2013

Cloud Computing-Based Forensic Analysis for Collaborative Network Security Management System

Zhen Chen  
*Research Institute of Information Technology and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing 100084, China*

Fuye Han  
*Department of Computer Science and Technology, Research Institute of Information Technology and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing 100084, China*

Junwei Cao  
*Research Institute of Information Technology and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing 100084, China*

Xin Jiang  
*Department of Computer Science and Technology, Research Institute of Information Technology and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing 100084, China*

Shuo Chen  
*Department of Automation, Research Institute of Information Technology and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing 100084, China*

Part of the [Computer Sciences Commons](https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology), and the [Electrical and Computer Engineering Commons](https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology)

**Recommended Citation**


This Research Article is brought to you for free and open access by Tsinghua University Press: Journals Publishing. It has been accepted for inclusion in Tsinghua Science and Technology by an authorized editor of Tsinghua University Press: Journals Publishing.
Cloud Computing-Based Forensic Analysis for Collaborative Network Security Management System

Zhen Chen*, Fuye Han, Junwei Cao, Xin Jiang, and Shuo Chen

Abstract: Internet security problems remain a major challenge with many security concerns such as Internet worms, spam, and phishing attacks. Botnets, well-organized distributed network attacks, consist of a large number of bots that generate huge volumes of spam or launch Distributed Denial of Service (DDoS) attacks on victim hosts. New emerging botnet attacks degrade the status of Internet security further. To address these problems, a practical collaborative network security management system is proposed with an effective collaborative Unified Threat Management (UTM) and traffic probers. A distributed security overlay network with a centralized security center leverages a peer-to-peer communication protocol used in the UTMs collaborative module and connects them virtually to exchange network events and security rules. Security functions for the UTM are retrofitted to share security rules. In this paper, we propose using cloud storage to keep collected traffic data and then processing it with cloud computing platforms to find the malicious attacks. As a practical example, phishing attack forensic analysis is presented and the required computing and storage resources are evaluated based on real trace data. The cloud-based security center can instruct each collaborative UTM and prober to collect events and raw traffic, send them back for deep analysis, and generate new security rules. These new security rules are enforced by collaborative UTM and the feedback events of such rules are returned to the security center. By this type of close-loop control, the collaborative network security management system can identify and address new distributed attacks more quickly and effectively.

Key words: cloud computing; overlay network; collaborative network security system; computer forensics; anti-botnet; anti-phishing; hadoop file system; eucalyptus; amazon web service

1 Introduction and Background

With the Internet playing an increasingly important role as our information infrastructure, the e-business and e-pay sector is booming due to its convenience and benefits for users. However, Internet security remains a big challenge as there are many security threats. The underground economics based on Internet scams and frauds is also booming. Attackers increasingly initiate e-crime attacks and abuses[1-5], such as spam, phishing attacks, and Internet worms. Firewalls, Intrusion Detection Systems (IDS), and Anti-Virus Gateway are now widely deployed in edge networks to protect end-systems from attack. When malicious attacks have fixed patterns, they can be easily identified by matching them to known threats[6-9]. However, sophisticated attacks are distributed over the Internet, have fewer characteristics, and evolve quickly. For example, a Distributed Denial of Service (DDoS) attack contains...
very few, if any, signature strings to identify. Nowadays, DDoS attacks are likely to be launched by a large volume of bots—a botnet—controlled by a bot-master. The bots are commanded to create zombie machines and enlarge the botnet as well as to disseminate spam or to launch DDoS attacks on victim hosts. To countermeasure botnets, a secure overlay is proposed. To prevent distributed attacks, collaboration is required. Collaborative intrusion detection system is reviewed by researches in Ref. [10]. By collaborating, the network security system embraces scalability and teamwork and has a better overview of events in the whole network. A collaboration algorithm is presented to improve the alert events accuracy by aggregating information from different sources in Ref. [11]. A similar alert correlation algorithm[12] was put forward, which is based on Distributed Hash Tables (DHT).

The Collaborative Network Security Management System (CNSMS)[13] aims to develop a new collaboration system to integrate a well deployed Unified Threat Management (UTM) such as NetSecu[14]. Such a distributed security overlay network coordinated with a centralized security center leverages a peer-to-peer communication protocol used in the UTMs collaborative module and connects them virtually to exchange network events and security rules. The CNSMS also has a huge output from operation practice, e.g., traffic data collected by multiple sources from different vantage points, operating reports and security events generated from different collaborative UTMs, etc. As such data is huge and not easy to analyze in real-time, it needs to keep them archived for further forensic analysis[15-18].

In this paper, we evaluate a cloud-based solution in the security center for traffic data forensic analysis. The main contribution of our work is that we propose a practical solution to collect data trace and analyze these data in parallel in a cloud computing platform. We propose to use cloud storage to keep huge volume of traffic data and process it with a cloud computing platform to find the malicious attacks. As we already operate a CNSMS that has a big data output, a practical example of phishing attack forensic analysis is presented and the required computing and storage resources are investigated. We have concluded that this phishing filters functions can be effectively scaled to analyze a large volume of trace data for phishing attack detection using cloud computing. The results also show that this solution is economical for large scale forensic analysis for other attacks in traffic data.

2 Collaborative Network Security Management System

2.1 System design and implementation

CNSMS[13] is deployed in a multisite environment as shown in Fig. 1 and includes Beijing Capital-Info network, IDC Century-Link, an enterprise network, and a campus network to demonstrate the workability of our system. These four sites are all managed by CNSMS in the remote security center. In each site, there are several NetSecu nodes[14, 19] that take charge in different network environments to adapt to different physical links.

During the systems operation, the collaborative mechanism runs as expected to share security events and rulesets, and new rulesets are enforced on demand as instructed by the security center. Operating reports from each NetSecu node and Prober are collected and sent back to the security center. Many network security events are observed and recorded in the deployment, such as DDoS reflected attacks, spam scatter and ad hoc P2P protocols, etc.

Figure 2 illustrates the whole procedure of network
security events processing. In general terms, it is an information control cycle that divides into several steps. Collaborative UTM and Probers act as sensors and report the security events and traffic data to the security center, which aggregates all the events and investigates the collected traffic data. After a detailed analysis, and with the assistance of expertise, the security center generates new policies or rulesets to disseminate to each collaborative UTM and Prober for enforcement and to receive feedback information.

2.1.1 Traffic prober

A traffic probe is the building block for recording the raw Internet traffic at connection level. Hyperion\(^{[20]}\), Time Machine\(^{[21, 22]}\), and NProbe\(^{[23]}\) are all well-known representative projects in this function area. The traffic probe can be designed to focus on specific traffic occasioned by certain security events when needed. We adapted Time Machine and deployed with TIFA\(^{[24, 25]}\) acting as prober in either a separate device or collaborative UTM. The key strategy for efficiently recording the contents of a high volume network traffic stream comes from exploiting the heavy-tailed nature of network traffic; most network connections are quite short, with a small number of large connections (the heavy tail) accounting for the bulk of the total volume\(^{[22]}\). Thus, by recording only the first $N$ bytes of each connection (the cutoff is 15 KB), we can record most connections in their entirety, while still greatly reducing the volume of data we must retain. For large connections, only the beginning of a connection is recorded, as this is the most interesting part (containing protocol handshakes, authentication dialogs, data item names, etc.).

2.1.2 Collaborative UTM

Treated as a collaborative UTM, NetSecu is introduced in Ref. [14]. A NetSecu node consists of the following features:

1. Incrementally deployable security elements,
2. Can dynamically enable/disable/upgrade security functions,
3. Policy-instructed collaboration over the Internet.

NetSecu node contains Traffic Prober, Traffic Controller, Collaborator Element, and Reporting Element to fulfill the above design goals.

A collaborator element in NetSecu manages other security elements based on the security centers command. It unites individual NetSecu platforms into a secure overlay network. The communication command between NetSecu nodes and the security center is transmitted in an SSL channel to ensure security. A collaborator can start or stop a security element at runtime and can respond to security events by, for example, limiting the DDoS traffic on demand.

NetSecu integrates security functions such as firewall, Intrusion Detection System (IPS), and Anti-Virus (AV). These functions can be loaded in NetSecu nodes at runtime and can be dynamically enabled, disabled, and upgraded. Based on commodity hardware and commonly used Java with Linux, and with a mature multi-core technology, NetSecu has a comparable Maximum Loss-Free Forwarding Rate (MLFFR\(^{1}\)) with bare Linux forwarding performance. Most of the security functions can run in a multi-thread model to accelerate the flow processing and pattern matching needed for UTM.

NetSecu is also equipped with bypass and self-protection capability to resist DoS attacks in case of faults or malicious attacks occurring, to ensure high availability and survivability.

2.1.3 Security center

CNSMS is proposed in Ref. [13] and operated in the security center. As NetSecu nodes can manage security problems in a subdomain and provide P2P...
communication interfaces\textsuperscript{[26]}, CNSMS orchestrates the communication between these nodes. More specifically, CNSMS will achieve the following objectives:

1. Security policy collaborative dissemination and enforcement,
2. Security ruleset dissemination, enforcement, and update,
3. Security event collaborative notification,
4. Trust infrastructure,
5. Scalability.

Another key function in the security center is the forensic analysis of the collected traffic and network security events. We used cloud computing in the security center to store a large volume of traffic data of different origins and conducted data analysis to generate new security rulesets as shown step 6 in Fig. 2.

To further inform the UTM how to defeat new attacks, such as a botnet, we must investigate the traffic in depth, acquire the communication graph of the botnet, and generate security rules for enforcement in the UTM to suppress the communication between bots and botmaster.

This makes it possible to resist a DDoS attack launched by a botnet. As we equipped the NetSecu node with open source application protocol identification and bandwidth management technology, the security center could instruct the system to be a collaborative distributed traffic management system, which detects and manages the traffic collaboratively after the analysis of collected traffic in the security center. It could effectively improve the identification ratio of unknown botnet protocols and throttle the DDoS traffic.

2.2 System application-botnet suppression

A botnet is a typical distributed attack, which is extremely versatile and is used in many attacks, such as sending huge volumes of spam or launching DDoS attacks. The work principle of a botnet is shown in Fig. 3. Suppressing botnets is increasingly difficult because the botmaster will keep their own botnets as small as possible not only to hide themselves but also to spread the botnets in an easy way. Additionally, bots can automatically change their Command and Control server (C&C) in order to hide and rescue themselves.

Based on an overlay network, Collaborative Network Security System can be used for a distributed botnets suppression system, automatically collecting network traffic from every collaborative UTM in a distributed mode and then processing these collected data in the security center. The detection algorithm proposed by Refs. [27, 28] is based on behavior features of botnets so the system will generate and distribute rules when botnets are detected in processing. The most important feature of this system is its close loop control characteristics, i.e., gathering the feedback events resulting from the deployed rules, processing and analyzing in control nodes, removing invalid rules to make the system more efficient and reliable.

3 Cloud-Based Forensic Analysis in Security Center

3.1 Cloud storage and computing platform

We focus on traffic data storage and forensic analysis. The underground cloud storage and computing platform is based on Hadoop and Eucalyptus cloud computing. We also give some analysis of the use of cloud computing platforms based on Eucalyptus and Amazon EC2 respectively.

3.1.1 Cloud storage with Hadoop

The Hadoop file system\textsuperscript{[29, 30]} with version 1.0.1 is used for the cloud storage system of collected traffic. The master node is working as NameNode, SecondaryNameNode, JobTracker, Hamster, while the other nodes are working as DataNode, TaskTracker, RegionServer.

There are 4 racks of machines with 5,5,4,4 in each rack making 18 slave nodes in total. The topology is shown in Fig. 4.

The Hadoop system is used for traffic analysis whereby the traffic collected in each individual collaborative UTM is aggregated and uploaded to this cloud platform. Each node has an Intel 4 core CPU with 800 MHz, 4 GB memory, and a 250 GB hard disk.

We tested the writing throughput for our Hadoop system with Hadoops TestDFSIO utility\textsuperscript{2}. We also


diagram

\[\text{Botmaster} \quad \text{Bot} \quad \text{C&C server} \quad \text{Bot} \quad \text{Bot} \quad \text{Target}\]

\[\text{Fig. 3 Botnet structure.}\]
tested two scenarios where we wrote 18 files of 300 MB each and 36 files of 100 MB each. The final results are shown in Table 1.

3.1.2 Cloud computing IaaS platform

3.1.2.1 Cloud computing based on Eucalyptus

In this section, we introduce our cloud computing platform based on Eucalyptus, an open-source platform used by NASA and Ubuntu Enterprise Cloud.

Figure 5 shows the Eucalyptus cloud computing platform we used. As shown in Fig. 5, Eucalyptus Compute consists of seven main components, with the cloud controller component representing the global state and interacting with all other components. An API Server acts as the web services front end for the cloud.

Table 1 The average writing throughput of Hadoop files system in the cloud platform.

<table>
<thead>
<tr>
<th>Number of files</th>
<th>File size (MB)</th>
<th>Total write throughput (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>95</td>
<td>100</td>
<td>215.120</td>
</tr>
<tr>
<td>95</td>
<td>200</td>
<td>378.630</td>
</tr>
<tr>
<td>95</td>
<td>300</td>
<td>460.055</td>
</tr>
<tr>
<td>57</td>
<td>200</td>
<td>153.390</td>
</tr>
<tr>
<td>57</td>
<td>100</td>
<td>324.830</td>
</tr>
<tr>
<td>19</td>
<td>200</td>
<td>59.670</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>119.190</td>
</tr>
</tbody>
</table>

Fig. 4 Cloud storage for traffic data collected with collaborative UTM and prober.
controller. The compute controller provides compute server resources, and the object store component provides storage services. An AuthManager provides authentication and authorization services. A volume controller provides fast and permanent block-level storage for the servers while a network controller provides virtual networks to enable servers to interact with each other and with the public network. A scheduler selects the most suitable controller to host an instance.

Our computer cluster consists of 10 heterogeneous servers. Each server is equipped with the following hardware parameters:

(a) Intel Core 2 Quad processor with 1.333 GHz FSB and 2 MB cache, double channel 4 GB DDR3 with 1.066 GHz, Intel G41 + ICH7R Chipset and Intel 82574L Network Chipset;
(b) Dual Intel Xeon5X00 series processors with Intel 5000P+ESB2 chipset, E5330 + 8 GB;
(c) Intel Xeon 5X00 series with FSB - 4.8/5.86/6.4 GT/s QPI Speed with Intel 5520 + ICH10R chipset, 24 GB.

In Eucalyptus terms, there is one cloud controller and the others are nodes. The cloud controller acts as the computing portal, task assigner, and result aggregator. There is an instance affiliated with each node. In our usage scenario, we ran 4 VM instances in each node, hence there were about 24 running instances simultaneously. Each computing instance runs the pipeline divided into the following phases: data fetcher, data processing, and posting results. Using this method, we could achieve the best working efficiency of hardware and software resources usage.

3.1.2.2 Cloud computing based on Amazon

Amazon EC2 and S3 were used for comparative analysis. The main reason for using Amazons service was to compare it to our bespoke Eucalyptus system. In consideration of user privacy and legal issues, we ensured all data was made anonymous before uploading to the Amazon S3 service.

3.1.3 Forensic analysis of phishing attack

Phishing is an intriguing practical problem due to the sensitive nature of stolen information (e.g., bank user account names and passwords) and is responsible for an estimated of billions of dollars loss annually. Not only users but the backing financial institutions such as e-banks and e-pay systems are impacted by phishing attacks.

There is already much research into phishing attack countermeasures. To protect web browser users from phishing attacks, plugins to compare visited URLs with blacklisted URLs are already provided by main-stream web browsers. Google also provides the Safe Browser API for checking an URL in Googles collected phishing database.

Some research on the life-cycle of phishing web sites is given in Ref. [2], and the results show that phishing URLs are quite ephemeral, making the collection of forensics difficult. Most internet users are oblivious to the dangers of phishing attacks, making combating them even harder.
Maier et al.\cite{36} proposed a traffic archiving technology for post-attack analysis in Bro IDS. Using Time Machine, the network trace data is archived and can be fed back to the IDS at a later date when more current data is available to use updated forensic details of attacks. Thomas et al.\cite{37} proposed the Monarch system for real-time URL spam filtering for tweets and spam mail streams, whereas we put emphasis on phishing forensic analysis of large volumes of offline trace with cloud computing platforms\cite{38}.

Similarly, we proposed an offline phishing forensic collection and analysis system. This system was targeted to solve the following challenging problems:
(1) How to collect the original data to search the phishing attack forensics therein;
(2) How to handle the huge volume of data in a reasonably short time.

A cloud computing platform\cite{39-41} was used for offline phishing attack forensic analysis. First, our CNSMS collected the network trace data and reported to the security center. Then we constructed an IaaS cloud platform\cite{42} and used existing cloud platforms such as Amazon EC2 and S3\cite{43-45} for comparison. All phishing filtering operations were based on cloud computing platforms and run in parallel with a divide and conquer scheme.

### 3.1.4 Data trace collection

Our trace data was an un-interrupted collection of about six months worth of multiple vantage points deployed by the UTM. The total size of traffic passed through our vantage points was about 20 TB. The total data was about 20 TB and divided into 512 MB data blocks. Figure 6 gives a daily traffic graph from one vantage point.

Typically, as shown in Fig. 6, HTTP traffic account for most of the daily traffic. A typical 512 MB of collected data block consists of about 40 K HTTP URLs. Counting the HTTP URLs visited by users, an explored URLs distribution is as shown in Fig. 7.

The experimental data was about 1 TB when collected in a cut-off mode in a collaborative UTM. The data trace was still growing in size during our experiments.

#### 3.1.5 Data anonymization

To protect users privacy and avoid legal issues in the research, the trace data was anonymized by replacing IP and other user information before data processing in Amazon EC2.

#### 3.1.6 Data processing

1. **File splitting:**
   
   Each packet capture file created by Time Machine is 512 MB, and is further divided into smaller parts for processing by using tcpdump\cite{46}. This is due to the amount of memory used during the extraction of data from TCP streams that would exceed the maximum physical memory.

2. **TCP stream reassembly:**
   
   This stage is to restore the TCP streams in the captured pcap files using tcptrace\cite{46}.

3. **URL extraction:**
   
   After extracting data from TCP streams, grep is used to find all URLs contained in the data by

![Fig. 6 Daily traffic observed and collected by Traffic Prober.](image-url)
searching for lines starting with “Referer: http://.”

(4) URL check:
URLs found are stored in a file to be checked for phishing by using Google’s Safe Browsing API[3]. In order to check URLs for phishing sites, we use phishing URL database of Google. Google provides the first 32 bits of phishing sites’ SHA256 values for users to use. If a match is found, the full 256 bits hash value is sent to Google to check the site. More details on data provided by Google can be found in Google Safe Browsing API’s documentation[3]. During the process of comparing URLs’ hash values, a prefix tree is used for matching because the data provided by Google is only 32 bits long and a prefix tree can do the matching of a URL’s SHA256 value with Google’s data in $O(1)$ time.

(5) Result reporter:
This stage collects the final results in different machines and aggregates the final report.

3.2 Experimental results
We conducted our evaluation experiments both on Eucalyptus and Amazon AWS for comparative purposes.

3.2.1 Eucalyptus
We ran the phishing data block processing task in our bespoke Eucalyptus platform with Intel Core 2 Quad Processor with 1.333 GHz FSB and 2 MB cache, double channel 4 GB DDR3 with 1.066 GHz, Intel G41+ICH7R Chipset, and Intel 82574L Network Chipset.

Times taken in different process stages in the Eucalyptus platform were measured and concluded as shown in Table 2.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP stream reassembly</td>
<td>1520</td>
</tr>
<tr>
<td>URL extraction</td>
<td>1620</td>
</tr>
<tr>
<td>URL check</td>
<td>5</td>
</tr>
</tbody>
</table>

It seems the prefix tree comparisons speed is quite fast and this time spending can be almost ignored. However, before checking the URL, it takes some time to download the Google Safe Browsing signature libraries. This time spending is quite undetermined due to the network status and Google servers response latencies.

It is also pointed out that the m1 small instance in EC2 is memory constrained without swap partition support. It will cause problems when consuming a large volume of memory (exceeding the memory usage limit) during trace data analysis.

3.2.2 Amazon AWS
Trace file processing is written in Python and executed on an EC2 small instance running Ubuntu Linux 10.04. As Linux command shows, the host CPU is Intel(R) Xeon(R) CPU E5430 @ 2.66 GHz with a cache size of 6 MB and 1.7 GB memory (with HighTotal: 982 MB, LowTotal: 734 MB).

Different processing stages incur different time consumptions and are measured in Table 3. TCP stream reassembly procedures still cost most of the processing time as it needs more logic in processing.

Compared with the Eucalyptus case, it seems that the CPU used in the Amazon instance has better performance than the QX9400 quad core CPU in our physical server as shown at the URL check stage. Because of large IO operations in reassembly and extraction, the Amazon case costs much more time than the Eucalyptus case.

3.2.3 Estimating the number of instances
Assume the time spent in an instance to handle a $k$-byte data block in stage (2), stage (3), and stage (4) is $t_1$, $t_2$, and $t_3$ (in seconds), respectively. Assume there are $m$ collaborative UTMs or probers to collect traffic data, the average traffic throughput is $f$ during the last 24 hours.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP stream reassembly</td>
<td>287</td>
</tr>
<tr>
<td>URL extraction</td>
<td>47</td>
</tr>
<tr>
<td>URL check</td>
<td>12</td>
</tr>
</tbody>
</table>
hours and the traffic cut-off factor is \( h \).

The number of total instances \( L \) in parallel needed to handle all of the last 24 hours traffic is calculated as follows:

\[
T = t_1 + t_2 + t_3, \\
L = (m \times f \times T \times h)/k.
\]

\( L \) is also affected by several factors such as the percentage of HTTP streams in the traffic, number of URLs in HTTP streams, the users behavior in exploring web sites, etc.

In the Eucalyptus case, we only ran one instance in each physical server. Assume \( m = 4, f = 100 \text{ MB/s} \) (800 Mbps) in 1 Gbps link, \( h = 0.2 \) (means 20% traffic is captured), each block is 200 MB, \( T = 40 \text{ s} \), then the number of physical servers (or instances) in parallel is calculated as follows:

\[
L = (m \times f \times T \times h)/k = 4 \times 100 \times 40 \times 0.2/200 = 16.
\]

In the Amazon EC2 case, \( T = 330 \text{ s} \), and the required number of EC2 m1 small instances in parallel is calculated as follows:

\[
L = (m \times f \times T \times h)/k = 4 \times 100 \times 330 \times 0.2/200 = 132.
\]

4 Conclusions

The CNSMS is very useful to countermeasure distributed network attacks. Its operation resulted in big data outputs, such as network traffics, security events, etc. In this paper, we proposed using cloud computing systems to explore the large volume of collected data from CNSMS to track the attacking events. Traffic archiving was implemented in collaborative UTMs to collect all the network trace data and the cloud computing technology was leveraged to analyze the experimental data in parallel. An IaaS cloud platform was constructed with Eucalyptus and existing cloud platforms such as Amazon EC2 and S3 were used for comparison purposes. Phishing attack forensic analysis as a practical case was presented and the required computing and storage resource were evaluated by using real trace data. All phishing filtering operations were cloud-based and operated in parallel, and the processing procedure was evaluated. The results show that the proposed scheme is practical and can be generalized to forensic analysis of other network attacks in the future.

Acknowledgements

This work is supported by the National Key Basic Research and Development (973) Program of China (Nos. 2011CB302805, 2011CB302505, 2012CB315801, and 2013CB228206), and the National Natural Science Foundation of China (No. 61233016). This work is also supported by Intel Research Councils UPO program with the title of Security Vulnerability Analysis Based on Cloud Platform.

References


J. Li, S. Ding, M. Xu, F. Han, X. Guan, and Z. Chen, TIFA: Enabling real-time querying and storage of massive stream data, in *Proc. 1st International Conference on Networking and Distributed Computing (ICNDC)*, Hangzhou, China, 2011, pp. 61-64.


T. Li, F. Han, S. Ding, and Z. Chen, LARX: Large-scale anti-phishing by retrospective data-exploring based on a cloud computing platform, in *Proc. 3rd Workshop on Grid and P2P Systems and Applications (GridPeer)*, Maui, Hawaii, 2011, pp. 1-5.


Zhen Chen is an associate professor in Research Institute of Information Technology at Tsinghua University. He received his BEng and PhD degrees from Xidian University in 1998 and 2004. He works as postdoctoral research in Network Institute of Department of Computer Science and Technology in Tsinghua University during 2004 to 2006. He is also a visiting scholar in UC Berkeley ICSI in 2006. His research interests include overlay networking architecture, Internet security, P2P systems and Trusted Computing. He has published around 60 academic papers.

Fuye Han is a master student in Department of Computer Science and Technology in Tsinghua University. He graduated from PLA Information Engineering University in 2008, major in Information Engineering. His research interests include botnet, traffic archiving and other network security issues. He joined the Cloud Computing and IoT lab in 2010.

Junwei Cao is currently Professor and Deputy Director of Research Institute of Information Technology, Tsinghua University, China. He is also Director of Open Platform and Technology Division, Tsinghua National Laboratory for Information Science and Technology. His research is focused on advanced computing technology and applications. Before joining Tsinghua in 2006, Junwei Cao was a Research Scientist of Massachusetts Institute of Technology, USA. Before that he worked as a research staff member of NEC Europe Ltd., Germany. Junwei Cao got his PhD in computer science from University of Warwick, UK, in 2001. He got his MEng and BEng degrees from Tsinghua University in 1998 and 1996, respectively. Junwei Cao has published over 130 academic papers and books, cited by international researchers for over 3000 times. Junwei Cao is a Senior Member of the IEEE Computer Society and a Member of the ACM and CCF.

Xin Jiang is working as computer security researcher. He received the PhD degree in Computer Science from Institute of Computer Network of Department of Computer Science and Technology in Tsinghua University in 2010. He got BEng degree in PLA Univ. of Sci. & Tech in 1998. His main research interests include computer network security, performance evaluation, and wireless networks.

Shuo Chen is a master student supervised by Prof. Junwei Cao from Department of Automation. He got BEng degree in Tsinghua University, Beijing, China, in 2012. His research interests include distributed computing and content centric networking.