Πανεπιστήμιο Πειραιώς

Τμήμα Ψηφιακών Συστημάτων

Π.Μ.Σ. " Τεχνοοικονομική Διοίκηση & Ασφάλεια Ψηφιακών Συστημάτων "

Master Thesis

“A flexible distributed network forensic evidence acquisition framework “

Μεταπτυχιακός Φοιτητής:
Παρασκευόπουλος Ιωάννης (ΑΜ: MTE1218)

Επιβλέπων:
Αναπληρωτής Καθηγητής Χρήστος Ξενάκης

Table of Contents

A flexible distributed network forensic evidence acquisition framework

Abstract

A flexible network forensic evidence acquisition framework is introduced which is composed by two main factors, the agent (portable network evidence acquisition device) and a cloud database. The agent is based on the inexpensive credit card–sized single-board computer "Raspberry Pi 2 Model B" and uses open source software. The cloud database is the MySQL Database which can be deployed in a virtual machine or as Database as a Service (DbaaS). It is described which of these two cloud databases deployment methods is chosen and in which cases. The main scopes of this design are firstly, to provide flexibility and scalability in the storage management of network evidence. This will be succeeded due to two reasons, the agent does not store data locally rather than sending them directly to the cloud database and the other one is the cloud database itself (theoretically due to cloud infinite storage capacity). Secondly, it is introducing the use of a small factor, relatively cheap hardware collector. Furthermore, two working modes will be described, wired and RF mode.

Introduction

Network forensics is the capture, storage, and analysis of network events. It is sometimes also called packet mining, packet forensics, or digital forensics. Regardless of the name, the idea is the same: record every packet of network traffic (all emails, all database queries, all Web browsing–absolutely all traffic of all kinds traversing an organization's network) to a single searchable repository so the traffic can be examined in detail. But monitoring and managing networks has become increasingly difficult for reasons such as:

- faster networks and greater data volumes
- richer data
- subtler and more malicious security threats

Network forensics solutions must provide three essential capabilities: capturing and recording data, discovering data, and analyzing data. Every network forensic solution has its limitations, including sustainable throughput, packets per second, data management, search functions, etc. The proposed framework focuses on the sector of capturing and recording data and through this thesis will be

examined the limitations of its design.

Network-Based Digital Evidence

"Network-based digital evidence" is digital evidence that is produced as a result of communications over a network. The primary and secondary storage media of computers (e.g., the RAM and hard drives) tend to be fruitful fodder for forensic analysis. Due to data remanence, persistent storage can retain forensically recoverable and relevant evidence for hours, days, even years beyond file deletion and storage reuse. In contrast, network-based digital evidence can be extremely volatile. Packets flit across the wire in milliseconds, vanish from switches in the blink of an eye. Web sites change depending on from where they're viewed and when. The requirements for admissibility of network-based digital evidence are murky. Often, the source that generated the evidence is not obtainable or cannot be identified. When the evidence is a recording of a chat log, blog posting, or email, the identity of the parties in the conversation (and therefore the authors of the statements) may be difficult to prove. When the evidence is a web site, the litigant may need to provide supporting evidence to demonstrate that the image presented in court is what actually existed at the time and location that it was supposedly viewed.

There is little case precedent on the admissibility of network packet captures. Depending on the method of capture and the details of the case, packet captures of network traffic may be treated as recordings of events, similar to a taped conversation.

Examples of "network-based digital evidence" can include:

- Emails and IM sessions
- Browser activity, including web-based email
- Routinely kept packet logs
- /var/log/messages etc.

Challenges Relating to Network Evidence

Network-based evidence poses special challenges in several areas, including acquisition, content, storage, privacy, seizure, and admissibility as described below:

- **Acquisition**

It can be difficult to locate specific evidence in a network environment. Networks contain so many possible sources of evidence—from wireless access points to web proxies to central log servers—that sometimes pinpointing the correct location of the evidence is tricky. Even when you do know where a specific piece of evidence resides, you may have difficulty gaining access to it for political or technical reasons.

- **Content**

Unlike filesystems, which are designed to contain all the contents of files and their metadata, network devices may or may not store evidence with the level of granularity desired. Network devices often have very limited storage capacity. Usually, only selected metadata about the transaction or data transfer is kept instead of complete records of the data that traversed the network.

- **Storage**

Network devices commonly do not employ secondary or persistent storage. As a consequence, the data they contain may be so volatile as to not survive a reset of the device.

- **Privacy**

Depending on jurisdiction, there may be legal issues involving personal privacy that are unique to network-based acquisition techniques.

- **Seizure**

Seizing a hard drive can inconvenience an individual or organization. Often, however, a clone of the original can be constructed and deployed such that critical operations can continue with limited disruption. Seizing a network device can be much more disruptive. In the most extreme cases, an entire network segment may be brought down indefinitely. Under most circumstances, however, investigators can minimize the impact on network operations.

- **Admissibility**

Filesystem-based evidence is now routinely admitted in both criminal and civil proceedings. As long as the filesystem-based evidence is lawfully acquired, properly handled, and relevant to the case, there are clear precedents for authenticating the evidence and admitting it in court. In contrast, network forensics is a newer approach to digital investigations. There are sometimes conflicting or even nonexisting legal precedents for admission of various types of network-based digital evidence. Over time, network-based digital evidence will become more prevalent and case precedents will be set and standardized.

This master thesis is referring to the first three areas and that is because the acquisition is covered by the agent, the storage is covered by the database and content is the area that occupies both the agent and the database.

Passive vs Active Evidence Acquisition

Ideally, we would like to obtain perfect-fidelity evidence, with zero impact on the environment. For copper wires, this would mean only observing changes in voltages without ever modifying them. For fiber cables, this would mean observing the quanta without ever injecting any. For radio frequency, this would mean observing RF waves without ever emitting any. In the real world, this would be equivalent to a murder investigator collecting evidence from a crime scene without leaving any new footprints. Obviously, we don't live in a perfect world, and we can never achieve "zero footprint." Detectives analyzing a murder scene still cannot avoid walking on the same floor as the killer. However, network investigators can minimize the impact.

Network forensic techniques often refer to "passive" versus "active" evidence acquisition. Passive evidence acquisition is the practice of gathering forensic-quality evidence from networks without emitting data at Layer 2 and above. Traffic acquisition is often classified as passive evidence acquisition. Active or interactive evidence acquisition is the practice of collecting evidence by interacting with stations on the network. This may include logging onto network devices via the console or through a network interface, or even scanning the network ports to determine the current state. Although the terms "passive" and "active" imply that there is a clear distinction between two categories, in reality, the impact of evidence acquisition on the environment is a continuous spectrum. In this thesis, we introduce the physical media that can be leveraged to passively acquire network-based evidence.

It is possible to obtain network traffic without sending or modifying any data frames on the network. While it is never possible to have absolutely zero impact on the environment, the process of capturing (or sniffing) traffic can often be conducted with very little impact. There are many ways to transmit data over physical media, and just as many ways to intercept it. The simplest case is a station connected to another station over a physical conduit, such as a copper cable. Voltage on copper can easily be amplified and redistributed in a one-to-many configuration. Hubs and switches are designed to extend the physical media in order to share the baseband with additional stations. Forensic

investigators can passively acquire network traffic by intercepting it as it is transmitted across cables, through the air, or through network equipment such as hubs and switches. The framework proposed in this master thesis is based on a passive evidence acquisition mode.

Network Forensic Analysis Tools (NFATs)

As mentioned above the framework presented here offers more than basic portable packet sniffing, in that it is specifically designed with digital forensics and network evidence collection in mind. The framework is not intended to compete with full featured, permanently deployed NFAT systems.

NFATs help administrators do forensic work. They can look at the logs and use the replay feature to see what exactly happened in a certain event. Companies that need NFATs would be those that get attacked frequently or those under regulation to protect its assets like healthcare and e-commerce. Although free tools are available that do the same things as NFATs, the benefit with the commercial products is their reporting ability. NFATs like SilentRunner, for example, have the ability to show the trends of network traffic. Usually companies use outside companies to do vulnerability assessments for them. These companies also use penetration tests against the security structure of the enterprise.

With NFATs, companies have these resources in house and save time and money. They also help out the security administrators since they are able to have a better picture of the environment that they are protecting. Administrators can start focusing on prevention by understanding where they are most vulnerable. They can spend more time in expanding user education. NFATs can test the IDSs to ensure that they are seeing the signatures they are supposed to be detecting. This way, they can also focus on improving the tools already in place in the environment so that all the security tools are working together for one purpose. Security and network administrators can use the information from the NFATs to know how well the network is performing. They can detect slow downs that may be a warning of a pending DOS attack. They can see rogue servers. And have a better tool to prevent internal attacks. They can better monitor internal threats by seeing all the events in the network. Emails can be read and administrators can be warned if certain thresholds are crossed. If certain confidential information is going out of the company, this can be detected. Activities that violate Internet and email use policies can also be detected. The great part of this is that administrators have a better way to record all the events. Evidence gathering is not as difficult as it once was. There are many options in deploying a NFAT either open source or commercial. In this thesis three examples of

Commercial Network Forensic Analysis Tools are examined in more detail since they have a large portion in the market and these are:

- RSA Security Analytics (former NetWitness Investigator)
- AccessData SilentRunner Sentinel
- Niksun's NetDetector.

RSA Security Analytics (NetWiteness Investigator)

The NetWitness Investigator tool, a Microsoft Windows-based application, enables the network forensics examiner to audit and monitor the network traffic by analyzing captured network traffic through the unique investigative lenses. The investigative lenses allow the network forensics examiner to conduct different network traffic analysis through the use of various different types of customizable filters. To achieve this objective, the NetWitness Investigator application is divided into six components as presented in the following figure

The successful use of the NetWitness Investigator application for the analysis of captured network traffic requires the network forensics examiner to be skilled with each of the six components.

Import/Live Capture Network Traffic

Importing or the live capturing of network traffic is the first NetWitness Investigator component. NetWitness Investigator provides four possible ways to insert network packet data into the network forensics application. The first option allows the downloading of captured network data from previously deployed NetWitness remote devices (for example, decoder, concentrator). The second

option allows the real-time capturing of network data through the use of a local wired or wireless network interface. The network interface can be configured in stealth mode to make the device appear logically invisible. The network-capturing process uses the WinPcap capture driver. For wireless captures, the NetWitness Investigator supports various Institute of Electrical and Electronics Engineers (IEEE) 802.11 standards (for example, Wired Equivalent Privacy, 802.11i). The types of wireless capture devices supported are listed as follows:

■ Microsoft Netmon (packet_netmon_)

■ Linux mac80211 (packet_mac80211_)

■ Mac OS X Airport (packet_airport_)

The third option, the importing of previous captured network data, allows precaptured network traffic to be read as file-based input. This option supports the various file types listed in Table 7.1.

Table 7.1 File-Based Formats Supported by the NetWitness Investigator

Type of File	Common File Extension
tcpdump	.tcp, .tcp.gz, .pcap, .pcap.gz
NetMon	.cap, .cap.gz
EtherPeek	.pkt, .pkt.gz
IPTrace	.ipt, .ipt.gz
NAIDOS	.enc, .enc.gz
RAW	.raw, .raw.gz
NetWitness Data	.nwd
Network Instruments Observer	.bfr

Collections

The second component is used after capturing or importing the network data. The NetWitness Investigator will store the network traffic into a component known as a collection. Collections are logical grouping containers (which maps to a local file system folder/directory storage structure) that store unique sets of captured network packet data before processing the network traffic. Since the collections are logical groupings, named by using alpha-numeric characters and symbols (except the following / \ * ? : " < > |), the naming convention for collections should represent the type of network data captured (see Figure 7.2). The following are examples of possible collection-naming categories:

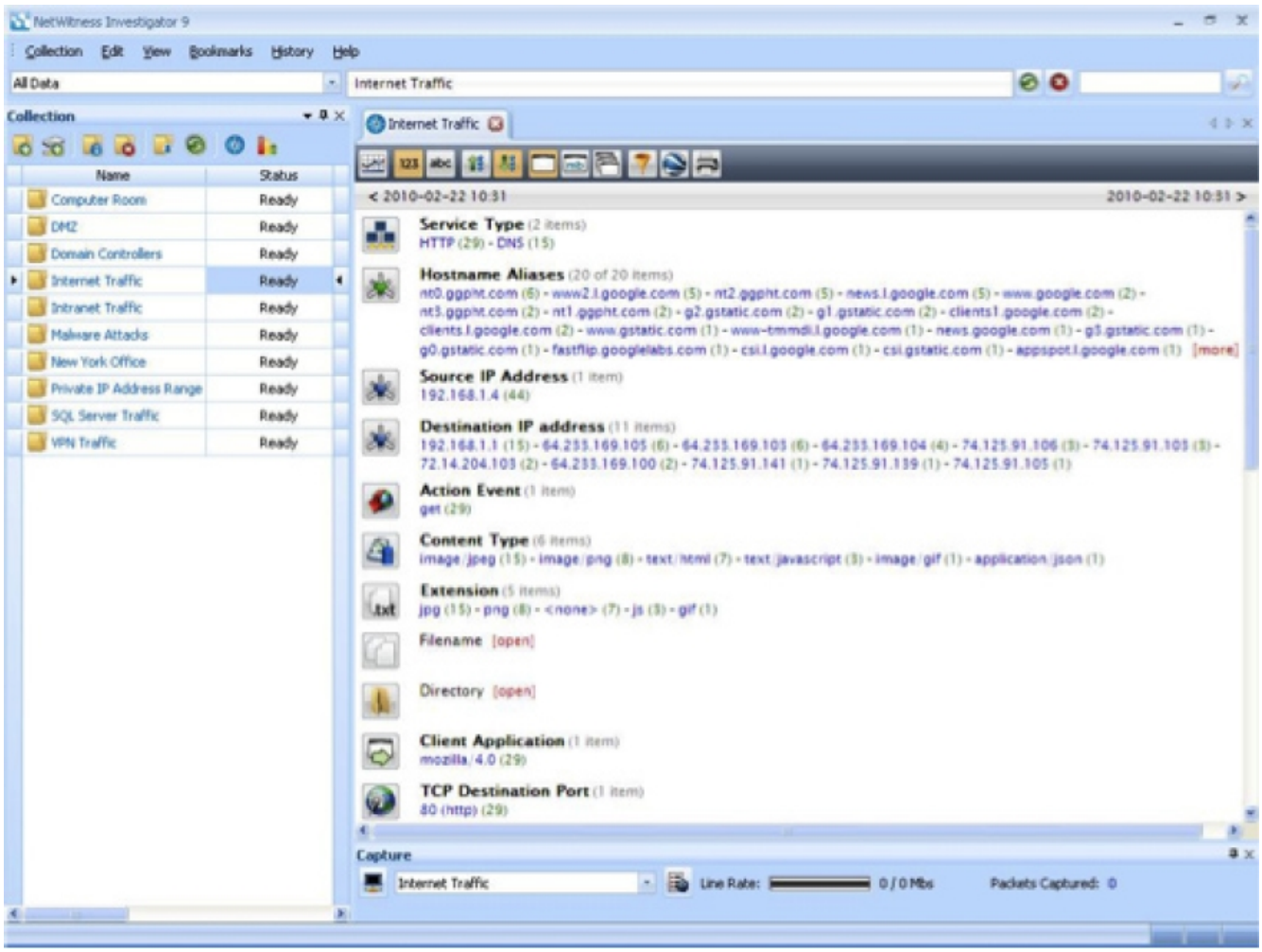
■ Specific type of network traffic captured (for example, Structured Query Language [SQL] Server,

Domain Controllers, Malware)

■ Location-based traffic (for example, Computer Room, New York Office)

■ Network traffic captured from security zones (for example, demilitarized zone [DMZ], Intranet, Internet, virtual private network [VPN], Data Center)

The implementation of a naming convention for collections will allow the network forensics examiner to store the captured network traffic based on a more meaningful representation.



AccessData SilentRunner Sentinel

SilentRunner is the network forensic tool by AccessData. It is a suite of applications designed to work together, offering data capture, analysis, and visualization of the data. This includes the loading of the data into a relational database to provide complex query and correlation abilities. The

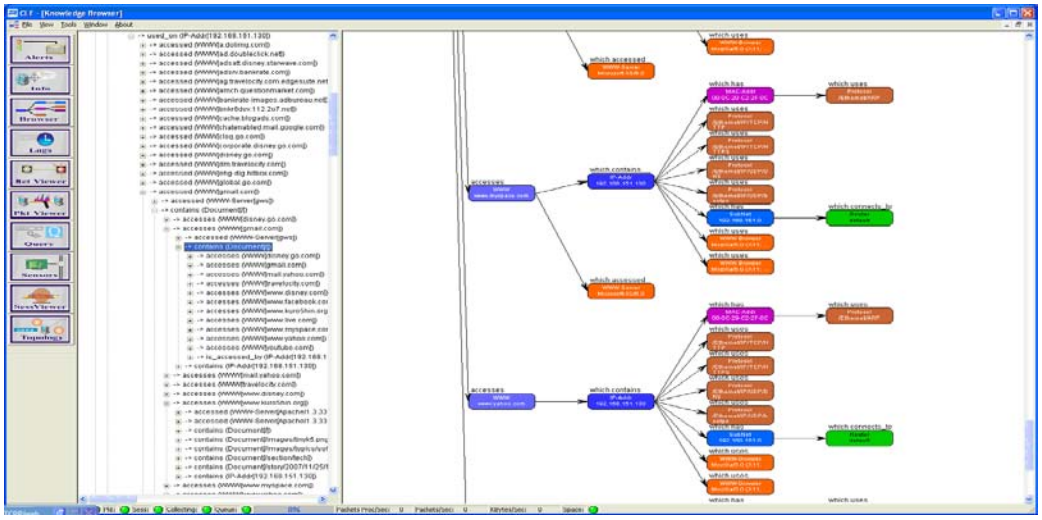
supported databases today are Microsoft Structured Query Language (SQL) and Oracle, and they support a variety of architectures and deployment strategies. The major parts of the SilentRunner system are the Collectors, Loaders, Database, and Analysis workstations. The Collectors capture the network traffic through their available network adapters. The Loader facilitates the transfer of the data from the Collectors into the Database. The Analysis workstations either perform queries against the database or import logs files and create visualizations and reconstructions of the data. The product supports approximately 2000 protocols including Voice over Internet Protocol (VoIP).

Parts of the SilentRunner System

The SilentRunner system of applications is made up of seven parts: the Collector, Forwarder, Loader, Database, Data Manager, Analyzer, and Context Management.

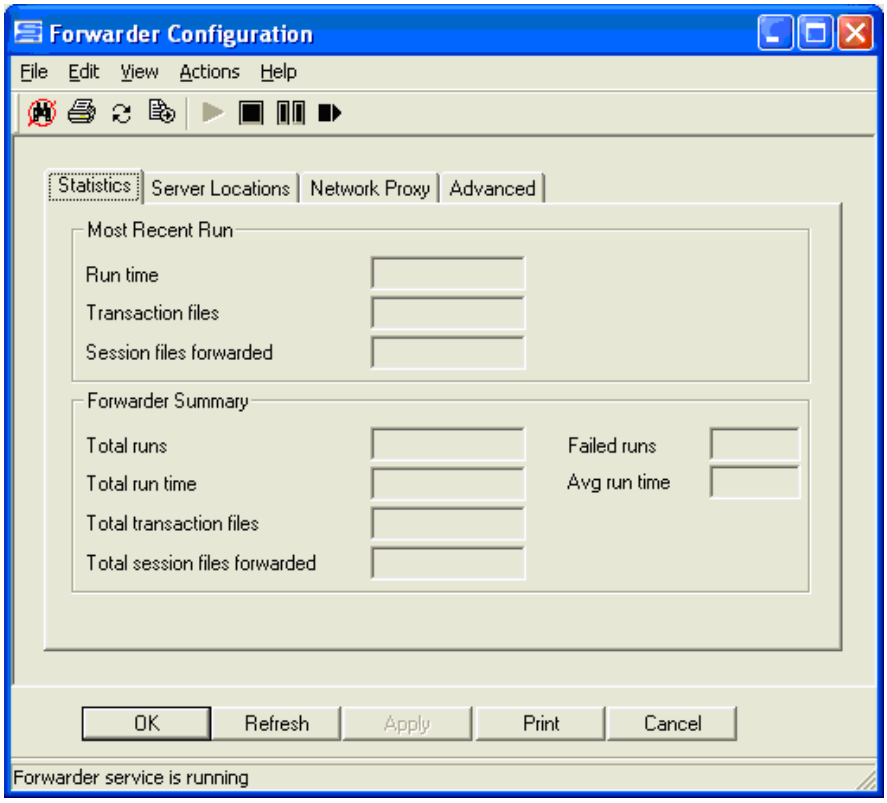
Collector

The Collector is basically a network sniffer with enhanced features. It also uses a network river in promiscuous mode to capture raw traffic from the network. It is able to gather data on all the layers of the Open Systems Interconnection (OSI) model. The Collector is able to capture data in tcpdump format in the event the user would want to export the packets for use in other tools. The Collector also allows the import of tcpdump captures from other tools for playback and imports into SilentRunner. The Collector also loads the data it captures into the Knowledge Base, which can provide some reporting and analysis functions showed in the following figure.



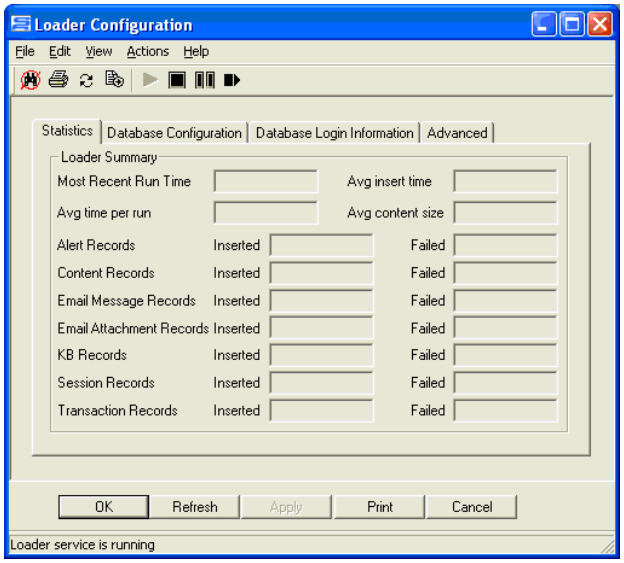
Forwarder

The Forwarder resides with the Collector and sends the data to the Loader. The Forwarder is responsible for the encryption of the data and sending it to the Loader.



Loader

The Loader receives the data from the Forwarder that was captured by the Collector, decrypts it, and performs the database import and insertions.



Database

SilentRunner ultimately stores the data it captures in a relational database. This allows the data to be queried using standard SQL statements. The use of a database also allows for the data returned from a query to be exported into a structured file to be leveraged by the analysis tools. SilentRunner supports Microsoft SQL and Oracle .

Data Manager

The Data Manager is a set of utilities that assist with the query of the database and the export to files to be used by the analysis tools. It also has tools to assist with the manipulation of log files to be imported from other applications.

Analyzer

The Analyzer performs the correlation, visualization, and reporting of the data. It can graphically display the interactions on a variety of data. The Analyzer also has tools to animate the network traffic. Another set of analysis tools called the Data Investigators recreate traffic like Instant Messaging (IM), Web, and e-mail sessions.

Context Management

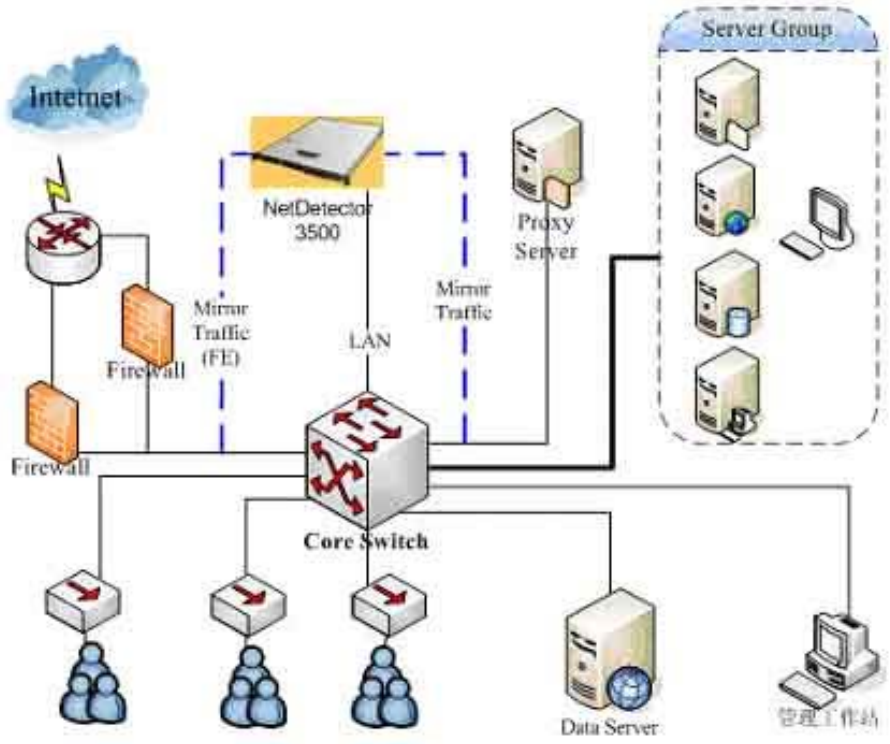
The SilentRunner Context Management determines relationships between like types of information using n-gram models. The Context Management then clusters the files based on their similarity, with a tighter cluster indicating the closer the match.

Niksun's NetDetector

On the other side of the NFAT scale is Niksun's NetDetector. It is a hardware appliance (FreeBSD) focused on capturing network traffic and forensics but lacks the same visual capability of SilentRunner. Its advantage is on capturing traffic and analyzing actual traffic. It is an example of a "Catch-it-as-you-can system" because it analyzes traffic in batches and has the ability to store a large amount of data. It has the ability to do replays so that administrators can do investigative work when an attack has been detected.

NetDetector has three functions: to capture network traffic, analyze network traffic and perform data discovery. NetDetector continually captures traffic but does not save the traffic. That is, older data is replaced by new data unless it is saved by the administrator. Analysis is started by an administrator who chooses a certain time interval to study. That batch of traffic is reassembled into

meaningful information so that analysis can be performed by the analysis engine. The analysis engine attempts to understand the content of the data stream rather than report on packet information. Finally, with data discovery, an administrator is able to perform trend analysis because prior records are archived into NetDetector. Administrators can also generate different types of reports based on network traffic (bandwidth usage), content (trend analysis on Internet traffic), user behavior and breaches (helping administrators increase network security where it is needed).



NetDetector's analysis is not as powerful or as detailed as SilentRunner's, but it does it in a way that it is actually looking at the traffic rather than making conclusions based on header information or protocols. However, a noteworthy advantage of NetDetector over SilentRunner is its ability to decrypt SSH-2 sessions. Additionally, NetDetector only accepts secure remote administration into the appliance. This is a huge difference to SilentRunner which does not provide remote administration of the collectors. NetDetector also allows for its own log files to be exported and analyzed by different tools like tcpdump. Network Forensic Analysis Tools may not represent the silver bullet everyone is looking for, but they do alleviate some of the laborious functions of investigation and analysis work for data security.

Moreover, except those recommended NFAT, reference should be made to the following tools which are described below:

Iris

Collects network traffic and reassembles it in its native session based format, reconstructs actual text of the session, replays traffic for audit trial of suspicious activity, provides a variety of statistical measurements and has advanced search and filtering mechanism for quick identification of data.

Infinistream

Utilizes intelligent Deep Packet Capture (iDPC) technology and performs real-time or back-in-time analysis. It does high-speed capture of rich packet details, statistical analysis of packet or flow based data and recognizes hundreds of applications. It uses sophisticated indexing and Smart Recording and Data Mining (SRDM) for optimization.

Solera DS 5150

DS 5150 is an appliance for high-speed data capture, complete indexed record of network traffic, filtering, regeneration and playback. DeepSee forensic suite has three software – Reports, Sonar and Search – to index, search and reconstruct all network traffic.

OmniPeek

Provides real-time visibility into every part of the network. It has high capture capabilities, centralized console, distributed engines, and expert analysis. Omnipliance is a network recording appliance with a multi-terabyte disk farm and high-speed capture interfaces. OmniEngine software captures and stores network traffic. OmniPeek interface searches and mines captured data for specific information.

NetworkMiner

Captures network traffic by live sniffing, performs host discovery, reassembles transferred files, identifies rogue hosts and assesses how much data leakage was affected by an attacker.

Xplico

Captures Internet traffic, dissects data at the protocol level, reconstructs and normalizes it for use in manipulators. The manipulators transcode, correlate and aggregate data for analysis and present the results in a visualized form.

PyFlag

An advanced forensic tool to analyze network captures in libpcap format while supporting a number of network protocols. It has the ability to recursively examine data at multiple levels and is ideally suited for network protocols which are typically layered. PyFlag parses the pcap files, extracts the packets and dissects them at low level protocols (IP, TCP or UDP). Related packets are collected into streams using reassembler. These streams are then dissected with higher level protocol dissectors (HTTP, IRC, etc) (Cohen, 2008).

Network Security and Monitoring (NSM) tools

These tools described above offer a powerful range of analysis options for network monitoring or assessing insider threats, zero-day exploits, and targeted malware focused for commercial organizations. There are other tools which merely focus on traffic analysis or embed traffic analysis engine for the investigation. They are described below.

TCPDump

A common packet sniffer and analyzer, runs in command line, intercepts and displays packets being transmitted over a network. It captures, displays, and stores all forms of network traffic in a variety of output formats. It will print packet data like timestamp, protocol, source and destination hosts and ports, flags, options, and sequence numbers.

Wireshark

Most popular network protocol analyzer. It can perform live capture in libpcap format, inspect and dissect hundreds of protocols, do offline analysis, and work on multiple platforms. It can read and write files in different file formats of other tools.

TCPFlow

Captures data transmitted as part of TCP connections (flows) and stores it for protocol analysis. It reconstructs actual data streams and stores in a separate file. TCPFlow understands sequence numbers and will correctly reconstruct data streams regardless of retransmissions or out-of-order

delivery.

Flow-tools

Library to collect, send, process and generate reports from NetFlow data. Important tools in the suite are – flow capture which collects and stores exported flows from a router, flow-cat concatenates flow files, flow report generates reports for NetFlow data sets, and flow-filter filters flows based on export fields.

NfDump

A suite of tools working with NetFlow format: nfcapd – NetFlow capture daemon reads the NetFlow data from the network and stores it. NfDump – NetFlow dump reads the NetFlow data from these files, displays them and creates statistics of flows, IP addresses, ports etc. nfprofile – NetFlow profiler filters the NetFlow data according to the specified filter sets and stores the filtered data. nfreplay – NetFlow replay sends data over the network to another host.

PADS

PADS is a portable, lightweight and intelligent network sniffer. It is a signature-based detection engine used to passively detect network assets. It can sniff TCP, ARP and ICMP traffic packets. It can move information about unique assets and services seen on the network into permanent storage, output it in CSV or MySQL format or present an user friendly report.

Argus

Processes packets in capture files are live data and generate detailed status reports of the 'flows' detected in the packet stream. The flow reports capture the semantics of every flow with a great deal of data reduction. The audit data are good for network forensics, non-repudiation, detecting very slow scans, and supporting zero-day events.

Nessus

Vulnerability scanner featuring high-speed discovery, configuration auditing, asset profiling, sensitive data discovery and vulnerability analysis

Sebek

Kernel based data capture tool designed to capture all activity on a Honeypot. It records keystrokes of a session that is using encryption, recover files copied with SCP, capture passwords used to log in to remote system, and accomplish many other forensics related tasks.

TCPTrace

Produce different types of output containing information, such as elapsed time, bytes/segments which are sent and received, retransmissions, round trip times, window advertisements, and throughput.

Ntop

Used for traffic measurement, network traffic monitoring, optimization, planning, and detection of network security violations. It provides support for both tracking ongoing attacks and identifying potential security holes including IP spoofing, network cards in promiscuous mode, denial of service attacks, trojan horses and port scan attacks.

TCPStat

Reports network interface statistics like bandwidth, number of packets, packets per second, average packet size, standard deviation of packet size and interface load by monitoring an interface or reading from libpcap file.

IOS NetFlow

Collects and measures IP packet attributes of each packet forwarded through routers or switches, groups similar packets into a flow, to help understand who, what, when, where and how the traffic is flowing. It also detects network anomalies and security vulnerabilities

TCPDstat

Produces per-protocol breakdown of traffic, for a given libpcap file, like number of packets, average rate and its standard deviation, number of unique source and destination address pairs. It is also useful in getting a high-level view of traffic patterns.

Ngrep

A pcap-aware tool that allows specifying extended regular or hexadecimal expressions to match against data payloads. It can debug plaintext protocol interactions to identify and analyze anomalous network communications and to store, read and reprocess pcap dump files while looking for specific data patterns.

TCPXtract

Extracts files from network traffic based on file signatures. It can also be used to intercept files transmitted across networks.

SiLK

Supports efficient capture, storage and analysis of network flow data based on Cisco NetFlow. The tool suite, consisting of collection and analysis tools, provides analysts with the means to understand, query, and summarize both recent and historical traffic data in network flow records. SiLK supports network forensics in identifying artifacts of intrusions, vulnerability exploits, worm behavior, etc. SiLK has performance as a key element and manages the large volume of traffic by storing only the security-related information.

TCPReplay

Suite of tools with ability to classify previously captured traffic as client or server, rewrite layer 2, 3 and 4 headers and finally replay the traffic back onto the network. TCPPrep is a multi-pass pcap file pre-processor which determines packets as client or server, TCPRewrite is a pcap file editor which rewrites packet h

eaders, TCPReplay replays pcap files at arbitrary speeds onto the network and TCPBridge bridges two network segments.

P0f

Passive OS fingerprinting by capturing traffic coming from a host to the network. It can also detect the presence of firewall, use of NAT, existence of a load balancer setup, distance to the remote system and its uptime.

Nmap

Utility for network exploration and security auditing. It supports many types of port scans and can be used as on OS fingerprinting tool. It uses raw IP packets in novel ways to determine hosts available on the network, services being offered, operating systems running, firewalls in use and many other characteristics.

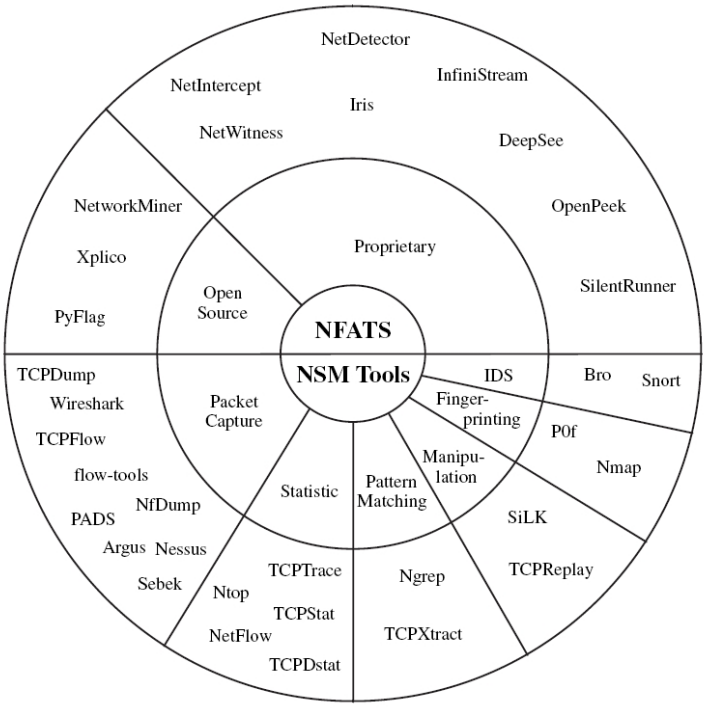
Bro

NIDS that passively monitors network traffic for suspicious activity. It detects intrusions by first parsing network traffic to extract its application-level semantics and then executing event-oriented analyzers that compare this activity with patterns deemed troublesome. It is primarily a research platform for IDS, traffic analysis and network forensics.

Snort

Network intrusion prevention/detection system capable of performing packet logging, sniffing and real-time traffic analysis. It can perform protocol analysis, content searching, matching and application-level analysis. It can capture the traffic in libpcap format.

The various classifications of tools are visualized here

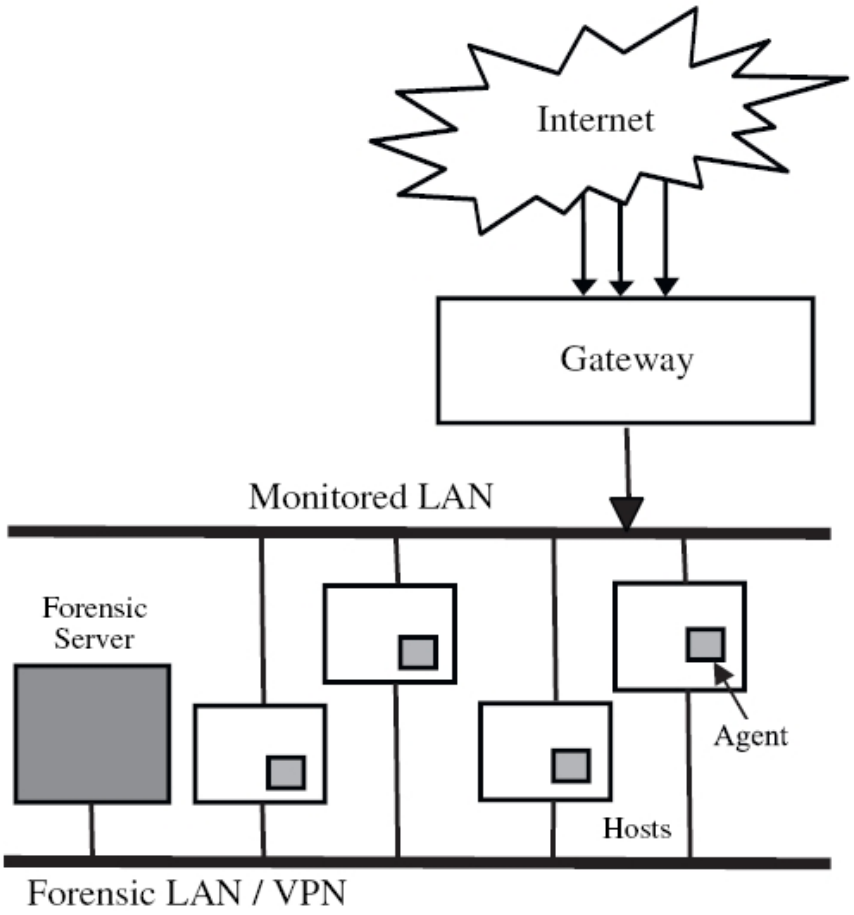


Network forensic frameworks

Researchers developed many frameworks which implement the phases of network forensics and an exhaustive survey is presented category wise to highlight the specific gaps and identify the research challenges.

Distributed systems based frameworks

Internet and LANs are distributed in nature and networks attack events are logged in clients at various locations. There is a need to collect these logs, fuse them and analyze on a central server. A general scheme for the frameworks is shown in here.



The following implementations were proposed:

Shanmugasundaram et al. (2003) propose ForNet, a distributed network logging mechanism to aid digital forensics over wide area networks. It has two functional components – Syn- App, designed to summarize and remember network events for a period of time and a Forensic Server, which is a

centralized authority for a domain that manages a set of SynApps in that domain. A forensic server receives queries from outside its domain, processes them in co-operation with SynApps and returns query results back to the senders after authentication and certification. The overall architecture involves a network filter, synopsis engine, synopsis controller, configuration manager, security manager, storage management and query processor. Evidence of crimes can be found in packet headers or application dependent payloads. ForNet can identify network events like TCP connection establishment, port scanning, connection record details and uses bloom filters to track other events.

Ren (2004a) proposed a reference model of distributed cooperative network forensic system. It is based on client– server architecture. The server captures network traffic, builds mapping topology database, filters, dumps and transforms the network traffic stream into database values, mines forensic database, and replays network behaviour. It also does network surveying, attack statistic analysis and visualization. The distributed agent clients integrate data from firewall, IDS, Honeynet and remote traffic. The goal of this model is dumping misbehaviour packets based on adaptive filter rules, analyzing the overall cooperative database to discover the potential misbehaviour, and replaying the misbehaviour for forensic analysis. It can discover the profile of the attacker and obtain clues for further investigation.

Ren and Jin (2005b) further extended their previous model as distributed agent-based real-time network intrusion forensics system. The goals of this framework include log system information gathering, adaptive capture of network traffic, active response for investigational forensics, integration of forensics data and storing the historical network misuse pattern. The four elements in the system are network forensics server, network forensics-agents, network monitor, and network investigator. Network forensic agents are engines of the data gathering, data extraction and data secure transportation. Network monitor is a packet capture machine which adaptively captures the network traffic. Network investigator is the network survey machine. Network forensics server integrates the forensics data, analyzes it and launches an investigation program on the network investigator. The model can expedite the investigation of an incident and improve the ability of emergence response.

Tang and Daniels (2005) proposed a simple framework for distributed forensics. It is based on distributed techniques providing an integrated platform for automatic evidence collection and efficient data storage, easy integration of known attribution methods and an attack attribution graph generation mechanism. The model is based on proxy and agent architecture. Agents collect, store, reduce, process and analyze data. Proxies generate the attack attribution graph and perform stepping

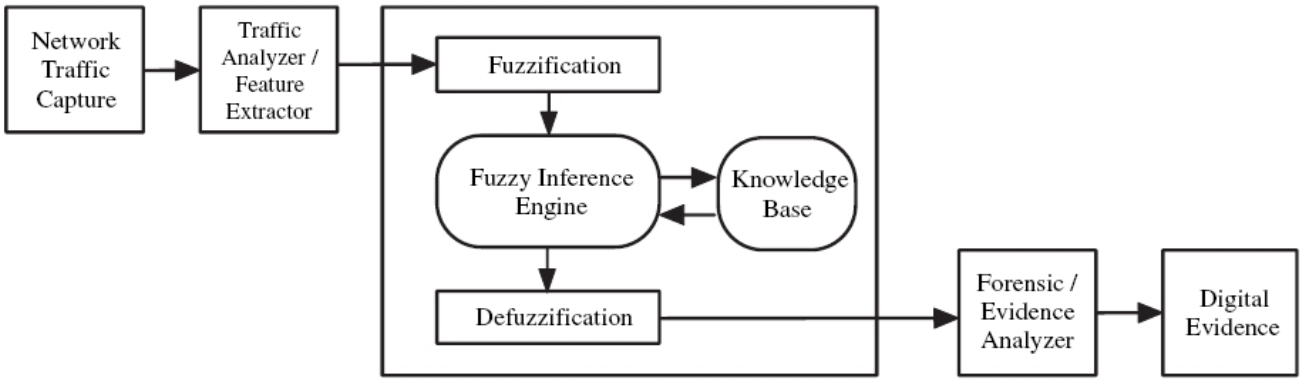
stone analysis. This model aims at providing a method to collect, store and analyze forensic information. It also provides automatic evidence and quick response to attacks.

Nagesh (2007) implemented a distributed network forensics framework using JADE mobile agent architecture. A node acting as a server, hosting the network forensics-agent, dispatches mobile agents to monitored heterogeneous locations. They gather network traffic logs, examine them and return the results which will be displayed on a user interface. The interface enables the analyst to specify data to be collected and analyze the resultant network events displayed. The solution automates collection of network data from distributed heterogeneous systems using mobile agents. The implementation is scalable, reduces network traffic, addresses single point of failure and provides real-time monitoring.

Wang et al. (2007) developed a dynamical network forensics (DNF) model based on the artificial immune theory and the multi-agent theory. The system provides a real-time method to collect and store data logs simultaneously, provide automatic evidence collection and quick response to network criminals. The system includes a Forensic Server and three agents namely Detector-Agent, Forensics-Agent and Response-Agent. The Detector-Agent captures real-time network data, matches it with intrusion behavior and sends a forensics request message to the Forensics-Agent. The Forensics-Agent collects the digital evidence, creates a digital signature using a hash function and transmits the evidence to the Forensic Server. The Forensic Server analyzes evidence and replays the attack procedure. The Response-Agent is being developed. The implementations discussed above use client–server/ agent–proxy architectures to collect the attack features and analyze them. They are limited in identifying the features correctly and some components are still being developed.

Soft computing based frameworks

The soft computing implementations are used to analyze captured data and classify the attack data. Neural network and Fuzzy tools are used for validation of attack occurrence. A general scheme for the fuzzy logic based frameworks is shown here.



The following implementations were proposed:

Kim et al. (2004) develop a fuzzy logic based expert system for network forensics to aid the decision making processes involving sources of imprecision that are non-statistical in nature. The system can analyze computer crime in networked environments and make digital evidences automatically. It can provide analyzed information for a forensic expert and reduce the time and cost of forensic analysis. The framework consists of six components. Traffic analyzer captures network traffic and analyses it using sessionizing. Knowledge base stores rules which are used by the fuzzy inference engine. The rules are written for various attacks using linguistic variables and terms. Membership functions are defined for each fuzzy set and a crisp value of degree of membership is determined. Each input variables crisp value is first fuzzified into linguistic values. Fuzzy inference engine derives output linguistic values using aggregation and composition. Defuzzification defuzzifies the output values into crisp values and the forensic analyzer decides whether the captured packets indicate an attack.

Liu and Feng (2005) proposed the Incremental Fuzzy Decision Tree-Based Network Forensic System (IFDTNFS). This is an efficient way to create a classification model by extracting key features from network traffic by providing the resulting fuzzy decision tree with better noise immunity and increasing applicability in uncertain or inexact contexts. IFDTNFS consists of three components: network traffic separator, traffic detector, and forensic analyzer. The network traffic separator component is responsible for capturing network traffic and separating it according to the service type, and directing the separated traffic to corresponding traffic detector. The traffic detector consists of four components: feature extractor extracts features from network traffic, fuzzy rule base is the knowledge base using which fuzzy decision trees are built, rule base updater adds new samples to the

fuzzy decision tree that has been constructed and also adjusts the optimal tree size, and fuzzy inferencer fuzzifies input values and processes them with the rule base. Forensic analyzer includes collecting relative event data, analyzing correlated information relating with the event, and establishing digital evidences.

Zhang et al. (2007) propose network forensic computing based on Artificial Neural Network and Principal Component Analysis (ANN–PCA). The major challenge faced in network forensics is massive information to be stored and analyzed. Extraction of key features reduces storage by correlating the features with attacks. ANN–PCA techniques are used to identify all possible violations, extract features and build signatures for new attacks. Classification is done using FAAR algorithm to mine association rules and calculate the PCA values. Classification accuracy increases and information storage size decreases after feature extraction is performed using ANN–PCA.

Neurofuzzy Techniques (Anaya et al., 2009) were used by Anaya et al., to address the challenges of enormous data to be logged and analyzed for network forensic computing. The Neurofuzzy solution is based on Artificial Neural Network (ANN) and fuzzy logic and is used for evidence differentiation into normal and abnormal flows. ANNs are used in information processing to learn from the data and generalize a solution. Fuzzy logic is used to generate a grade of membership to different behaviors so that attacks are determined. The model consists of four modules. The Monitor control module stores all the network information. Information preprocessing module is made up of syntax sub module and correlation sub module. Syntax module is responsible for normalizing the inputs and correlation sub module aggregates the different flow formats and groups the PDUs into a flow. Dependencies module relates all network element logs and takes decision about the flows. The decider module distinguishes between normal and abnormal flows.

Liao et al. (2009) propose a Network Forensic System based Fuzzy Logic and Expert System (NFS-FLES), an effective and automated analysis system, which guarantees evidence reliability by collecting information from different sensors. It also analyzes computer crimes and makes automatic digital evidence using the approach of fuzzy logic and expert systems. The NFS-FLES consists of the following components – traffic capture, feature extractor, fuzzification, fuzzy inference engine, knowledge base, defuzzification and forensic analyzer. The whole operation is done in four parts – real-time forensic data acquisition and preprocessing, knowledge base construction and dynamic rule generation, fuzzy linguistic operation of input attack data and computing aggregation fuzzy value and total fuzzy score of every kind of attack. The forensic result is then output in time.

The soft computing tools give desirable results provided the rules to differentiate attack and legitimate traffic can be generated. Detecting zero-day attacks is also a challenge.

Honeypot based frameworks

Honeypot frameworks are used to attract the attackers so that their process methodology can be observed and analyzed to improve defense mechanisms. The following implementations were proposed:

Honeytraps (Yasinsac and Manzano, 2002) were proposed as a deception tool to collect information about blackhat activities and learn their techniques so that protection and defense mechanisms can be formulated. Honeytraps are Honeypot or Honeynet systems which attract intruders to enter a host by emulating a known vulnerability. Once an attacker penetrates a honeytrap, data are captured to detect and record his actions. This data can be used to profile the tools and tactics used by the attackers putting the investigators in an offensive mode. Two architectures, serial and parallel, facilitate the forensic investigation. The serial architecture places honeytrap between the Internet and the production system. Recognized users are filtered to the production systems and blackhats are contained in the honeytrap. The parallel architecture allows honeytrap to be independent of the production system. Once the system detects the presence of blackhat, the forensic alert system is activated. If the attack is detected, forensic processes are activated on the honeytrap and production systems. Once the attack is contained, the investigation process is begun to determine the identity of the intruder on the production system.

Thonnard and Dacier (2008) propose a framework for attack patterns' discovery in Honeynet data. Their work aims at finding groups of network traces sharing various kinds of highly similar patterns within an attack data set. They design a flexible clustering tool and analyze one specific aspect of the Honeynet data, time series of attacks. Malicious network traffic is obtained from the distributed set of Honeynet responders. Time signature is used as a primary clustering feature and attack patterns are discovered using attack trace similarity. Attacks are detected as a series of connections, zero-day and polymorphic attacks are detected based on similarity to other attacks and knowledge from the Honeynet data is used in intrusion detection efforts. The clustering method does feature selection and extraction, defines a pattern proximity measure and groups similar patterns. The result of clustering applied to time series analysis enables detection of worms and botnets in the traffic collected by Honeypots.

Merkle (2008) investigates automated analysis of network based evidence in response to cyberspace attacks. The two major challenges of network forensics, namely 'complexity' problem of analyzing raw traffic data and 'quantity' problem of amount of data to analyze are addressed in his solution. The model integrates results of data logged by various tools into a single system that exploits computational intelligence to reduce human intervention. This integrated tool is referred as 'automated network forensic' tool. An isolated network of virtual machines is built into a Honeynet. Open source forensic tools are used for collecting data. The information produced by various tools in one stage is characterized and transformed for use by other tools in the succeeding stages. Time consuming and error prone processes are identified and automated. The data sets are partitioned, system is trained and then tested.

Honeynets meet the expectations of forensic analyzers but they cannot be used for investigative purposes as evidence generated is not valid in legal system.

Attack graph-based framework

Wang and Daniels (2008) developed a graph-based approach toward network forensics analysis. An evidence graph model facilitates evidence presentation and automated reasoning. The basic architecture has six modules: evidence collection module collecting digital evidence from heterogeneous sensors deployed, evidence preprocessing module transforms evidence into standardized format, attack knowledge base provides knowledge of known exploits, assets knowledge base provides knowledge of the networks and hosts under investigation, evidence graph manipulation module generates the evidence graph and attack reasoning module performs semiautomated reasoning based on the evidence graph. A hierarchical reasoning framework consisting of two levels – local reasoning (functional analysis) aims to infer the functional states of network entities from local observations and global reasoning (structural analysis) aims to identify important entities from the graph structure and extract groups of densely correlated participants in the attack scenario. The results from both levels are combined and attacks further analyzed.

5.5. Formal method based framework

Rekhis et al. (2008) develop a system for Digital Forensic in Networking (DigForNet) which is useful to analyze security incidents and explain steps taken by the attackers. DigForNet uses the expertise of intrusion response teams and formal reasoning tools (I-TLA and I-TLC) to reconstruct

potential attack scenarios. They integrate analysis performed by the IRT on a compromised system through the use of Incident Response Probabilistic Cognitive Maps (IRPCMs). They provide a formal method framework to identify potential attack scenarios using Investigation-based Temporal Logic of Actions (I-TLA). They generate executable potential attack scenarios and show progress of the attack using Investigationbased Temporal Logic Model Checker (I-TLC), automatic verification tool. Unknown attacks are handled by generating hypothetical actions. The generated executable potential attack scenarios are used to identify risk scenarios that have compromised the system, entities which originated the attacks, different steps taken to conduct the attacks and to confirm the investigation.

Aggregation framework

Network forensic analysis involves many phases. The aggregation frameworks harness the strength of these tools to facilitate forensic investigation rather than building a new tool from scratch. The following implementations were proposed:

Almulhem and Traore (2005) propose a Network Forensics System (NFS) that records data at the host level and network level. The system consists of three main modules – marking, capture and logging. Marking module decides whether a passing packet is malicious. One or more sensors (like IDS) report suspicious IP addresses. Capture module is a collection of lightweight capture modules which wait for the marked packets. They arrange to reliably transport them to the logging module for archival. Logging module is a system repository where the attack data are being stored. It uses three types of loggers – host logger stores data sent by capture module, sensor logger stores sensors' alerts and raw logger is optional and is used when other loggers fail. The capture module was implemented using Sebek, marking module used Snort IDS, and logging module used server-side Sebek, Snort's barnyard tool, ACID Lab and TCPDump.

Nikkel (2006) proposed a Portable Network Forensic Evidence Collector (PNFEC) which was built using inexpensive embedded hardware and open source software. The compact and portable device has been designed for traffic collection between a network and a single node, having specific modes of operation, rapid deployment and stealthy inline operation. The traffic on the Ethernet Bridge is promiscuously captured using various pcap based capture tools and stored on a hard disk. The operating system, additional software, configuration files and investigator activity logs are stored on a compact flash. Administrative access controls various aspects of the device like startup, scheduling, configuration of capturing filters, forensic functions such as preserving and transferring

the evidence. The PNFEC is easy to deploy and operate (plug and play). The network traffic collected can be stored in encrypted form. PNFEC also controls filtering of captured data using TCPDump to ensure there are no privacy violations. A script is used to create a cryptographic hash of the packet capture files and preserved. OpenBSD is the operating system used, as many of the functionalities like secure access, packet capture, encrypted file system, evidence preservation, disk wiping and formatting tools, are included by default. Tools for trouble shooting (TCPFlow) and pcap management (tcpslice) are also added. PNFEC operates in three modes – investigator, server and user.

Vandenberghe (2008) proposed a Network Traffic Exploration (NTE) Application being developed by Defense Research and Development Canada (DRDC) for security event and packet analysis. This tool combines six key functional areas into a single package. They are intrusion detection (signature and anomaly based), traffic analysis, scripting tools, packet playback, visualization features and impact assessment. NTE has three layers with MATLAB as development environment, low level packet analysis library and unified application front end. It provides an environment where statistical analysis, session analysis and protocol analysis can exchange data.

Motivation

The main thought behind this master thesis is derived by the following three speculations which are described below:

- If I don't have network data stored through the network evidence acquisition process then I am not able to proceed to a network forensic analysis. So I need to introduce a portable, small factor, as affordable as possible device that will be able to capture network traffic with the best possible outcome that is overcoming in the best possible grade the challenges relating to network evidence, as described above.
- Storage capacity is another obvious factor in collecting large amounts of live network evidence. High-speed networks (Gigabit Ethernet, OC-24 speeds and higher, etc.) cannot realistically be captured in their entirety beyond a certain length of time, and large repositories of remotely accessible data may be impossible to store on a local forensic collection device due to finite storage capacity. Cloud services are introduced here as a thought for the solution to this barrier and in particular Database as a Service. Due to the technical implementation of cloud services, every time our framework reaches its capacity limit, extra storage is adjusted without interrupting the process

of network evidence acquisition. By this method theoretically we can achieve "infinite" storage capacity combined with the advantage of process interruption avoidance.

- The use of cloud services, in order to overcome the storage capacity barrier, in combination with the portable agent, unlocks the possibility of using more than one agent in any different network location. In other words, an integrated, distributed network forensic evidence acquisition framework is introduced where each agent sends the captured data to a single point of reference (cloud database) as showed in Figure 1

The technologies and concepts used in designing this framework are not new (Ethernet bridging, promiscuous packet sniffing, cloud database as a service etc.). However, the design goals and intended usage satisfy a current need in the area of network forensic evidence collection. This framework fills the gap between basic portable network packet analyzers and full featured NFAT systems bridging all the above with cloud storage.

Portable network packet sniffers such as Network General's Sniffer Portable, or portable analyzers from Fluke Networks, have existed for many years. These portable devices are designed more for short-term collection, troubleshooting, and network traffic analysis. They are not specifically designed for network forensic evidence collection. NFAT devices such as Sandstorm's NetIntercept, CA's eTrust Network Forensics (formerly SilentRunner) and Niksun's NetDetector, have also been on the market for some time. While these are intended for network forensic evidence collection, they are relatively expensive, and designed more for permanent deployment on a network. These devices are very feature rich in areas of forensic analysis, and offer far more than simple packet collection.

The framework presented here offers more than basic portable packet sniffing, in that it is specifically designed with digital forensics and network evidence collection in mind. The framework is not intended to compete with full featured, permanently deployed NFAT systems. The focus is limited to acquisition and storage, not analysis. The intended use could be discerned in two case:

- temporary situations involving single network nodes. In this cases it is expected the use of a single agent working in online or offline mode
- permanent, fully deployed set of agents in online mode

The network forensic evidence agent is portable, inexpensive, and built using existing open source software. It can be used on demand or in full deployment with more than one, and driven by specific cases or incidents. Deployment can be done very quickly by staff not specifically trained in digital forensics, and without reconfiguration of the surrounding network. The device has several intended

modes of operation which are described in detail.

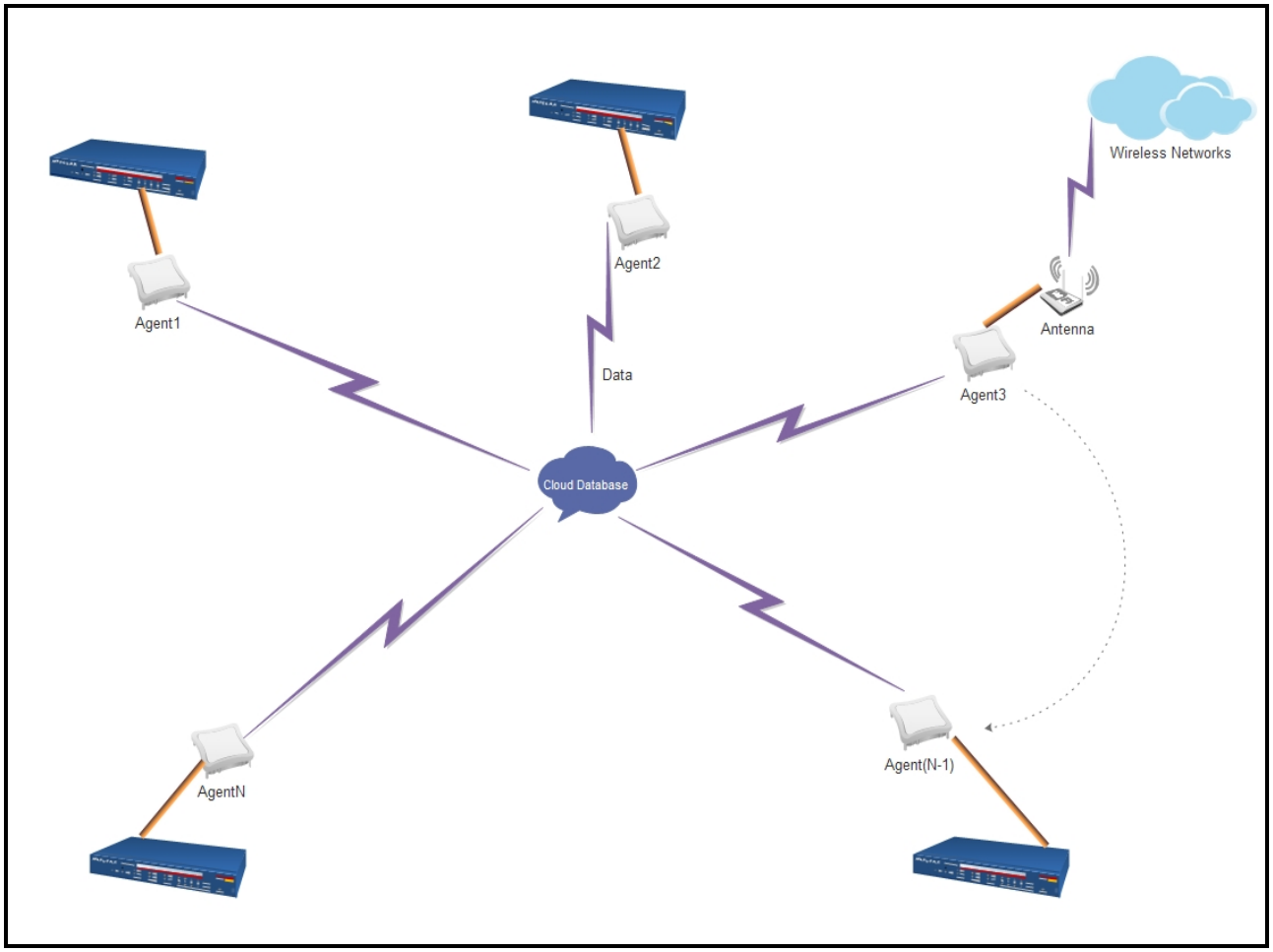


Figure 1 - Main Architecture of network forensic evidence acquisition framework

Implementation

As mentioned above the framework consists of two main elements, the agent and the cloud database. The agent's hardware is based on the inexpensive credit card–sized single-board computer "Raspberry Pi 2 Model B" (Fig. 2) and it makes use of promiscuous packet capturing to enhance evidence collection from remote network sources. For this to happen two network adapters are needed. One for the packet capturing which will operate in promiscuous mode and the other one for management and data transfer operations. Nevertheless, the Raspberry Pi 2 has built in one network adapter and four USB 2.0 ports. Therefore one USB port will be used by a USB to Ethernet Adapter

(Fig. 3).

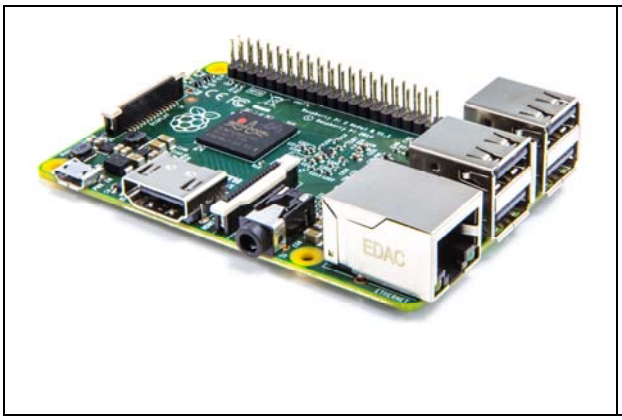


Figure 2 - Raspberry Pi 2 Model B

Figure 3 - USB to Ethernet Adapter

The operating system that is used in building and testing the agent is Debian 8.1 (jessie). This operating system was chosen for its stability, security, and robustness and partly due to the author's familiarity with the OS. Debian supports transparent bridging of Ethernet frames between two network interfaces. It has extensive security and cryptography support, and the Debian code base is audited to reduce the number of vulnerabilities.

One of the main goal of this thesis is the development and coding of the software that handles the two main processes:

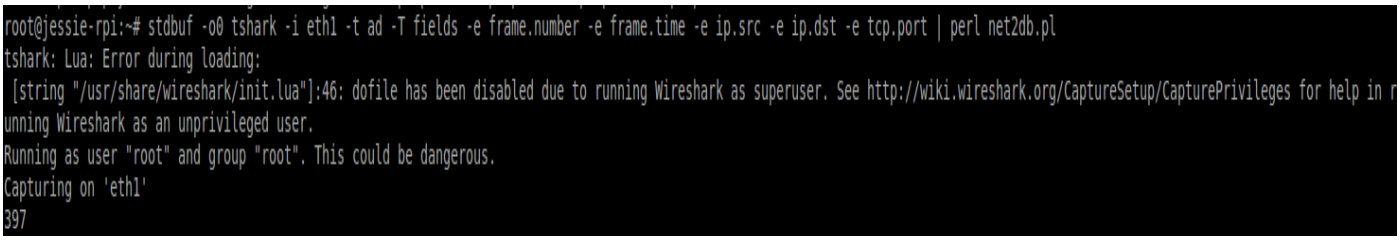
1. the data capturing
2. the transformation of the output of the data captured in a database query format in order to be sent to the cloud database

In figure 4 it is shown the procedure and the processes which is consisted of.

As far as the capturing is concerned, Tshark was used. The reason that finally TShark was selected is that it is a network protocol analyzer that lets you to capture packet data from a live network or read packets from a previously saved capture file, either by printing a decoded form of those packets to the standard output or by writing the packets to a file. TShark's native capture file format is libpcap format, which is also the format used by tcpdump and various other tools. Read filters in TShark, which allow you to select the packets that are to be decoded or written to a file, are very powerful. More fields are filterable in TShark than in other protocol analyzers, and the syntax you can use to create your filters is richer.

Subsequently, in order to achieve the transformation of the Tshark's output to a database query form in order to be sent to the database, a Perl script was written which was piped to the Tshark's output

like it is shown in the next figure



Interpreting the command given at the CLI we could say the following:

Command: stdbuf -o0 tshark -i eth1 -t ad -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.port | perl net2db.pl

- stdbuf -o0 : This command is set for performance reasons and it ensures that there will be no delays in capturing the packets due to buffering operations. It actually provide an unbuffered STDOUT of the Tshark command
- tshark: the network protocol analyzer used for capturing
- -i eth1: the network interface from which the packets will be captured
- -T fields: filtering and exporting fields of tshark capture. In this example we record the following fields
    - frame.number
    - frame.time
    - ip.src
    - ip.dst
    - tcp.port
- | perl net2db.pl : this is the pipe that takes as input the output of tshark.

The net2db.pl is the perl script that reads the output of tshark and sends the capturing data to the cloud database. Inside the script it is set a unique id value which describes the agent. This is very important in the case of multiple agents usage.

The second factor of this framework, the cloud database, will be a relational database management system (RDBMS) and specifically MySQL in the form of cloud service (DbaaS). In the laboratory environment a virtual machine located in a different network will replace the cloud service for economic reasons. Nevertheless this replacement does not affect at all the main concept. The use of RDBMS is an added value to the framework since evidence can be queried instantly and in an

oriented manner.

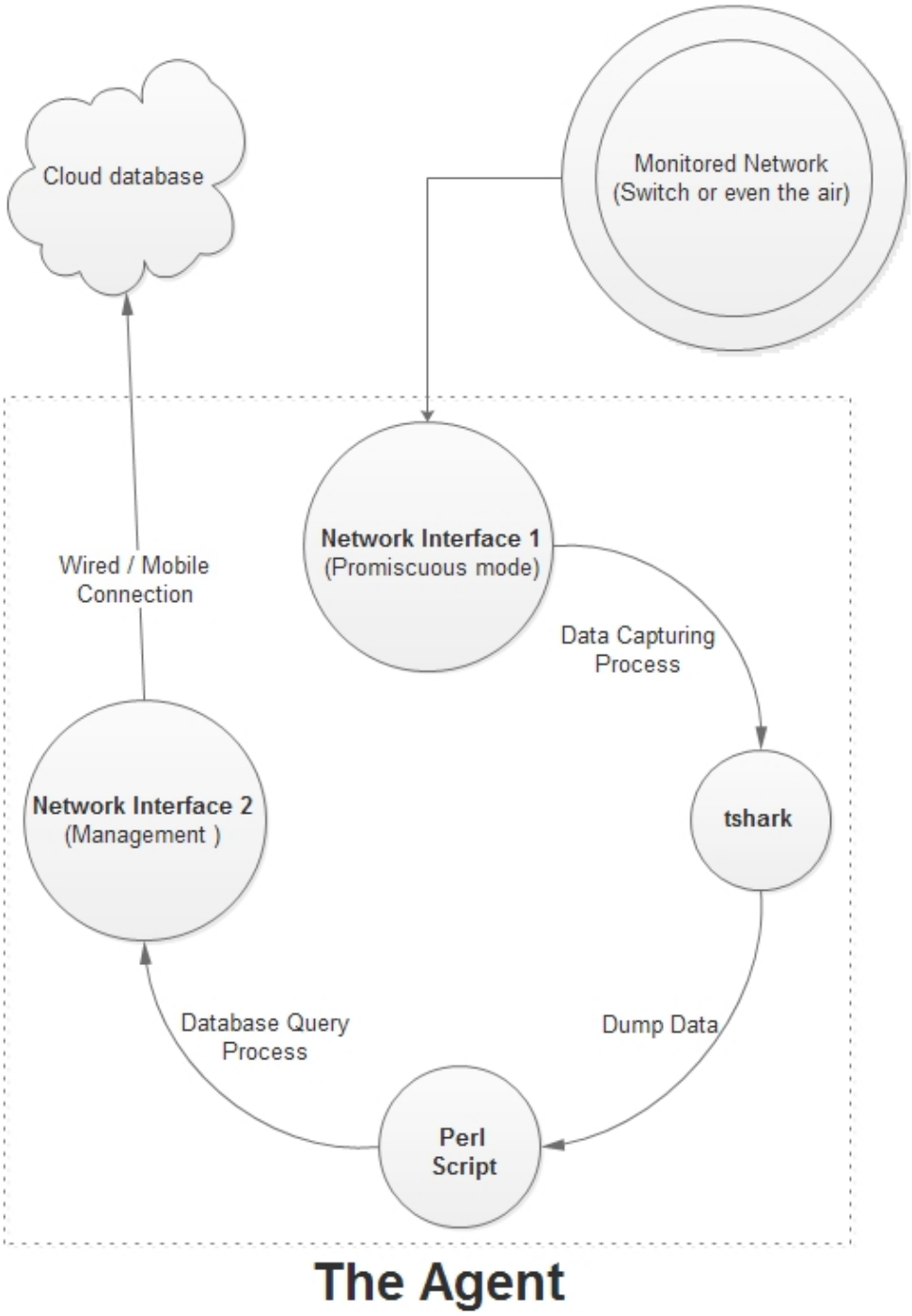


Figure 4 – Logical Diagram of Agent's Procedure

Modes of Operation

The agent has two basic intended modes of operation:

1. Online Mode: It is the case when the agent is using two ethernet ports, one in promiscuous mode and the other for management and transmission purposes.

2. RF Mode: In this case the agent is using the built in ethernet port of raspberry pi for management and transmission purposes and a wireless adapter in monitor mode for capturing purposes.

Given that the hardware of the agent "Raspberry Pi" is a low energy consumption device, offers the framework the privilege of autonomy. The fact that the agent could be fully functional if connected with a power bank gives the framework the ability to be applied practically everywhere at anytime.

Advantages

There are a number of advantages the distributed network forensic evidence acquisition framework offers for network forensic evidence collection. Regarding the agent we have the following:

- It is compact and portable making it useful for crowded and cramped wiring closets and racks.
- It is also invisible on the network, causing no disruption, and requiring no reconfiguration of the surrounding infrastructure.
- In case it is only capturing traffic from a single node, it can be used in a switched or VLAN environment without mirrored ports or additional port configuration on the switching device.
- It can be preconfigured in a forensic lab, with the insertion or deployment delegated out to IT staff or network administrators. This is useful to reduce travel by forensic investigators (the preconfigured device can be shipped), and also makes better use of investigator time and resources.
- The agent is prebuilt with inexpensive hardware and uses free open source software.
- The agent does not modify or inject traffic as it acts as a bridge at the link layer. Ethernet frames are forwarded unchanged through the agent (MAC address, hop-count, fragmentation, MTU, etc. remain untouched).
- It supports the collection of multiple protocols, including IPv4, IPv6, IPX, NetBios, or any other Ethernet based network traffic.
- Low Energy Consumption Hardware

Regarding the framework:

- Many agents capturing data which are stored in a single point of reference, that is the cloud database. This advantage actually sums up the most of what this framework may offer.
- Continuous data capturing with no interruptions due to storage outage.
- You can query your data offline.
- Databases are storing their data optimized, so they are quicker than pure disk I/O.
- You can embed intelligence in your data by using a database.
- You can distribute your data in many databases.
- You can automatically use the reporting tools that support your database.

- If you already know SQL, querying your network data is easy.
- A remote user can easily access your data using the network.
- The ability of instant correlation of captured data between different agents. As a notice, different agents could mean different networks, different locations, same network in different locations and so on.

Further Work

The framework has been tested in low network load conditions and had excellent results, that is zero data loss both in capturing and cloud database querying (writing). Since the framework is functional and produces results, it can be tested in different modes so as to find out its sustainable throughput, its limitations and any potential yield factors.

Other fields, that could be examined and generate interesting results through the testing processes in laboratory, are:

- Data rate limits
- Packets lost per second during capture process
- Records lost per second through transfer to the database process
- Yield factor of transform script etc.

In order to take the framework to its limits it is suggested to be applied in high usage computer rooms and in a laboratory environment through the usage of packet traffic generators.
Studying all produced data and results will help in conclusion of the proposed framework possibilities. Based on its possibilities we can decide in which network cases it addresses, since it seems obvious that an enterprise class network is out of his potentials due to the enormous volume of data.

Conclusion

Undoubtedly there has been much work done in the area of network forensics and especially in the distributed frameworks. This thesis is focused in the network forensic evidence acquisition procedure and is aiming to three goals. The first one is to take advantage of the characteristics of Raspberry Pi 2 in creating a flexible network forensic evidence acquisition device, that is:

- Relatively cheap (about 45 €)
- Small factor

- Portable
- Flexible (4 usb 2.0 ports, HDMI, Ethernet, 1 GB RAM, till 32GB SD card)
- Compatible with many OS (in our case Linux)
- Low Energy Consumption etc

Moreover it focuses on developing code that would transform the dumped data from the capturing process into SQL queries in order to be sent to cloud database. Finally, using the technological implementation of cloud services (and in combination with the two goals mentioned above) proposes an alternative solution in storage limitation in the field of network forensic evidence acquisition.

References

- Bruce J. Nikkel, A portable network forensic evidence collector, Digital Invastigation 3 (2006) 127-135
- Raspberry Pi - Teach, Learn, and Make with Raspberry Pi, https://www.raspberrypi.org/
- Bruce J. Nikkel, Improving Evidence Acquition from Live Network Sources, The International Journal of Digital Forensics and Incident Response Vol. 3, No.2
- Bruce J. Nikkel, Generalizing Sources of Live Network Evidence
- NIST, Guide to Integrating Forensic Techniques into Incident Response, Special Publication 800-86
- Network Forensics: Tracking Hackers through Cyberspace, Sherri Davidoff -Jonathan Ham, Prentice Hall
- Almulhem A. Network forensics: notions and challenges. In: Proceedings of the ninth IEEE international symposium on signal processing and information technology (ISSPIT 2009), UAE; Dec. 2009.
- Almulhem A, Traore I. Experience with engineering a network forensics system. In: Proceedings of the international conference on information networking (ICOIN 2005), Korea, LNCS 3391. Berlin, Heidelberg: Springer-Verlag; 2005. p. 62–71.
- Anaya EA, Nakano-Miyatake M, Meana HMP. Network forensics with neurofuzzy techniques. In: Proceedings of the 52nd IEEE international Midwest symposium on circuits and systems (MWSCAS 2009); 2–5 Aug. 2009, p. 848–52.
- Argus, http://www.qosient.com/argus.

- Baryamureeba V, Tushabe F. The enhanced digital investigation process model. In: Proceedings of the fourth digital forensic research workshop (DFRWS); 2004.
- Beebe NL, Clark JG. A hierarchical, objectives-based framework for the digital investigations process. Digital Investigation (The International Journal of Digital Forensics & Incident Response) Jun 2005;2(2):147–67.
- Berghel H. The discipline of Internet forensics. Communications of the ACM 2003;46(8):15–20.
- Bro, http://www.bro-ids.org.
- Network forensic frameworks: Survey and research challenges Emmanuel S. Pilli*, R.C. Joshi, Rajdeep Niyogi, DIGITAL INVESTIGATION 7(2010)14–27
- Broucek V, Turner P. Forensic computing: developing a conceptual approach for an emerging academic discipline. In: Fifth Australian Security Research Symposium; July 2001.
- Digital Forensics for Network, Internet, and Cloud Computing: A Forensic Evidence Guide for Moving Targets and Data, Clint P Garrison
- TCPDstat, http://staff.washington.edu/dittrich/talks/core02/tools.
- TCPDump, http://www.tcpdump.org.
- TCPFlow, http://www.circlemud.org/jelson/software/tcpflow.
- TCPReplay, http://tcpreplay.synfin.net/trac/.
- TCPStat, http://www.frenchfries.net/paul/tcpstat.
- TCPTrace, http://www.tcptrace.org.
- TCPXtract, http://tcpxtract.sourceforge.net.
- Sebek, http://projects.honeynet.org/sebek/.
- Shanmugasundaram K, Memon N, Savant A, Bronnimann H. ForNet: a distributed forensics network. In: Proceedings of the second international workshop on mathematical methods models and architectures for computer networks security (MMM–ACNS 2003), LNCS 2776. Springer; 2003. p. 1–16.
- SilentRunner, http://www.accessdata.com/silentrunner.html.
- SiLK, http://tools.netsa.cert.org/silk/.
- Sira R. Network forensics analysis tools: an overview of an emerging technology, GSEC, version 1.4; Jan. 2003.
- Snort, http://www.snort.org.

- Solera DS 5150, DeepSee, http://www.soleranetworks.com.
- Flow-tools, http://www.splintered.net/sw/flow-tools.
- Garfinkel S. Network forensics: tapping the Internet, http://www. oreillynet.com/pub/a/network/2002/04/26/nettap.html.
- Guan Y. Network forensics. In: Computer and information Security handbook. Morgan Kaufmann Publishers; 2009. p. 339–47.
- Infinistream, http://www.netscout.com/Products/infinistream.asp.
- Iris, http://www.eeye.com/Iris.