SIP-based VoIP Traffic Behavior Profiling and Its Applications *

Hun Jeong Kang, Zhi-Li Zhang
University of Minnesota
{hkang, zhzhang}@cs.umn.edu

Supranamaya Ranjan, Antonio Nucci
Narus Inc.
{soups, anucci}@narus.com

ABSTRACT

With the widespread adoption of SIP-based VoIP, understanding the characteristics of SIP traffic behavior is critical to problem diagnosis and security protection of IP Telephony. In this paper, we propose a general methodology for profiling SIP-based VoIP traffic behavior at multiple levels: SIP server host, server entity and individual user levels. Using SIP traffic traces captured in a production VoIP service, we illustrate the characteristics of SIP-based VoIP traffic behavior in an operational network and demonstrate the effectiveness of our general profiling methodology. In particular, we show how our profiling methodology can help identify performance anomalies through a case study.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Applications —*SIP*; C.2.3 [**Network Operations**]: Network monitoring

General Terms

Management, Measurement, Reliability, Security

Keywords

Session Initiation Protocol, SIP, Traffic Profiling

1. INTRODUCTION

Voice over IP (VoIP) allows users to make phone calls over the Internet, or any other IP network, using the packet switched network as a transmission medium. The maturity of VoIP standards such as SIP [1] and quality of service (QoS) on IP networks maximize network efficiency, streamline the network architecture, reduce capital and operational costs, and open up new service opportunities such as Web-enabled multimedia conferencing and unified messaging.

VoIP offers compelling advantages, but it also presents a security paradox. The very openness and ubiquity that make IP networks such powerful infrastructures also make them a liability. Risks include Denial of Service (DoS), Service Theft, Unauthorized Call Monitoring, Call Routing Manipulation, Identity Theft and Impersonation, among others. Not only does VoIP inherit all data security risks, it also introduces new vehicles for threats related to the plethora of new emerging VoIP protocols that have yet to undergo detailed security analysis and scrutiny. There have been several reported incidents and many alerts regarding VoIP attacks or vulnerabilities (e.g., [2]). It is therefore imperative for VoIP service operators to deploy scalable monitoring and defense systems to effectively shield their VoIP infrastructure and protect their services and users against potential attacks. In addition, problem diagnosis is also essential to ensure the robustness of VoIP services. Despite the importance of VoIP problem diagnosis and security, there is relatively little research on analysis of behavior characteristics of SIP traffic – the critical control flow of VoIP services – to help design effective problem diagnosis tools and attack detection mechanisms.

This paper is the first attempt at understanding SIP traffic behavior based on traces from an operational VoIP service. In particular, we develop a novel multi-level profiling methodology for characterizing SIP traffic behavior, with the objective to help identify behavior anomalies for problem diagnosis and attack detection. Our methodology characterizes VoIP service activities by extracting and profiling a large variety of traffic features and metrics at three different levels in a progressively refined fashion: (i) SIP *server host* characterization, which provides a broad view of their behavior by monitoring and keeping statistics related to only the message types (`request` vs `response`) and user activity diversity; (ii) *server entity* characterization, which provides a functional analysis of server activities by separating their logical roles into *registrar*, *call proxy*, and so forth; and (iii) *individual user* characterization, which maintains more detailed profiles of individual user activities. The multi-level profiling enables VoIP service operators may choose to profile server/user activities at different levels depending on their needs/requirements. In other words, our methodology allows us to balance the speed of profiling, the resource consumption, the desired sophistication of behavior characteristics, and finally the level of security to be offered, based on the specific objectives and needs of the VoIP service op-

*Hun Jeong Kang and Zhi-Li Zhang were supported in part by NSF grants CNS-0435444 and CNS-0626812, a University of Minnesota Digital Technology Center DTI grant, and a Narus Inc. gift grant.

MineNet'07, June 12, 2007, San Diego, California, USA.

erator. In order to demonstrate the effectiveness of our general profiling methodology, we illustrate the characteristics of SIP-based VoIP traffic behavior using real-network SIP traffic traces, and show how our profiling methodology can help identify performance anomalies through a case study.

Related Work. While there is a considerable volume of white papers and surveys regarding various vulnerabilities and security threats towards VoIP services (see, e.g., [3]), there is relatively few research studies on these topics. Most focus on defense against specific attacks, e.g., malformed SIP message format attacks [4, 5], DoS and other call disruption attacks [6, 7, 8], and voice spams [9], albeit these studies are not based on real-network SIP traces. To the best of our knowledge our study is the first analysis of SIP traffic from an operational VoIP service and the first attempt at profiling SIP-based VoIP traffic behavior based on real-network traces.

Paper Organization. Section 2 provides some background on SIP, and briefly describes the problem setting and data sets. In Section 3, we first introduce a heuristic for discovering SIP servers from passively monitored SIP traffic, and then present our general multi-level profiling methodology for characterizing SIP traffic behavior. Section 4 applies our methodology to analyze the SIP traffic behavior using the real network SIP traces. In Section 5, we use a case study to illustrate how our methodology can help detect performance anomalies. The paper is concluded in Section 6.

2. BACKGROUND AND DATA SETS

We first provide a quick overview of SIP-based IP telephony. We then briefly touch on the challenges in profiling SIP traffic behaviors based on *passive packet monitoring*, and describe the SIP data sets used in our study.

2.1 SIP-based VoIP Service

The session initiation protocol (SIP) [1] is the Internet standard signaling protocol for setting up, controlling, and terminating VoIP sessions[1]. SIP-based VoIP services require *infrastructure* support from entities such as SIP registrars, call proxies, and so forth (see Fig. 1) – we collectively refer to these entities as *SIP servers*. A SIP registrar associates SIP users (e.g., names or identities called *SIP URIs*) with their current locations (e.g., IP addresses). A SIP call proxy assists users in establishing calls (called *dialogs* in the SIP jargon) by handling and forwarding signaling messages among users (and other SIP servers). In practice, a physical host (SIP server) may assume multiple logical roles, e.g., functioning both as registrars and call proxies.

SIP is a text-based `request-response` protocol, with syntax very similar to HTTP. SIP messages are of type either `request` or `response`. The `method` field is used to distinguish between different SIP operations. The most common `methods` include `REGISTER` (for user registration), `INVITE`, `ACK`, `BYE`, `CANCEL` (these four used for call set-up or teardown), `SUBSCRIBE`, and `NOTIFY` (for event notification). `Response` messages contain a `response code` informing the results of the requested operations (e.g., `200 OK`). The `FROM` and `TO` fields in an SIP message contain respectively the SIP URIs of the user where a `request` message is originated from (e.g., the caller of a call) or destined to (e.g., the callee of a call). The reader is referred to [1] for details.

[1]In addition to IP telephony, it can also be used for teleconferencing, presence, event notification, instant messaging, and other multimedia applications.

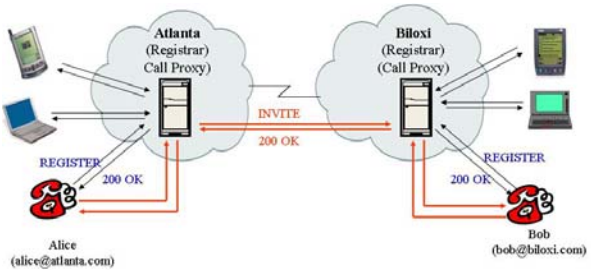


Figure 1: SIP servers and clients

2.2 Problem Discussion and Data Sets

In this paper, we focus on characterizing and profiling SIP-based VoIP traffic behavior by using *passive traffic monitoring*, with the objective to identify anomalies to help diagnose problems and detect potential attacks on critical VoIP services (and their infrastructure). We assume that passive packet monitoring and capturing devices are deployed in the underlying network hosting VoIP services. In addition to the standard layer-3 (IP) and layer-4 (TCP/UDP) header information, portion of layer-7 payload containing appropriate application protocol (SIP) fields are also captured. The captured packet header and payload information is then processed and parsed for our analysis and profiling. Unlike the layer 3/4 header fields which generally have well-defined and *limited* semantics, the layer-7 application protocol such as SIP has a variety of fields, with *rich* semantics that are often context-sensitive and sometimes even implementation-specific. Hence a major challenge in performing layer-7 protocol analysis and behavior profiling is to determine how to judiciously incorporate application-specific semantics or "domain knowledge" to select appropriate set of key features to capture the essential behavior characteristics of the application in question. In the next section we present such a general methodology for characterizing and profiling SIP-based VoIP traffic behavior.

Our profiling methodology is motivated and substantiated by in-depth analysis of SIP traffic traces captured in an operational network of a commercial wireless VoIP service provider. The results reported in this paper use three SIP traces from this network, referred to as *Trace I (13:55-14:30)*, *Trace II (19:00-19:40)* and *Trace III (19:55-20:30)*, respectively (the numbers within the parentheses indicate the start and end time of the traces). They are of about 40 minutes or so long, captured between 13:00 h and 21:00 h within a single day.

3. GENERAL METHODOLOGY

In this section, we present a multi-level profiling methodology for characterizing SIP traffic behavior using layer-3 to layer-7 protocol information obtained from *passive network monitoring*.

3.1 Discovering SIP Servers

In order to characterize and profile SIP server behaviors by using passively collected SIP traffic traces, we need to discover SIP servers. In this section, we introduce a simple heuristic for identifying the IP addresses associated with SIP servers.

The key observation behind our heuristics is based on the role of SIP servers in SIP-based VoIP communications: typically users must register with SIP registrars; and users' call signaling must get through SIP call proxies (see Fig. 1). Hence the IP address associated an SIP server will consistently see a large number of SIP messages going through it (i.e., with the said IP address as either the source or destination IP addresses); furthermore, we will also see a large number of distinct `FROM` and `TO` fields in the appropriate SIP messages (e.g., `INVITE`, `REGISTER`) associated with this IP address. The baseline algorithm for SIP call proxy discovery is given in Algorithm 1 examining the SIP `INVITE` messages. By examining the SIP `REGISTER` messages, we have a similar algorithm for SIP registrar discovery.

Algorithm 1 Algorithm for SIP Call Proxy Discovery

```
1: Parameters: message set M, threshold α;
2: Initialization: IPSet := ∅; ProxyIP := ∅;
3: for each m ∈ M do
4:   if m.method == INVITE then
5:     x = m.sourceIP; y = m.destinationIP;
6:     from = m.FROM; to = m.TO;
7:     if x ∉ IPSet then
8:       x.Out_FROM = {from}; x.Out_TO = {to};
9:       x.In_FROM = ∅; x.In_TO = ∅;
10:    else
11:      x.Out_FROM = x.Out_FROM ∪ {from};
12:      x.Out_TO = x.Out_TO ∪ {to};
13:    end if
14:    if [|x.In_FROM|, |x.In_TO|, |x.Out_FROM|, |x.Out_TO|]
       > [α, α, α, α] then
15:      ProxyIP = ProxyIP ∪ {x}
16:    end if
17:    if y ∉ IPSet then
18:      y.In_FROM = {from}; x.In_TO = {to};
19:      y.Out_FROM = ∅; y.Out_TO = ∅;
20:    else
21:      y.In_FROM = y.Out_FROM ∪ {from};
22:      y.In_TO = y.In_TO ∪ {to};
23:    end if
24:    if [|y.In_FROM|, |y.In_TO|, |y.Out_FROM|, |y.Out_TO|]
       > [α, α, α, α] then
25:      ProxyIP = ProxyIP ∪ {y}
26:    end if
27:  end if
28: end for
```

In Algorithm 1, for each IP address a in the SIP messages (either as the source or destination IP) we maintain four records, $a.In_{\texttt{FROM}}$, $a.In_{\texttt{TO}}$, $a.Out_{\texttt{FROM}}$ and $a.Out_{\texttt{TO}}$, which maintain, respectively, the set of unique users (or rather their URIs) seen in the `FROM` and `TO` fields of the SIP `INVITE` messages *received* (In) by or *sent* (Out) from a. If the number of distinct users in each of the four records exceeds a threshold α [2] for an example, then a is included in the SIP call proxy candidate set $ProxyIP$. By ensuring the diversity of callers (`FROM`) and callees (`TO`) in both the SIP `INVITE` messages originating from and destined to a given IP, we minimize the chance of misclassifying of a user in the *forward* mode in which incoming `INVITE` messages are forwarded to another location, or similarly, when a user is in a conference mode. In both cases, the `TO` field of the `INVITE` messages will contains the URI (or its variants) of the forwarder. Hence the size of corresponding $In_{\texttt{TO}}$ and $Out_{\texttt{TO}}$ will be small. We have extended the baseline algorithm to incorporate additional mechanism to address the effect of NAT boxes, and illustrate the effectiveness of our baseline algorithm using the real SIP traffic traces, the details of which can be found in [10].

[2]The threshold can be determined, for example, by first plotting $In_{\texttt{FROM}}$ vs. $In_{\texttt{TO}}$ and $Out_{\texttt{FROM}}$ vs. $Out_{\texttt{TO}}$ in a scatter plot in [10].

3.2 Profiling SIP Server and User Behaviors

Once we have identified the IP addresses associated with the SIP servers, we characterize and profile the behaviors of SIP servers by examining the SIP messages going through them. We characterize and profile the behaviors of SIP servers (and their associated users) at three levels – *server host*, *server entity* and (*individual*) *user* – by introducing a range of features and metrics from coarser granularity and finer granularity in terms of the amount of application-specific (i.e., SIP) semantic information. This *multi-level, progressively refined* methodology allows us to balance the speed of profiling, resources required, desired sophistication of behavior characteristics, and level of security, an so forth based on the objectives and needs of a SIP-based VoIP operator.

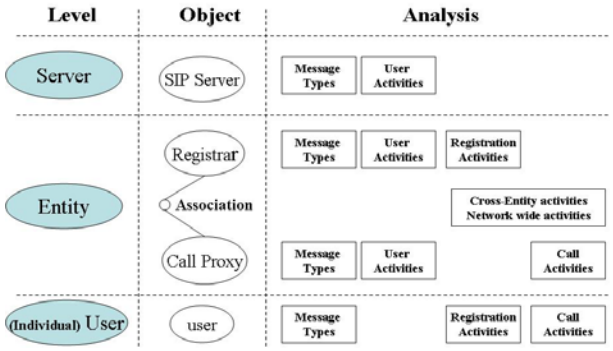


Figure 2: Multilevel Profiling

Fig. 2 is a schematic depiction of our multi-level profiling methodology. At the *server host* level we maintain only aggregate features and metrics to provide a broad view of a SIP server behavior and its "health" by examining only the message types (`request` vs. `response`) into and out of a SIP server and extracting only coarse-grain user statistics information. At the *server entity* level, we separate the (logical) role of a SIP server into *registrar* and *call proxy*, as these two separate entities require different sets of features and metrics to characterize their respective behaviors. Based on the SIP semantics, we examine the *method* field of a SIP message to attribute it to either the SIP registrar or call proxy, and compute appropriate features and metrics for the corresponding registrars and call proxies. We also cross-examine the activities of SIP registrars and call proxies to build cross-entity associations. At the (*individual*) *user* level, we attribute the SIP messages to individual users, and maintain statistics and features to characterize individual user behaviors. In the following we provide a more detailed description of our multi-level profiling methodology.

a. Server Host Level Characterization.

We characterize the aggregate behaviors of a SIP server by maintaining two types of (aggregate) statistics and features: i) we count the number of `request` and `response` messages received (i.e., *fan-in*) and sent (i.e., *fan-out*) by each SIP server (and derivatively their corresponding ratios) over a given period of time T (say, 5 or 15 minutes); ii) we count

the number of unique users (URIs) seen in the FROM and TO fields of SIP request messages, and compute an aggregate *user activity diversity* (*UAD in short*) metric from the distribution of such data over T. This UAD metric is computed as follows: Let m be the total number of SIP request messages over T, and n is the total number of distinct users seen in the message. For each unique user i, m_i is the number of SIP messages with i in either the FROM or TO field of the messages. Then $p_i = m_i/m$ is the frequency that user i is seen in the SIP messages. The user activity diversity metric, UAD, is then given by

$$UAD := (-\sum_i p_i \log p_i)/\log m \in [0, 1], \quad (1)$$

where the numerator is the entropy of the distribution $\{p_i\}$ while the $\log m$ is its maximum entropy – the ratio of the two is the standardized entropy (or *relative uncertainty*). UAD thus provides a measure of "randomness" of user activities as captured by the distribution $\{p_i\}$: for $n >> 1$, if $p_i \approx 0$, a few users dominate the SIP activities (in other words, they appear in most of the messages), whereas $p_i \approx 1$ implies that $p_i = O(1/m)$ and thus each user only appears in a few number of SIP messages (hence overall the user activities appear random).

b. Server Entity Level Characterization.

Registrar: Using the method field of SIP messages, we separate registrar-related messages (e.g., the REGISTER messages and their responses) and use them to generate statistics and features for registrar behavior profiling. Similar to the server level analysis, we maintain *aggregate statistics* regarding the number (and ratios) of REGISTER and other registrar-related requests and responses received and sent by a registrar. In terms of *user activities*, we maintain the number (and list) of users that are successfully registered, and compute a similar user activity diversity (UAD) metric with respect to the registrar. In addition to these aggregate statistics and features regarding the message types and user activities, we also perform more detailed registration analysis. We examine the response codes in the response messages to maintain statistics about the number of *successful* and *failed* registrations. We also calculate the registration periods of users (i.e., the time lapses between two consecutive REGISTER messages from the same user) and the inter-arrival times of any two consecutive REGISTER request messages (from different users). From the former we compute the (average) registration period of the registrar and from the latter we derive a (fitted) model for the user REGISTER request arrival process. Together, they not only reveal the configuration of the registrar but also the temporal behavior of the registrar.

Call Proxy: By analyzing the SIP messages related to call activities (e.g., SIP messages with the INVITE, BYE methods and their responses), we generate statistics and features for call proxy behavior profiling. Similar as before, we maintain aggregate statistics regarding the numbers and ratios of various call requests (INVITE, BYE, CANCEL, etc.) and their responses received and sent by a registrar. We maintain several user activity diversity (UAD) metrics regarding the aggregate user call activities: UAD_{caller}, UAD_{callee} and $UAD_{\text{caller-callee}}$, which measure the UAD of callers, callees and caller-callee pairs. Each of these metrics is computed using equation (1) with appropriate defined parameters: m is the number of SIP call request messages (SIP INVITE, BYE and CANCEL) requests, and i) for UAD_{caller}, m_i is the number of SIP call request messages with user i in the FROM field, ii) for UAD_{callee}, m_i is the number of SIP call request messages with user i in the TO field, and iii) for $UAD_{\text{caller-callee}}$, we replace m_i by m_{ij} where m_{ij} is the number of SIP call request messages with user i in the FROM field and user j in the TO field.

Furthermore, we perform a more detailed call analysis to maintain various call statistics and features of a call proxy. These include the number of on-going calls, completed calls (calls ended by BYE only), canceled calls (calls ended by CANCEL only), *failed* calls (calls receiving a response with a Request Failure (400-499) response code), and so forth, in a given time period. We also compute statistics (average, standard deviation or distribution) regarding call durations and call request arrival rates.

Cross-Entity Association: we also correlate statistics and features to generate a cross-entity and network-wide view of the SIP traffic. The detailed description is provided in [10] due to space limitation.

c. Individual User Level Characterization.

If needed, we can also maintain statistics and features regarding the individual user activities. For example, from the user call activities we can maintain the (typical or average) number of calls made or received by each user u, and compute the diversity of callees ($UAD^{(u)}_{\text{callee}}$) of the calls made by the user as well as the diversity of callers ($UAD^{(u)}_{\text{caller}}$) of the calls received by the user u. Other statistics such as (average) call durations may also be maintained. Due to space limitation, we do not elaborate them here.

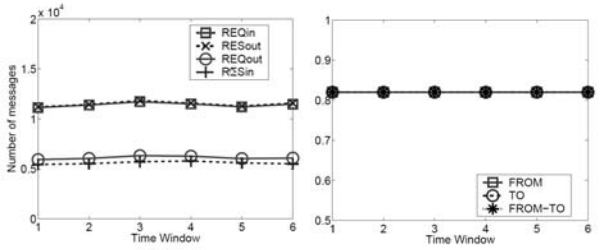
4. CHARACTERISTICS OF SIP TRAFFIC BEHAVIOR

We apply the general profiling methodology presented in the previous section to analyze the SIP traces to illustrate the characteristics of SIP traffic in a real VoIP network, and use them to justify the statistics and features we have taken for profiling SIP traffic behavior. In particular, we show that in normal operational environments SIP traffic behavior tends to be very stable both in terms of various SIP message types, user registration, call, and other related activities. Throughout this section, we primarily use *TRACE II* and server-1 as an example to illustrate the results. More detailed and various results are provided in our technical report [10].

4.1 Overall Server Level Characteristics

For the server level characteristics, we process SIP messages of all method types. Fig. 3(a) shows the numbers of request and response messages received (*REQin*, *RESin*) and sent (*REQout*, *RESout*) over 5-minute time intervals. We see that overall the total numbers of request and response messages received and sent by the SIP server do not vary significantly. In particular, for every one request message received/sent by the SIP server, on the average there is approximately one response message sent/received by it – this is generally expected. There are roughly twice as many request messages received by the SIP server than sent by it. This is primarily due to the REGISTER messages which comprise a large portion of the total request messages received by the SIP server. Unlike many SIP request messages of other methods (e.g., INVITE), a REGISTER request message does not trigger the SIP server to generate another request message except a response message.Fig. 3 (e) shows the user

activity diversity (UAD) metric of the total SIP messages (both received and sent) by the SIP server over 5-minute time intervals for caller, callee, and caller-callee pair separately. We see that the UAD metrics are stable and close to 1 over all 5-minute time intervals,which indicate that there are no individual users who dominate the generation of SIP messages. As seen in the next subsection, this is primarily due to the periodic exchanges of the `REGISTER`, `SUBSCRIBE`, and `NOTIFY request` messages and their `responses` between the SIP server and users.



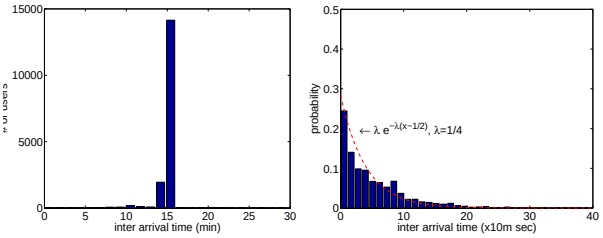
(a) No. of message types (b) User activities diversity

Figure 3: Analysis on server behaviors

Our results show that the aggregate SIP traffic behavior is in general fairly stable and the aggregate statistics/features chosen in our profiling methodology provides a good summary of these stable characteristics. The same observations also hold true for *TRACE III*. *TRACE I*, on the other hand, contains an interesting *anomaly* which is detected by our profiling methodology. We will discuss and dissect this anomaly in more detail in Section 5.

4.2 Registrar Behavior Characteristics

We now focus on the `REGISTER request` messages and their responses(functioning in the role of a registrar), and in particular, examining how `REGISTER` messages are generated by users. We have observed that `REGISTER` messages consist of 60% of the total `request` messages received by the SIP server and the ratio of the number of `REGISTER request` messages vs. their responses is approximately 1. From the examination of users seen in the `FROM` field, we see that the total number of (distinct) users (about 17800) seen in the trace is almost the same to the number of users seen in 15-minute intervals. As we will see, this is primarily due to registration periods and a `REGISTER` arrival process.



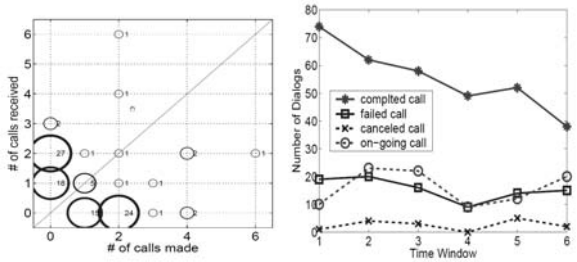
(a) Period of registration updates (b) Requests inter-arrival times

Figure 4: Analysis on registrar behaviors

To further illustrate how `REGISTER` messages are generated, we calculate the time lapses between two consecutive `REGISTER` messages from each user, the distribution of which is shown in Fig. 4(a). The distribution clearly reveals that users generate `REGISTER` messages roughly periodically with a mean of 15 minutes. In Fig. 4(b) we plot the distribution of the *inter-arrival* times between two consecutive `REGISTER` messages (from two different users). The distribution can be well fit into an exponential distribution of the form $p(x) = \lambda e^{-\lambda x}$, where $\lambda = 0.27$. Hence we see that the number of `REGISTER` messages seen by the SIP server (registrar) follows approximately a Poisson process.

4.3 Call Proxy/User Call Behavior Characteristics

We now analyze characteristics of calls and call-related user activities. Comparing with the number of `REGISTER`, we observe that call-related messages consist of a much smaller portion (less than 5%), indicating that while there are a large number of users (or more aptly, SIP phone devices) in the network, only a very small number of the users actually make phone calls in *a specific* period. Fig. 5(a) is a scatter plot showing the number of calls made vs. calls received per user over 5-minute intervals. Again we see that at individual user level, the numbers of calls made and received are generally very small and consistent.



(a) Calls made vs received (b) Call types

Figure 5: Analysis on call proxy activities

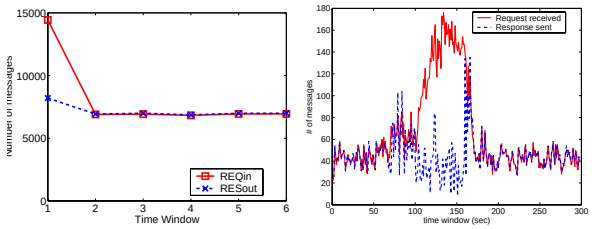
The number of various call types (on-going, completed, failed, and canceled calls) over 5-minute intervals is shown in Fig. 5(b). We see that the number of calls in a typical 5-minute interval is fairly small, and the number of *failed* calls is relatively high due to user mobility or receiver statuses (busy or not available).We observe that call duration typically lasts between 0-3 minutes, while failed and canceled calls tend to last very short. Not surprising, these statistics are similar to traditional telephony, indicating that these call activities are human-generated.

5. APPLICATIONS: PROBLEM DIAGNOSIS

We have applied the general SIP traffic profiling methodology to diagnosing performance problems as well as detecting potential attacks against VoIP service and infrastructure. In particular, we have developed a novel profiling-based feature anomaly detection algorithm for these purposes, and demonstrate its efficacy through testbed experiments. Due to space limitation, we omit the details here, and refer the interested reader to the technical version of the paper [10]. Instead in this section we use a case study to illustrate the usefulness and applicability of our general profiling methodology in helping diagnose performance problems.

As reported earlier, we see that overall the numbers of SIP `REGISTER` *request* and *response* messages and their ratios (over 5-minute intervals) stay fairly stable, and this can be mainly attributed to the fact that users generate `REGISTER`

messages periodically and these messages are generated randomly from the users. These observations hold for almost all 5-minute intervals for both servers in the traces except for one 5-minute interval of server 1 in Trace I, where we have found an interesting "*anomaly*". As evident in Fig. 6(a), the number of `REGISTER` messages received by server 1 in the very first 5-minute interval in this trace segment is significantly larger than in other time intervals, and while the number of the responses sent by the server also increases slightly – in particular, the ratio of the numbers of *requests* vs. *responses* increases drastically.



(a) Message types (b) Total requests and responses

Figure 6: Analysis on anomaly

To figure out what causes this anomaly, we perform a more in-depth analysis of the SIP messages in this anomalous 5-minute interval. Fig. 6(b) shows the number of `REGISTER` messages received vs. the responses generated by in each second of the anomalous 5-minute interval. We see that between around the 100th second to 160th second of this 5-minute interval, the number of `REGISTER` *requests* from users shots up quickly, while the *responses* returned by the server first dips for about 50-60 seconds before it shots up also, catching up with the number of `REGISTER` *requests*, after which everything returns to the norm. We examine the number of `REGISTER` *requests* generated vs. number of *responses* received per user in the 1-minute time period from the 100th second to 160th second. Then, we see that instead of the normal one `REGISTER` *request* and one *response* per user, many users send from 2-7 `REGISTER` requests while receiving one or two *responses*.

Closer investigation reveals that the problem is caused by the SIP server not responding to the user registration requests immediately, triggering users to repeatedly re-transmit their requests within a few seconds until they either gives up or receive a response with either response code `404 Not Found`, `408 Request Timeout`, or (eventually) `200 OK`. Since all these users were eventually able to successfully register with the SIP server, the surge of the `REGISTER` *requests* is unlikely caused by denial-of-service attacks with spoofed or frivolous `REGISTER` messages (as were originally suspected by us). That the SIP server failed to respond to the user registration requests in a timely fashion may be caused by delay or slow response from some remote (user/call) database with which the SIP server was interacting. [3] This performance anomaly can be easily detected using a simple anomaly detection algorithm included in [10].

[3]This problem points to a potential implementation flaw in the SIP client software: when a registration request times out, the client immediately retransmits the request, thereby causing a surge of requests and thus aggravating the problem. A better solution would have been to use an exponential back-off mechanism to handle the retransmission of the registration requests.

6. CONCLUSIONS

In this paper, we have presented a general profiling methodology for characterizing SIP-based VoIP traffic behaviors at multiple levels: the SIP server host, service entity (registrar, call proxy, etc.) and individual user levels. Applying knowledge about application protocol semantics and expected system/user behaviors, an ensemble of statistics and features are selected at each level to capture the essential and stable characteristics of SIP message exchanges, types, volumes, user activities, and so forth. Through our analysis of SIP traffic traces obtained from an operational VoIP service, we show that overall SIP-based VoIP traffic exhibit stable characteristics and behavior that are well captured by the statistics and features selected in our profiling methodology, thereby justifying the selection of these statistics and features. Finally we illustrate how our profiling methodology can be used to help identity anomalies for problem diagnosis and attack detection.

7. REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, P. J. Johnston, A., R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, June 2002.

[2] N. Wosnack. A Vonage VoIP 3-way call CID spoong vulnerability, 2003. http://www.hackcanada.com/canadian/phreaking/voip-vonage-vulnerability.html.

[3] S. McGann and D. C. Sicker. An analysis of security threats and tools in SIP-Based VoIP Systems. In *2nd Workshop on Securing Voice over IP*, June 2005.

[4] D. Geneiatakis, T. Dagiuklas, C. Lambrinoudakis, G. Kambourakis, and S. Gritzalis. Novel Protecting Mechanism for SIP-Based Infrastructure against Malformed Message Attacks: Performance Evaluation Study. In *Proc. of the 5th International Conference on Communication Systems, Networks and Digital Signal Processing (CSNDSP'06)*, July 2006.

[5] D. Geneiatakis, G. Kambourakis, T. Dagiuklas, C. Lambrinoudakis, and S. Gritzalis. SIP message tampering: The SQL code injection attack. In *Proc. IEEE of SoftCOM*, Sept. 2005.

[6] B. Reynolds, D. Ghosal, C.-N. Chuah, and S. F. Wu. Vulnerability analysis and a security architecture for IP telephony. In *IEEE GlobeCom Workshop on VoIP Security: Challenges and Solutions*, Nov. 2004.

[7] B. Reynolds and D. Ghosal. Secure IP telephony using multi-layered protection. In *Proc. of Network and Distributed System Security Symposium(NDSS'03)*, Feb. 2003.

[8] Y.-S. Wu, S. Bagchi, S. Garg, and N. Singh. SCIDIVE: a stateful and cross protocol intrusion detection architecture for Voice-over-IP environments. In *Proc. of the 2004 International Conference on Dependable Systems and Networks (DSN'04)*, pages 433-442, June 2004.

[9] R. Dantu and P. Kolan. Detecting spam in VoIP networks. In *Proc. of USENIX, SRUTI Workshop*, pages 31-37, July 2005.

[10] H. J. Kang, Z.-L. Zhang, S. Ranjan, and A. Nucci. SIP-based VoIP traffic behavior profiling and its applications. Technical report, NARUS, July 2006.