchetto risposta eco[10] contenente lo stesso identico contenuto della di richiesta. L'attaccante può però forgiare il pacchetto di risposta con un indirizzo sorgente non corretto e ovviare a questo inconveniente. Un aspetto importante da considerare è che il contenuto del pacchetto in questione è raramente ispezionato e non ha rilevanza nel normale utilizzo di questo protocollo. Lo standard prevede che la dimensione della richiesta eco sia di 56 *bytes*, tuttavia può essere anche maggiore. Rendere il pacchetto troppo grande aumenta la possibilità di rilevamento e quindi i dati dovranno essere frammentai e poi inviati. Il sistema ricevente ricomporrà l'informazione ricevuta e provvederà ad inviare la risposta eco per far apparire la comunicazione più autentica.

[9] echo request, tipo 8
[10] echo replay, tipo 0

Rilevazione

Il maggiore inconveniente per l'utilizzo di questo metodo per l'esfiltrazione dei dati risulta essere il grande quantitativo di traffico ICMP generato nella rete rispetto ad un normale utilizzo, in cui è solitamente basso. Algoritmi di rilevazione possono inoltre ispezionare i pacchetti e rilevare la presenza di dati sensibili. Anche la dimensione non comune dei pacchetti eco è un aspetto che può essere considerato sospetto in una rete e quindi generare allarmi.

3 Analisi e classificazione contromisure

Con la parola contromisura si definisce l'insieme delle azioni compiute per difendersi contro il problema dell'esfiltrazione dei dati. Queste, come già riportato in precedenza, sono solo una parte di un intero sistema di sicurezza di una rete. Ad un livello astratto le contromisure si possono dividere in tre categorie: tecniche di prevenzione, rilevamento e investigazione. In alcuni casi queste categorie si possono sovrapporre, una contromisura può prevenire e rilevare un esfiltrazione oppure un'altra può rilevare e investigare un tentativo di esfiltrazione. Da un esame del numero di studi effettuati in questo campo[19] è possibile riconoscere che la comunità di ricercatori si è concentrata maggiormente sulle tecniche di prevenzione e rilevamento invece che su quelle di investigazione. La mancanza di attenzione nel campo

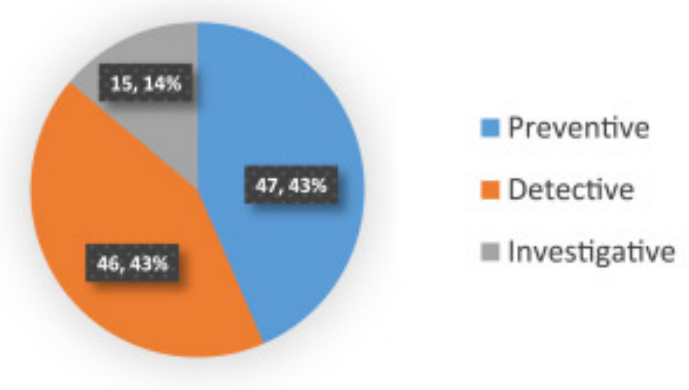


Figura 3: Percentuale di studi per ogni categoria

dell'investigazione può essere motivata dall'interesse primario da parte degli individui o delle organizzazione nel proteggere i dati e identificare quando e come possono essere rubati. L'obiettivo primario è quello di prevenire o stoppare eventuali processi di esfiltrazione.

Un altro aspetto da considerare nella classificazione delle contromisure è lo stato dei dati: in uso, in transito o in archivio (Figura 4). I dati in transito sono quelli che vengono trasmessi da un nodo della rete ad un altro nella stessa. Questi dati possono attraversare una o più reti oppure essere trasmessi direttamente tra due dispositivi. I dati in archivio sono quelli

immagazzinati per un utilizzo futuro. Generalmente si riferiscono a dati contenuti in database o file di sistema, utenti, applicazioni o di backup. Tali dati sono fisicamente allocati in sistemi di immagazzinamento fisici come hard drive o in infrastrutture remote come cloud. I dati in uso sono quelli presenti in memoria o che sono utilizzati in un qualsiasi processo(da un applicazione o da un utente). Per esempio i dati presenti in memoria di un dispositivo che fornisce un interfaccia per una applicazione.

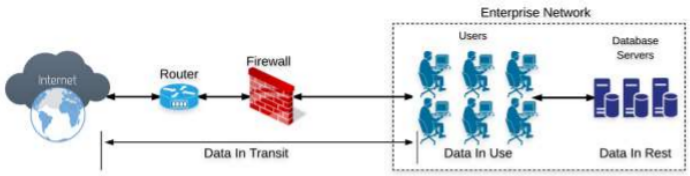


Figura 4: I tre stati dei dati

Nel seguito di questo capitolo presenterò una classificazione (Figura 5) delle contromisure come descritto nell'articolo[19]. Descriverò brevemente le categorie di prevenzione e investigazione, approfondendo in maggior dettaglio quella delle tecniche di rilevamento con particolare riferimento al monitoraggio della rete con utilizzo della tecnica di Deep Packet Inspection (DPI).

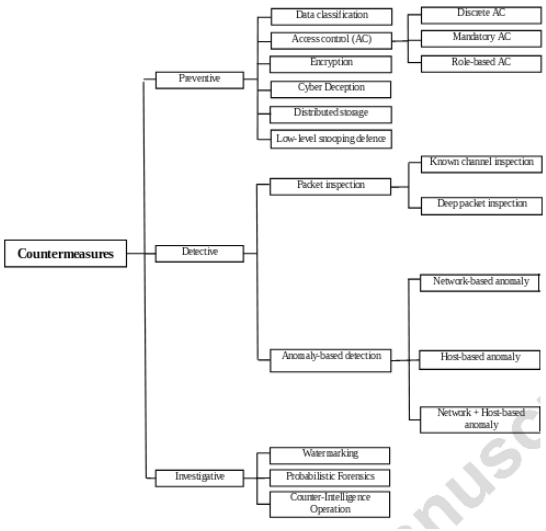


Figura 5: Classificazione contromisure per l'esfiltrazione dei dati

3.1 Prevenzione

Questa categoria include le contromisure per prevenire che un attaccante si impossessi di dati sensibili. Le tecniche in questione sono di natura preventiva e restrittiva con l'obiettivo di impedire i tentativi di esfiltrazione dei dati. Queste contromisure sono incorporate nei sistemi come: PC, Laptop o server. Hanno l'obiettivo di controllare l'accesso a dati che risiedono in questi sistemi o applicare alcune particolari tattiche di sicurezza(crittografia o classificazione dati) per rendere i dati più protetti. Le contromisure preventive sono applicabili principalmente ai dati in uso o a quelli immagazzinati in un database (dati archiviati). I dati sono suddivisi in differenti classi a seconda del loro valore, quelli più sensibili sono solitamente criptati per aumentare il livello di sicurezza. Inoltre è anche possibile inserire porzioni di dati fasulli in modo da depistare un possibile attaccante. Come mostrato in Figura 5 esistono diversi tipologie di contromisure preventive[19] che si suddividono in base alle tecniche utilizzate per proteggere i dati.

3.2 Investigazione

Una esfiltrazione di dati è un processo che nella maggior parte dei casi, ha delle conseguenze irreversibili. Nonostante ciò esistono sistemi che possono identificare quando, come e chi ha esfiltrato i dati. Questo tipo di analisi forense può essere utile per correggere alcune falle di sicurezza e attuare determinate misure dopo che una violazione è avvenuta. Le contromisure di investigazione analizzano come e quando i dati sono stati esportati e, se possibile, quali contromisure possono aiutare a tracciare l'attaccante. Investigare un esfiltrazione di dati può essere utile per mitigare gli effetti di un attacco. Inoltre le informazioni raccolte possono essere preziose per altre organizzazioni. Ovviamente, non è possibile condividere informazioni riservate, nonostante ciò alcuni aspetti dell'investigazione possono essere divulgati per migliorare le politiche di sicurezza. Queste contromisure sono state divise in tre categorie: Watermarking, Probabilistic Forensics e Counter-Intelligence Operation[19].

3.3 Rilevamento

Le contromisure in questa categoria hanno l'obiettivo di individuare i tentativi di esfiltrazione dei dati.

Le problematiche che queste contromisure affrontano sono rappresentate dall'individuare eventi rari che provocano conseguenze disastrose. Inoltre risulta complicato distinguere questi eventi rispetto ad un utilizzo legittimo dei servizi offerti dalla rete.

Presenterò due diverse categorie di contromisure: quelle che utilizzano il monitoraggio del traffico alla ricerca di specifiche condizione all'interno di esso e quelle che invece sfruttano un modello di utilizzo legittimo della rete per rilevare eventuali anomalie.

3.3.1 Ispezione Pacchetti

Un semplice approccio per rilevare un tentativo di esfiltrazione dei dati è rappresentato dall'ispezione del traffico in uscita dalla rete. L'obiettivo della ricerca è quello di rilevare la presenza, nel traffico, di dati sensibili attraverso l'utilizzo di pattern,parole chiave, informazioni personali o signatures. Per ogni canale di comunicazione vengono analizzati, in tempo reale, i pacchetti e se è riscontrata la presenza di dati sensibili viene generato un allarme per una possibile tentativo di esfiltrazione.

E' possibile ispezionare determinati canali di comunicazione che presentano un altro rischio di esfiltrazione di dati sensibili(come le Email). Un approccio di questo tipo non monitora tutto il traffico in uscita da una rete ma solo determinati canali in base a quale protocollo di comunicazione è utilizzato.

Un esempio di questo tipo è presentato nel lavoro di Jaskolka[21]. La tecnica proposta si basa su due fasi: la prima prevede il monitoraggio di canali di comunicazione conosciuti, il secondo calcola una relazione tra i dati osservati sul canale e le informazioni sensibili. Investigando queste relazione è possibile stabilire se c'è stata esfiltrazione o meno. La caratteristica principale di questo metodo è rappresentata dall'ispezione dell'intestazione dei

pacchetti ma non il contenuto (payload) degli stessi. Risulta critica anche il ritardo tra l'individuazione di un esfiltrazione e la rilevazione della stessa.

Un approccio diverso per questo problema si basa sull'utilizzo della tecnica di Deep packet Inspection che descriverò nel dettaglio nel prossimo paragrafo.

3.3.2 Deep Packet Inspection(DPI)

La Deep Packet Inspection (DPI) è una forma di filtraggio dei pacchetti dati che esamina il contenuto(payload) alla ricerca di informazioni non aderenti a determinati criteri prestabiliti.

La ricerca può essere effettuata per identificare e eventualmente agire su anomalie dei protocolli, intrusioni, propagazione virus o per ottimizzare il traffico sulle reti o per raccogliere dati statistici sull'uso della stessa. A differenza della Packet Inspection, gli apparati che implementano DPI non si limitano a controllare l'intestazione(header) dei pacchetti, ma anche il contenuto. La DPI opera spaziando dal livello 2 al livello 7 del modello OSI, analizzando il payload del pacchetto e identificando il contenuto in base a delle signatures contenute nel database di apparati specifici per identificare il tipo e la natura del traffico a prescindere dalle informazioni contenute nell'intestazione del pacchetto. Un pacchetto una volta classificato può essere rediretto, bloccato, taggato (QOS), limitato a livello di banda o semplicemente salvato e analizzato successivamente.

L'efficienza e diffusione delle tecniche di DPI ha innescato in alcuni ambiti polemiche legate all'utilizzo di questa tecnologia. Infatti l'utilizzo da parte di alcuni operatori di telecomunicazioni di sistemi di Deep Packet Inspection, per privilegiare il proprio traffico rispetto a quello di altri operatori o per limitare alcuni tipi traffico degli utenti sulla rete, è contestata nel mondo dai sostenitori della Neutralità della rete. In risposta tali operatori sostengono che la DPI sia necessaria per ragioni economiche o per ragioni di sicurezza (per bloccare la trasmissione di malware, virus, trojan o spam) e per proteggere gli utenti. Inoltre l'ispezione del contenuto(payload) dei pacchetti, all'interno di una rete privata, rende visibile l'informazione trasmessa. Que-

sto ha implicazioni nell'ambito della privacy degli utenti che utilizzano la rete stessa.

Come descritto in precedenza questa tecnologia può essere utilizzata per diversi scopi. Nell'ambito della sicurezza di una rete presenta molteplici utilizzi e, in particolare, può rivelarsi molto utile anche per il problema dell'esfiltrazione dei dati.

Con la DPI si possono realizzare diverse tecniche di tracciamento del traffico a seconda di come vengono raggruppati i pacchetti. Si può considerare ogni pacchetto singolarmente(Packet-Based No State) oppure considerare i pacchetti per flusso, ossia rappresentare la sessione di comunicazione instaurata da un applicazione tra due host con la quintupla: Ip sorgente, IP destinazione, Porta sorgente, Porta Destinazione, (Protocollo Packet-Based per Flow State).

La DPI opera abitualmente su costosi dispositivi, firwall con caratteristiche IDS/IPS su router avanzati, con elevate capacità di calcolo.

Sebbene i flussi di dati, siano normalmente verificati a velocità collegamente, nel caso non sia possibile capire il contenuto del payload all'istante, si può scegliere di raccogliere i pacchetti in memoria per poi esaminarli successivamente.

Il meccanismo della DPI per l'identificazione dei contenuti è basato sulla signature. Questa, è un'associazione delle caratteristiche di un applicazione software o di un protocollo, come un impronta digitale. Le referenze devono essere aggiornate periodicamente di pari passo con l'aggiornamento delle applicazioni e i nuovi sviluppi su protocolli esistenti. esistono diversi metodi per costruire una signature e si basano su criteri differenti, riassunti con i seguenti[23]: analisi sule stringhe, analisi sulle proprietà numeriche o analisi comportamentale e statistica. Nessuno di questi metodi, usato singolarmente, provvede all'identificazione di tutte le applicazioni o protocolli software, vengono infatti tipicamente utilizzati simultaneamente.

La diffusione di protocolli di comunicazione che utilizzano canali criptati è in costante aumento. Questo trend rende più difficoltoso l'utilizzo della DPI, che non ha la possibilità di leggere il contenuto(payload) dei pacchetti. Tuttavia è ancora possibili utilizzare questa tecnologia per il monitoraggio della

rete con ottimi risultati. Un possibile utilizzo della DPI per rilevare eventuali esfiltrazioni di dati, con riferimento a comunicazioni criptate, è presentato nell'articolo "A Novel Method to Detect Encrypted Data Exfiltration"[22]

3.3.3 Rilevamento di anomalie

Le tecniche basate sul rilevamento delle anomalie confrontano i dati osservati con dei modelli di comportamento(della rete o dell'host) precedentemente stabiliti come legittimi. Quando viene rilevata una deviazione rispetto al modello prestabilito, questa viene considerata un anomalia. I dati ispezionati comprendono diversi aspetti del traffico di rete o del comportamento di un host. L'ispezione delle attività della rete può essere attuata con diverse modalità. Si può adottare una strategia basata sul monitoraggio dell'host, una basata sul monitoraggio della rete oppure una combinazione tra le due.

Il monitoraggio basta sulla rete si concentra sul traffico per determinare se le comunicazioni differiscono dalle condizioni stabiliti nei termini delle comuni metriche come: volume, indirizzi sorgente/destinazione, uso di determinati protocolli o differenze negli orari di utilizzo della rete. Gli aspetti considerati per determinare un modello di utilizzo legittimo possono riguardare anche dettagli particolari riguardanti le connessioni. Una deviazione rispetto al comportamento atteso viene considerata come un anomalia e come un possibile segno di esfiltrazione. Le difficoltà che riscontrano le tecniche utilizzate in questa categoria riguardano la stima di un utilizzo 'normale' della rete. Non è solo molto difficile stimare i comportamenti legittimi in un organizzazione, che spesso producono impredicibili picchi nell'utilizzo della rete. C'è anche un altro rischio di falsi positivi che generano allarmi, causando un danno alle attività degli utenti o dell'organizzazione.

Queste contromisure si basano sull'analisi del traffico di rete(analizzando i pacchetti) e utilizzando la DPI o un analisi teorica delle informazioni. Nel lavoro [24] è proposto un framework per il rilevamento di esfiltrazione tramite un network bridge[11] trasparente al limite esterno della rete. Gli autori

[11]Il bridge (letteralmente ponte) è un dispositivo di rete che si colloca al livello datalink del modello ISO/OSI e che traduce da un mezzo fisico a un altro all'interno di una stessa rete locale

applicano un analisi statistica e alcune tecniche di signal processing sul flusso del traffico per generare delle signature e/o estrarre caratteristiche per fini di classificazione. In [25] si utilizza la teoria degli N-grammi[12] per classificare i dati. Il metodo si basa sulla frequenza degli N-grammi per classificare i documenti in modo da prevenire eventuali esfiltrazioni. In [26] gli autori esplorano l'utilizzo di alcune caratteristiche dell'entropia[27] del traffico di rete per determinare se la comunicazione è criptata. In una seconda fare decide, in base a modello di utilizzo del traffico criptato, se è presente un anomalia o meno.

Invece di monitorare il traffico della rete(i pacchetti) è possibile avere una visibilità delle comunicazioni monitorando le attività, a livello di sistema, dei singoli host. L'aumento della velocità della rete combinato da un crescente uso di crittografia dei dati rende più difficoltosa l'ispezione dei pacchetti non solo in termini di prestazioni ma anche in termini di visibilità delle caratteristiche del traffico di ree da parte del sistema di monitoraggio. Oltre a queste problematiche si aggiunge il crescente utilizzo di virtualizzazione a livello di sistema operativo, pratica che crea difficoltà aggiuntive ad un monitoraggio basato sui pacchetti. Infatti, la virtualizzazione, è utilizzata anche a livello di rete, dove le strutture fisiche possono essere sostituite da componenti software. E' quindi possibile che il traffico scambiato tra processi in esecuzione in ambienti virtualizzati diversi, ma su uno stesso sistema fisico, non sia visibile monitorando l'interfaccia fisica del sistema stesso. Sarebbe teoricamente possibile adattare il monitoraggio basato sui pacchetti ad un ambiente virtualizzato. Così facendo si avrebbe però un degrado delle prestazioni molto alto. Inoltre una delle caratteristiche principali di questi ambienti è la struttura dinamica, aspetto che non si lega agevolmente con la struttura statica della configurazione dei sistemi di monitoraggio. Le problematiche appena descritte hanno spinto la ricerca di tecniche di monitoraggio che si concentrano sugli eventi al livello dell'host per determinare se sono presenti anomalie. Un possibile approccio è quello che si basa sulla sequenza delle system call e delle operazioni su file come presentato le lavoro di J. Beaver e F. Jewell[20].

Un utilizzo combinato dei due approcci può aumentare le capacità di rile-

[12]Un n-gramma è una sottosequenza di n elementi di una data sequenza.

vamento di anomalie, che possono essere determinate in base ad un modello che trae i dati l'utilizzo della rete che quello a livello di sistema degli host presenti in essa.

4 Progetto

Nel capitolo 3 ho presentato una classificazione di alcune delle possibili contromisure per il problema dell'esfiltrzione concentrandomi i particolare sul rilevamento di questo fenomeno attraverso il monitoraggio della rete.

In questo capitolo presenterò un semplice progetto da me sviluppato con l'obiettivo di studiare un caso concreto di utilizzo di un sistema di rilevamento del traffico di rete applicato al problema dell'esfiltrazione dei dati.

La soluzione che propongo rientra nella categorie delle contromisure di rilevamento della presenza di un processo di esfiltrazione. Utilizzerò la tecnica della DPI per visionare il contenuto del traffico di rete alla ricerca di testo in formato leggibili per un umano. In caso affermativo verrà generato un allarme relativo al flusso dei pacchetti considerati. I dati che sono obiettivo di questa ricerca sono quelli classificati nel capitolo precedente come in transito.

Il linguaggio utilizzato per sviluppare il progetto è il C, ho sfruttato la libreria nDPI per l'implementazione della deep packet inspection e per eseguire i test sul codice. Inoltre ho fatto uso del programma Wireshark per ottenere e decodificare i file da utilizzare come input per test.

4.1 Strumenti

Segue una descrizione degli strumenti utilizzati in fase di sviluppo e di test del software.

4.1.1 nDPI

Nel seguente paragrafo presenterò una breve descrizione della libreria utilizzata basandomi sulla documentazione ufficiale[30].

La libreria OpenDPI è stata il punto di partenza per lo sviluppo della libreria nDPI. Scritta in linguaggio C, prevede due componenti principali:

il *nucleo*, responsabile della cattura del pacchetto, decodifica dei livelli 3/4 del modello OSI e estrazione delle informazioni di base come indirizzi IP e porte. I *dissector* sono invece i componenti che consentono il riconoscimento dei protocolli.

nDPI ha mantenuto questa architettura di base estendendo e migliorando alcuni aspetti della libreria OpenDPI come:

- Flessibilità delle strutture dati che permette di estenderle in caso di aggiunta di nuovi protocolli o funzionalità.
- Supporto per protocolli criptati.
- Supporto per il multiThreading con implementazione efficiente.
- Gerarchia dei *dissector* che permette di seguire un ordine 'intelligente' di utilizzo degli stessi.
- Possibilità di estrarre metadati, essenziali per capire i protocolli applicativi utilizzati con HTTP.
- Configurazione a runtime per i protocolli.

La libreria nDPI può essere utilizzata da qualsiasi applicazione grazie ad la scelta di non utilizzare un determinato set hardware, caratteristica che la rende più flessibile e scalabile.

Lo scopo principale della libreria è la classificazione del traffico di rete su due livelli: uno per i protocolli standard e l'altro per quello proprietario. Sono stati creati molti *dissector* per i protocolli proprietari come ad esempio quelli per riconoscere traffico di Skype, Whatsapp o Netflix.

Un protocollo di livello applicativo è definito in nDPI da la coppia $(ProtocolID, ProtocolName)$. L'insieme dei protocolli riconosciuti è molto ampio e spazia da quelli di livello rete, come SMNP o DNS, fino a quelli applicativi tra i quali sono presenti protocolli proprietari come Facebook o Twitter o altri.

Il *dissector* è un modulo, scritto in C, che è utilizzato per riconoscere se il pacchetto è classificabile o meno con il protocollo associato. Si può utilizzare

anche una classificazione basata su porta/protocollo, indirizzo IP o metadati estratti.

La struttura base su cui la libreria esegue le operazioni è il flusso. Un flusso è una $5 - tupla$ composta da (Source IP address, Destination IP address, Source Port, Destination Port, Transport Protocol ID). Ogni flusso contiene diverse informazioni aggiuntive ricavate dai pacchetti. Inoltre viene mantenuto lo stato dei *dissector* utilizzati e successivamente scartati attraverso una maschera di bit. Il flusso viene suddiviso in due sotto-strutture distinte, una per i flussi TCP e l'altro per quelli UDP.

Il ciclo di vita del processo di classificazione può essere schematizzato nel modo seguente

- Quando un pacchetto viene consegnato all'applicazione, si valuta la presenza dell'Header di rete o meno(pacchetti ARP scartati).
- Successivamente è prevista la decodifica dei livelli 3/4 (Livelli Trasporto e Rete modello OSI), da cui si estraggono le informazioni su porte e indirizzi IP.
- In caso ci sia un *dissector* registrato per il protocollo/porta del pacchetto, questo viene applicato per primo.
- In caso di insuccesso, tutti i *dissector* registrati per il protocollo del pacchetto sono utilizzati. Se l'esecuzione di uo di questi non ha successo si aprono due opzioni: Settare il *dissector* nella maschera dei protocolli esclusi oppure lasciare la classificazione al pacchetto successivo del flusso. In caso di fallimento di tutte i *dissector* il pacchetto verrà etichettato come *Unknown* e si tenterà la classificazione con il metodo del guessig, ovvero valutando se le porte o gli indirizzi sono noti
- Il riconoscimento del protocollo termina se l'applicazione di un *dissector* ha successo.

Un domanda che viene spontaneo farsi è :quanti pacchetti servono alla libreria per poter classificare il traffico? La risposta, determinata dall'e-

sperienza di utilizzo, è dipendete dal protocollo. Per la maggior parte dei protocolli che utilizzano UDP basta solitamente un solo pacchetto. Quelli basati su TCP richiedono invece un maggior numero di pacchetti. Esistono però diverse eccezioni come ad esempio per il protocollo BitTorrent.

4.1.2 Wireshark

Wireshark[31] è un software gratuito per l'analisi dei pacchetti. E' usato per la risoluzione di problemi e l'analisi di una rete, per lo sviluppo di software o protocolli di comunicazione.

Il programma è utilizzabile su tutti i più comuni sistemi operativi e, a differenza di altri strumenti come tcpdump, ha un interfaccia grafica e diverse funzionalità per l'ordinamento e il filtraggio dei pacchetti. Esiste anche una versione basata sul terminale (non-GUI) chiamata TShark. La distribuzione di questo software è realizzata sotto i termini della licenza GNU General Public License.

Wireshark è u programma di cattura che decodifica la struttura di diversi protocolli di rete. E' in grado di analizzare e visualizzare i campi dei pacchetti di rete. Per catturare i pacchetti viene utilizzata la API pcap[32]. Questa è implementata da diverse librerie a seconda del sistema operativo utilizzato (in Linux è la libreria libpcap[32]).

Alcune delle principali funzionalità che mette a disposizione sono:

- I dati possono essere catturati da una connessione live o letti da un file di pacchetti precedentemente catturati
- I dati live possono essere catturati da differenti tipi di reti come Ethernet, IEEE 802.11, PPP, o loopback.
- I dati visualizzati possono essere raffinati utilizzando dei filtri.
- Plug-ins possono essere creati per rilevare nuovi protocolli

Nel mio lavoro questo strumento è stato utilizzato per ottenere e studiare i file di cattura(.pcap o .pcapng), contenenti il traffico di rete, usati nella fase di test del progetto.

4.2 Implementazione

Il progetto è composto dalla funzione di rilevamento e dal tool che, sfruttando la libreria nDPI, esegue la funzione sul contenuto dei pacchetti. In base al risultato della ricerca di testo in forma leggibile ad un umano e dal protocollo di comunicazione utilizzato può essere generato un allarme che marca la connessione come non sicura.

4.2.1 ndpi_has_human_readeable_string

la funzione ndpi_has_human_readeable_string è utilizzata per scansionare il contenuto dei pacchetti alla ricerca di testo in forma leggibile da un umano. Il payload del pacchetto viene passato alla funzione sotto forma di una buffer di byte. Il contenuto è estratto un singolo byte alla volta per formare una stringa. Questa ha una dimensione minima per poter essere considerata tale. Nel mio codice la dimensione è impostata a quattro caratteri. Ho scelto questo valore perché è risultato sperimentalmente il più adatto. Valori più bassi di questo porta ad un altra probabilità di casi favorevoli, scegliendo invece valori grandi non si ottiene un riscontro nella maggior parte dei casi.

Per determinare se la stringa è in un formato leggibile ho utilizzato due livelli di filtraggio. Il primo ripulisce la stringa da tutti i caratteri che non sono alfanumerici(utilizzando la codifica ASCII). Il secondo invece utilizza la teoria dei $bigrammi$(caso particolare della teoria degli $N-grammi$[28]) per eliminare stringhe di caratteri senza senso. Un bigramma è una sequenza di due caratteri consecutivi. Ho utilizzato un vocabolario abbastanza ristretto di bigrammi che non possono essere presenti in parole in lingua inglese. Una corrispondenza di due caratteri con un elemento di questo vocabolario determina lo scarto dei caratteri che non apparterranno dunque ad una stringa sensata per un umano. Questo approccio è un euristica e quindi non rende il filtraggio esatto in termini di corrispondenza con le parole del vocabolario inglese. L'utilizzo dell'euristica è reso necessario dalle necessità di efficienza che deve avere la funzione per poter essere utilizzata all'interno della DPI. Per questo motivo non è possibile utilizzare un vero e proprio dizionario di parole.

4.2.2 ndpiReader

L'applicazione ndpiReader è una demo sviluppata come esempio di utilizzo della libreria nDPI. Questo tool implementa alcune delle principali funzionalità che offre la libreria.

Sono partito da questa implementazione di base aggiungendo l'utilizzo della funzione ndpi_has_human_readeable_string(integrata nella libreria) che viene applicata ad ogni pacchetto. I risultati vengono inseriti all'interno delle informazioni che sono associate ad ogni flusso. Al termine dell'elaborazione è possibile visualizzare, per ogni flusso, tutte le informazioni raccolte con l'aggiunta del numero di pacchetti che contengono testo in formato leggibile da un umano. Inoltre viene generato un warning nel caso in cui la comunicazione è ritenuta parte di un fenomeno di esfiltrazione di dati.

NdpiReader può essere utilizzato per la cattura live da una delle interfacce di rete oppure può leggere il contenuto di un file di cattura (.pcap o .pcapng).

Il flusso di esecuzione di questo tool, nella parte di interesse per i miei scopi, può essere schematizzato come segue.

In caso di lettura da un file viene creato un thread con il compito di scansionare tutti i pacchetti contenuti all'interno di questo utilizzando la funzione di libreria pcap_loop. Ogni pacchetto viene processato applicando la funzione ndpi_workflow_process_packet. Questa ha il compito di decodificare i livelli livelli 3/4 e collezionare le informazioni in apposite strutture determinando anche il protocollo applicativo utilizzato e a quale flow appartiene il pacchetto.

Prima di passare alla fase di elaborazione del pacchetto applico la funzione di scansione ndpi_has_human_readeable_string e in caso di risultato positivo viene settato un flag nelle struttura di supporto al lavoro del thread.

Nella fase di decodifica vengono raccolte tutte le informazioni disponibili nel pacchetto(indirizzi IP e porte mittente/destinatario). Si cerca di determinare il protocollo di livello applicativo e in base a questo, oltre a indirizzi e porte, è possibile stabilire l'appartenenza ad un flusso esistente oppure la necessità di crearne uno nuovo.

E' in questa fase che viene incrementato il contatore di pacchetti con testo

in formato leggibile ad un umano.

Ho scelto di non considerare in pacchetti che appartengono ad un flusso che utilizza il protocollo SSL. Questa scelta deriva dal fatto che SSL è un protocollo che utilizza la crittografia per proteggere i dati che vengono trasmessi. Non è quindi possibile determinare la presenza di testo nel formato stabilito inoltre l'applicazione della funzione ndpi_has_human_readeable_string produce un risultato negativo nella quasi totalità dei pacchetti appartenenti a un flusso SSL. I pochi casi in cui viene rilevato del testo in formato leggibile ad un umano sono quelli in cui vengono analizzati i primi pacchetti di un flusso SSL, quelli che appartengono alla fase di handshake. All'interno di questi sono presenti le informazioni utili alla definizione dei parametri crittografici che possono essere riconosciute dalla funzione che ho sviluppato. L'informazioni che si ricavano nell'ispezione dei pacchetti SSL, con il metodo che ho stabilito, non sono di nessun valore al fine di rilevare un eventuale presenza di un processo di esfiltrazione dei dati.

4.3 Scenari di utilizzo

L'obiettivo di questo lavoro è la rilevazione di testo in formato leggibile da un umano all'interno del traffico. I protocolli più comunemente utilizzati in una rete si basano su messaggi di testo, come ad esempio HTTP. Ovviamente monitorare le comunicazioni che utilizzano questi protocolli, con lo strumento che ho sviluppato, non può dare informazioni utili su la possibile presenza del fenomeno di esfiltrazione in quanto è normale rilevare contenuti in forma leggibile per un umo all'interno dei pacchetti.

Esistono tuttavia diversi contesti in cui è possibile riscontrare che il traffico contiene dei dati che normalmente non dovrebbero essere presenti, in caso di utilizzo legittimo del protocollo di comunicazione.

Con riferimento ai metodi descritti nel capitolo 2 è possibile applicare il software presentato in caso di utilizzo del protocollo ICMP. Questo rientra nella categoria dei canali di comunicazione in chiaro che non dovrebbero contenere testo in forma leggibile ad un umano. In caso contrario è molto

alta la probabilità che sia in atto un processo di esfiltrazione di dati da una rete.

4.4 Validazione e Test

Per effettuare un processo di validazione del software presenterò i risultati dell'esecuzione su diversi file di cattura[13] che rappresentano alcuni specifici scenari di utilizzo della rete.

Per verificare se il tool riesce ad individuare effettivamente la presenza ti testo in forma leggibile ad un umano utilizzerò un file di cattura SNMP[29]. L'informazione a cui sono interessato è contenuta all'interno del campo *community* presente in ogni unita dati del protocollo(PDU). Questa stringa è utilizzata dal router, che riceve la richiesta, per determinare se l'accesso ai dati è consentito. Rappresenta quindi un ID utente o una password che deve essere fornita per le comunicazioni SNMP. La stringa cercata è trasmessa in chiaro è quindi dovrebbe risultare presente nei pacchetti scansionati con ndpiReadr.

Come possiamo vedere utilizzando Wireshark (Figura 6), all'interno della richiesta *get_next*, nel campo *community* è presente la stringa *timer*.

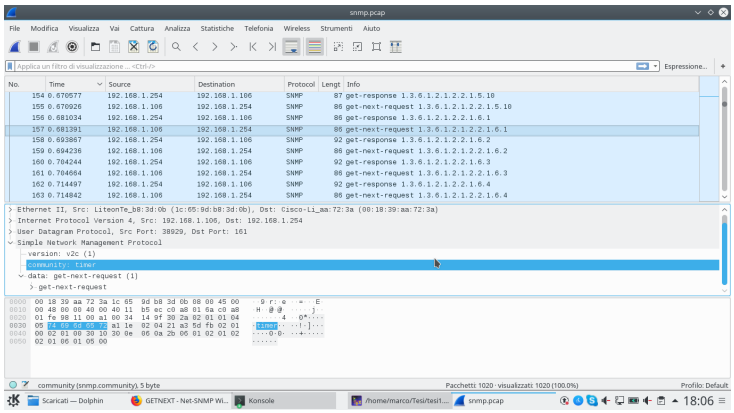


Figura 6: Analisi traffico SNMP

[13]ottenuti dal sito https://wiki.wireshark.org

Eseguendo ndpiReader [14] con il file di cattura precedentemente mostrato in ingresso, viene riconosciuto il flusso di pacchetti SNMP e inoltre è identificata la stringa cercata all'interno delle richieste(Figura 7).

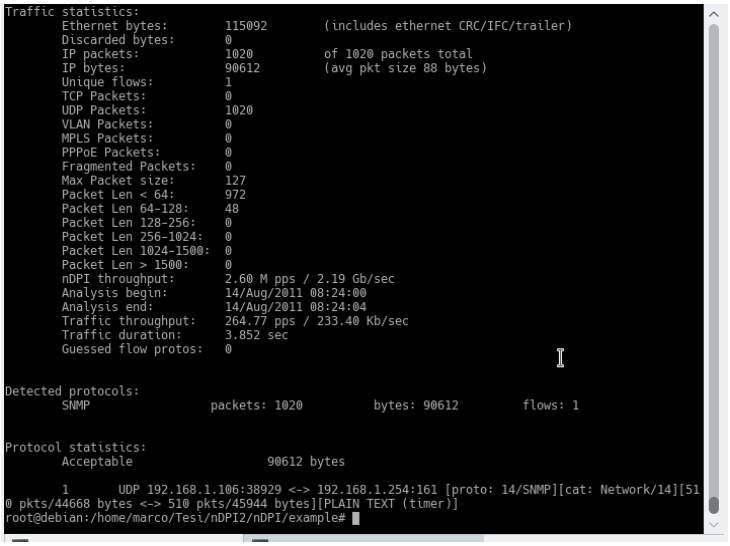


Figura 7: Flussi rilevati

Per verificare le funzionalità del progetto in uno scenario riguardante l'esfiltrazione dei dati ho effettuato il seguente test.

I due file di cattura presentano l'utilizzo del protocollo ICMP. Il primo riguarda un tipico uso del comando *ping*. Nel secondo invece, come spiegato nel capitolo 2.6, si utilizza la richiesta *eco* come vettore per trasportare dati.

Nel normale utilizzo del comando(Figura 8) non è prevista la presenza di dati all'interno del pacchetto, in particolare nel campo *data*. Si può infatti notare come i pacchetti abbiano una dimensione non superiore a 102 bytes, che rappresenta un valore tipico per la grandezza di questi pacchetti.

[14] lo specifico comando da eseguire è ./ndpiReader -i file.pcap -v 2

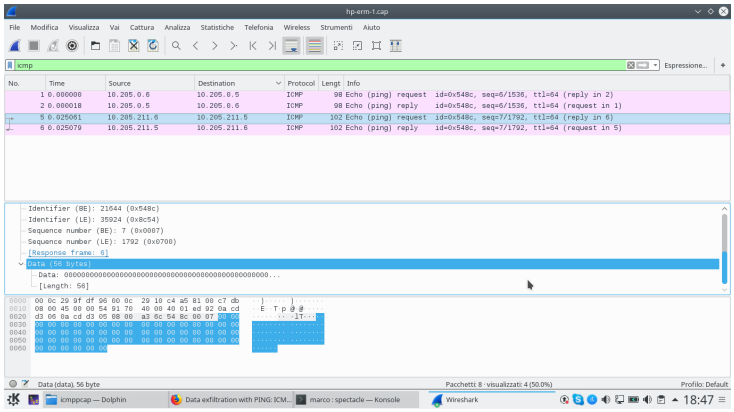


Figura 8: Analisi traffico ICMP

Di seguito(Figura 9) sono presentati i risultati dell'utilizzo del tool ndpiReader con in ingresso il file di cattura. Vengono individuati due differenti flussi, entrambi composti da due pacchetti. Il protocollo applicativo individuato è ICMP e il numero di pacchetti contenenti stringhe leggibili da un umano è zero.

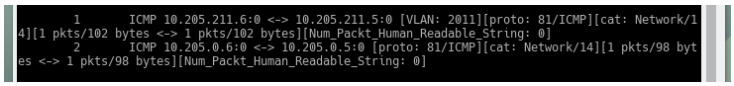


Figura 9: Statistiche e Flussi rilevati

Nel secondo file di cattura (Figura 10) invece è presente un utilizzo del protocollo ICMP con il fine di trasmettere dati. Si nota subito che la dimensione dei pacchetti è più grande(542 bytes) del previsto. Inoltre con l'aiuto di Wireshark si può vedere che nel campo *data* sono inseriti i dati da esfiltrare.

Eseguendo ndpiReader con questo file in ingresso otteniamo l'output mostrato in (Figura 11/12). Si può notare che, in riferimento al singolo flusso individuato, il numero di pacchetti che presentano testo in formato leggibile è uguale a 108. Questo riscontro, in combinazione con il riconoscimento del protocollo ICMP, genera un *warning* che segnala la presenza di un esfiltrazione nella connessione di riferimento.

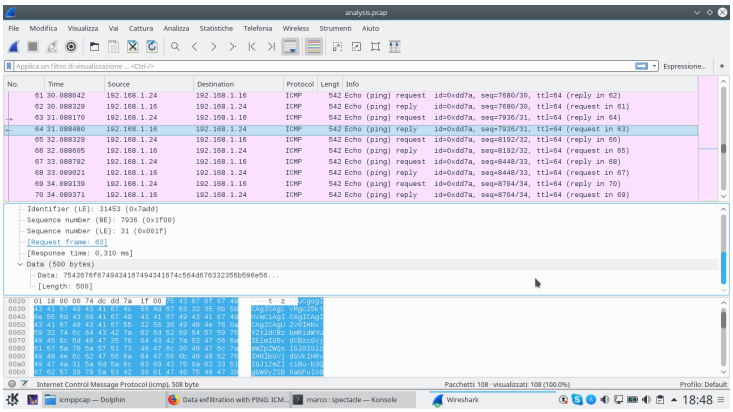


Figura 10: Analisi traffico ICMP durante esfiltrazione

```
Traffic statistics:
        Ethernet bytes:         61128           (includes ethernet CRC/IFC/trailer)
        Discarded bytes:        0
        IP packets:             108             of 108 packets total
        IP bytes:               58536           (avg pkt size 542 bytes)
        Unique flows:           1
        TCP Packets:            0
        UDP Packets:            0
        VLAN Packets:           0
        MPLS Packets:           0
        PPPoE Packets:          0
        Fragmented Packets:     0
        Max Packet size:        508
        Packet Len < 64:        0
        Packet Len 64-128:      0
        Packet Len 128-256:     0
        Packet Len 256-1024:    108
        Packet Len 1024-1500:   0
        Packet Len > 1500:      0
        nDPI throughput:        41.63 K pps / 179.79 Mb/sec
        Analysis begin:         10/Jun/2018 18:08:33
        Analysis end:           10/Jun/2018 18:09:36
        Traffic throughput:     2.03 pps / 8.99 Kb/sec
        Traffic duration:       53.096 sec
        Guessed flow protos:    0


Detected protocols:
        ICMP                    packets: 108            bytes: 58536            flows: 1


Protocol statistics:
        Acceptable                      58536 bytes
```

Figura 11: Statistiche del traffico rilevate

```
        1       ICMP 192.168.1.24:0 <-> 192.168.1.16:0 [proto: 81/ICMP][cat: Network/14][54 pkts/2
9268 bytes <-> 54 pkts/29268 bytes][Num_Packt_Human_Readable_String: 108]!WARNING!
root@debian:/home/marco/Tesi/nDPI/example#
```

Figura 12: Flussi rilevati

5 Lavoro futuro

Una naturale estensione del mio lavoro riguarda la raffinazione delle tecniche utilizzate per il riconoscimento di testo in formato leggibile da un umano. La tecnica dei bigrammi e solo un possibile approccio utilizzato nel campo dei modelli linguistici e riconoscimento di frasi. L'utilizzo di queste è però limitato dalla necessaria efficienza che il riconoscimento deve avere per poter essere utilizzato nel monitoraggio di una rete.

La generazione di allarmi di sicurezza avviene in base alla combinazione della presenza di testo nel formato cercato e del protocollo di comunicazione utilizzato. Nel progetto ho considerato un insieme ristretto di protocolli. E' necessario estendere questo insieme aggiungendone altri all'interno dei quali non è previsto il passaggio di informazioni sotto forma di testo comprensibile per un umano.

Un'altra importante sfida da affrontare riguarda l'utilizzo della crittografia. I lavoro che ho sviluppato non è applicabile nei casi un cui la comunicazione dei dati esfiltrati avviene attraverso un canale criptato.

6 Conclusioni

L'esfiltrazione dei dati è un problema importate e in continuo sviluppo nel campo della sicurezza informatica. Il numero e la complessità dei metodi di esfiltrazione è in costante aumento. l'esistenza e la progressiva affermazione di questi fenomeni rende le contromisure adottate per proteggersi dall'esfiltrazione critiche per la sicurezza di un organizzazione.

In ambito privato e accademico è stato svolto un grande lavoro di sviluppo delle tecniche di difesa contro questo fenomeno. Tra le soluzioni proposte si individuano diversi obiettivi come la prevenzione, il rilevamento e l'investigazione di esfiltrazioni. Ognuno di questi aspetti è importante per stabilire un perimetro di sicurezza adeguato per una rete. A causa della natura dinamica e mutevole del problema non è però possibile ottenere una soluzione che riduca a zero il rischio di furto dei dati.

Il costante sviluppo delle tecnologie utilizzate all'interno della rete richiede un parallelo impegno nello studio approfondito e nello sviluppo degli strumenti di prevenzione e rilevamento nel campo della sicurezza. Risulta quindi importante l'utilizzo di diverse tecniche in modo combinato e coordinato tra loro. Inoltre ognuna di queste va sfruttata in base alle caratteristiche della rete e con una profonda conoscenza delle comunicazioni che avvengono in essa.

Nel mio lavoro mi sono concentrato sull'aspetto del rilevamento di un esfiltrazione di dati. Anche in questo campo specifico le difficoltà sono molte e in evoluzione costante. Il sempre maggiore utilizzo di nuove tecnologie come IoT o cloud comporta un aumento dei rischi dovuti alla sicurezza. Anche lo sviluppo delle rete in termini di velocità, utilizzo di protocolli criptati e ambienti virtualizzati sono aspetti che complicano il compito di rilevare eventuali processi di esefiltrazione.

Tuttavia sono stati sviluppati molti strumenti in grado di compiere un azione di monitoraggio delle comunicazioni in una rete. Tra questi, per sviluppare il mio progetto, ho utilizzato la DPI. L'uso di questa tecnologia in combinazione con una semplice idea di controllo sul contenuto del traffico mi ha permesso di produrre un risultato ristretto ma accettabile in termini di utilizzo ed efficienza.

Come chiarito in questa tesi, i metodi di esfiltrare i dati sono potenzialmente infiniti, non è quindi consigliabile lo sviluppo di contromisure che fanno riferimento alle specifiche tecniche che sono state utilizzate o teorizzate per questo scopo. Inoltre il monitoraggio della rete è una attività che si prefigge di individuare eventi molto rari e nascosti all'interno del traffico, che è di sua natura è molto grande e complesso.

A mio parere l'approccio migliore per l'implementazione di nuove tecniche per il rilevamento dell'esfiltrazione dei dati si basa su due principali aspetti. Il primo è un dettagliato studio del tipo di comunicazione che avvengono all'interno di una rete in termini di caratteristiche specifiche dei protocolli utilizzati e dei sistemi in gioco. L'altro è la struttura stessa dei componenti della rete. Con l'avanzamento della tecnologia questi aspetti sono soggetti ad una sempre più veloce trasformazione e richiedono un aggiornamento costante

per una comprensione adeguata. Queste conoscenze devono essere combinati con l'utilizzo consapevole di strumenti pratici per il monitoraggio della rete. Molti di questi già esistenti e spesso estendibili per adattarsi a specifiche necessita(es: nDPI). Uno schema di sviluppo di questo tipo può generare soluzioni con alto grado di efficienza ed efficacia nell'ambito del rilevamento delle minacce di esfiltrazione di dati sensibili.

Riferimenti bibliografici

[1] R. Broadhurst,P. Grabosky, M. Alazab, B. Bouhours, S. Chon. *Organizations and Cybercrime: An Analysis of the nature of groups engaged in cybercrime*. 2014. International Journal of Cyber Criminology. 8. 1-20. 10.2139/ssrn.2345525.

[2] Ideentity Theft Resource Center. *End of year data breanch report*. 2018. Available from: https://www.idtheftcenter.org/2018-data-breaches/

[3] Ponemon Institute. *2018 Cost of a Data Breach Study*. 2018. Available from: https://www.ibm.com/security/data-breach

[4] K. InghamIngham, S. Forrest. *A history and survey of network firewalls*. 2002. ACM Journal Name. 1-42.

[5] D. Ashok Kumar,S.R. Venugopalan. *Intrusion Detection Systems: A Review*. 2017. International Journal of Advanced Research in Computer Science. 8. 10.26483/ijarcs.v8i8.4703.

[6] F. Ullah,M. Edwards,R. Ramdhany,R. Chitchyan, M.A. Babar, A.Rashid. *Data Exfiltration: A Review of External Attack Vectors and Countermeasures*. 2017. Journal of Network and Computer Applications. https://doi.org/10.1016/j.jnca.2017.10.016.

[7] J. Vukalović D. Delija, *Advanced Persistent Threats - detection and defense* 2015. 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija. pp. 1324-1330. doi: 10.1109/MIPRO.2015.7160480

[8] J. Postel,J. Reynolds. File Transfer Protocol (FTP). October. 1985. Type RFC. Number: 959. Available from: http://www.rfc-editor.org/rfc/rfc1654.txt.

[9] T. Holz1, C. Gorecki1, K. Rieck, F. C. Freiling. *Measuring and Detecting Fast-Flux Service Networks*. 2008.

[10] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. June. 1999. Type RFC. Number: 2616. Available from: http://www.ietf.org/rfc/rfc2616.txt.

[11] E. Nebel, L. Masinter Form-based File Upload in HTML. November. 1995. Type RFC. Number: 1867. Available from: http://www.ietf.org/rfc/rfc1867.txt.

[12] T. Liu, Y. Sun, A. X. Liu, L .Guo, B. Fang. *A Prefiltering Approach to Regular Expression Matching for Network Security Systems*. 2012. pages 363–380.

[13] A. Freier, P. Karlton, P. Kocher. The Secure Sockets Layer (SSL) Protocol Version 3.0. August. 2011. Type RFC. Number: 6101. Available from: https://tools.ietf.org/html/rfc6101.

[14] E. Rescorla. HTTP Over TLS. May. 2000. Type RFC. Number: 2818. Available from: https://tools.ietf.org/html/rfc2818.

[15] J. Postel. Simple Mail Transfer Protocol. August. 1982. Type RFC. Number: 821. Available from: https://tools.ietf.org/html/rfc821.

[16] S. Deering. ICMP Router Discovery Messages. June. 1991. Type RFC. Number: 1256. Available from: https://tools.ietf.org/html/rfc1256.

[17] P. Mockapetris. Domain Names - Concepts and Facilities. November. 1983. Type RFC. Number: 882. Available from: https://tools.ietf.org/html/rfc882.

[18] S. Josefsson. The Base16, Base32, and Base64 Data Encodings. October. 2006. Type RFC. Number: 4648. Available from: https://tools.ietf.org/html/rfc4648.

[19] F. Ullah, M. Edwards, R. Ramdhany, R. Chitchyan, M. A. Babar, A. Rashid. *Data Exfiltration: A Review of External Attack Vectors and Countermeasures*. 2017 11. Journal of Network and Computer Applications. 101. 10.1016/j.jnca.2017.10.016.

[20] J. M. Beaver, F. C. Jewell. *Host-based data exfiltration detection via system call sequences*. 2011. 01. Tennessee Technological University, Cookeville, TN, U.S.

[21] Jaskolka, J., R. Khedri, and K.E. Sabri. *Investigative support for information confidentiality*. 2015. Journal of Ambient Intelligence and Humanized Computing, 6(4): p. 425-451.

[22] B. Xu, G. He, T. Zhang, Y. Ma. *A Novel Method to Detect Encrypted Data Exfiltration*. 2014. 2014 Second International Conference on Advanced Cloud and Big Data. pp. 240-246. Huangshan. doi: 10.1109/CBD.2014.40

[23] Cisco Systems Inc. *Wan and application optimization solution guide*. 2008.

[24] Yali Liu, C. Corbett, Ken Chiang, R. Archibald, B. Mukherjee, D. Ghosal. *SIDD: A Framework for Detecting Sensitive Data Exfiltration by an Insider Attack*. 2019. 42nd Hawaii International Conference on System Sciences. Big Island. HI. pp. 1-10. doi: 10.1109/HICSS.2009.390.

[25] S. Alneyadi, E. Sithirasenan, V. Muthukkumarasamy. *Word N-Gram based classification for data leakage prevention*. 2013. 12th IEEE International Conference on Trust Security and Privacy in Computing and Communications (TrustCom). pp. 578-585.

[26] T. Fawcett. *ExFILD: A tool for the detection of data exfiltration using entropy and encryption characteristics of network traffict*. 2010.

[27] J. Olivain, J. Goubault-Larrecq. *Detecting subverted cryptographic protocols by entropy checking*. 2006. Laboratoire Specification et Verification ENS Cachan France. pp. 06-13.

[28] B. Maia, R. Silva, A. Barreiro, C. Fróis *N-grams in search of theories*. 2008. University of Porto e Linguateca.

[29] J. Case, M. Fedor, M. Schoffstall, J. Davin. A Simple Network Management Protocol (SNMP). May. 1990. Type RFC. Number: 1157. Available from: https://tools.ietf.org/html/rfc1157.

[30] L. Deri, M. Martinelli, A. Cardigliano *nDPI: Open-Source High-Speed Deep Packet Inspection*. 2014. pp. 617-622. Nicosia. 2014 International Wireless Communications and Mobile Computing Conference (IWCMC). doi: 10.1109/IWCMC.2014.6906427.

[31] www.wireshark.org

Available from: https://www.wireshark.org/docs/.

[32] www.tcpdump.org

Available from: https//www.tcpdump.org/documentation.