Chapter 11

INTEGRATING DIGITAL FORENSICS IN NETWORK INFRASTRUCTURES

Kulesh Shanmugasundaram, Hervé Brönnimann and Nasir Memon

Abstract This paper explores the idea of integrating digital forensic capabilities into network infrastructures. Building a forensic system for wide area networks has generally been considered infeasible due to the large volume of data that potentially has to be processed and stored. However, it is opportune to revisit this problem in the light of recent advances in data streaming algorithms, the abundance of cheap storage and compute power and, more importantly, increased threats faced by networked infrastructures. This paper discusses the challenges involved in building reliable forensic systems for wide area networks, including the Internet itself. Also, it describes a prototype network forensic system that is currently under development.

Keywords: Network forensics, wide area networks

1. Introduction

The Internet was created using simple rules to serve the communications needs of a well-defined community. During the past ten years, however, it has transformed itself to accommodate a much wider and varied community of users and services with conflicting interests. Although the benefits of a forensic system on a critical infrastructure are obvious, thus far no attempts have been made to incorporate forensic capabilities within the Internet. One reason is the technical difficulty involved in handling large volumes of data on wide area networks. Another is the lack of economic incentives for network operators to deploy and maintain forensic systems. Hence, state-of-the-art network forensic systems simply incorporate augmented packet capture tools for collecting data and *ad hoc* analysis tools for dissecting data. This approach has many inherent problems. Besides being an economic burden on net-

work operators, such systems cannot share data across networks and are, therefore, not very effective. They also lack explicit support for user privacy. Therefore, few incentives exist for network operators to deploy and maintain such systems.

We believe that the lack of forensic capabilities in networks can no longer be ignored; it is the right time to address the problem with forethought and scientific rigor. Advances in streaming algorithms [1] and the availability of cheap storage and computing power can be leveraged to handle large volumes of data. Renewed threats on network infrastructures, increases in cyber crime and litigation in which operators and service providers are held liable for damages are creating the business case for network forensic systems.

This paper presents an approach to implementing network forensics. Also, it discusses the challenges involved in building reliable forensic systems for wide area networks, including the Internet itself.

1.1 System Overview

We are currently building a prototype system to support wide area network forensics. We begin by briefly describing a simple use case of the system. We envision each network domain to have a monitoring policy, privacy policy, and a forensic server associated with it. Further, we assume that network components, e.g., routers and switches, are instrumented to collect data pertaining to various network events. A monitoring policy for the domain describes the events that are monitored and the data collected by the system. The privacy policy for the domain describes access control information on the data. A forensic server is responsible for enforcing the monitoring policy in cooperation with network components in its domain and forensic servers in neighboring domains. It is also responsible for sharing information with appropriate entities in accordance with the privacy policy.

When a user connects to a network, the monitoring and privacy policies are presented to the user, say, via a DNS-like query from the application layer. The policies inform the user of the data collected by the network and how the data are shared. As the user utilizes various networks (enterprise LANs, WLANs, ISP networks, etc.) that form the Internet, each network component collects and records data according to its domain's monitoring policy. The data are either stored within the network components or are transferred to forensic servers for archiving.

When an investigation is launched, e.g., to trace an attack, or even to verify a simple transaction like establishing a connection, an analyst queries the network for relevant information. The forensic servers route

these queries to the appropriate domains, verify the authenticity of the queries, and return the relevant information. For example, suppose the analyst suspects that a malicious connection has been established from a neighboring network, he/she can query the network to assert that it is, indeed, originating from the network and is not being spoofed. Ultimately, if the analyst wants specific details or the connection history of the remote host he/she may have to present an electronic subpoena as defined in the domain's privacy policy. Upon verifying the subpoena, the forensic server reveals the relevant information to the analyst.

1.2 Applications

A network forensic system has several useful applications which are highlighted below.

- **Forensics:** The primary application of a reliable network forensic system is for the *post mortem* analysis of security incidents, especially when host logs are compromised and deemed useless. Compared to the *ad hoc* methods in use today, having a systematic method to collect data across networks would provide reliable evidence as well as valuable corroborating evidence from various vantage points on the network. It would assist in answering questions like: Where did this virus first appear in the network? When and how was confidential information stolen? How did this malware arrive on this host?

- **Network Management and Security:** Forensic data is very valuable for billing, provisioning and making various network management decisions. It can also play an important role in network security. Intrusion detection systems (IDs), for instance, are not effective against slow attacks because they have narrow windows of history. The ability of networks to retain historical event data can enhance the effectiveness of IDSs by providing the contextual information needed to detect slow attacks. For example, IDSs would be able to better detect distributed slow port scans because they can use the forensic system to gather historical data and statistics from their network as well as from neighboring networks.

- **Compliance:** Many regulations (e.g., Sarbanes-Oxley, HIPAA, etc.) and standards (e.g., International Security Standard ISO 17799) pertaining to information systems require that comprehensive audit data be maintained. Obviously, the presence of a reliable network forensic system would significantly support compliance-related activities. The integration of monitoring policies into net-

work infrastructures facilitates the verification of compliance that, in turn, allows users to make their own assessments of the trustworthiness of systems.

In the following subsection, we examine existing solutions to highlight what is missing and what is needed to implement a robust network forensic system.

1.3 Current Solutions and Needs

At present, collecting forensic data is at best an *ad hoc* process. Most networks do not have any means of collecting forensic data aside from firewall logs and intrusion detection logs. The problem is that the scope of the data collected is too narrow, and novel attacks and network abuses may go unnoticed.

There are, however, some innovative solutions that capture and record traffic passing through concentration points in networks [4, 5, 9]. The idea is that the captured data would be available for *post mortem* analysis if needed. Since these tools usually sniff traffic at a single point in a network, they lack the advantages of having multiple views from different vantage points, which would, for example, help spot spoofing attacks. Network Flight Recorder [8] uses distributed sensors to capture and record events. Most solutions, however, take the brute force approach of simply recording raw network traffic. On large networks the amount of traffic would make such solutions infeasible. Furthermore, storing raw data reduces the longevity of data as the storage requirements can be overwhelming, which in turn limits how far back investigations can go. Existing tools do not share the stored information across domains nor do they address privacy issues. Therefore, their deployment has been limited to small LANs, and they do not scale well for large, heterogeneous networks like the Internet. Although distributed intrusion detection systems [6, 7, 13] and traffic measurement systems [2] can be adapted to collect audit data, they lack many of the features required by a network forensic system. Finally, as these solutions augment the existing infrastructure, they add a burden on network operators by incorporating extra hardware and software.

What is needed is a forensic solution that is seamlessly integrated into the very fabric of a network and is transparent to network operators and users, which would make it easy to deploy, maintain and use. Integrating resource-intensive data collection into network infrastructures is challenging as it has to make do with limited memory and processing power. However, a network forensic system is only as effective as the comprehensiveness and longevity of the data it collects. This raises sev-

eral important questions: What events should the forensic system keep track of? How much storage will it take? How much of the existing infrastructure can be reused? What new functionality is expected of networks? How will the added functionality affect routine operations? How should the system handle privacy and security?

2. Research Challenges

An effective network forensic system should monitor and record numerous events on a network to facilitate end-to-end pictures of these events in both macroscopic and microscopic forms. For example, suppose we are interested in the connections between two hosts during some time interval. A macroscopic view of these connections would provide details about the number of connections, the average lifetime of connections and the total amount of data transferred. A microscopic view, on the other hand, would provide details about the individual connections between the hosts, e.g., lifetime of each connection, amount of data transferred, size of individual packets, type of content (audio, text, etc.) associated with each connection and, perhaps, the contents of the connections as well.

This raises an interesting question. How is a network forensic system different from approaches proposed for embedding measurement capabilities in network infrastructures? The difference is that a network forensic system is more general in its purpose and in the information that can be inferred from it (of course, a system that provides fine-grained information can be used for measurements as well). Furthermore, a network forensic system functions without or with only minimal *a priori* knowledge on how the collected data will be used: Will it be used for billing (macroscopic view)? Or will it be used to solve a crime (microscopic view)? Clearly, the data collected by a network forensic system covers a much wider spectrum of events than that collected by a measurement system. Consequently, a network forensic system is more general than existing systems. Much work needs to be done to realize this vision. The following sections discuss some of the challenges that lie ahead.

2.1 What to Collect?

What kind of "events" should a network forensic system track? Ideally, the data collected should help determine the exact path and payload of an arbitrary network packet in the distant past. But this is the ideal case. The challenge then lies in identifying important events and keeping track of these events in an efficient manner. Although the Internet is built on stateless IP, applications and protocols make it a stateful

network. A network forensic system must, therefore, keep track of state changes in this network and its associated entities over time. Information required to keep track of these states is grouped into two categories.

- **Network Dynamics:** The Internet is a very dynamic environment. Different vantage points have different views of the same set of networks and these views may also depend on the protocol levels from which they are viewed. For example, a DNS request for some arbitrary domain may result in two different IP addresses depending on the geographic location of the request and the time of request. Also, two neighboring networks during working days may not be neighbors at nights and weekends due to peering agreements among service providers. The network dynamics are the result of mappings used by the protocols (ARP, DNS, BGP, OSPF, etc.) that glue thousands of networks to form the Internet. A network forensic system must keep track of all this information and use the appropriate information to process forensic queries. For example, it would be incorrect to use current DNS entries to resolve the domain name of an event from the previous month.
- **Traffic Dynamics:** A network forensic system must keep track of information about billions of packets. This information, which is useful for providing macroscopic and microscopic views, includes various global quantities and network traffic trends, packet counts and/or volume broken down by categories (service, source, destination), packet headers, content types, and the payloads themselves.

2.2 How to Store?

Obviously, keeping track of all this information requires massive storage even by present standards. For instance, collecting 64-byte headers of all packets traversing a partial OC3 line in a moderately large network requires 100 GB of storage a day! Therefore, the next logical question is how should the collected data be stored? One approach is to create a "synopsis" of data [12]. A synopsis is a compact representation of the underlying data that is easy to compute. An ideal synopsis would not have any information loss. However, a good synopsis would have a small memory footprint and still capture enough information to be useful for forensic investigations.

Recent advances in streaming algorithms provide mechanisms for creating synposes. Measurement algorithms can be used to develop synopses, but the information they capture is not of sufficient granularity to support forensic queries. Therefore, synopses used in network forensic systems call for a different breed of algorithms. Several streaming

algorithms have been proposed for measuring traffic flow properties in networks. Most of the algorithms provide answers to only a subset of flows, known as "heavy hitters." A network forensic system, however, needs to keep track of all flows, not just heavy hitters. Therefore, these algorithms cannot be used in a network forensic system. A suitable algorithm for collecting traffic dynamics employs space-code bloom filters [3], which help keep track of all flows in a memory-efficient manner.

2.3 How to Retrieve?

Data on wide area networks are widely dispersed. How should the network forensic system locate the data needed for analysis? How could the data be transported efficiently across networks? How should the system present this information to an investigator? The details of the types of data being collected by different nodes in a network, what they collect and their location should be transparent to the investigator. To meet these challenges, novel protocols and distributed query processing mechanisms must be developed to permit the appropriate information to be located within a reasonable amount of time. Moreover, the network must be aware of the privacy requirements of network domains and make sure that these requirements are met. Note that standard relational databases cannot be used because different types of events may be captured in different synopses. For example, TCP connection establishment (connection records) could be stored as a table whereas traffic characteristics could be stored in a tree-based histogram. Even simple arithmetic operations can differ from one synopsis to another. This requires the development of novel database management systems for handling a variety of data types transparently. Since the information presented by the system can be overwhelming for an investigator, appropriate data analysis and visualization tools must be developed.

2.4 Infrastructure Issues

Changing a network infrastructure to accommodate a forensic system presents a set of challenges on its own. How will this system scale on the Internet? What about deployment and coverage? On large networks, e.g., national defense networks and transcontinental private networks, deployment and coverage are relatively easy as there is centralized administrative control. The Internet, however, presents major challenges in terms of deployment and coverage. Still, we believe that even with minimum adaptation to legacy systems, a network forensic system can facilitate the capture of valuable information. For example, if all the upstream paths to a rogue network are covered, it is still possible to in-

fer substantial information about the network using macroscopic views. Support for incremental deployment is another key factor as the adaptation of a network forensic system could go on for several years.

2.5 Privacy and Security

How should a network forensic system support user privacy? How can anonymity on networks be maintained? Network users may have their own privacy requirements (in addition to the prevailing privacy policies) and the network infrastructure should be attentive to these user requirements. Privacy-aware routing protocols must be developed and deployed to assist privacy negotiations in networks. Furthermore, policy monitoring must be integrated into the existing network infrastructure so that user applications can perceive it as being an integral part of the network. Currently, we plan to extend DNS to support monitoring policies. However, privacy negotiations and privacy-aware routing are open questions.

How can the network forensic system be made secure? How should trust be established among nodes? How can the risk of system compromise be minimized? The hierarchical nature of the Internet makes it possible to establish trust using certificates. Proper access control mechanisms and compartmentalization can minimize the risk of system compromise. An interesting research problem is to develop protocols and algorithms that enable network components to collectively decide on the types of events they should monitor to minimize exposure.

2.6 Rethinking Existing Solutions

Once network forensic systems become widely available it would be necessary to rethink many existing solutions so that they can take advantage of the wealth of information available in networks. For example, IDSs can be made more effective by using data collected by a network forensic system. Routing protocols can use historic data to improve their services, and network measurements will be more accurate because data can be gathered from multiple points in a network.

3. The ForNet System

This section describes the architecture of ForNet [12], a network forensic system that articulates the vision outlined in this paper. ForNet has two major functional components: (i) SynApp, which is integrated into networking components (e.g., routers and switches) to provide a secure, scalable and modular environment for collecting data, and (ii) ForensicServer, which functions as a centralized administrative controller for the

domain. A forensic-server is comparable to a DNS server; in fact, we propose to extend DNS to support network forensics.

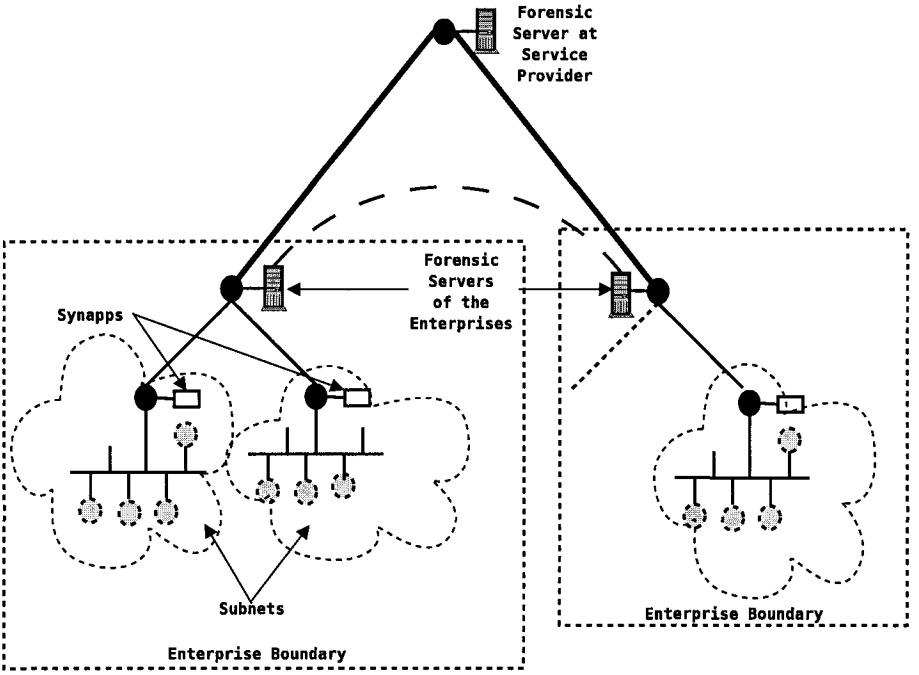

Figure 1. Hierarchical architecture of ForNet with forensic-servers in each network domain and networking components with embedded SynApps.

3.1 System Architecture

The network components instrumented with SynApps and forensic-servers are interconnected in a hierarchy (Figure 1). All SynApps within a domain form a network and are associated with the forensic-server of the domain. The forensic-server receives queries from outside domain boundaries, authenticates the requests and passes them on to appropriate SynApps for processing. Also, it certifies the responses from the agents and send them back to the originators. In addition, the forensic-server advertises the monitoring and privacy policies of the domain and enforces them by verifying that the SynApps and their responses adhere to the policies. Finally, the forensic-server functions as an archiver, receiving collected data from SynApps and storing them for prolonged periods of time (Figure 2).

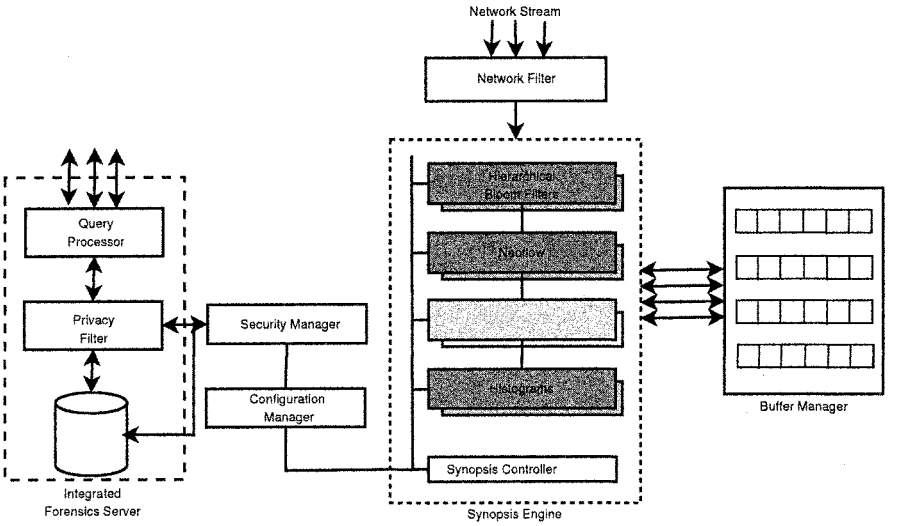

Figure 2. Architecture of a SynApp with an integrated forensics-server.

3.2 Data Architecture

As discussed earlier, data must be collected to facilitate macroscopic and microscopic views. Note, however, that creating microscopic views at the core of Internet is a resource-intensive task due to high traffic volumes. ForNet leverages the hierarchical nature of the Internet to solve this problem. Specifically, SynApps at the leaves of the hierarchy collect fine-grained information necessary for microscopic views whereas SynApps closer to the core collect coarse information necessary for macroscopic views. For instance, a switch at a subnet collects packet content details, an upstream router collects only connection records and an edge router at a service provider collects simple traffic statistics.

This so-called "cascading data collection" has two advantages. First, the data collection workload is spread across the network: SynApps closer to the core only collect macroscopic data and are not overwhelmed by the added functionality. Second, the privacy of users is maintained to a certain extent as SynApps outside a network can only generate macroscopic views and rely on SynApps within the network for detailed microscopic views. This feature naturally helps reduce the amount of storage space required at the network components closer to the core of the network as the traffic volumes are much higher than that at the leaves of the hierarchy.

```
<request>
  <command name="query" scope="module"
      module="ConnectionRecord"
  <event>
    <aspect key="StartTime">X</aspect>
    <aspect key="EndTime">Y</aspect>
  </event>
</request>
```

Figure 3. Query for connections between time X and time Y.

```
<request>
  <command name="query" scope="module"
      module="ConnectionRecord">
  <event>
    <aspect key="StartTime">X</aspect>
    <aspect key="EndTime">Y</aspect>
    <aspect key="DestinationPort">445</aspect>
  </event>
</request>
```

Figure 4. Query for events where destination port is 445.

3.3 Data Structures

Collecting data in a hierarchical manner balances the workload but it does not alleviate storage requirements. A set of synopses is used to reduce the amount of stored data. ForNet uses synopses for capturing compact payload digests in a manner outlined in [10] and for identifying content-types of flows as described in [11]. In addition, ForNet keeps track of connection records, histograms of various traffic statistics, and DNS resolutions. The modular design of SynApp allows the loading and unloading of synopses as necessary. This, in turn, permits cascading data collection in the network. For example, the switches or routers in the subnet level create extended connection records (approx. 30 bytes per record) and require stateful inspection of packets. The routers at the edge create simple connection records with source, destination ports and IP addresses (approx. 12 bytes per record). Upstream routers, on the other hand, uses bloom filters to store simple connection records, which reduce the storage requirement to about 2 bytes per record.

Clearly, cascading data collection reduces the storage requirements at different levels of the network hierarchy while preserving forensic information. Note also that the coarseness of data increases as the network core is approached. This helps strike a balance between security and privacy as it is difficult (if not impossible) to gain meaningful correlations without using information stored at the leaves of a network hierarchy.

3.4 Query Processing and Routing

Cascading data collection results in distributing the collected data evenly across networks. This approach necessitates a reliable query mechanism that can "crawl" a network for data relevant to an event or an investigation. In the ForNet system, a query is defined as a partial description of an event. ForNet uses XML for query expressions and query routing protocols. Figure 3 shows a sample ForNet query for information about all connections during a time interval. Figure 4 shows a more specific query where the destination port is 445.

Of course, we would like to have a query processor where the details of where (which SynApp) and how (in which synopsis) the data are stored are transparent to investigators. ForNet makes query processing transparent using a distributed query processor capable of efficiently locating data within a single SynApp scattered in multiple synopses and in multiple SynApps across networks. ForNet queries can be divided into three major categories: Synopsis Scope Queries, SynApp Scope Queries and Network Scope Queries. The "scope" of a query determines where the data lies in the vast network for SynApps.

- **Synopsis Scope Query:** In a Synopsis Scope Query, an investigator specifically asks ForNet to query a particular synopsis in a SynApp and/or set of SynApps. This is the simplest type of query (see Figures 3 and 4).

- **SynApp Scope Query:** A SynApp Scope Query is produced when an investigator is not sure which synopses must be queried. Instead, he/she sends a general query to a SynApp. The query processor generates a query plan which describes the synopses required to answer the query, the order of chaining of the synopses, and the appropriate merge order of query results from synopses. Then, the query processor executes this query plan and returns the results to the investigator.

- **Network Scope Query:** A Network Scope Query is produced when an investigator wants ForNet to crawl multiple networks for a particular event. Network Scope queries are useful for correlating various events.

During an investigation, queries from a network or set of networks are sent to the appropriate forensic-servers to retrieve evidence. A forensic query is simply a description of one or more events in a set of networks within a certain time interval. A query may describe an event partially and request that the details be filled in by the network. A

```
<request>
  <event>
    <aspect key="payload">0xCAFEBABE</aspect>
  </event>
  <select>
    <aspect key="src" type="ip"/>
    <aspect key="dst" type="ip"/>
    <aspect key="src" type="port"/>
    <aspect key="dst" type="port"/>
    <aspect key="any"/>
  </select>
</request>
```

Figure 5. Network scope query for payload 0xCAFEBABE.

query may be sent to a forensic-server of a network or it may be propagated to forensic-servers in neighboring networks to gather additional information. Queries that cross domain boundaries must go through forensic-servers.

Note that the route a query takes depends on the evidence being sought and is independent of the network topology. At each hop (in this case, a hop is one forensic-server), the query is evaluated and the results are returned. Then, the query moves to the next hop as determined by the results. This propagates until the query terminates within another network. In practice, queries generally begin in one enterprise network and terminate at another enterprise network. In between, the queries travel through all the forensic-servers in the hierarchy until the two endpoints are linked.

Figure 5 presents a sample ForNet query that asks for the source IP, destination IP, port numbers and "any" related information on packets that carried the string "0xCAFEBABE." Note that an "event" is not necessarily related to network changes. Rather, it is an event of interest to an investigator; in this case, it is the occurrence of string "0xCAFEBABE." Readers are referred to [10] for a discussion of similar queries that were used to trace the MyDoom virus in a campus network.

4. Conclusions

This paper deals with the issue of integrating forensic capabilities into network infrastructures. It discusses the research and implementation challenges, and describes an approach for addressing the problem. The prototype system, ForNet, which is currently under development, incorporates novel strategies such as synopses and cascading data collection. The applications of synopses to tracing payloads and detecting network abuses demonstrate its potential for wide area network forensics.

References

[1] B. Babcock, S. Babu, M. Datar, R. Motwani and J. Widom, Models and issues in data stream systems, *Proceedings of the ACM Symposium on Principles of Database Systems*, 2002.

[2] C. Cranor, T. Johnson, O. Spataschek and V. Shkapenyuk, Gigascope: A stream database for network applications, *Proceedings of the ACM SIGMOD Conference on Management of Data*, 2003.

[3] A. Kumar, L. Li and J. Wang, Space-code bloom filter for efficient traffic flow measurement, *Proceedings of the Internet Measurement Conference*, pp. 167-172, 2003.

[4] A. Mitchell and G. Vigna, Mnemosyne: Designing and implementing network short-term memory, *Proceedings of the International Conference on Engineering of Complex Computer Systems*, 2002.

[5] Network General, Infinistream (www.networkgeneral.com).

[6] V. Paxson, Bro: A system for detecting network intruders in real-time, *Proceedings of the Seventh Annual USENIX Security Symposium*, 1998.

[7] P. Porras and P. Neumann, Emerald: Event monitoring enabling responses to anomalous live disturbances, *Proceedings of the National Information Systems Security Conference*, 1997.

[8] M. Ranum, K. Landfield, M. Stolarchuk, M. Sienkiewicz, A. Lambeth and E. Wal, Implementing a generalized tool for network monitoring, *Proceedings of the Eleventh Systems Administration Conference*, 1997.

[9] Sandstorm Enterprises, Netintercept (www.sandstorm.net).

[10] K. Shanmugasundaram, H. Bronnimann and N. Memon, Payload attribution via hierarchical bloom filters, ISIS Technical Report, Polytechnic University, Brooklyn, New York, 2004.

[11] K. Shanmugasundaram, M. Kharazi and N. Memon, Nabs: A system for detecting resource abuses via characterization of flow content type, *Proceedings of the Annual Computer Security Applications Conference*, 2004.

[12] K. Shanmugasundaram, N. Memon, A. Savant and H. Bronnimann, Fornet: A distributed forensics system, *Proceedings of the Second International Workshop on Mathematical Methods, Models and Architectures for Computer Network Security*, 2003.

[13] V. Yegneswaran, P. Barford and S. Jha, Global intrusion detection in the DOMINO overlay system, *Proceedings of the Network and Distributed System Security Symposium*, 2004.