SYSTEM AND METHODS FOR DETECTING UNWANTED VOICE CALLS

Prakash Kolan, B.Tech

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

December 2007

APPROVED:

Ram Dantu, Major Professor
Steve Tate, Committee Member
Parthasarathy Guturu, Committee Member
Krishna Kavi, Chair of the Department of Computer Science and Engineering
Oscar N. Garcia, Dean of the College of Engineering
Sandra L. Terrell, Dean of the Robert B. Toulouse School of Graduate Studies

Kolan, Prakash. System and Methods for Detecting Unwanted Voice Calls. Doctor of Philosophy (Computer Science), December 2007, 228 pp., 36 tables, 73 illustrations, 101 references.

Voice over IP (VoIP) is a key enabling technology for the migration of circuit-switched PSTN architectures to packet-based IP networks. However, this migration is successful only if the present problems in IP networks are addressed before deploying VoIP infrastructure on a large scale. One of the important issues that the present VoIP networks face is the problem of unwanted calls commonly referred to as SPIT (spam over Internet telephony). Mostly, these SPIT calls are from unknown callers who broadcast unwanted calls. There may be unwanted calls from legitimate and known people too. In this case, the unwantedness depends on social proximity of the communicating parties.

For detecting these unwanted calls, I propose a framework that analyzes incoming calls for unwanted behavior. The framework includes a VoIP spam detector (VSD) that analyzes incoming VoIP calls for spam behavior using trust and reputation techniques. The framework also includes a nuisance detector (ND) that proactively infers the nuisance (or reluctance of the end user) to receive incoming calls. This inference is based on past mutual behavior between the calling and the called party (i.e., caller and callee), the callee's presence (mood or state of mind) and tolerance in receiving voice calls from the caller, and the social closeness between the caller and the callee. The VSD and ND learn the behavior of callers over time and estimate the possibility of the call to be unwanted based on predetermined thresholds configured by the callee (or the filter

administrators). These threshold values have to be automatically updated for integrating dynamic behavioral changes of the communicating parties. For updating these threshold values, I propose an automatic calibration mechanism using receiver operating characteristics curves (ROC). The VSD and ND use this mechanism for dynamically updating thresholds for optimizing their accuracy of detection.

In addition to unwanted calls to the callees in a VoIP network, there can be unwanted traffic coming into a VoIP network that attempts to compromise VoIP network devices. Intelligent hackers can create malicious VoIP traffic for disrupting network activities. Hence, there is a need to frequently monitor the risk levels of critical network infrastructure. Towards realizing this objective, I describe a network level risk management mechanism that prioritizes resources in a VoIP network. The prioritization scheme involves an adaptive re-computation model of risk levels using attack graphs and Bayesian inference techniques. All the above techniques collectively account for a domain-level VoIP security solution.

ACKNOWLEDGMENTS

Writing a dissertation is a tedious and time taking process. I have gone through it with the help of many people around me. Thus my sincere appreciation and gratitude goes to all these people.

First and foremost, I express my greatest gratitude to my major professor and dissertation advisor Dr. Ram Dantu for his constant supervision and valuable time. His advice and suggestions have helped tremendously to find quicker and better solutions to my research problems. I thank my dissertation advisors Dr. Steve Tate, and Dr. Parthasarathy Guturu for their helpful suggestions and feedback throughout my Ph.D study. I also thank Peggi for her help in reviewing all my papers and chapters. This was a great time saver.

Thanks to my parents and to my wife Padma for their support and understanding. I owe my sincere gratitude to my friends who have helped me in different ways. Further, I would like to thank all my colleagues who have worked with me at the Network Security Laboratory for their time and assistance.

Finally, I am very grateful, and thank all the people who have directly or indirectly helped me during my Ph.D study. My journey to graduation was certainly not possible without their help.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

VoIP (Voice over IP) also termed as IP telephony has fast emerged as a standard for distant-voice communication using the Internet. Using VoIP, voice and fax can be transported on the same packet data network that is used for transmitting traditional data packets, i.e., IP(Internet Protocol) network. As VoIP uses the existing IP network, it is possible to reduce long distance charges typical with traditional PSTN(Public Switched Telephone Network) networks. In addition, ease of deployment and reduced communication hardware make VoIP a compelling solution for voice communication on the Internet. Further, VoIP provides a flexibility of value-added and personalized services for defining customized solutions. As a result, most of the control which existed in PSTN network's central infrastructure has been transferred to the end devices by deploying the VoIP communication infrastructure.

With the advent of VoIP technology, an increasing number of telecommunication service providers have started to integrate VoIP solutions into their systems and provide VoIP services to their customer base. Equipment manufacturers and end users have greatly benefited from performance advancements, cost reduction, and feature support provided by the VoIP technology. Forecasts for VoIP show a massive increase in revenue (see Figure 1.1). Forecasts for residential-community users are also high. Analysts estimate that by 2010, the number of residential subscribers will reach 140 million [62].

1.1. VoIP

Voice over IP is a technology for transmitting voice packets on the existing IP network between two communicating parties who both have access to the Internet.

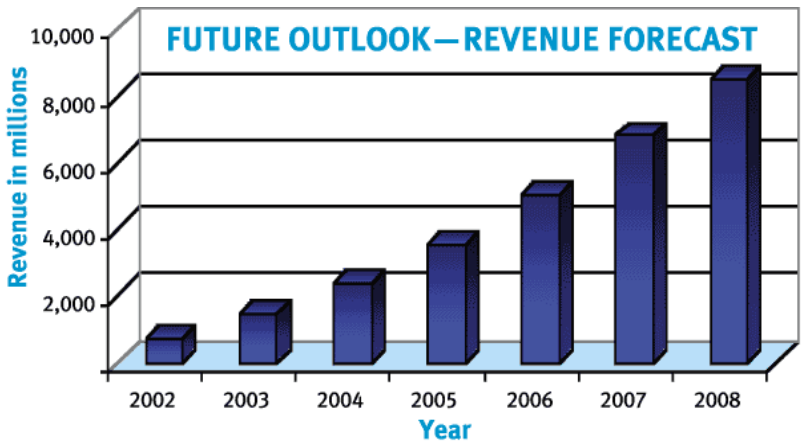

FIGURE 1.1. VoIP revenue forecast[88] (Source: Frost and Sullivan [25])

Unlike PSTN networks, an IP network is packet switched. In PSTN networks, when the calling party calls the called party, there exists a physical circuit connecting the two parties. After the call is established, the parties communicate and the circuit is reserved until the parties finish the communication. However, on an IP network, all communication is carried out using IP packets. When a calling party communicates with a called party, the analog signals are digitized, encoded, and packed into an IP packet at the transmitting end and converted back to analog signals at the receiving end.

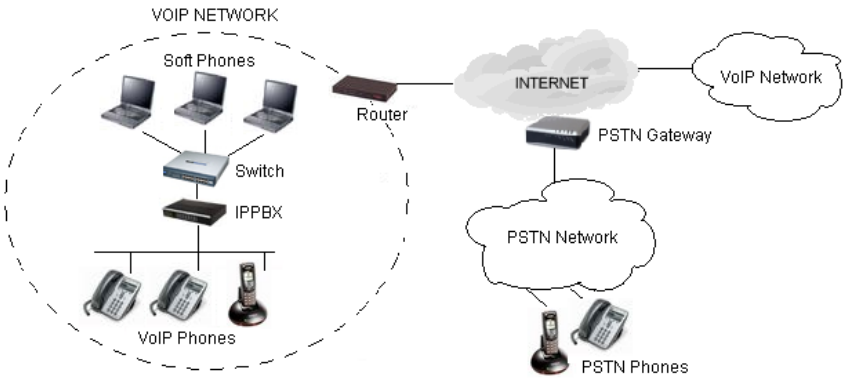

FIGURE 1.2. VoIP network interconnectivity

Figure 1.2 represents a VoIP network's basic interconnectivity. Calls generated from the VoIP end devices (such as IP phones, soft-phones) are transmitted on the IP network to destination IP devices. Gateways separating disparate networks convert the incoming packets' format so that the destination network carries forward the communication. The network components through which the call travels and the communication format are defined by the specific VoIP implementation, based on signaling protocols.

1.1.1. VoIP Protocols

The two major protocols used for voice communication on the IP network are

(1) H.323: An ITU Telecommunication standard for voice communication on packet networks, H.323 defines a suite of protocols such as H.225 (for call signaling), H.235 (security), H.245 (multimedia communication) and H.261, H.263 and H.264 for Video encoding. This suite of protocols establishes standards for communication between the H.323 Terminals and network components such as Gateways, Gatekeepers, and Multipoint Control Units (MCU's).

(2) SIP(Session Initiation Protocol): SIP is an application layer protocol for creating, modifying, and termination sessions between the communication parties. The protocol, which specifies a set of signaling messages for session establishment and termination, is used along with other transport protocols such as RTP(Real-time Transport Protocol), RTCP(Realtime Control Protocol) for enabling voice-communication services between two communication parties. We have discussed the SIP protocol in detail in Chapter 2.

In addition to the above protocols, VoIP networks may use protocols such as MGCP (Media Gateway Control Protocol) and SCCP (Skinny Client Control Protocol) for communication among various VoIP network elements. The MGCP protocol permits controlling VoIP devices such as media gateways from control elements such as Call

agents. Some VoIP infrastructures use SCCP, a Cisco proprietary protocol, for communication between Cisco end-IP phones running skinny clients with a Cisco Call manager.

1.1.2. VoIP Security

Based on their VoIP implementation, network components are interconnected to provide end-to-end VoIP services on both the public Internet and private enterprise Intranets. However, before successful deployment and management of VoIP networks, a number of issues have to be taken care of. Only then can VoIP service providers and recipients reap the advantages of VoIP. One of the most important issues which need immediate attention is securing VoIP networks and their communication elements. As VoIP is still at its inception, service providers are deploying their network infrastructures without adequate security. Such inattention to implementing precise and adequate security measures allows malicious individuals to cause damage to critical communications networks. In addition, inadequate security within the VoIP network's infrastructure elements can lead to exploitation of network infrastructure of other connected networks such as PSTN.

Current PSTN voice services provide excellent voice quality and high reliability (99.999%), carry critical services such as E911, provide federal agencies with the ability to carry out lawful intercept, and operate on well-established and secured PSTN networks. The PSTN network's security, mainly based on a closed network principle, is often physically remote and/or inaccessible to users. End-users do not directly connect to PSTN signaling networks (e.g., SS7). Hence, it is reasonable to assume that they have adequate protection from malicious users. On the contrary, in case of VoIP, attackers can either have a physical (by being near) or logical access (by exploiting and gaining access) to the network elements. These attackers can access most of the network infrastructure on the IP network. Once gaining access, they can

create a major telecommunications disruption.

1.2. Motivation

Unfortunately, researchers have reported very little work on how to defend VoIP against attacks such Denial of Service (Dos), session hijacking and termination, monitoring and eavesdropping, service disruption, toll fraud, identity fraud, spamming, and other attack methods. Also, as a discipline, we do not yet understand the impact of vulnerabilities on a large-scale VoIP network (e.g., several millions of IP phones). Hence, it is imperative that we investigate the vulnerabilities and threats to residential communities that arise with the wide-spread adoption of real-time services like VoIP. All possible threats need to be addressed before we deploy VoIP services on a mass scale. Our current lack of adequate security has the potential of delaying and disrupting next-generation voice communications.

With the use of IP network for VoIP communication services, the problems that exist on the present IP networks hold also for the VoIP networks. One such problem that present IP networks face is spam. A recent study [22] indicates that spam makes up over 40% of the email circulating on the Internet nowadays. Analysts predict daily traffic will rise above 68 billion messages a day by 2007, and more than half63%will be spam. The study estimates the spam costs for US corporations will reach $8.9 billion. With this quantity of spam messages circulating through the Internet each day, problems such as low availability and network congestion would not be a surprise.

In VoIP networks, SPIT(Spam over IP Telephony) refers to unsolicited or unwanted voice calls (the equivalent to spam in email). Initially, it was perceived that such unwanted calls are calls from unknown people, e.g., strangers. But, over time, the meaning of "unwanted" has changed to include subjectively labeled unwanted calls, e.g., telemarketing calls from sales or marketing personnel in legitimate corporations and organizations or calls from people calling about previously opted-in services such

as mortgages, financial opportunities, newsletters, and discussion groups. People interested in those services may view them as wanted calls, but other recipients may view those same calls as unwanted.

So far, we lack documented cases of SPIT because few people use VoIP services. But, with the rapid pace of deployment, SPIT poses a major threat if the VoIP industry fails to adequately investigate and implement prevention mechanisms now. An analysis [71] indicates that spammers find IP-based SPIT roughly three orders of magnitude cheaper to send than traditional circuit-based telemarketer calls. For example, consider a VoIP spammer generating random user-names, IP addresses, or domain names to form a SIP (Session Initiation Protocol) URI (SIP identity) in the form sip:usernameip_address such as in *sip:aliceabcdef.com*(similar to e-mail addresses in e-mail domain). The spammer can then use this randomly generated SIP URI to generate SPIT calls (similar to the way an e-mail spammer generates random addresses for sending spam emails). In addition, spammers can use botnets for relaying VoIP spam calls just as e-mail spammers use botnets to generate and relay spam e-mails. By using botnets, the spammers do not reveal their identity, and, therefore, tracing back to the source would be useless. The spammers can also spoof their identity while generating VoIP calls. Thus, the Caller-ID of the incoming voice call does not reveal the actual individual who has generated the call. It is also not uncommon that spammers confide with each other and exchange phone-numbers among themselves for monetary benefit. Using the available SIP identities and phone numbers, the spammers could likewise deluge the end-users' voice-mail boxes. Certainly, end-users will become frustrated when they know that their voice-mail box are filled with junk voice calls, thus, denying legitimate callers access or space to leave a message.

Most of the present day spam filters use blacklists and whitelists for access control. The set of users from which the calls have to be blocked are encoded in blacklists and the set of users from which the calls have to be forwarded are encoded in whitelists. However, especially for VoIP, this level of quarantining proves

insufficient. Frequently, we receive wanted calls from unknown callers. For example, legitimate callers might call us from new phone numbers. Using lists as our prime filtering method denies us calls from these kinds of callers. Additionally, it is also possible that we stop communicating with some acquaintances for various reasons (e.g., an acquaintance has a new phone number or has not been in communication with us for a while). In this case, if a spammer obtains this phone number and generates VoIP spam calls, a filter would allow the call to reach the end user; the filter, thus, mistakenly forwards a SPIT call to the VoIP recipient. Few spam filters employ challenge response methodology for filtering spam calls. When a spam filter receives a suspicious call, the spam filter can challenge the calls's source (e.g., by requesting to perform a simple mathematical operation). Upon response from the source, the spam filter quarantines the call or forwards it, depending on its legitimacy. Using challenge response SPIT filters, it would be feasible to identify calls from automated machines that fail when responding back to the filter. However, challenge response filters are inefficient with calls generated by humans (e.g., calls from a creditor of an unknown credit card company).

Besides SPIT(Spam over Internet Telephony) calls from unknown callers being unwanted, there can be unwanted calls from known people, too. Thus, call recipients may find it extremely useful if their VoIP system can detect an unwanted call well in advance of reaching the callee and stop it at a perimeter controller (such as proxy server or gateway). Preventing unwanted calls into the network limits the the misuse of network resources which can be better used for legitimate call-processing functions. An inference as to whether an incoming call is wanted can be estimated from past calling patterns between the caller and the callee. We can reasonably assume that the more frequent the callee's communication with the caller has been in the past, higher are the chances that the callee wants to receive the incoming call. On the other hand, if the callee has rejected calls from the caller in the past, then the chances that the callee wants to receive calls from that caller gets significantly reduced.

To date, we have found no reported work on estimating a called party's eagerness or reluctance when receiving an incoming voice call. However, we believe, pro-actively computing a callee's eagerness helps when limiting unwanted calls. We argue that every incoming voice call is associated with some amount of nuisance which depends on several factors

(1) The closeness between the caller and the callee: A caller's closeness represents the proximity of the caller in the callee's social network. The callee's social network constitutes all the people with whom the callee previously had VoIP communication. People with whom the callee had more communication are closer; people who had fewer communications with the callee are considered to be farther away from the callee if we were to map the callee's social network. Thus, closeness depends on the frequency and duration of communication with the callee.

(2) Presence of the callee: The callee's presence is the callee's mood or state of mind at the time of receiving a call. Presence depends upon the the callee's current spatial and temporal contexts. Depending on presence, a callee perceives certain calls to be wanted and certain others to be unwanted. For example, a callee may consider calls from family members to have a comparatively high nuisance value when the callee is in the office as compared to when the callee is at home.

Integrating nuisance computation along with inferring SPIT behavior can be used to limit unwanted calls. Filters can be configured with anti-SPIT and nuisance computation mechanisms for allowing legitimate and necessary calls to reach a callee while deleting or diverting unwanted or illegitimate calls. These filters can learn the characteristic behavior of specific callers over a period of time and effectively quarantine unwanted calls. However, apart from learning the callers' behavior, the filters must also adapt to a callee's changing preferences. For example, a callee who is looking to buy a house may initially welcome calls from mortgage companies or realtors.

But, once the callee has completed a purchase, mortgage and realtor calls become unwanted. In this context of changing end user preferences, it is always a challenging task for adaptive filters to re-evaluate the callers' behavior based on the new preferences and re-learn the wantedness of any specific caller. This learning should be fast enough so that the filter's performance is unaffected.

In addition to being unwanted, SPIT calls can threaten the VoIP network's components. Intelligent attackers can create malicious VoIP traffic for compromising a VoIP network's infrastructure. For example, an attacker can deluge a domain-level VoIP network device such as an H.323 or SIP server with call set-up requests (Dos attack). By exploiting the network components' vulnerabilities, the attacker may attain administrative privileges on them. Using these privileges, the hacker can manipulate device configurations and overwrite routing tables. Thus, the hacker can perform attacks such as session termination, session hijacking, toll fraud[2][87], Dos[46][79][52][34], phishing[28], and eavesdropping[34]. Using these attacks, the attacker can bring down the network and deny service to legitimate users, or the attacker can use the exploitation for financial benefit. In this context, it can be extremely useful to frequently perform risk-assessment procedures to infer the risk levels of critical network infrastructure. Based on the assessed risk, respective network components can be patched to reduce or even avoid risk from attackers. Frequently assessing the risk of network devices and fixing their vulnerabilities would render a lower opportunity to a attacker to compromise network infrastructure.

1.3. Contributions

Here, in this section, I outline the contributions of this dissertation in the field of SPIT research and VoIP/ Network security as a whole. This dissertation makes five major contributions

1. *A behavior learning mechanism for filtering spam calls in VoIP networks*: We developed a filtering mechanism using social notions of trust and reputation for filtering SPIT calls in VoIP networks. In developing this mechanism, we have defined

(1) A model for computing and updating trust based on past transactions. The model uses a Bayesian learning mechanism that takes into account a callee's preferences and past mutual communication between the callers and the callee when updating knowledge about the callers' SPIT behavior.

(2) A model for inferring and updating reputation based on recommendations. For inferring reputation, we developed a Bayesian Network inference model that takes into account the caller's history of the caller with users in different VoIP domains.

(3) A trust and reputation formalism based on human intuitive behavior for detecting SPIT based on the called party's direct and indirect relationships with the calling party.

(4) A model for inferring domain-level trust information based on the calling party's history with domain-level users.

2. *Pro-active eagerness estimation based on previous communication*: We developed a model for proactively inferring the callee's eagerness when receiving voice calls from different callers. For this inference, we have defined

(1) A model for inferring social closeness of callers with the callee on the voice communication network. This inference is based on previous communication patterns between the callers and the callee such as frequency and talk-time. Using the inferred social closeness, we group callers into four groups of a) Socially Close callers b) Socially near callers c) Opt-ins and d) Opt-outs.

(2) A model for computing incoming calls' nuisance. A measure of nuisance for a given caller depicts the callee's eagerness to receive incoming voice calls from that caller. For this nuisance computation, we have developed a model based on communication patterns between a caller and the callee such as frequency, talk-time, periodicity, and reciprocity along with other factors such as social closeness and presence (based on present context). Quarantining calls from unauthorized callers, and unnecessary calls based on present context of callee

helps in realizing the objective of making the phone to ring only when the callee would like to receive it.

3. *Automatic calibration mechanism for optimizing filter performance*: We have developed an automatic parameter calibration mechanism for optimizing a filter's performance. The automatic calibration mechanism can be used with any adaptive filter to enable the filter to quickly converge to optimal values for configured parameters. This automatic calibration is extremely useful with filters in dynamic environments (where configurations change frequently) particularly when the filters involve multiple dynamically changing parameters.

4. *Formalism based on voice communication*: A formalism that represents possible communication patterns between different callers and callees. The formalism can be integrated with SPIT filtering and the nuisance-estimation mechanism to provide evidential information regarding the type of communication between the callers and the callees. The formalism includes

(1) Enumeration of algebraic rules based on real-life communication patters for a given callee with respect to callers who belong to socially close, socially near, opt-in, and opt-out groups.

(2) Enumeration of operations using communication patterns between multiple callers and callees (e.g., using matrices). Matrices based on communication patterns such as frequency and talk-time can be used for defining meaningful calling constructs such as connectivity, reputation, and periodicity.

(3) Enumeration of applications that can be solved using derived operations. This enumeration provides insight into applications that can be solved by examining the communication patterns between the callers and the callees.

5. *Adaptive risk computation*: We have developed a model for adaptively inferring the risk to network devices in a given VoIP network. Deriving network devices' risk levels proves to be useful when prioritizing vulnerabilities and exploits during the patch-management process. In addition, the adaptive risk-computation mechanism

can be used to dynamically infer risk levels after implementing suitable security policies. Therefore, the adaptive-computation mechanism forms an effective basis for a network administrator to appropriately change network configuration. Thus, suitable vulnerability analysis and risk-management strategies can be formulated to efficiently curtail the risk from different types of attackers (script kiddies, hackers, criminals and insiders).

To sum up, the work presented in this dissertation extends our understanding of how to limit unwanted calls on a VoIP network. The solution can be deployed at any level, i.e., on the end VoIP phones, at an enterprise network's perimeter gateway, or at the higher level ISP domain. At any level, the solution helps in quarantining unwanted calls and prevents congestion at that level. Further, the work helps in minimizing the threats due to unwanted calls on the VoIP infrastructure.

1.4. Dissertation Road-map

The dissertation discusses a framework for limiting unwanted calls coming into a VoIP network. The content in this dissertation is organized into nine chapters.

Chapter 1 (Introduction): In chapter 1, we give an overview about voice over IP (VoIP) and VoIP security in general. In addition, we discuss the problem of spam in VoIP networks. Further, we give a brief overview about the necessity of risk management for minimizing network attacks on VoIP infrastructure.

Chapter 2 (Session Initiation protocol): In this chapter, we give a brief overview about Session Initiation Protocol. In this overview, we discuss about network elements in a SIP based network and a basic SIP call-flow. In addition, we also discuss SIP messages i.e. SIP requests and responses.

Chapter 3 (Unwanted voice call detection framework): In this chapter, we present a general overview of the detection framework for minimizing unwanted calls on a VoIP network. Different stages of the framework are discussed in brief and a general overview of statistical models incorporated into different stages is presented.

Chapter 4 (VoIP Spam Detection): In this chapter, we present a VoIP spam detection framework for filtering VoIP spam calls. The framework includes models for computing and updating trust, inferring reputation based on recommendations using a network topology graph, and a domain level trust information integration mechanism. Using all the models, the spam detection framework learns the behavior of the callers over a period of time and quarantines next incoming calls from them.

Chapter 5 (Pro-active Nuisance Estimation): In this chapter, we discuss a computation mechanism for nuisance of incoming calls. For this computation, we describe a model for inferring social closeness between a caller and a callee based on past communication patterns. Further, we discuss mathematical models for inferring other behavioral patterns such as periodicity, reciprocity, and presence for integrating them into nuisance computation mechanism.

Chapter 6 (Automatic Calibration using Receiver Operating Characteristics Curves): In this chapter, we discuss an automatic calibration mechanism for dynamically updating filter parameters (e.g., threshold values). Using this automatic calibration mechanism, the filter quickly converges to optimum parameter values after each transaction (e.g. call) and thereby delivers optimum overall performance.

Chapter 7 (Call Algebra): In this chapter, we discuss a formalism that enumerates algebraic rules that represent possible communication patterns between callers and callees, operations based on communication patterns such as frequency and talk-time for deriving calling constructs such as periodicity, reciprocity, and reputation. In addition, we discuss how the algebraic rules and operations can be used for solving a multitude of problems pertaining to real-time voice communication.

Chapter 8 (Network Level Risk Management): In this chapter, we discuss a network level risk management mechanism that can be used for adaptively inferring the risk of network resources of a given VoIP network. In addition, we discuss how attack behavior relates to risk (i.e. how much risk do different types of attackers such as hackers and explorer pose to network infrastructure) and network penetration (i.e.

the penetration an attacker can achieve).

Chapter 9 (Conclusion): In this chapter, we summarize the work presented in previous chapters. In addition, we discuss how the proposed framework can be used along with other filtering techniques to result in a comprehensive solution for limiting unwanted calls in a VoIP network.

CHAPTER 2

SESSION INITIATION PROTOCOL

2.1. SIP Protocol

SIP, an RFC(Request for Comments) standard (RFC 3261) from the IETF (Internet Engineering Task Force), provides signaling constructs for real-time IP communication. An increasing number of Internet Telephony Service Providers have started to provide value-added, SIP-based services to their subscribers. As a result, SIP is becoming a de-facto standard for telephony on the IP network. In addition, the SIP protocol's similarity to other text-based protocols such as HTTP(HyperText Transfer Protocol) and SMTP(Simple Mail Transport Protocol) has considerably eased its transition into traditional IP applications.

As an application layer protocol for creating, modifying, and terminating sessions between participants in communication, SIP protocol specifies a standard for digital communication on the IP network between the SIP end points. Sessions can be any of voice, video, chat, interactive games, and virtual reality [1]. The protocol provides capabilities for registration, location identification, and availability determination of SIP end points. In addition to providing a means for end-to-end multimedia communication, presence, and instant messaging, SIP provides an application layer framework for multi-user support such as multimedia conferencing. Using SIP, multiple users can establish multimedia sessions and exchange multimedia content.

When facilitating communication, SIP can work independently of the transport medium such as TCP(Transmission Control Protocol), UDP(User Datagram Protocol), or ATM(Asynchronous Transfer Mode) protocols. SIP's flexibility makes it transport independent, i.e., independent of underlying transport protocols. This independency from the transport medium when coupled with it's ability to provide capabilities

to the SIP end points to negotiate the media communication parameters makes the SIP protocol a highly scalable, an extensible, and a generic signaling framework that provides a wide variety of services. However, SIP alone cannot provide a complete framework for multimedia transfer across different participants. SIP works in conjunction with other protocols to provide communication services. The basic protocols with which SIP has to function for providing the IP communication services are

- DNS: SIP uses the DNS(Domain Name System) protocol's services to translate domain names to network IP addresses. The name resolution is needed for the signaling and the media information to be transferred from one communicating party to the other.
- DHCP(Dynamic Host Configuration Protocol): The DHCP protocol provides a standard for dynamic host configuration. The communicating parties' end points should have network identity (IP address) to communicate with other network participants. To obtain this identity, the end points communicate with a local DHCP server to dynamically obtain an IP address.
- SDP (Session Description Protocol): SIP provides a signaling framework for establishing sessions between the communication participants. However, for describing the underlying media, SIP uses standards provided by the SDP protocol. The SIP end points use the SDP standard for agreeing on the media information such as codec formats, timing, and transport information.
- RTP: The SIP end points use RTP protocol for real-time content transfer. RTP provides a standard for real-time data transfer over the IP network.

In addition to the above basic protocols, SIP also works with other protocols such as SOAP(Simple Object Access Protocol), HTTP, XML(Extensible Markup Language), and WSDL(Web Service Definition Language) to provide a wide variety of services [2]. Of course, each of the above protocols has a specific role to aid in IP communication.

2.2. SIP Elements

SIP performs similarly to SS7(Signaling System 7) in the present PSTN networks. However, while PSTN's SS7 uses a highly centralized architecture with dumb terminals (normal telephone sets) and intelligent central systems such as switches, the SIP world consists of intelligent entities towards the end. With SIP, most functionality is implemented at the end-user terminal with intermediate servers being used for call routing and user identification. Here is a list of SIP elements generally observed in a SIP network.

- SIP End Points: The SIP end points (IP Phones) are end devices that the communicating parties use for making and receiving calls. The communicating parties' terminals exchange signaling information to establish, modify, and terminate calls. In addition, each end device samples the analog media, digitizes it, and encodes it with a codec format (negotiated during call establishment process) to transfer the media to other parties. The end devices (or terminals) in the SIP world are intelligent with the ability to provide an interface for many services such as call waiting, three-way calling, and call transfer. These services, otherwise provided and managed by the service providers in the PSTN world, are integrated into end SIP devices for regular use. Integration of these services reduces the complexity involved in the central servers and provides a mechanism to customize and offer SIP-users on-demand access.
- SIP Proxy Server: The proxy server is used predominantly for call routing. When an end user (caller) would like to invite another user (callee) to communicate, the caller requests that the domain proxy server establish a session with the intended callee. The Proxy server after receiving the call request checks to determine whether the callee is in the same domain. If the callee is, the proxy server routes the call to the callee. Alternatively, if the callee is not within the proxy's domain, the caller's domain proxy routes

the call to the appropriate proxy of the callee's domain. The destination domain proxy would, thus, redirect the call to the callee's appropriate end device and the call is established. Similarly to the above call-establishment process, the proxy server may route signaling communication such as session modification and termination between the communicating parties.

- SIP Registrar Server: A SIP registrar is a domain-level server for receiving and acknowledging register requests of domain's end points. The registrar servers are generally collocated with the domain's proxy and redirect servers. The domain's SIP end points periodically (and whenever they boot-up) exchange registration requests with the registrar to let the server know that they are ready to receive incoming calls. The domain proxy server acknowledges calls only from users registered with the registrar.
- SIP Location Server: As a registered user can log in from multiple end points, the SIP registrar uses an abstract service known as location service managed by a Location server to provide their address bindings. When the domain proxy receives a call request for a callee in its domain, the proxy queries the location service to determine the callee's location.
- SIP Redirect Server: The redirect servers generally provide the next hop information to the SIP end points. To provide this information, the SIP end points can request an alternate set of addresses from the redirect server. The redirect servers are also sometimes used for reducing the proxy server's load by providing routing services.
- IP-PBX(IP based Private Branch Exchange): The IP-PBX facilitates switching VoIP calls in a domain and provides a multiplexing/de-multiplexing solution by enabling domain users to use a limited set of outgoing phone lines onto either an external IP or a traditional PSTN network. Traditional PBX's require separate networks for voice and data transfer. However, by using

IP-PBX's, a converged network providing voice and data services can be implemented.

- Soft-switches: Soft-switches facilitate routing services between VoIP networks. These soft-switches generally include signaling and media gateways for voice call transfer.
- SBC(Session Border Controllers): The SBC's, intermediate devices either or both on the signaling and media paths, traditionally have been used for providing a control function on the incoming VoIP traffic (i.e., call admission control). SBC's are VoIP-session aware and can filter unwanted calls. SBC's, although primarily used as a firewall (for limiting unwanted traffic) and a NAT(Network Address Translation) gateway when the VoIP network employs topology hiding, may also (depending upon the vendor), provide fault tolerance and Quality of Service (QoS) for media traffic.

2.3. SIP Messages

The SIP network elements described in Section 2.2 communicate among themselves using a set of SIP messages which constitute the requests the client makes to a server and the responses the server returns back to the client. Any SIP element can adopt a client or server role depending on the direction of the requests from the message-generating elements to the message receiving elements.

2.3.1. SIP Requests

The SIP network's elements exchange SIP requests for requesting a service from the server. These services can be session initiation, session modification, or session termination. In general, the following are the SIP messages that are exchanged when a client requests for a service

- INVITE: The SIP elements exchange INVITE messages when inviting other parties into a call. A client sends an Invite message to its domain's proxy server requesting to establish a call. The proxy server forwards the message

directly to the destination proxy (if it knows the destination proxy) or indirectly (by forwarding to the next hop proxy). The destination proxy in turn forwards the Invite message to the destination end point for call establishment.

- BYE: SIP elements exchange BYE messages to specify their intention to terminate a call. During a communication, any client or the server can generate such a message.
- ACK: An exchanged ACK (acknowledgment) message acknowledges receipt of a message such as INVITE or BYE. When the client or the server receives an ACK message, it is assumed that the other party has received its earlier message.
- CANCEL: The CANCEL messages are exchanged for canceling any pending searches. However, this exchange does not terminate an established call.
- REGISTER: A SIP end point registers with its domain SIP registrar by sending REGISTER messages. The end points periodically (and after boot-up) send REGISTER messages to indicate that they are up, running, and ready to receive calls.
- OPTIONS: A client can use the Options message to query server's capabilities. Depending upon the server's response, the client can initiate actions that modify session parameters.
- PRACK: A client may use the PRACK (Provisional Acknowledgement) message to request the server to keep re-transmitting the provisional responses until they are received by the client.
- REFER: The REFER message arises when client asks the recipient to issue a call transfer by initiating a call request.
- MESSAGE: The MESSAGE method is used when transmitting Instant messages using SIP.

- INFO: The INFO message is used for carrying session information. However, using INFO message does not change the session state.
- UPDATE: The client can use an UPDATE message to update session parameters. However, updating session parameters does not impact the established dialog.
- SUBSCRIBE: The SUBSCRIBE message can be used to request notification when an event happens.
- NOTIFY: The NOTIFY message is used for notifying an event's happening.

While the above messages request a service, SIP protocol also defines a set of responses for these requests.

2.3.2. SIP Responses

Unlike the SIP request messages, however, SIP identifies responses by status codes, a 3-digit number with the first digit representing the response's class [3]. Classes of responses defined by the SIP protocol are as shown in Table 2.1:

TABLE 2.1. SIP response codes

Class	Description	Used
1xx	Informative and Provisional Responses	To represent session status information
2xx	Success Responses	When the request is received and accepted
3xx	Redirection Responses	To request needs further action
4xx	Client Failure Responses	To represent client error messages
5xx	Server Failure Response	To represent Server error messages
6xx	Global Failure Responses	To represent errors that are nowhere acceptable

Many responses (thus response codes) defined for SIP fall into the above classes, and the SIP elements interact using the SIP request and response messages outlined

above. With the discussed elements and connectivity, a typical SIP network is as shown in Figure 2.1.

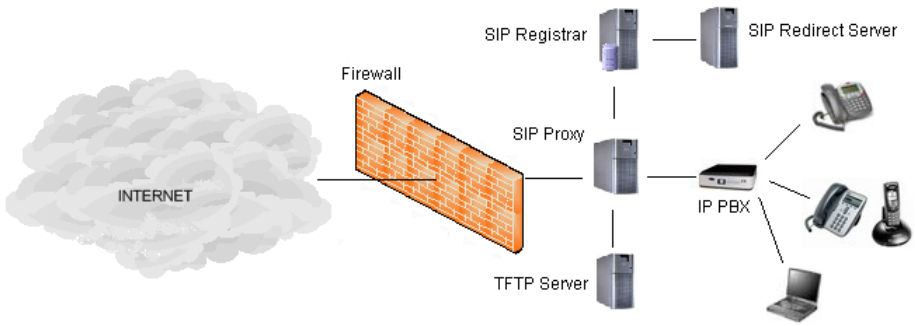

FIGURE 2.1. A basic SIP network

SIP defines a number of network components that are used for user-identification and call routing as shown in Figure 2.1. A caller generates a call request to a callee through the caller's domain SIP proxy server. The proxy server checks with the domain SIP registrar if the caller is a user registered to use the SIP services. After the caller is identified as a registered user, the proxy server uses the SIP location server's services to appropriately locate the destination domain and its proxy server. If the callee is in the same domain as the caller, the proxy server forwards the call directly to the callee's end device. However, if the callee is not in the same domain, the proxy server communicates with the redirect server to determine the next-hop proxy for sending the request. The next-hop proxy again checks whether the callee exists within its domain and, if not, forwards the call to the next hop proxy. This continues until the call request reaches the domain proxy in which the callee resides. Once the destination domain proxy determines the callee's end device, it forwards the call to that device. The callee acknowledges the call request, and the connection between caller and callee is established. A basic SIP call flow diagram is shown in Figure 2.2. Figure 2.2 presents the series of signaling messages for establishing and terminating a call. The caller's terminal generates an INVITE request to its domain proxy. The

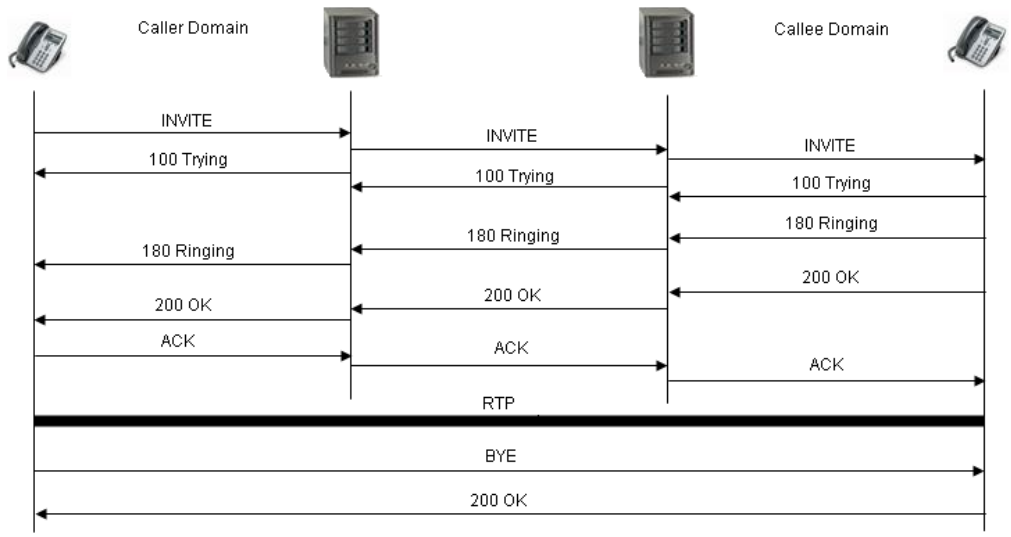

FIGURE 2.2. A basic SIP call flow diagram

proxy forwards the INVITE request to the destination SIP proxy which forwards the request to the callee. The callee acknowledges the call invitation using a 200 OK message. However, before the callee acknowledges the call request, the SIP devices exchange a set of status messages such as 100 Trying and 180 Ringing. After the 200 OK message reaches the caller's SIP terminal, the terminal sends back an ACK message for acknowledging the receipt of 200 OK message. When the ACK message reaches the callee's terminal, the call is established and the two parties exchange the media (RTP traffic). When the two parties finish the communication and intend to terminate the call, one of the two parties (usually the party that hangs up first) generates a BYE message from their terminal to the other party's terminal. Upon receiving the BYE message, the terminal sends back a 200 OK message (acknowledgment) for the call termination request, and thus the call is terminated.

CHAPTER 3

UNWANTED CALL DETECTION FRAMEWORK

A solution for filtering unwanted calls must include models that help in filtering calls which create nuisance to the callees, or attempt to exploit network level infrastructure. While junk VoIP calls from unauthorized callers and unnecessary calls from legitimate callers create nuisance to callees, malicious VoIP traffic can compromise network devices. For limiting these calls, we propose a framework that analyzes incoming VoIP calls for unwanted behavior.

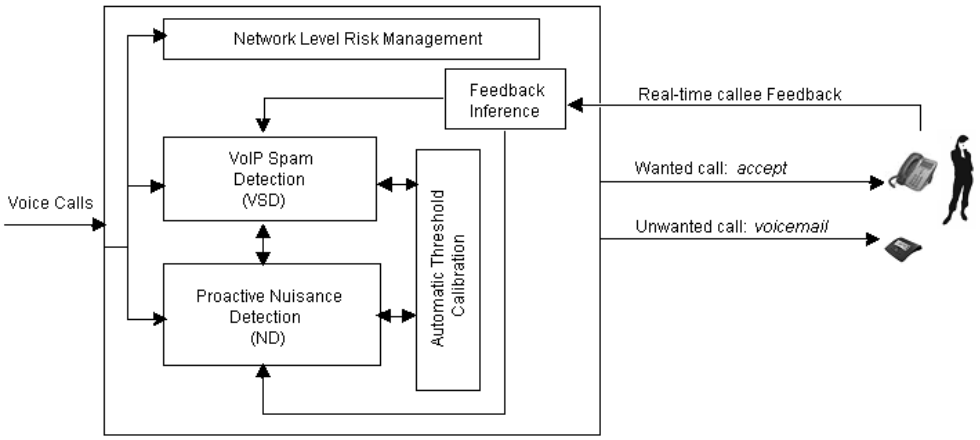

FIGURE 3.1. Unwanted voice call detection framework

The unwanted call detection framework given in Figure 3.1 primarily involves four behavior learning models for filtering unwanted calls that attempt to either exploit network level infrastructure or create nuisance to the callees. The models are explained as follows:

- VoIP Spam Detection: We propose a VoIP spam detector (VSD) that analyzes incoming VoIP calls for spam behavior. Spam detection involves a multi-stage adaptive model for detecting VoIP spam calls. The model includes static checking of wanted and unwanted calling entities encoded in white- and blacklists, social notions of trust and reputation based on history of calling parties, and domain level knowledge integration by learning from others experience (Chapter 4).
- Proactive Nuisance Detection: We propose a Nuisance Detector (ND) that screens incoming call traffic for filtering unwanted calls to the callee. ND proactively infers the nuisance of incoming calls based on caller's previous behavior, the callee's tolerance (inferred using past calls from the caller), and presence (mood or state of mind), and the social closeness between the caller and callee. In addition to these factors, ND considers other behavioral patterns of communication between the caller and the callee such as periodicity and reciprocity for inferring the nuisance of voice calls. Based on the inferred nuisance, the ND makes a decision to forward the call to the callee (Chapter 5)
- Automatic Threshold Calibration: We propose an automatic calibration mechanism for re-computing optimal threshold value of filters after each call. The calibration mechanism uses Receiver Operating Characteristics curves (ROC) for computing optimum threshold value. The computed optimum threshold is then used as the filter threshold for the next incoming call. Automatic calibration mechanism is even more helpful when the filter is configured with multiple thresholds where manually updating them for optimizing accuracy is a complex process. Dynamically updating the thresholds of the filter helps in fast learning of the caller and the callee's mutual behavior, and converging to the optimum performance (Chapter 6).

- Network Level Risk Management: We propose a network level risk management mechanism of prioritizing end host vulnerabilities for patch management and penetration testing processes. The mechanism involves adaptive computation of risk level of VoIP network components (such as SIP proxy, end VoIP phones). The computed risk level is then used for prioritizing vulnerabilities that need immediate patching. This prioritization scheme is particularly necessary in context of everyday emerging exploits and patches released by VoIP infrastructure vendors (Chapter 8).

All the above models are primarily driven by the feedback from the called parties. For an incoming call, VSD computes the probability of the call to be spam and then compares it to a predetermined threshold to make a decision to forward the call or quarantine it (thus establishing the authenticity of the caller). After the authenticity of the caller is established, ND infers the nuisance associated with the call using the nuisance detection model. ND then compares the inferred nuisance with a predetermined threshold to make a decision to forward the call to the callee's phone or to his voicemail. When the callee receives the call (either to the phone i.e., when the phone rings, or to his voicemail box), he can give a feedback about the validity of the call. This feedback is then used for updating the history of the caller with respect to the callee. This updated history is then used for inferring spam and nuisance behavior next time a call comes in from that caller. The feedback from the callee can also be used for dynamically updating the threshold (or thresholds) of VSD and ND using the automatic calibration mechanism. In addition, a network component can give feedback about abnormal behavior (e.g., that it is exploited). In this case, the network level risk management mechanism can be used for updating the risk levels of other network components, and a decision can be taken for patching network components that need immediate attention.

CHAPTER 4

VOIP SPAM DETECTION

4.1. Introduction

The possibility of VoIP network replacing the PSTN network depends on enhancing the existing IP network to carry voice traffic. With the usage of IP network to carry voice traffic, existing problems on the IP network holds for the VoIP network too. One of the major issues that current IP networks face is controlling spam - the unsolicited (bulk) e-mail. Spam control has been perceived to be one of the most important problems of research with the traditional e-mail systems. Many techniques have been designed to avoid e-mail spam. However, such techniques often have limited application to avoid voice spam because of real time considerations. For example, a spam call from a telemarketer if not filtered before it reaches the end user makes the phone to ring instantly. In this context, the end user has to answer the call in real-time and it would be very frustrating to know that the call is of least importance. On the contrary, a spam e-mail from a telemarketer sits in the inbox until the enduser checks his e-mails. In addition, content filtering is not useful in VoIP spam analysis as media flows in after the two parties (i.e., calling party and the called party) have agreed upon to start the communication and would be too late to filter the call. This inability to filter VoIP calls poses a serious challenge of detecting spam in real time with the available signaling messages.

To realize the objective of receiving only genuine VoIP calls from any person anywhere in the world, we must replace static junk-call filtering mechanisms with

adaptive learning systems. These systems, apart from learning spam behavior, should incorporate human behavioral models of how the call recipients (called parties) determine whether to answer call. For example, whenever a phone rings, depending on our state of mind, we determine whether the call is from a trusted party. If we do not know the calling party, we guess the calling party's reputation. After picking up the call, we ask the calling party some questions and move forward only when satisfied with the calling party's response. Therefore, a mechanism for filtering VoIP spam calls has to take into account the presence of the called party (e.g., state of mind, location, time), the rate of incoming calls from the calling party, trust and reputation between calling party and the called party. We describe a multi-state adaptive spam filtering mechanism that computes the trust (estimated using Bayesian theory) and reputation of the calling party (using reputation graphs based on called party's social network). We have integrated these factors within our adaptive learning system to facilitate deciding whether to accept/reject a call or forward it to voice mail.

4.2. Background

There exists lot of literature in the field of trust and reputation computation ([63],[45], [50], [6], [42], [101], [89], [90], [96], [97], [55], [100], [73], [37]). Rahman et al [63] presents a distributed trust model for computing trust of entities in online transactions. The trust model adopts a recommendation protocol where a source entity requests its trusted entities to give recommendations about a target entity for a given trust category. When the source entity receives all the recommendations, it computes a trust value to the target entity based on the received recommendations. Lei et al [45] presents a distributed trust model organized as a Trust Delegation Tree (TDT) in e-commerce applications. The paper presents a trust management model for computing different trust levels such as direct trust based on history, indirect trust based on trusted intermediaries' recommendations, and trust authorization levels using delegation certification chains. Marsh [50] describes trust formalism for computing and

updating trust between two agents. The trust mechanism involves the computation of types of trust such as basic trust (based on accumulated experiences), general trust independent of context, and situational trust that takes into account a specific situation. Cahill et al [6] presents a trust model for secure collaborations among pervasive entities. The paper presents a trust model in which a source principal makes a decision whether to interact with a target principal based on current trust value he holds with the target principal for that particular action, and the risk in communicating with the target principal. Krukow et al [42] discusses the importance of probabilistic models in logic reasoning. The paper presents a probabilistic trust model for computing the predictive probability of a principal behavior i.e., the probability with which the principal's next interaction will have a specific outcome. Zimmerman [101] describes a web of trust model where users can exchange PGP keys among themselves to trust each other. The exchanged keys are signed by each user, and at any time, the trustworthiness of users that have signed the keys can be established. However, the users are required to create their own security and trust policies. The web of trust model is not scalable and is not used with very large systems such as the Internet. Wang et al [89] and Wang et al [90] present a Bayesian trust model for inferring the trust of agents participating in an online transaction. The proposed trust mechanism involves deriving Bayesian Networks for trust inference and using the past transaction history to update the available trust information. In addition, the trust model incorporates a mechanism for evaluating the recommendations of other agents and updating its trust values towards them. Yu et al [96] and Yu et al [97] describe a trust framework for classifying agents to be trustworthy based on quality of recent transactions. The agents provide thresholds for differentiating trustworthiness of agents into trustworthy, non-trustworthy and unclear classification groups. The framework assumes that an agent belongs to one of these groups when the probability of service between that group and latest group is greater than a give threshold. In addition, the framework gives preference to direct interaction information before taking into account indirect

information from witnesses.

Mui et al [55] presents a computational model for inferring trust and reputation of a given source. The model infers trust as a dyadic quantity between trustor and trustee and is computed based on the reputation data of the trustee. The paper further defines the reputation score as a quantity embedded in the social network of the trustor and past transactions with the trustee. Zacharia et al [100] and Zzcharia et al [99] present two reputation mechanisms "Sporas" and "Histos" for inferring the reputation of agents in an online community. Sporas is a reputation computation system where the agents rate each other after the completion of the transaction and the reputation values are updated. This update is less when the agent's reputation is high. Histos is a reputation computation mechanism that performs a recursive personalized rating inference of agents that have communicated with the target agent. This recursive inference is achieved by deriving a weighted graph with nodes as the agents and the links connecting them as personalized ratings given by the parent node to the child node of the link. Sabater et al [73] presents a survey on computational trust and reputation models that are of specific application in the field of distributed Artificial Intelligence. The paper describes different classification aspects based on which the models can be classified and provides a review on sample models in the area of trust and reputation computation research. Josang et al [37] presents a comprehensive survey of trust and reputation models in Internet transactions. The survey describes the trust and reputation semantics existing in the literature of trust and reputation inference. The paper also describes the existing problems and solutions for aggregation trust and reputation metrics. All these trust and reputation inference techniques have been used to solve problems in different problem domains such as e-commerce, peer-to-peer networks, and spam filtering.

Spam filtering for the present day e-mail infrastructure has been well addressed in current literature ([74],[85], [75], [10], [67], [27], [80], [11], [91], [24]). Designers of

spam filters have used a wide variety of filtering mechanisms such as text classification, rule-based scoring systems, Bayesian filtering, pattern recognition, and identity verification. Sahami et al [74] describes a Bayesian trust model for filtering spam e-mails. The paper proposes that incorporating domain specific features in addition to identifying various textual phrases, and probabilistically inferring the spam behavior of the constructed message vector leads to a more accurate spam analysis. Soonthornphisaj et al [85] presents a spam filtering technique that constructs the centroid vector of the incoming e-mail and checks for its similarity between the centroid vector of spam class and legitimate class e-mails. Sakkis et al [75] suggests probabilistic inference for calculating the mutual information index (MI) for feature selection. Using this, a vector of attributes having the highest MI scores is constructed for spam identification. The memory-based algorithms then attempt to classify messages by finding similar previously received messages and using them for classification. Cohen [10] recommends spam filtering based on a set of rules for identifying the message body content. Features of the message are identified and scored to compute the total spam score of the e-mail spam message and the messages having a score more than a given threshold is identified as a spam e-mail. Large quantities of spam and legitimate messages are used to determine the appropriate scores for each of the rules in the rule-based scoring systems. Rigoutsos et al [67] suggests pattern discovery scheme for identifying unsolicited e-mails by training the system with a large number of spam messages. The system matches the e-mail message with the available patterns; more the patterns are matched more is likelihood that the message is spam.

Golbeck et al [27] presents an algorithm for inferring reputation relationships. This is achieved by constructing social network of people connected with each other. Every user in the social network is attributed with a reputation score. For a given message from the email sender, the receiver infers the weighted average of its neighbor's reputation ratings to the email sender. The neighbors in turn infer the weighted average of their neighbors' reputation rating for the e-mail sender and this continues

until the e-mail sender is reached. Seigneur et al [80] discusses the establishment of identities of e-mail senders using trustworthy e-mail addresses. The identities of the parties are established by exchanging hashes of previously exchanged e-mails. In turn, a Challenge Response system is discussed for challenging the party to establish its identity. The paper also discusses a Trust/Risk Security framework for inferring whether the incoming e-mail is spam or not. The framework proposes to use a Bayesian spam filter for trust inference. In addition, the framework uses a static model for choosing a finite set of recommenders for making a recommendation. Identity verification mechanisms help in establishing that the caller is the person who he claims to be. However, Identity based mechanisms are not a complete solution for filtering spam especially in cases of dynamically changing behavior and preferences of people involved in communication. Damiani et al [11] suggests a P2P framework for detecting e-mail spam messages. A P2P network consisting of user-tier nodes, mailers (mail servers), and super peers exchange communication among themselves for tagging and updating incoming e-mail messages as spam. This is achieved by constructing message digests of incoming messages and checking for similarity with other spam message digests. Wattson [91] presents a spam filtering mechanism based on sender identity verification and disposable e-mail addresses. The mechanism proposes a multi stage architecture consisting of black- and white-listing procedures, sender identity verification, and challenge response systems. The cumulative inference of all the stages dictates whether the incoming e-mail is spam or not.

Foukia et al [24] presents a collaborative framework for spam control in distributed administration domains. The framework involves mail servers collaborating with each other to exchange spam control information. The proposed framework involves spam control processes at both the incoming and outgoing mail servers. The servers that participate in this information exchange are rewarded and traffic restrictions are imposed on those e-mail servers that do not participate. All the above proposed techniques attempt to classify incoming email as spam based either on static

rule-checking, learning spam behavior from the email's contents, or through identity verification. However, these techniques cannot be directly used for filtering real-time junk voice. Unlike e-mail spam messages, VoIP calls are real-time and have to be filtered in real-time. In e-mail, people do not exhibit dynamic changes in behavior. Even if the people exhibit dynamic changes in behavior, the e-mails from them do not pose a nuisance to the end user. However, in case of VoIP, real-time voice calls from people exhibiting dynamic changes in behavior creates a lot of nuisance to the called party.

While spam in e-mail is effectively addressed, little work exists on analyzing spam in VoIP networks. The SIP spam draft (Rosenberg et al [71]) discusses spam in VoIP networks. Rebahi et al [65] present a spam filtering solution using reputation relationships. The mechanism proposes to build a social network of people that can issue recommendations regarding a given source. Macintosh et al [48] present a statistical detection technique by analyzing the incoming traffic distribution of the VoIP calls. An abnormal deviation from the normal call distribution is considered to be SPIT traffic. Shin et al [83] presents a spam filtering approach where the calls are analyzed based on their incoming rate. If the incoming rate is greater than predetermined short-term and long-term thresholds, the call is blocked and branded as spam. We believe that a spam solution for analyzing incoming voice calls is not only confined to limiting the rate of calls. The solution should also consider the social authenticity (trust and reputation), social connectivity, and the inherent need in accepting the incoming call based on called party's presence.

4.3. VoIP Spam Detector

The network diagram shown in Figure 4.1 is partitioned into network segments such as the call-receiving domain hereafter referred to as VSD_{domain} (circled part in Figure 4.1) i.e., the domain for which the VSD acts as a spam detector, and the call-generating domains i.e., the domains from which calls would be generated to an end

user in the VSD_{domain}. This VSD_{domain} consists of all the VoIP users whose calls are analyzed by the VSD for spam behavior. VSD_{domain} can be scaled to different network sizes. For example, the VSD_{domain} can be an enterprise having a Class B network such as the domain www.unt.edu (University of North Texas with an IP address range 129.120.xxx.xxx), or a scaled down domain such as the computer science department at UNT (with an IP address range 129.120.60.xxx or 129.120.61.xxx). At any level, VSD analyzes and filters the call for the users inside the network (or domain). Calls are generated from an end user outside or inside VSD_{domain} through the VSD. Each IP phone in VSD_{domain} includes a SPAM button to allow the called party (callee) to give feedback to the VSD.

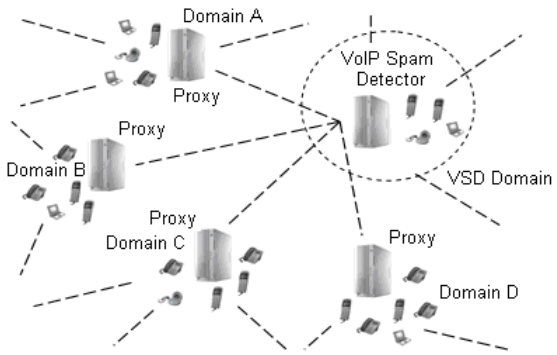

FIGURE 4.1. Voice Spam Detector for computing the spam probability of incoming calls. The VSD can be deployed either in a VoIP proxy server (e.g., at the enterprise perimeter) or in an end VoIP phone. In any case, VSD analyzes the incoming and outgoing calls based on end users' preferences

On receiving a call, VSD analyzes the spam level of the incoming call (the associated spam probability) using the VoIP spam-detection model presented in Section 4.4. The VSD then compares the call's computed spam level with a predetermined threshold value to decide whether to block or forward the call to the callee. The threshold value (permissible limit) is chosen by giving preference to legitimate calls over spam

calls, i.e. the number of spam calls that can be forwarded so as to minimize false positives (legitimate calls being blocked). The main aim of any spam-filtering technique should be to minimize false negatives (spam calls let in as legitimate) while keeping the false positives to zero.

The call processing depends on the callee's reaction to the incoming calls. The VSDdomain users are equipped with spam-recognition capabilities. The callee receives the call, experiences it, and gives feedback to the VSD about the nature of call (whether it is a spam call or a legitimate call). This feedback is as simple as pressing a spam button either during the call or just after termination. This feedback resembles the way e-mail users give feedback about spam e-mail by clicking a SPAM button on their web browser. The VSD learns about the spam behavior of call participants (such as user, host and domain) based on callee's feedback. If the callee responds with a feedback that the current call is spam, the VSD updates the calling party's (caller's) history for future spam analyses. Future calls from the caller will have a high spam probability and a higher chance of being stopped at VSD. On the other hand, if the callee responds with a positive experience (non-spam), the caller's history is updated to depict more legitimacy for next incoming calls from the caller.

4.4. VoIP Spam Detection Framework

VoIP spam detection cannot be achieved using a single detection method. The detection needs to occur at several stages of call processing to achieve a high degree of accuracy.

4.4.1. Architecture

The architecture for spam detection process should take into account the callee's preferences of wanted and unwanted calls, his presence of mind, the trust and reputation he has for the caller. The basic architecture for spam detection is shown in Figure 4.2. Each stage in the architecture diagram represents a technique based on

which the call would be quarantined by employing a specific set of mechanisms and user feedback. Each stage computes the possibility that the call is spam and the collective inference of all stages provides a measure of the call's probability of being spam. Based on this inference, a decision is made to forward or quarantine the call.

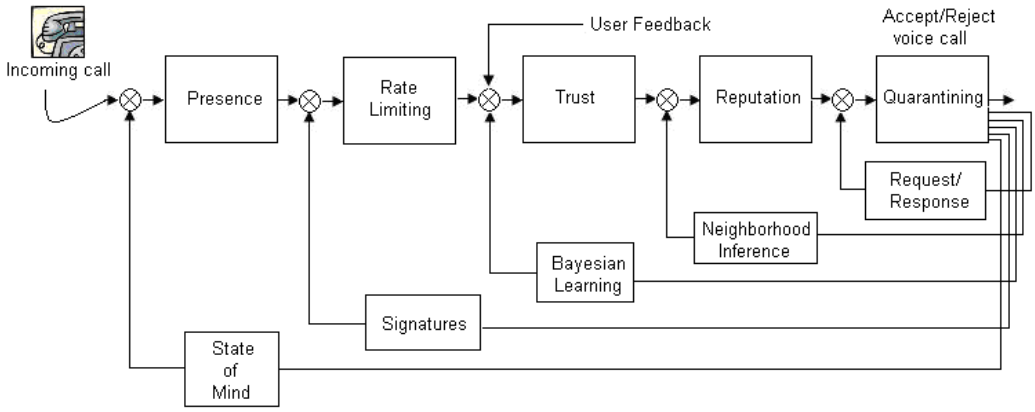

FIGURE 4.2. Functional Elements of VSD. The diagram describes a logical view of a multi-loop feedback control process. Each stage employs a specific set of mechanisms for computing the incoming call's spam level. Each stage computes a spam level based on its specific features. A collective inference from all of the stages represents an incoming call's spam level

4.4.2. Functional Elements in Voice Spam Detection

As depicted in Figure 4.2, a number of techniques are used to achieve a composite measure of a call's spam level. This section provides a description of these techniques. *Presence*: Whenever we receive a voice call, whether we answer it or not depends on our state of mind. This state of mind can change depending on several factors such as location, the time, and our mood. To provide the VSD with parameters for determining their state-of-mind, end users can configure their phones with modes such as

do-not-disturb-me, follow-me, and 911-emergency-modes depending on their preferences and needs. One such example of integrating state-of-mind with filtering process is to synchronize the system with the end user's calendar. The filtering process that takes place during this stage is based on static and dynamic rules configured by the end user.

Rate Limiting: Based on known traffic patterns, signatures can be used to detect the rate of incoming calls. For example, velocity and acceleration values (first and second order derivative) of the number of arriving calls from a given user/host/domain can be used as a detection mechanism. When the velocity/acceleration reaches a certain threshold, the drop rate can be updated through feedback control. As expected, the earlier we detect a change in the incoming pattern based on signatures, the earlier there will be a reduction in the spread of spam. Once spamming is identified, the filter can use Proportional Integral Control (PID), a feedback control, to reduce the velocity of spreading. This method of detection is useful not only in deterring spamming attacks but also in averting DOS attacks. Some preliminary investigation on the effectiveness of this detection method is presented in Dantu et al [16].

Black and White Lists: Most of the present day spam filters conduct static checking of a set of signatures (a set of valid and invalid entities such as a user, host and domain). The spam filter uses these signatures to allow or block calls. Whitelist represents the set of entities from which the callee is always ready to receive calls. Similarly, blacklist represents the set of entities from which the callee prefers not to receive calls. Depending upon the callee's specifications, the calls with the specified entities will be allowed or denied calling. Such lists are customized i.e. each callee has the flexibility of specifying a set of whitelist and blacklist entities. The legitimate and spam entities in each of the callee's whitelist and blacklists will differ from other callees' lists, and thus, will not influence the call forwarding or blocking of other callees i.e. each end user is guaranteed of forwarded or denied calls based on a customized list.

The VSD constructs black- and whitelists using callee's feedback. If the callee's

feedback for a forwarded call indicates spam, the VSD adds the call entities to the blacklist. Future calls from any entity on the blacklist are blocked at the VSD. On the other hand, if the callee responds with a legitimate-call feedback, the call entities are added to the callee's whitelist, and calls from them are forwarded to the callee.

Trust: Learning the caller's spam and legitimate behavior over time allows us to make many intelligent decisions regarding the call. This process of observing the caller's behavior constitutes the trust level the caller has built with the callee. Trust as such represents an abstract modeling of the caller's and the callee's past mutual behavior. This trust information can be used to classify an incoming call as spam or legitimate.

When the VSD needs to compute the trust of an incoming SIP voice call, it checks for previous trust information associated with the call participating entities such as the call source (calling user, calling host, call-generating domain), participating proxies in routing with the help of fields such as from, to, record route, and via. The call's trust level is then computed using Bayesian inference techniques (see Section 4.4.3.1). If the call is forwarded to the callee, VSD updates the history of the caller to appropriately reflect the callee's feedback. At times, it is possible that due to unavailability of previous transactions, VSD cannot compute a trust value. In this case, we infer the caller's reputation from callee's neighbors.

Social Networks and Reputation: Social networks can be used to represent user relationships that can be derived along the network paths. These social networks can be used to infer the associated relations between the elements in the network. These relationships are transitive and transparent. If Alice is related to Bob and Bob is related to Charles, then with a reasonable degree of confidence, Charles can derive the trust information of Alice from Bob. With respect to a VoIP service user, the user's social network represents the associated and trusted neighbors from whom the user is willing to receive calls. While the trust is computed based on history, we derive reputation from trusted peers. The reputation of the call source can be inferred based on the previous experience of those trusted peers (Section 4.4.3.2) with that

call source.

It is highly imperative for spam filters to be integrated with human behavioral aspects and principles to mimic the way humans' answer calls from wanted people. Applying these social notions of trust and reputation helps in identifying the social community and the relative closeness among the members of the community. Such information can then be used to improve the accuracy of identifying unwanted calls.

4.4.3. Trust and Reputation in Voice Calls

Formal models of trust have been proposed in security and in social sciences ([64], [32], [58]). These papers, however, do not address the social notions of trust and reputation for solving real-time problems such as spam. The trust and reputation formalism presented here precisely addresses this problem. In particular, we attempt to address the following

(1) Formalism structured on human intuitive behavior for detecting spam based on trust (direct) and reputation (indirect) relationships with the caller.
(2) A quantitative model based on the formalism that computes the number of spam calls required to move a caller from a whitelist to a blacklist and vice versa.

For defining the quantitative model, we use Bayesian inference techniques to compute and update the trust and reputation based on intuitive considerations. Variants of Bayesian estimation methodologies have been used in solving a multitude of problems relating to probabilistic reasoning and statistics. We believe that these Bayesian analysis and inference techniques would aid in automated and adaptive learning of spam behavior. This formalism for voice calls is integrated into VSD (Section 4.3). In Appendix A, we present the terminology adopted for the formalism. In Section 4.4.3.1, we present the trust formalism that describes a model for computing and updating trust based on callee's feedback. Section 4.4.3.2 presents a reputation formalism for inferring and updating reputation based on callee's feedback. Section 4.4.3.3 explains

the integration of the above models of trust and reputation for computing the spam probability of the call.

4.4.3.1. *Trust.* Trust has been traditionally used in solving the problem of authentication in application areas such as ad-hoc and peer-to-peer systems. Social notions of trust can be used in inferring the spam behavior of voice calls. To begin, we define trust in context of analyzing voice calls by modifying the definition given in Wang et al [90] as *a callee's belief in caller's capabilities, honesty and reliability based on his/her own direct experiences.* In context of voice calls, trust refers to caller's capability, honesty, and reliability in making legitimate calls to the callee. This trust of the incoming call is based on the trust of the individual participants of the call and the callee's experiences towards those call participants.

Property 1: Trust level of a voice call depends on the call participants.
The trust T of the incoming SIP voice call depends on the trust of individual call participants. A call participant can be a user, a host, a domain, or an intermediate proxy.

Property 2: Trust is derived from the caller's past behavior.
Trust for a call participant is accrued over a period of time based on its past behavior. For each call participant i, we denote a call set $C_i = \{N_{i,s}, N_{i,v}\}$ the spaminess and legitimateness of participant i. The spaminess $N_{i,s}$ represents the total number of past spam calls and legitimateness $N_{i,v}$ represents the total number of past legitimate calls from the call participant. The higher a call participant's spaminess, the higher are the chances that the call having this call participant will be filtered. Similarly, the higher the legitimateness, the higher the chances that VSD will forward the call having this call participant to the callee. The initial values of $N_{i,s}$, and $N_{i,v}$ of a call participant i are defined by $N_{i,s}$, $N_{i,v} = 1$ when VSD has no history for the call participant. For incoming calls from the caller, VSD increments $N_{i,s}$ for every spam

call and $N_{i,v}$ for every legitimate call with respect to the callee. This spaminess and legitimateness of individual call participants helps in computing the overall trust of the incoming call.

The trust of the incoming call is inferred from $D = f(C,\ N_S,\ N_V)$ where N_S and N_V represents the total number of spam and legitimate calls processed by VSD. C represents the set of call sets of all participants i.e., $C = \{C_1,\ C_2\ \ldots\ C_n\}$ where n is the number of call participants. D is defined as the distrust of the call. The higher the distrust of the call, the lower is the trust level associated with it. $D \in [0\ 1]$, i.e., the distrust of the call lies in the range [0 1]. The distrust D of the incoming call is dependent on the spaminess and legitimateness of all the call participants and is computed using Bayesian Analysis. This is represented by

$D = f(C,\ N_S,\ N_V)$

i.e. $D = f(\{N_{1,s},\ N_{1,v}\},\{N_{2,s},\ N_{2,v}\},\{N_{3,s},\ N_{3,v}\}.....\{N_{n,s},\ N_{n,v}\},\ N_S,\ N_V\ )$

where the function *"f"* is defined as (detailed probabilistic model is explained in Appendix B)

$$(1) \quad D = \frac{(\frac{\sum_{i=1}^{n} N_{i,s}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})})(\prod_{i=1}^{n}\frac{N_{i,s}}{N_{i,s}+N_{i,v}})}{(\frac{\sum_{i=1}^{n} N_{i,s}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})})(\prod_{i=1}^{n}\frac{N_{i,s}}{N_{i,s}+N_{i,v}}) + (\frac{\sum_{i=1}^{n} N_{i,v}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})})(\prod_{i=1}^{n}\frac{N_{i,v}}{N_{i,s}+N_{i,v}})}$$

Higher the value of D, higher is the chance that the call is going to be filtered. The simple Bayesian equation shown above helps us in computing the distrust of the incoming call. Though we have carried out a treatment of simple Bayesian analysis for processing calls, we believe that the results are valid even for different variants of Bayesian analysis and techniques.

Definition 1: Trust level T is a direct measure of distrust and is equal to 1 - D. The distrust D given in Equation (1) helps in computing the value of trust for the incoming call. This computation is a direct measure of distrust D and is equal to the value 1- D.

Axiom 1: Callers can be grouped based on their calling patterns.

In real world, we remember people such as our friends, family members, neighbors, and unwanted callers who can be grouped into white, grey and black lists. The membership of these lists varies depending upon our mood, past experience, current needs and distrust. In addition, we have different levels of tolerance for people belonging to different groups. For example we can assign some ranges of distrust levels to these lists as follows.

$$D = \begin{cases} 0 - 0.01 & Whitelist \\ 0.01 - 0.99 & Greylist \\ 0.99 - 1.0 & Blacklist \end{cases}$$

A caller in a callee's blacklist can be a user who has had a spam behavior for a long time such that the caller's distrust was as high as 0.99. However, a reasonable number of legitimate calls from the caller can decrease this distrust value. If the distrust falls below a threshold of 0.01, then the caller can be added to the callee's whitelist. In addition, the above scale can also be used when quarantining incoming calls. For example, with a distrust value more than 0.99, the incoming call can be directly blocked, and for distrust less than 0.01, the call can be directly forwarded. The computed distrust (Property 2) after integrating with reputation inference (Section 4.4.3.2) is compared with a predetermined threshold configured by the callee and a decision can be made whether to forward the call to the callee or to filter it. If the call is filtered, then it can be sent to the voicemail box or completely blocked depending upon the callee's customized options. Alternatively, if the call is forwarded to the callee, then the callee can answer the call and give feedback to the VSD. The callee

responding with a spam or legitimate call feedback constitutes the explicit feedback. However, another type of feedback can be inferred from the callee's calling behavior. It is the implicit feedback available to the VSD when the callee makes outgoing calls (here, the role of the callee changes to a caller). In this case, the VSD updates the history of the called parties with respect to him. There is a marked difference in the way trust is updated based on implicit and explicit feedback and is explained in the following property.

Property 3: Trust can be derived from incoming as well as out-going calls. Every wanted or legitimate incoming call translates to an additive increase in trust whereas an outgoing call results in an exponential increase.

For a given callee, trust is a function of both incoming and outgoing calls. It is normal that the trust we have towards people we call is greater than the trust we attribute to people who call us. The trust we bestow on people who receive calls from us increases exponentially with every outgoing call whereas the trust attributed to people who call us increases additively. We incorporate this behavior into VSD as follows.

For a positive feedback from the callee for the present call, the distrust D for the next call from the caller would be updated by

$D = f(\{N_{1,s},\ N_{1,v}+1\},\{N_{2,s},\ N_{2,v}+1\},\{N_{3,s},\ N_{3,v}+1\}.....\{N_{n,s},\ N_{n,v}+1\},\ N_S,\ N_V+1\ )$

And if the call is spam, as specified by the callee, distrust D is updated by

$D = f(\{N_{1,s}\ +1,\ N_{1,v}\},\{N_{2,s}+1,\ N_{2,v}\},\{N_{3,s}+1,\ N_{3,v}\}.....\{N_{n,s}+1,\ N_{n,v}\},\ N_S+1,\ N_V\ )$

But, for *an outgoing call*, trust is increased exponentially and is represented by

$D=f(\{N_{1,s},\ N_{1,v}\lfloor e^{k_1}\rfloor\},\{N_{2,s},\ N_{2,v}\lfloor e^{k_2}\rfloor\},\{N_{3,s},\ N_{3,v}\lfloor e^{k_3}\rfloor\}.....\{N_{n,s},\ N_{n,v}\lfloor e^{k_n}\rfloor\},\ N_S,\ N_V\ )$

We believe that $k_i \geq 0$ and is proportional to individual trust T_i of the call participant i for $i = 1..n$, i.e., the amount of exponential increase is in the order of trust of the respective call participant.

Therefore, $k_i \propto T_i$ for $i = 1..n$.

For defining the trust T_i of individual call participant i, we compute the distrust D_i of the call participant. Trust is then inferred directly from the distrust value and can be safely assumed to be equal to 1 - D_i. The distrust D_i for call participant i (variant of appendix B for one random variable) is given by

$$D_i = \frac{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}})}{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}}) + (\frac{N_{i,v}}{N_V})(\frac{N_{i,v}}{N_{i,s}+N_{i,v}})} \tag{2}$$

Computing individual distrust for each call participant helps to identify the amount of spam behavior associated with that call participant. In addition, this computation assists in reducing false alarms i.e. the total number of false positives and false negatives (for example, using the composite distrust computation D along with distrusts of individual call participants for filtering spam calls). Therefore, for $i = 1..n$, the lower the value of D_i from Equation (2), the higher is the value of k_i and, therefore, the higher the increase in trust level and vice versa. Trust can be updated by computing the distrust (as shown in Equation (2)) for every incoming call passing through the VSD.

Based on the above constructs, we can derive a quantitative model using Bayesian estimation that computes the number of spam or legitimate calls required for moving the caller among the lists defined in Axiom 1. To achieve this, the current behavior of a call participant, i.e., the spaminess and legitimateness of a call participant based on its past behavior must be computed. We, as humans, do this in our daily life as well. When receiving a voice call, we check the caller-id of the incoming call and intuitively estimate the likelihood of the call being spam based on the caller's trustworthiness and past behavior.

Lemma 1: A participant's spaminess can be inferred from its past calls and current distrust.

Let D_i be the distrust of a call participant "i" for its past total calls $N_{i,B}$ where $N_{i,B} = N_{i,s} + N_{i,v}$ where $N_{i,s}$ and $N_{i,v}$ represent the spaminess and legitimateness associated with the call participant; then,

$N_{i,s} = h1(N_{i,B},\, D_i,\, N_S,\, N_V)$

$N_{i,v} = h2(N_{i,B},\, D_i,\, N_S,\, N_V)$

where the functions *h1* and *h2* are derived by solving the two equations

$$N_{i,B} = N_{i,s} + N_{i,v} \tag{3}$$

and $D_i = \frac{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}})}{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}})+(\frac{N_{i,v}}{N_V})(\frac{N_{i,v}}{N_{i,s}+N_{i,v}})}$ from Equation (2)

From above we have

$$D_i = \frac{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}})}{(\frac{N_{i,s}}{N_S})(\frac{N_{i,s}}{N_{i,s}+N_{i,v}})+(\frac{N_{i,v}}{N_V})(\frac{N_{i,v}}{N_{i,s}+N_{i,v}})}$$

$$\implies D_i = \frac{N_V(N_{i,s})^2}{N_V(N_{i,s})^2+N_S(N_{i,v})^2}$$

$$\implies (N_{i,v})^2 = \frac{N_V(N_{i,s})^2(1-D_i)}{N_S D_i}$$

$$\implies \frac{(N_{i,v})^2}{(N_{i,s})^2} = \frac{N_V(1-D_i)}{N_S D_i}$$

$$\implies \frac{(N_{i,v})^2}{(N_{i,B}-N_{i,v})^2} = \frac{N_V(1-D_i)}{N_S D_i} \text{ from (3)}$$

$$\implies \frac{(N_{i,B}-N_{i,v})^2}{N_{i,v})^2} = \frac{N_S D_i}{N_V(1-D_i)}$$

$$\implies \frac{N_{i,B}}{N_{i,v}} = 1 + \sqrt{\frac{N_S D_i}{N_V(1-D_i)}}$$

$$\implies N_{i,v} = \frac{N_{i,B}}{1 + \sqrt{\frac{N_S D_i}{N_V(1-D_i)}}} \tag{4}$$

From Equation (3) $N_{i,s} = N_{i,B} - N_{i,v}$

$$\implies N_{i,s} = N_{i,B} - \frac{N_{i,B}}{1+\sqrt{\frac{N_S D_i}{N_V(1-D_i)}}}$$

$$\implies N_{i,s} = \frac{N_{i,B}}{1 + \sqrt{\frac{N_V(1-D_i)}{N_S D_i}}} \tag{5}$$

Therefore, given distrust D_i and past number of calls $N_{i,B}$ for a call participant i, we can find $N_{i,s}$ and $N_{i,v}$, i.e. the call participant's spaminess and legitimateness. Deriving $N_{i,s}$ and $N_{i,v}$ for a given distrust and total past calls helps us to compute

the number of spam or legitimate calls required to move a caller from one list (e.g., whitelist) to another (e.g., blacklist) defined in Axiom 1 as shown in lemma 2.

Lemma 2: The number of calls (N_{CF}) needed to identify spam depends on the caller's distrust and the callee's tolerance.

In reality, it is highly likely that people will change their behavior such as legitimate callers generating unwanted calls or unsolicited callers making legitimate calls. In this context, we derived a quantitative model that computes the number of spam calls required to categorize a caller to a blacklist. This is more necessary in VoIP than in e-mail. In the case of e-mail, it would not be a serious nuisance to the callee if the spam filter doesn't stop spam emails (false negatives). But, spam calls become a greater nuisance in case of VoIP because the callee must answer the call in real time. For deriving N_{CF}, assume that $N_S^{'}$ and $N_V^{'}$ represent the total number of spam and legitimate calls processed by the VSD. In this lemma, consider that three call participantscalling user, calling host, and call-generating domainare used to compute an incoming call's distrust value. Assume

Spaminess of user $(N_{1,s}) = N_{1,s}^{'}$, and legitimateness of the user $(N_{1,v}) = N_{1,v}^{'}$

Similarly, spaminess of host $(N_{2,s}\ ) = N_{2,s}^{'}$, and legitimateness of host $(N_{2,v}\ ) = N_{2,v}^{'}$

Spaminess of domain $(N_{3,s}\ ) = N_{3,s}^{'}$, and legitimateness of domain $(N_{3,v}\ ) = N_{3,v}^{'}$

Therefore, the current distrust (D_C) based on the spaminess and legitimateness of the three call participants of the incoming call derived using Equation (1) is given by,

$$D_C = \frac{(\frac{\sum_{i=1}^{3} N_{i,s}^{'}}{\sum_{i=1}^{3}(N_{i,s}^{'} + N_{i,v}^{'})})(\prod_{i=1}^{3} \frac{N_{i,s}^{'}}{N_{i,s}^{'} + N_{i,v}^{'}})}{(\frac{\sum_{i=1}^{3} N_{i,s}^{'}}{\sum_{i=1}^{3}(N_{i,s}^{'} + N_{i,v}^{'})})(\prod_{i=1}^{3} \frac{N_{i,s}^{'}}{N_{i,s}^{'} + N_{i,v}^{'}}) + (\frac{\sum_{i=1}^{3} N_{i,v}^{'}}{\sum_{i=1}^{3}(N_{i,s}^{'} + N_{i,v}^{'})})(\prod_{i=1}^{3} \frac{N_{i,v}^{'}}{N_{i,s}^{'} + N_{i,v}^{'}})}$$

$$\implies D_C = \frac{(\sum_{i=1}^{3} N'_{i,s})(\prod_{i=1}^{3} N'_{i,s})}{(\sum_{i=1}^{3} N'_{i,s})(\prod_{i=1}^{3} N'_{i,s}) + (\sum_{i=1}^{3} N'_{i,v})(\prod_{i=1}^{3} N'_{i,v})}$$

$$\implies D_C = \frac{N'_{1,s}N'_{2,s}N'_{3,s}(N'_{1,s}+N'_{2,s}+N'_{3,s})}{(N'_{1,s}N'_{2,s}N'_{3,s}(N'_{1,s}+N'_{2,s}+N'_{3,s}))+(N'_{1,v}N'_{2,v}N'_{3,v}(N'_{1,v}+N'_{2,v}+N'_{3,v}))}$$

$$\implies \frac{D_C}{1-D_C} = \frac{N'_{1,s}N'_{2,s}N'_{3,s}(N'_{1,s}+N'_{2,s}+N'_{3,s})}{N'_{1,v}N'_{2,v}N'_{3,v}(N'_{1,v}+N'_{2,v}+N'_{3,v})} \tag{6}$$

Now, assume that there were N_{CF} number of spam calls from the call participants (i.e., the calling user, calling host, and call-generating domain) after which the calls are filtered. Therefore, because of linearly updating the history of each call participant based on feedback from the callee, the history of the three call participants is given by

Spaminess of user ($N_{1,s}$) = $N'_{1,s} + N_{CF}$ and legitimateness of user ($N_{1,v}$) = $N'_{1,v}$

Spaminess of host ($N_{2,s}$) = $N'_{2,s} + N_{CF}$ and legitimateness of host ($N_{2,v}$) = $N'_{2,v}$

Spaminess of domain ($N_{3,s}$) = $N'_{3,s} + N_{CF}$ and legitimateness of domain ($N_{3,v}$) = $N'_{3,v}$

Total spam calls processed by VSD = $N_S + N_{CF}$ and total number of legitimate calls processed by VSD = $N_V + N_{CF}$.

Therefore, the final distrust level (D_F) after N_{CF} number of spam calls from the three call participants, is given by

$$D_F = \frac{(\frac{\sum_{i=1}^{3}(N'_{i,s}+N_{CF})}{\sum_{i=1}^{3}(N'_{i,s}+N_{CF}+N'_{i,v})})(\prod_{i=1}^{3}\frac{(N'_{i,s}+N_{CF})}{N'_{i,s}+N_{CF}+N'_{i,v}})}{(\frac{\sum_{i=1}^{3}(N'_{i,s}+N_{CF})}{\sum_{i=1}^{3}(N'_{i,s}+N_{CF}+N'_{i,v})})(\prod_{i=1}^{3}\frac{(N'_{i,s}+N_{CF})}{N'_{i,s}+N_{CF}+N'_{i,v}}) + (\frac{\sum_{i=1}^{3}N'_{i,v}}{\sum_{i=1}^{3}(N'_{i,s}+N_{CF}+N'_{i,v})})(\prod_{i=1}^{3}\frac{N'_{i,v}}{N'_{i,s}+N_{CF}+N'_{i,v}})}$$

$$\implies D_F = \frac{(N'_{1,s}+N_{CF})(N'_{2,s}+N_{CF})(N'_{3,s}+N_{CF})(N'_{1,s}+N'_{2,s}+N'_{3,s}+3N_{CF})}{(N'_{1,s}+N_{CF})(N'_{2,s}+N_{CF})(N'_{3,s}+N_{CF})(N'_{1,s}+N'_{2,s}+N'_{3,s}+3N_{CF})+(N'_{1,v}N'_{2,v}N'_{3,v}(N'_{1,v}+N'_{2,v}+N'_{3,v}))}$$

$$\implies \frac{(N^{'}_{1,s}+N_{CF})(N^{'}_{2,s}+N_{CF})(N^{'}_{3,s}+N_{CF})(N^{'}_{1,s}+N^{'}_{2,s}+N^{'}_{3,s}+3N_{CF})}{N^{'}_{1,v}N^{'}_{2,v}N^{'}_{3,v}(N^{'}_{1,v}+N^{'}_{2,v}+N^{'}_{3,v})} = \frac{D_F}{1-D_F}$$

Using Equation (6), we have

$$(7) \qquad \frac{(N^{'}_{1,s}+N_{CF})(N^{'}_{2,s}+N_{CF})(N^{'}_{3,s}+N_{CF})(N^{'}_{1,s}+N^{'}_{2,s}+N^{'}_{3,s}+3N_{CF})}{N^{'}_{1,s}N^{'}_{2,s}N^{'}_{3,s}(N^{'}_{1,s}+N^{'}_{2,s}+N^{'}_{3,s})} = \frac{D_F(1-D_C)}{(1-D_F)D_C}$$

For solving Equation (7) for N_{CF}, we require the values of $N^{'}_{i,s}$ and $N^{'}_{i,v}$ for i=1..3. $N^{'}_{i,s}$ and $N^{'}_{i,v}$ are dependent on the distrust of participant i and its past calls $N^{'}_{i,B}$ for i=1..3 as presented in Lemma 1. Substituting $N^{'}_{i,s}$ and $N^{'}_{i,v}$ for i = 1..3 with the current distrust D_C and final distrust D_F in Equation (7), the number of calls N_{CF} required to move from distrust D_C to distrust D_F can be computed. Using Equation (7), the number of spam calls required to move a caller from a given list (e.g., whitelist) to another list (e.g., blacklist) can be computed by assuming the values of current distrust (D_C) and final distrust (D_F) based on the values defined for the lists in Axiom 1.

Corollary 1: The number of spam calls (N_{CF}) required by VSD to identify a new spammer is 3.

Equation (7) is used for computing the number of spam calls required by the VSD for moving the distrust of an incoming call from D_C to distrust D_F. For a new spammer from a new host and domain, the spaminess and legitimateness for each call participant in the call is equal to 1 i.e. $N^{'}_{1,s} = N^{'}_{1,v} = N^{'}_{2,s} = N^{'}_{2,v} = N^{'}_{3,s} = N^{'}_{3,v} = 1$ (Property 2). These values if substituted in Equation (1) would result in an initial distrust value $D_C = 0.5$. Therefore, substituting above values, the value of D_C, and threshold T = 0.99 (therefore $D_F = 0.99$) to directly get filtered in Equation (7), we get a value of $N_{CF} = 3$ i.e. VSD takes 3 spam calls to move from initial probability

of 0.5 to the threshold probability $D_F = 0.99$. This is experimentally validated in Section 4.5.2.1. In practice, the value of N_{CF} depends on the threshold value.

Note 1: In this lemma, we assume that the number of spam calls from the host and domain are same as the number of spam calls from the user (N_{CF}). However, it is quite possible that there might be different user accounts on the same IP phone and many hosts in the same domain. In this situation, depending upon the spam behavior of other users, the spam histories of the host and domain change. Therefore, at any instant, the number of calls required to cross the threshold would be dependent on the spaminess and legitimateness of each of the call participants as is modeled using Equation (1).

Note 2: The quantitative model computes the number of spam calls required to categorize a caller into blacklist by increasing the call participants' spaminess. A similar model can be used to compute the number of legitimate calls required to categorize the caller into a whitelist. However, in this case, the spaminess remains the same and legitimacy of the call participants change i.e., after N_{CF} number of legitimate calls, the spaminess $N_{i,s}$ remains $N'_{i,s}$, but the legitimateness $N_{i,v}$ changes to $N'_{i,v} + N_{CF}$ for each call participant i for $i = 1..3$.

Note 3: The number of call participants for analysis can be extended from the three participants of calling user, calling host, and call-generating domain to include other call participants such as the source and intermediate proxies. In this case, to categorize a caller into blacklist, N_{CF} would be a function of the individual spaminess of all the "n" call participants i.e., given spaminess $N_{i,s}$ for $i = 1..n$ and the distrusts D_C and D_F, the number of spam calls N_{CF} can be computed. Similarly, to categorize a caller into a whitelist, N_{CF} would be a function of the individual legitimateness of all the "n" call participants, i.e., given legitimateness $N_{i,s}$ for $i = 1..n$ and the distrusts D_C and D_F, the number of legitimate calls N_{CF} can be computed.

Note 4: Corollary 1 is a specific case for a new spammer who does not have history of calling any of the users inside the VSD_{domain}. Due to this, we initialize the spaminess

and legitimateness of the call participants from this new caller to be 1 and derive a value of $N_{CF} = 3$. However, for calls from other callers that have a history of calling users inside the VSD_{domain}, the value of N_{CF} depends on the previous spaminess and legitimateness of the incoming call's participants. Higher spaminess compared to legitimateness of the call participants results in a value of N_{CF} less than 3, and higher legitimateness compared to spaminess of call participants results in a value of N_{CF} greater than 3. To generalize, for a call from new spammer with n call participants, the spaminess and legitimateness of each call participant can be initialized to a value 1 for computing the value of N_{CF}.

The above quantitative model is derived based on an assumption that there are spam or legitimate calls from the caller to the callee. However, it is possible that a callee does not receive a call from the caller for a long time. In this case, the trust level of the caller fades over time.

Property 4: We forget bad and good experience over time, and as a result, in the absence of any transactions, trust fades.

Trust is accrued over a period of time. This accrued trust is a representation of the caller's past behavior. But, in absence of calls from the caller over a period of time, the trust decreases, i.e., trust fades with time. This fading of trust is exponential (Palla et al [59]). If the last transaction with the caller was at time t_p in the interval $\{t_1,t_n\}$ such that $t_1,,t_p,,t_n$, then, the trust value will decay over the time period $\Delta = t_n - t_p$. This can be represented by $T_{i,n} = T_{i,p}e^{-\Delta t}$ where $T_{i,p}$ and $T_{i,n}$ represent the trust for a given call participant "i" for $i = 1..n$ at time periods t_p and t_n respectively. This fading of trust over a period of time for a caller is as shown in Figure 4.3.

All the above notions of trust can be applied only when there is an available caller history for the incoming call. But, in everyday life, we receive calls from individuals who are calling for the first time (e.g., unknown callers - strangers). We lack a prior

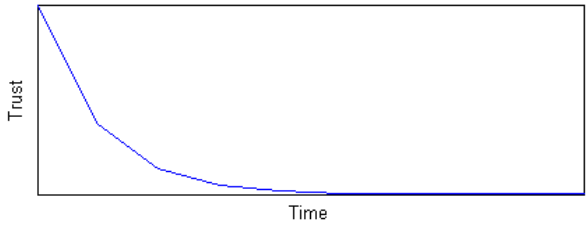

FIGURE 4.3. Trust fades with time. In the absence of transactions, trust decreases exponentially in the order of elapsed time

calling history for them. In this case, we must rely on reputation or on recommendations based on word of mouth.

4.4.3.2. *Reputation.* It is a human tendency to rely on the opinions of trusted people regarding an individual's trustworthiness (usually referred to as the individual's reputation) in addition to one's own experience. Knowing the individual's reputation becomes even more necessary when we have no previous experience with that individual. Inferring reputation for detecting the spam behavior of a call is useful particularly in cases when neighbors have first-hand experience with the caller. Here we present a model for inferring reputation and updating it based on callee's feedback. But, first, we define reputation of a call participant *as a notion or report of its propensity to fulfill the trust placed in it (during a particular situation); its reputation is created through feedback from individuals who have previously interacted with the call participant* (Goecks et al [26]).

The reputation of a call participant is inferred based on the recommendations of the neighbors of the callee (e.g., other employees in the enterprise) as given in Ono et al [57]. These neighbors can in turn poll their neighbors for the call participant's reputation. This reputation mechanism can be integrated into the functionality of VSD. For this integration, instead of the actual callee seeking the recommendations regarding each call participant, the VSD seeks recommendations about the caller's

domain proxy from its own neighboring domain proxies. The neighboring proxies, in turn, seek the recommendation of their neighboring proxies until the caller's domain proxy is reached. VSD then infers the reputation of the caller's domain proxy based on these recommendations using Bayesian Networks inference techniques. For this, with respect to a domain proxy, a graph can be generated using the neighboring proxies that can be used in deriving the caller's domain reputation. For example, consider an example ring proxy network topology (Lancaster [44]) graph given in Figure 4.4.

For the given topology graph in Figure 4.4, reputation is inferred using

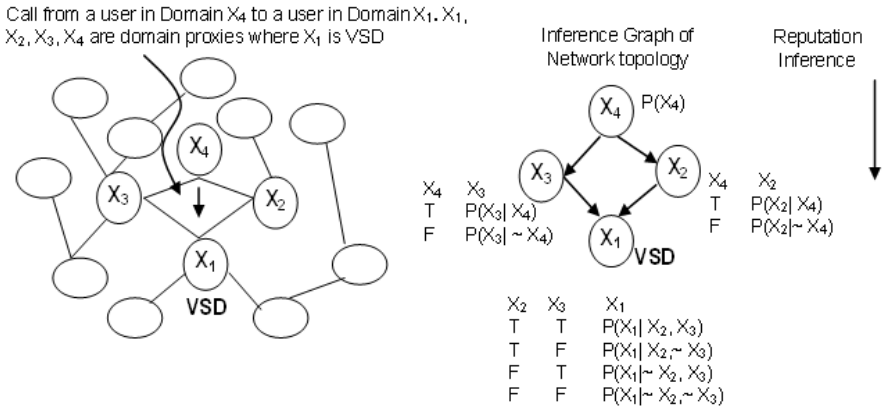

FIGURE 4.4. Reputation inference for a call from domain X4 to domain X1. The proxy topology for reputation takes into account the interconnection among domain proxies that are in all the possible paths from the caller's domain proxy to VSD. This topology can then be used for propagating and updating reputation information using Bayesian Networks based on an observed evidence of spam based on callee's feedback

Bayesian networks. For a call from a domain (X_2, X_3 or X_4) to a user inside X_1 (VSD_{domain}), the reputation of the domain can be updated by feedback from the end user i.e., the evidence is propagated throughout the Bayesian network (detailed probabilistic model is explained in Appendix C). For a call from domain X_4 to domain

X_1, the reputation of domain X_4 can be inferred by computing $P(X_1|X_4)$, i.e. the posterior probability of X_1 given an event that a call has been generated at X_4.

$$P(X_1|X_4) = P(X_1, X_2|X_4) + P(X_1, \sim X_2|X_4)$$

$$i.e. P(X_1|X_4) = P(X_1|X_2)P(X_2|X_4) + P(X_1| \sim X_2)P(\sim X_2|X_4) \tag{8}$$

where

$$P(X_1|X_2) = P(X_1, X_3|X_2) + P(X_1, \sim X_3|X_2)$$

$$P(X_1|X_2) = P(X_1|X_2, X_3)P(X_3) + P(X_1|X_2, \sim X_3)P(\sim X_3) \tag{9}$$

and

$$P(X_1| \sim X_2) = P(X_1, X_3| \sim X_2) + P(X_1, \sim X_3| \sim X_2)$$

$$= P(X_1| \sim X_2, X_3)P(X_3) + P(X_1| \sim X_2, \sim X_3)P(\sim X_3) \tag{10}$$

$$P(X_3) = P(X_3, X_4) + P(X_3, \sim X_4) = P(X_3|X_4)P(X_4) + P(X_3| \sim X_4)P(\sim X_4) \tag{11}$$

Solving Equations (8)- (11) gives the reputation $P(X_1|X_4)$ of domain proxy X_4. The inferred reputation of caller's domain is then updated based on callee's feedback. The reputation for the caller domain that is inferred can then be used either for increasing or decreasing the trust level (See Section 4.4.3.3) of the incoming call.

Property 5: Trust and reputation levels increase additively and decrease multiplicatively.

In our daily life, we slowly gain trust but we develop distrust quickly. We model this intuitive behavior in updating the trust and the reputation values. However, previously in the chapter we have adopted a linear model in updating trust (or distrust). Here, we present an alternative model for updating trust. The type of update model

to be used depends on the sensitivity of the underlying application.

For a legitimate-call feedback from a callee, the distrust D is decreased as shown below (increasing the legitimateness decreases the distrust)

$D = f(\{N_{1,s},\ N_{1,v} + T_1\},\{N_{2,s},\ N_{2,v} + T_2\},\{N_{3,s},\ N_{3,v} + T_3\}.....\{N_{n,s},\ N_{n,v} + T_n\},$ $N_S,\ N_V + 1\ )$

Similarly, reputation is additively increased for good behavior. For the graph topology in Figure 4.4, the parameters for each node are updated additively for a legitimate call. For example, for a legitimate-call feedback from callee for a call from domain X_4 to domain X_1, the parameters of each node are updated as follows:

Node $X_2 : P(X_2|X_4) = P(X_2|X_4) + r_1 \qquad P(X_2| \sim X_4) = P(X_2| \sim X_4) - r_1$

Node $X_3 : P(X_3|X_4) = P(X_3|X_4) + r_1 \qquad P(X_3| \sim X_4) = P(X_3| \sim X_4) - r_1$

Node $X_1 : P(X_1|X_2, X_3) = P(X_1|X_2, X_3) + \frac{P(X_1|X_2)}{S_{inf}} s_1 + \frac{P(X_1|X_3)}{s_{inf}} S_1$

$P(X_1|X_2, \sim X_3) = P(X_1|X_2, \sim X_3) + \frac{P(X_1|X_2)}{S_{inf}} s_1 - \frac{P(X_1|X_3)}{S_{inf}} s_1$

$P(X_1| \sim X_2, X_3) = P(X_1| \sim X_2, X_3) - \frac{P(X_1|X_2)}{S_{inf}} s_1 + \frac{P(X_1|X_3)}{S_{inf}} s_1$

$P(X_1| \sim X_2, \sim X_3) = P(X_1| \sim X_2, \sim X_3) - \frac{P(X_1|X_2)}{S_{inf}} s_1 - \frac{P(X_1|X_3)}{S_{inf}} s_1$

where r_1 and s_1 are constants and $S_{inf} = P(X_1|X_2) + P(X_1|X_3)$.

Alternatively, for a spam call, both the trust and the reputation levels decrease multiplicatively. This is represented as follows

$D = f(\{N_{1,s} + J_1 D_1,\ N_{1,v}\},\{N_{2,s} + J_2 D_2,\ N_{2,v}\},\{N_{3,s} + J_3 D_3,\ N_{3,v}\}.....\{N_{n,s} + J_n D_n,$ $N_{n,v}\},\ N_S + 1,\ N_V\ )$

where D_i is the associated distrust and J_i is multiplicative constant for updating distrust for a call participant i for $i = 1..n$. In the basic case, $J_i D_i = 1$ for $i = 1..n$ for experimenting with a linear increase in distrust.

Similarly for reputation, the parameters for each of the nodes in the graph topology would be updated for a spam-call feedback from the callee as follows

Node $X_2 : P(X_2|X_4) = P(X_2|X_4) - r_2 \qquad P(X_2| \sim X_4) = P(X_2| \sim X_4) + r_2$

Node $X_3 : P(X_3|X_4) = P(X_3|X_4) - r_2 \qquad P(X_3| \sim X_4) = P(X_3| \sim X_4) + r_2$

Node $X_1 : P(X_1|X_2, X_3) = P(X_1|X_2, X_3) - \frac{P(X_1|X_2)}{S_{inf}} s_1 - \frac{P(X_1|X_3)}{S_{inf}} s_1$

$$P(X_1|X_2, \sim X_3) = P(X_1|X_2, \sim X_3) - \frac{P(X_1|X_2)}{S_{inf}} s_1 + \frac{P(X_1|X_3)}{S_{inf}} s_1$$
$$P(X_1| \sim X_2, X_3) = P(X_1| \sim X_2, X_3) + \frac{P(X_1|X_2)}{S_{inf}} s_1 - \frac{P(X_1|X_3)}{S_{inf}} s_1$$
$$P(X_1| \sim X_2, \sim X_3) = P(X_1| \sim X_2, \sim X_3) + \frac{P(X_1|X_2)}{S_{inf}} s_1 + \frac{P(X_1|X_3)}{S_{inf}} s_1$$

where $S_{inf} = P(X_1|X_2) + P(X_1|X_3)$. r_2, s_2 are constants such that $r_2 > l_1 r_1$ and $s_2 > l_2 s_1$ and $l_1, l_2 > 1$. The two constants l_1 and l_2 are the multiplicative constants and are configured based on callee's preferences. The other constants r_1, r_2, s_1, s_2 are configuration parameters of VSD and can be initialized based on criteria such as the maximum number of calls that can be allowed from a spam domain and spam host. The updated values for the reputation parameters are in turn substituted in Equations (8)- (11) to result in a new set of updated reputation values for the nodes X_2, X_3, and X_4 (represented by $P(X_1|X_2), P(X_1|X_3)$ and $P(X_1|X_4)$ respectively). For a given set of initial or prior probabilities for the topology graph nodes representing the reputation of those domains, and for a spam call from domain X_4 to domain X_1, the Bayesian inference calculations shown above would decrease the reputation for X_2 , X_3 and X_4 proxies and increase the reputation for a legitimate call for the same domain proxies. For every incoming call, this adaptive update of reputation is derived for all the domains in the probable path from the source domain proxy to VSD.

Property 6: With no prior experience, we rely on reputation. After multiple transactions with the caller, trust takes precedence and the influence of reputation decreases.

Many a times, trust and reputation are used to represent human belief. Trust represents a caller's past behavior whereas reputation signifies social status. While trust is computed, reputation is derived. Figure 4.5 presents a trust-and-reputation-influence plot based on human intuitive behavior in estimating the belief we place in individuals.

With no available history or experience, we rely mostly on the caller's reputation. Once we start receiving calls from the caller, trust would have more influence than

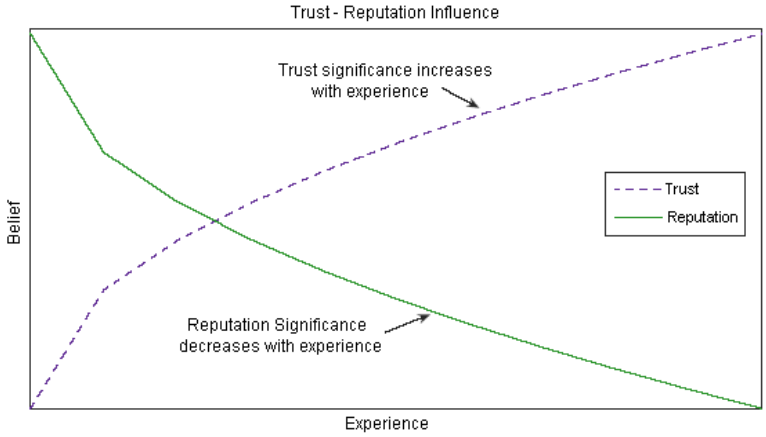

FIGURE 4.5. Real life influence of trust and reputation. With no previous experience one relies mostly on reputation or recommendations. With increasing experience, the influence of trust increases and that of the reputation decreases

reputation. This is particularly useful when our main goal is to define customized filters as the needs and perceptions of people change.

4.4.3.3. *Integrating Trust and Reputation.* The trust and reputation models defined in Section 4.4.3.1 and Section 4.4.3.2 are integrated as functional working modules in our voice-spam-filter analysis as shown in Figure 4.6.

For an incoming call, the spam level of the call can be computed using distrust D of the call by taking into account the spam and legitimate histories of the call participants. The reputation module infers the reputation (R) of the source domain and then augments the distrust (D) based on inferred reputation. As shown in Figure 4.6, the final spam behavior (p_c) in filtering is a result of the analysis of the incoming call by applying the principles of distrust and reputation. For the above analysis, the filter formal notation can be summarized as follows,

$D = (C,\ Ns,\ Nv)$: Distrust computation based on history of call participants.

$R = [P(X_1|X_2), P(X_1|X_3), P(X_1|X_4)]$ - Reputation analysis for the 4-node graph

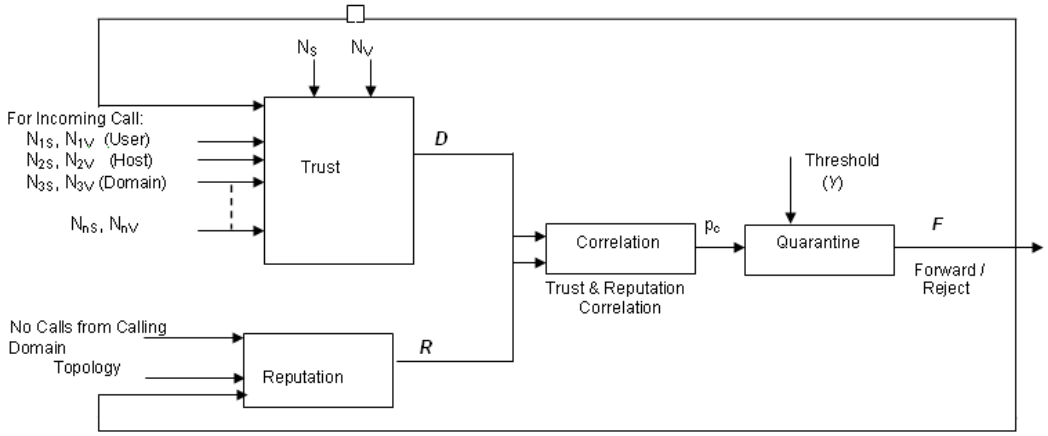

FIGURE 4.6. Integration of trust & reputation. The trust is either increased or decreased based on reputation of caller's domain. The collective inference of these two stages results in a decision as to whether forward or quarantine the call

shown before where X_1 is the VSD and X_2, X_3, X_4 are the domain proxies.

$p_c = (D,R)$: a correlation between the trust and reputation analysis. This tells us how the distrust D is updated based on reputation analysis. In this chapter we have used a simple linear update based on the extent of deviation in reputation of a domain from its initial reputation.

$F = (pc,\ Y)$: F represents a Forward / Reject decision by comparing the final spam level of the call (pc) with the assumed threshold Y (defined in Axiom 1). F can be a simple Boolean function resulting in True (call is spam - filter the call) or False (call is not spam - forward the call).

In the above sections we have discussed an evidence-based filtering technique for filtering VoIP spam calls. In the next section, we compare our methodology with the existing evidence based techniques.

4.4.3.4. *Comparison of Evidence Based Filtering Techniques.* The VoIP spam detection framework discussed previously in the chapter can be compared with some

of the existing evidence based spam filtering approaches used in e-mail. Table 4.1 presents a tabular qualitative comparison of the features and techniques supported by different approaches.

TABLE 4.1. Qualitative comparison of techniques used by existing spam filtering approaches.

Features	P1	P2	P3	P4	P5	P6
Trust computation framework using past history		✓		✓	✓	✓
Reputation inference based on recommendations	✓	✓			✓	✓
Rate Limiting						✓
Presence or context information for real-time analysis						✓
Blacklists and whitelists for signature based detection		✓	✓			✓
Community experience	✓					✓
Feedback control				✓		✓
Identity Verification		✓	✓			
Collaborative analysis		✓			✓	✓
Adaptive and real-time usage		✓				✓
Distributed Solution					✓	✓
Challenge and Response techniques		✓	✓			

P1: Golbeck et al [27] P2: Seigneur et al [80]

P3: Wattson[91] P4: Damiani et al [11]

P5: Foukia et al [24] P6: Our proposed VoIP spam filtering framework

4.5. Experimental Results

VoIP deployment is still at its inception. No VoIP corpus exists for testing a detection mechanism. So, to test our proposed VoIP spam-detection framework, we use randomly generated data for the network setup defined in Figure 4.1. The end

users in the call-generating domains and VSD_{domain} (the enterprise network) are either real SIP IP phones or soft clients compliant with SIP RFC ([70]). End users outside the VSD_{domain} use randomly generated usernames and IP addresses to form a from SIP URI. The end users inside the VSD_{domain} can receive calls forwarded by the VSD and also generate a call to a randomly selected user outside the VSD_{domain}. The call-generation process uses a Bernoulli distribution. Calls are generated with an average rate of 8 calls/minute. Neither the VSD nor the VSD_{domain} end users have any knowledge of the call-generation process. A random subset of users, hosts, and domains outside the VSD_{domain} are configured to be spam entities before the start of experiments. We ran the experiments with 6 users inside the VSD_{domain} and 40 users outside the VSD_{domain}. The VSD_{domain} is configured to be a domain with 5 Class C networks. For a given user inside the VSD_{domain}, Figure 4.7 compares the number of total calls and spam calls from all users outside the VSD_{domain}, and the number of filtered calls by the VSD.

4.5.1. VSD Architecture: Collaboration between Different Filtering Techniques

In our architecture, filtering techniques employed at each stage of spam analysis include spam and legitimate signatures (black- and whitelists), trust, and reputation of the calling party. While most current spam filters employ blacklisting as their sole means of stopping junk calls, blacklisting coupled with trust and reputation inference techniques increases the filter accuracy as shown in Figure 4.8. The figure presents the number of spam calls blocked using the three stages of analysis. It can be observed that the number of spam calls blocked using blacklisting, trust and reputation is approximately 97.16% compared to 4.25% if only blacklisting is implemented.

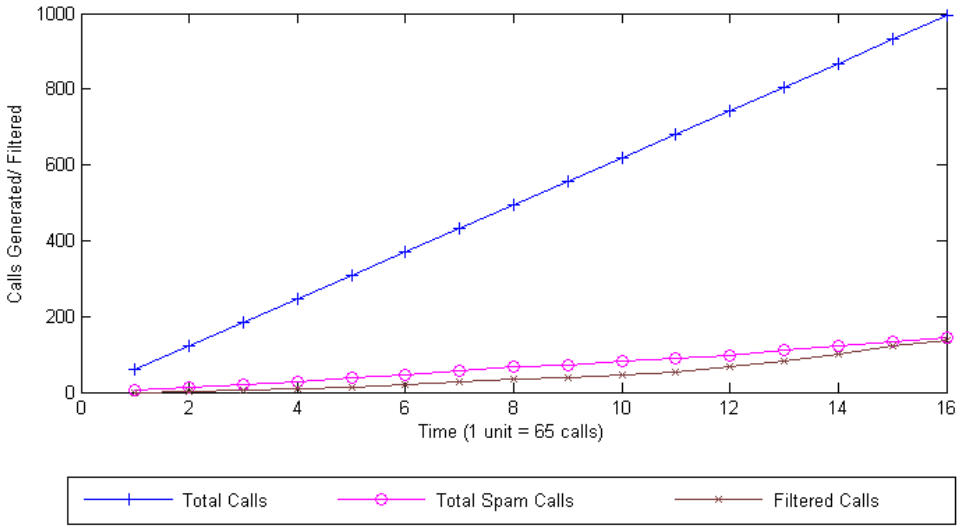

FIGURE 4.7. Comparing the total calls generated, generated spam calls and filtered calls. The VSD continuously tries to catch up with generated spam

4.5.2. Accuracy of the Voice Spam Detector(VSD)

The VSD's accuracy can be estimated by comparing the rate of spam with the rate of filtered calls.

4.5.2.1. *Accuracy during learning period.* Initially, VSD has no knowledge of spam but learns the callers' behavior using feedback from the end users. This learnt behavior is used for blocking spam calls. The number of spam calls filtered by the VSD increases with time and, ultimately, tries to catch up with the generated spam calls. Figure 4.9 presents the total number of spam and filtered calls from all the callers to a particular user inside the VSD_{domain}. VSD catches up with spammers during its learning period. After the learning period, VSD has an accuracy of 97.6%, a false positive percentage of 0.4% with 2% of spam calls forwarded to the end user (false negatives). After 16 time units (Figure 4.9), the filter locks-in with the spammers, thus, improving the accuracy of detection.

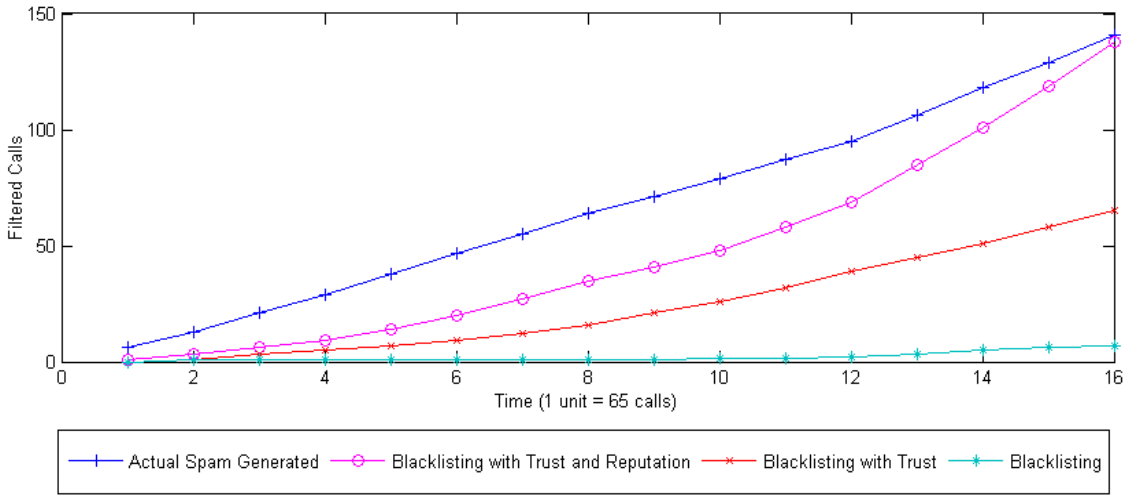

FIGURE 4.8. Spam Calls blocked by VSD for different stages of analysis. Filter performance improves significantly when the three stages (blacklisting, trust, and reputation inferences) are used collectively to infer the spam behavior of incoming calls. Data for the plots include spam calls generated by 40 users outside the VSD_{domain} to a user inside the VSD_{domain}

Similar behavior of filter locking-in with a spammer for a given user inside the VSD_{domain} can be observed in Figure 4.10. The figure depicts the VSD's accuracy during the lock-in period for an end user inside the VSD_{domain} from a particular caller. It presents the learning period when the spam calls are generated from that caller. For a caller who is repeatedly spamming the end user, the filtered-calls curve catches up with the spam-calls curve after the 3rd call, i.e., the caller's 3rd spam call is automatically filtered. The rate of filtered calls from then on equals the rate of generated spam calls, resulting in minimum false alarms (validating Corollary 1). Next we study the accuracy of the filter during the lock-in period.

4.5.2.2. *Accuracy during lock-in period.* After the learning period, the number of filtered calls will be close to the number of spam calls. During this time, VSD

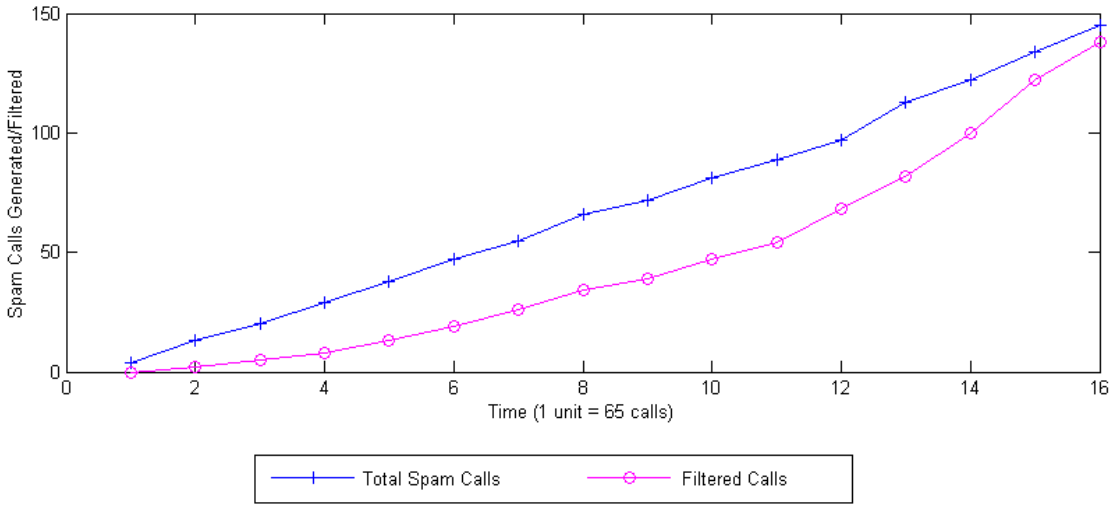

FIGURE 4.9. Spam Detection Accuracy increases with time. The number of filtered spam calls increase with time as VSD learns the behavior of calling entities. This learned knowledge results in the VSD filtering more and more spam until it catches up with the generated spam

may filter other calls (creating false positives) or may let the spam calls reach their destination (false negatives).

Figure 4.11 is a magnified version of Figure 4.9 during the lock-in period. This graph describes the rate of spam calls versus the rate of filtered calls from all callers to a given user inside the VSD_{domain} during the lock-in period. At any time, the difference between the spam and filtered calls provides the number of false alarms. Initially, the filter starts learning the spam behavior and, therefore, fewer spam calls are filtered, resulting in false negatives. After considerable learning, the rate of filtered calls will almost be equal to the rate of spam calls. Later, it is also possible that the VSD will filter more calls than the actual generated spam calls in that time period

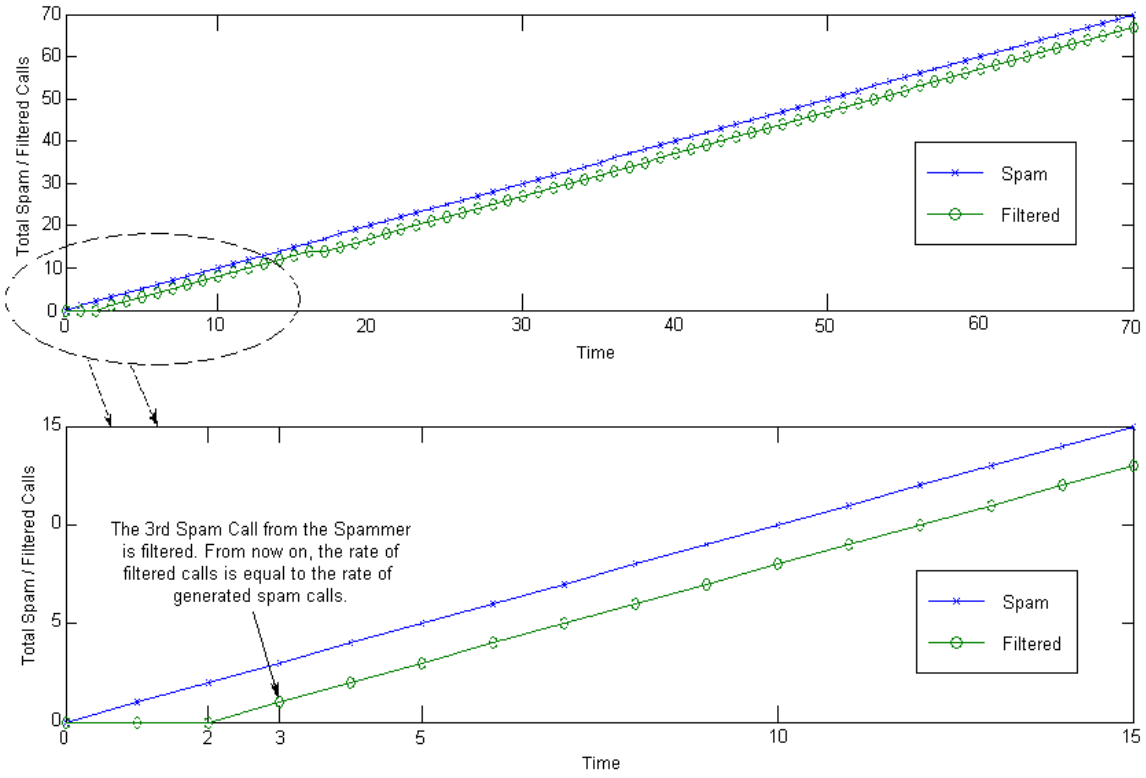

FIGURE 4.10. Spam Detection Accuracy increases with time. The graph presents the learning period for the user with respect to a particular caller outside the VSD_{domain}. Initially, spam calls are forwarded, but the VSD learns the caller's spam behavior during the learning period and starts to filter the caller's calls. All spam calls starting with the caller's 3rd spam call are directly filtered.

resulting in false positives. This can happen in a random setting because some non-spam users can accrue spam behavior by sharing resources (e.g., hosts, domains) with the spammers.

4.5.2.3. *Improving the accuracy of VSD.* Filtered calls are fewer than the number of spam calls before lock-in because the VSD's knowledge regarding the spammers is insufficient. This is the period where false negatives appear. This is represented by

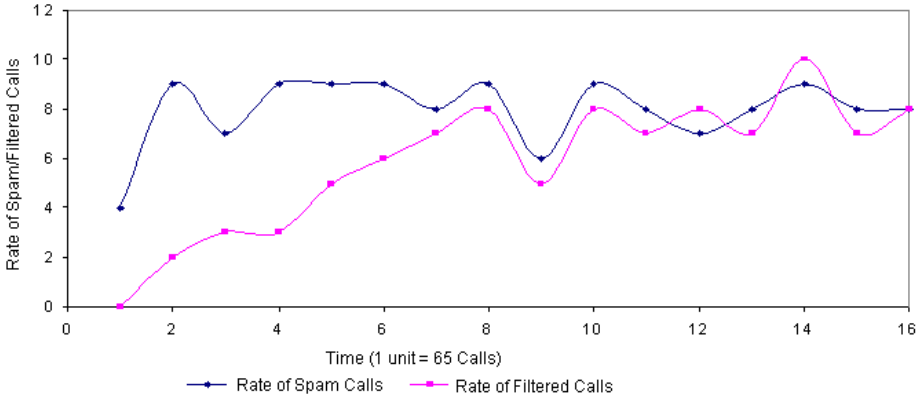

FIGURE 4.11. Small number of false alarms after the learning period. VSD, after learning the caller's spam behavior, filters most of the spam. After the learning period, the calls filtered by VSD are almost same as the spam calls but with very few false alarms (we say that the filter locks in with the spammer

the false alarms curve below the x-axis as shown in Figure 4.12. The curve tends to zero-in when the VSD has the complete history of all the callers so that every spam call can be right away stopped. At times, it can happen that some of the legitimate callers accrue spam behavior by using spam resources (e.g., spam host, spam domain etc.). Because of this, it is possible that the filter blocks more calls than actual spam calls thereby creating false positives represented by the error curve above the x-axis as shown in Figure 4.12.

In view of the above scenarios, we believe that in addition to using feedback from the end user regarding false negatives (like a spam button), using feedback about the false positives would prove to be equally effective for spam learning. This feedback mechanism is similar to e-mail where a filtered email is routed to a junk-mail folder. Instead of directly blocking the filtered calls, the VSD would forward the suspicious call to the callee's voice-mail box. The callee would have added flexibility of looking at the calls in his voice-mail box and reporting the validity of the calls to the

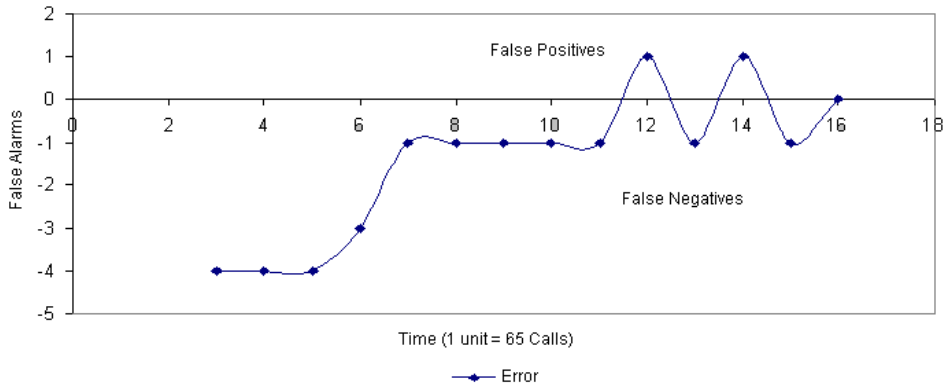

FIGURE 4.12. Small number of false alarms after the learning period. A false negative (spam classified as legitimate) results in ringing the phone and a false positive (legitimate classified as spam) is diverted to the voice mail box. In both cases, the VSD updates the history based on the end-user's feedback. The feedback can either be explicit feedback (when end-user presses the spam button) or implicit feedback (when the end-user calls back the caller). Hence, user feedback can be used to reduce error and to keep false alarms to a minimum.

VSD. For example, the callee could inform the VSD that a call in the voice-mail box is legitimate. The VSD can then update the legitimate history of that caller. This update can be a linear decrease in distrust or complete spam history reset. In this chapter we have used the linear decrease in distrust update procedure. Alternatively, the callee could also respond saying that the call is a spam call or could immediately purge the call from the voice-mail box. In this case, the VSD will implicitly understand that the call was indeed spam and that the filter accurately judged the spam behavior of the call. With this kind of callee response, the filter can reduce the number of false positives. As a result, the curve in Figure 4.12 drops to the actual spam-call curve. This process repeats and, eventually, the filter converges with the

zero line locking-in closely with the spammers.

4.6. Sensitivity Analysis

While considering the accuracy of the filter, it is important to analyze the impact of the configuration parameters. In this section, we analyze the effect of parameters such as spam volume and the network size on the filter's accuracy.

4.6.1. Spam volume versus Accuracy

Property 7: More the amount of spam, the easier it is to detect it. We may receive more spam because of an increasing number of spammers in the call-generating domains outside the VSD_{domain} (e.g., telemarketing company with huge employee base) or because the amount of traffic generated by spammers is significantly higher than that of legitimate callers. We have observed that for a given number of calls, VSD takes less time to learn spam behavior when the volume of spam calls is high than when the number of spam calls is low. This relationship between spam volume and the VSD's spam-detection capability can be proved using an analytical model. For proving this relation, two different percentages of spam (x% and y% of spam processed by VSD such that $x < y$) can be individually substituted in Equation (1) as shown in Appendix D. We further support the analytical model in D by presenting our experimental results for spam-detection capability for varied amounts of spam. To measure this capability, we plot the rate of false negatives for varying amounts of spam.

Figure 4.13 describes the plot for rate of false negatives versus the amount of spam generated from the calling domain. As the amount of spam increases, the number of false negatives decreases, i.e., the filter's spam-detection capability increases. It can be inferred that the detection capability increases with increasing spam encountered, but at the same time the filter shows as much capability with a

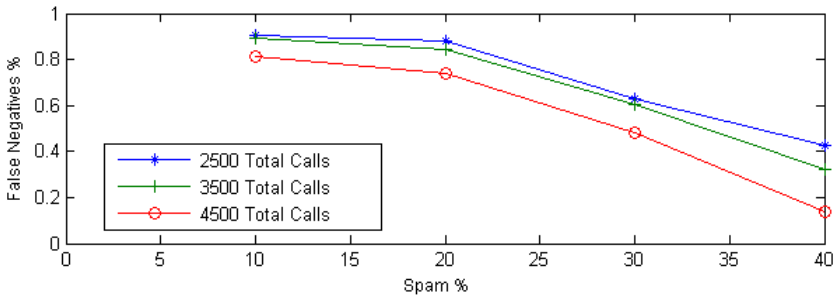

FIGURE 4.13. False negatives decrease when the VSD encounters more spam. The greater the number of spammers among the users generating calls, the higher is the probability that the incoming call is spam. Therefore, the chances that the VSD will filter the call are higher. This larger number of spammers also reduces the chances that the VSD will forward the spam, i.e., there are fewer false negatives

smaller percentage of spam when it has more time (more calls) for learning. The detection capability based on the experimental results can be compared with that of the analytical model. The detection capability based on experimental results is directly derived from the false negative rate shown in Figure 4.13. The detection capability based on an analytical model can be obtained by substituting different values for $N_{i,s}$ and $N_{i,v}$ (such that the sum of them is always a constant) for a call participant i for $i = 1..n$ in Equation (1).

Figure 4.14 presents the plot for a comparison between the spam-detection capability based on the analytical model and our experimental results. The spam-detection capability based on experimental results shows similarity with that of the analytical model. However, the increase in the spam-detection capability for larger amounts of spam might result in a few more false positives. This increase can be represented by plotting the false positive rate using RoC curves for two amounts of spam as shown in Figure 4.15. The figure represents the false positive rate for 20% and 40% of spam in the total traffic processed by VSD. It can be observed that as

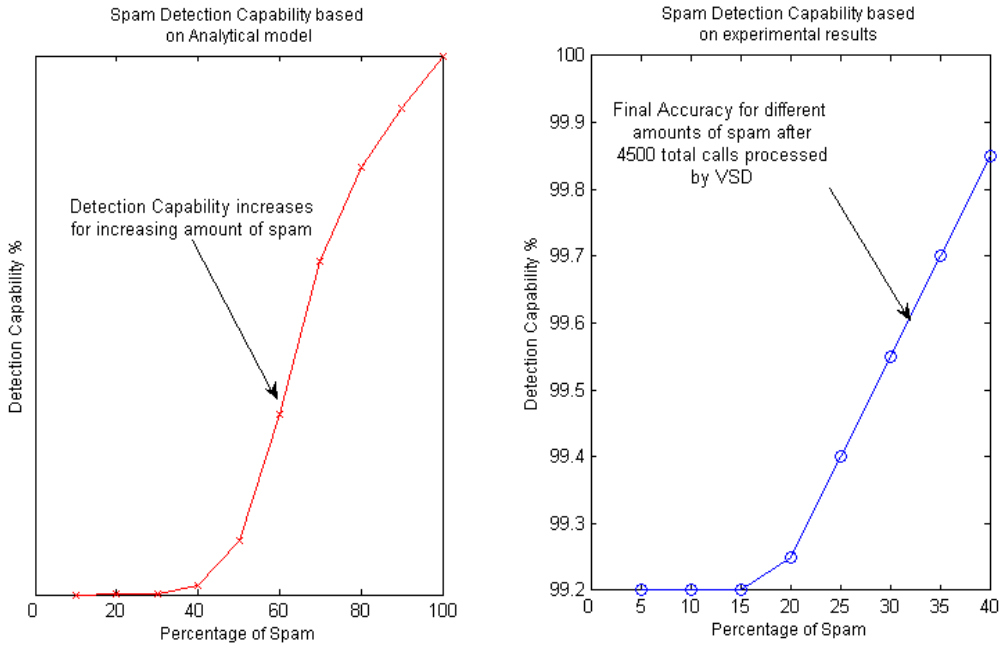

FIGURE 4.14. Experimental Accuracy vs. Analytical Detection Capability with respect to amounts of spam.

the amount of spam increases the false positive rate increases. This happens in a random setting because as the number of spammers increase, the percentage of non-spam or legitimate users decreases. Due to this, the legitimate users begin to accrue spam behavior by using resources (e.g., host and domain etc.) used by the spammers. This results in the legitimate users getting filtered at the VSD thus resulting in more false positives. Taking into account the feedback about filtered calls as described in Section 4.5.2.3 can reduce these false positives.

4.6.2. Network Size versus Accuracy

Another important parameter that can affect the filter's accuracy is network size. The users registering for VoIP services might have dynamic addresses because of the end hosts running the DHCP protocol. Because of this, every time a user connects to the VoIP network, it might have a different Class B or Class C network address (mostly a different Class C). In this event, the number of available IP addresses in

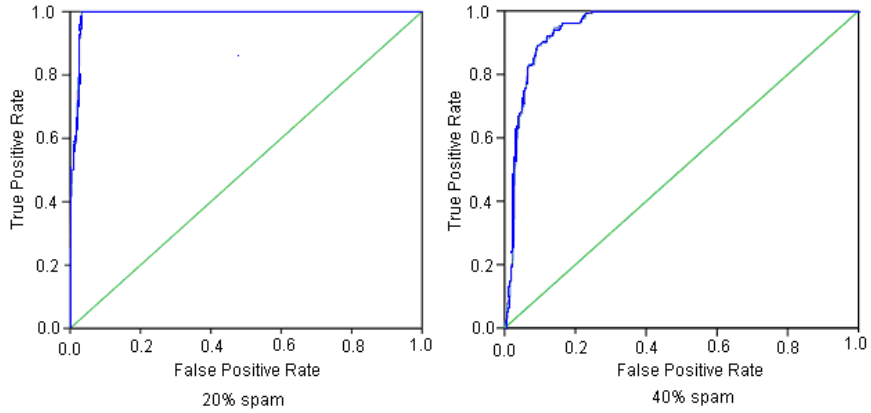

FIGURE 4.15. Experimental Accuracy vs. Analytical Detection Capability with respect to amounts of spam.

the address space greatly affects estimating the spam behavior of the users in the networks inside a VoIP domain. For a smaller number of networks inside a VoIP domain, VoIP users have a more limited set of IP addresses to use for connecting to the network. In this case, the behavior of a spammer having an IP address within this address space can be learned more quickly than is the case when the spammer is in a network with more available IP address space. Catching a spammer who is within a large IP address space becomes more difficult than in the case of a smaller IP address space. We validate this by plotting the number of filtered calls by VSD from the same set of spammers for two sizes of call-generating domains (5 and 10 Class C network domains).

4.6.2.1. *Network Size versus Blocked or Filtered Calls.* Figure 4.16 gives the number of spam calls filtered for two different sizes of networks. For the same set of spammers, the time taken by VSD for learning spam behavior from a network size of 10 Class C networks is higher when compared to time taken for a network of size 5 Class C networks.

The detection capability for network sizes based on the above experimental results and analytical model can be plotted as shown in Figure 4.17. The graph for the

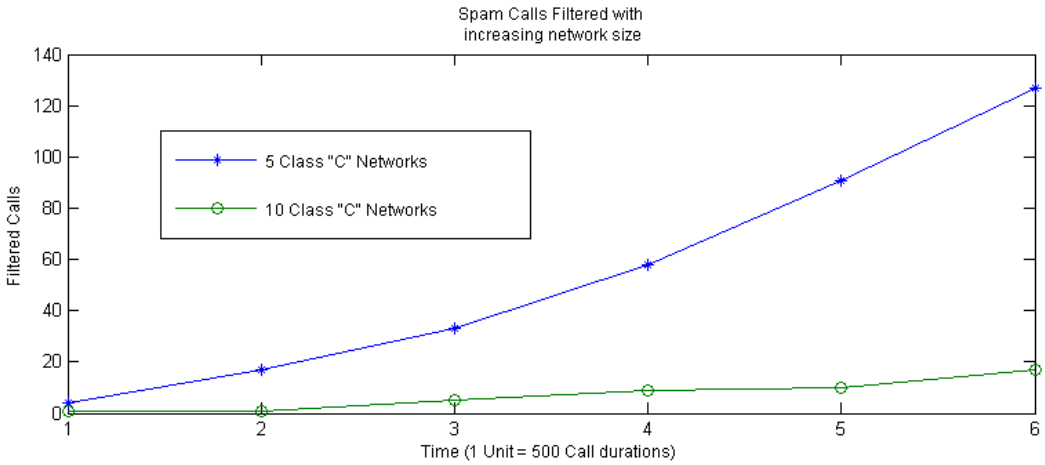

FIGURE 4.16. Spam calls filtered with increasing network size. For a same set of spammers distributed across differently sized networks, VSD is more capable of differentiating spam for smaller-sized networks than larger networks

analytical model is generated by assuming that a fixed number of spammers are distributed over different network sizes. Due to this distribution, the amount of spam varies for each unit of network size i.e., for the same set of spammers, the spam generated for each unit of network size (e.g., 1 Class C network) in smaller-sized networks is higher when compared to the amount of spam generated for each unit of network size in large-sized networks. Alternatively, the detection capability based on experimental results is directly inferred from the experimental results presented in Figure 4.16.

Figure 4.17 shows the relationship between detection capability and network size based on our analytical model and experimental results. From Figure 4.16 and Figure 4.17, we note that, for a given time period and number of calls, spam detection capability is lower for large-sized networks when compared to small-sized networks. However, spam detection capability is better even for large-sized networks

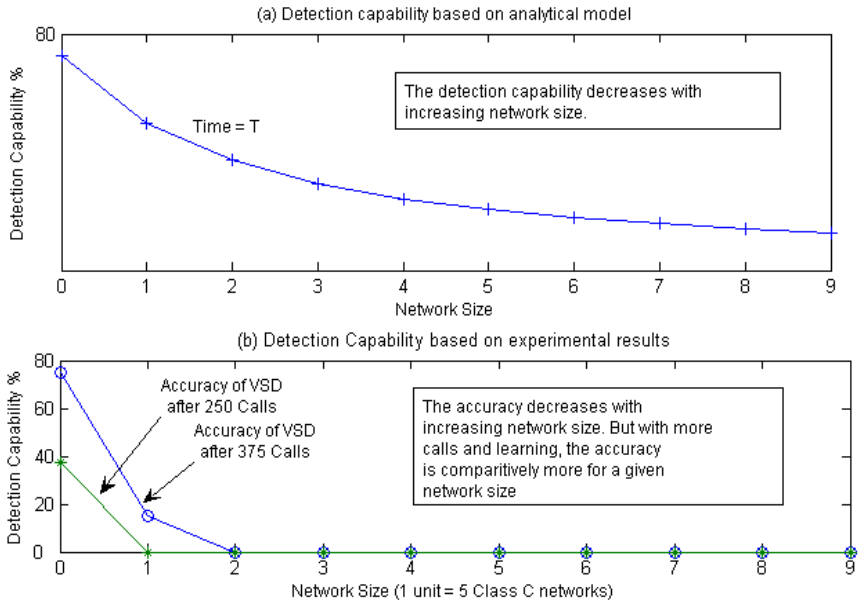

FIGURE 4.17. Blocked spam calls for increasing scalability on call generation. The detection capability based on experimental analysis is similar to that of the analytical model.

when the filter encounters large number of calls from spammers. In any case, irrespective of the network size, VSD identifies spammers if it receives a sufficient number of calls from those spammers. This can be observed in our daily life as well. It is easy to detect spam from telemarketers from the same company when compared to the telemarketers spread across the country.

4.6.2.2. *Network Size versus False Alarms.* Here we consider the effect of network size with respect to false alarms.

Figure 4.18 represents false negative rates for differing network sizes. As the network size increases, VSD takes more time in learning the spam. We need to note that the spam probability of an incoming call depends on the call participants' history. Due to a smaller number of spam calls in each unit-sized networks (e.g., 1 Class C network), it is highly likely that most of the non-spammers have created good will and a legitimate behavior. This will decrease the spam-probability of an incoming

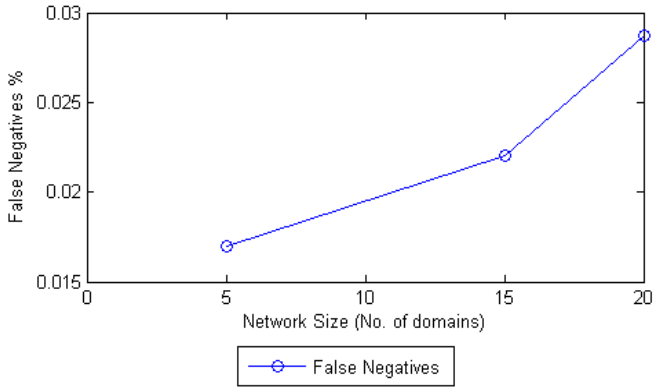

FIGURE 4.18. Increasing false negatives with increasing network size. The false negatives for small-sized networks are fewer than those identified within larger-sized networks due to the smaller number of spam calls at any given time.

call and allow spam to go through VSD. Therefore, with an increasing network size, the allowed spam will increase, i.e. number of false negatives increases. We believe we can further reduce the false negatives if we consider the experiences of others (i.e. neighbors) along with the experiences of the caller. It is highly likely that if a set of users receive spam from a source, the same source will spam other users in the same domain. So, integrating domain-level knowledge can improve the performance of VSD.

4.7. Integrating Domain Level Knowledge(We Learn from Others' Experiences)

It is frequently observed that spammers usually spam more than one user in a domain. This is certainly true for broadcast spammers who spam all the users in a domain. For example, a telemarketer would broadcast messages to many users in the domain. Nearly all the recipients of the messages will consider them to be spam. VSD can take advantage of this behavior to identify more spam calls A domain-level database (e.g., stored in a proxy server located in the enterprise's perimeter) can be

used for computing the distrust of incoming calls with respect to the callee's community. This domain-level distrust can then be used to either increase or decrease the distrust perceived by the callee (Section 4.4.3.3). However, in few cases, a callee might want to receive calls that others in the community have categorized as spam. For example, a callee in search of mortgage rates may want to receive broadcast calls about mortgages even though the calls are considered to be spam by many other members in the callee's community. In this case, the customized filter options of the callee (e.g., whitelist) can allow reception of calls of specific interest even when the domain-level spam analysis reports that incoming call is spam. It is also possible that the callee might not have a prior history of calls from such spammers. In this context, if the call is filtered and directed to the callee's voice-mail box, the callee can provide feedback telling the VSD that the call is legitimate. This ensures that future calls from that caller are directly forwarded. On the other hand, if the call is allowed through to the callee and if the callee is interested in taking the call, the callee can provide positive feedback about its legitimacy to the VDS and, thus, increase the caller's trust level relative to the callee.

Property 8: Trusting a caller depends on the social experience of the callee's community.

Upon receiving a call, if the callee doesn't have a prior experience with the caller, the callee can use the social experiences of neighbors to determine if a call should be accepted. These neighbors constitute the callee's community. Some neighbors may have first-hand experience with the caller. This can be taken advantage of for identifying spammers who make spam calls to more than one member in the callee community. In this scenario, the distrust of the incoming call with respect to all the members in the callee's community is given by

$D = f(\{N^d_{1,s},\ N^d_{1,v}\},\{N^d_{2,s},\ N^d_{2,v}\},\{N^d_{3,s},\ N^d_{3,v}\}.....\{N^d_{n,s},\ N^d_{n,v}\},\ N_S,\ N_V\ )$

where $N^d_{1,s}$ and $N^d_{1,v}$ represents the spaminess and legitimateness of call participant i

with respect to all the users inside the VSD_{domain} i.e. callee's community. Simply, for m number of callee community members, $N^d_{1,s} = \sum_{k=1}^{m} N_{i,s}$ where $N_{1,s}$ is the spaminess of the call participant i with respect to the community member k. Similarly, for m number of callee community members, $N^d_{1,v} = \sum_{k=1}^{m} N_{i,v}$ where $N_{1,v}$ is the spaminess of the call participant i with respect to the community member k. However, $N_{1,s}$ and $N_{1,v}$ may take different values when considered for different callees (community members). In order to have a lighter notation, we avoided indexing $N_{1,s}$ and $N_{1,v}$ by callee in the rest of the chapter, but $N^d_{1,s}$ and $N^d_{1,v}$ actually are $\sum_{k=1}^{m} N^k_{i,s}$ and $\sum_{k=1}^{m} N^k_{i,v}$ respectively.

Note: The above scenario assumes that each member of the callee's community is of equal importance to the callee. It is quite possible that the callee himself has different trust relationships with each community member. In this context, the callee may find it more useful to obtain recommendations from all the community members. The callee can then weigh the recommendations based on the trust relationships the callee holds with each member who has responded to the request. The community feedback to callee a about caller b can thus be represented by
$T(a,b) = \Phi(T(u_1,b), T(u_2,b), T(u_3,b)T(u_m,b))$ where $T(u_k,b)$ for $k = 1..m$ represents the trust of member u_k towards caller "b".
The function Φ can be a simple weighted function as below
$T(a,b) = W_1T(u_1,b) + W_2T(u_2,b) + W_3T(u_3,b) + W_mT(u_m,b)$ such that $W_1 + W_2 + W_3 + W_m = 1$.
The weight constants $W_1, W_2, W_3, , W_m$ represent the trust for the callee towards each of the community members.

Property 9: The larger the community, the smaller the impact of an individual. The callee requests the neighbors the trust values they associate for a caller when the callee lacks first hand information about the caller. If the set of callee's neighbors who respond is large, the individual significance of each recommendation decreases.

On the other hand, if the set of neighbors who respond to the request is small, the weighted response of each neighbor has a more influential effect on the callee's decisions.

4.7.1. Increase in Performance of VSD by Integrating Domain Level Knowledge

To measure the increase in performance of VSD by integrating domain level knowledge, we ran the experiment for 1500 calls with 6 users in the VSD_{domain}. VSD logs the spam and legitimate calls to all the callees inside the VSD_{domain}. In addition to computing the callee specific distrust value, the VSD also computes the call's distrust with respect to the callee's community. The VSD then uses both the distrust values for inferring the spam behavior of the call.

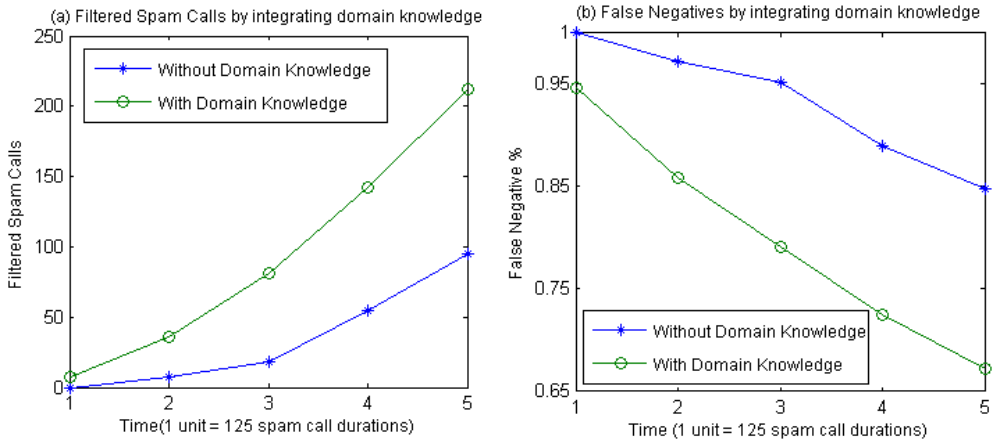

FIGURE 4.19. Increase in filter accuracy by integrating domain level information. The broadcasting mechanism followed by many spammers can be taken advantage of to block spam across the domain. Integrating this knowledge helps in identifying true spam and, therefore, in reducing false negatives

Figure 4.19 represents the increase in the accuracy during the learning period by integrating domain knowledge for a user inside the VSD_{domain}. It can be seen that by integrating the domain level spam information for distrust computation, the increase in performance of the VSD is as high as 100% (Figure 4.19a). In addition, as shown in Figure 4.19b, a marked improvement is observed in the false negative rate by integrating domain knowledge.

As described in the previous sections, VSD learns the spam behavior of the people calling the callees. However, it is possible that when the available information for spam filtering is insufficient, some of the spam calls (3 as proved in 4.4.3.1) are let through the VSD. All the calls that are let through the VSD are analyzed by nuisance detector (ND) as a result of which the number of spam calls reaching the callee decreases further. In addition to spam calls, the callee would not like to receive calls from legitimate callers at a given instant. In this case, the nuisance detector filters unwanted calls from legitimate callers too.

CHAPTER 5

PROACTIVE NUISANCE COMPUTATION

5.1. Introduction

We communicate with people around us in different ways. Prominent among these communication methods are face-to-face interactions, e-mail and telephonic communications. All the three types of communication methods possess favorable and unfavorable facets in their ease of use and comfort to the people communicating. Face-to-face interactions provide a flexibility of real-time communication but require the two parties to be in the same location. While e-mail eliminates geographic limitations, it is not real time. Telephonic conversation solves both the problems of geographic limitations and real-time communication.

With a flexibility of comfort and ease of use, the telephone mode of communication is widely preferred over other communication modes. However, this ease of use in real-time communication brings challenges that are not really pertinent in e-mail communications and face-to-face interactions. One problem that the people using a telephone communication mode experience is eliminating unwanted calls. Initially, unwanted calls were understood to be voice calls from unknown people i.e., strangers. But, over time, the meaning of "unwanted" changed and became subjective e.g., telemarketing calls from valid sales personnel in different corporations and organizations, calls from people calling about previously opted-in services like mortgage, newsletters, discussions etc. People interested in those services viewed them as wanted calls, but those calls are unwanted to others.

While a solution for inferring the associated spam behavior for incoming voice calls on a VoIP network is explained in Chapter 4, there is no reported work on estimating the end user's eagerness or reluctance in receiving the incoming voice call. We believe that each voice call is associated with certain degree of nuisance to the callee. This degree of nuisance increases if the callee does not know the caller. However, voice calls from known people too are associated with some nuisance. In this case, the nuisance depends on the presence of the callee. For example, corporate executives may resent a call from an employee when at home, but would like to receive such calls when at office. As predicted in Canny [7], future communication systems will be context-aware and built based on the end user's behavior, needs and wants. Proactively estimating the nuisance based on the caller's past behavior, the tolerance and presence of the callee helps limit unwanted calls. To realize this objective, we propose a model for computing the nuisance for incoming voice calls to a callee. To test our model, we collected real-time voice calling patters of several individuals at our university. We, then, validated our computational model using the collected calling patterns and individuals' feedback.

5.2. Background

Literature on spam filtering exists for e-mail infrastructures([74], [76], [53], [85], [75], [10], [67]) and VoIP ([71], [83]). But, neither the e-mail nor the VoIP anti-spam solutions advocate the computation of nuisance due to unwanted messages or calls by using end user's behavior and context. Some literature in social science ([30], [49]) discusses the social closeness of people based on amount of time and intensity (frequency) of communication. Granovetter et al [30] suggested that the time spent in a relationship and the intensity along with the intimacy and reciprocal services form a set of indicators for social tie. The paper predicts that the strength of an interpersonal tie is a linear combination of amount of time, the emotional intensity, the intimacy (mutual confiding) and the reciprocal services in a relationship. Marsden et al [49]

evaluates the indicators and predicators of strength (tie) described in Granovetter et al [30]. The paper concludes that "social closeness" or "intensity" provides the best indicator of strength or tie. However, none of these researchers based their discussion of social tie or degree of nuisance on the amount of connectivity occurring within voice calling patterns. Eagle [21] describes how the telephone communication data can be used to infer structural relationships between the subjects in communication. A detailed study of social network dynamics using the telephone communication data is presented. Khalil et al [38] and Khalil et al [39] describe the need of context aware mobile devices for reducing cell-phone interruptions. All these solutions discuss the context awareness of mobile phones. But, none of these solutions advocate the learning of mutual communication behavior between the caller and the callee to infer the callee's interest or reluctance in receiving voice calls from that caller.

We believe that a solution for estimating the callee's reluctance in receiving incoming voice calls depends on the caller's past behavior with the callee and the callee's tolerance based on his mental and physical presence. One way of inferring this reluctance in receiving voice calls is to compute a degree of nuisance associated with them. To our knowledge, no work reports computing the nuisance of a voice call using socio-technical methods. In this chapter, we present a model for estimating the nuisance value of an incoming call by inferring the social closeness between the caller and the callee, caller's past behavior, callee's presence and tolerance. Section 5.3 presents the architecture of a Nuisance Detector (ND) that computes the nuisance of incoming voice calls. Section 5.4.2 describes a model for inferring the social closeness of callers using real-life calling patterns. Section 5.4.3 describes a nuisance computation model based on the inferred closeness and behavioral patterns between the caller and the callee.

5.3. Nuisance Detector

The Nuisance Detector (ND) for computing the nuisance of an incoming voice call can be deployed either in conjunction with perimeter controllers such as voice spam filters or firewalls, or in end systems such as voice phones. The basic architecture of the Nuisance Detector is shown in Figure 5.1.

For every incoming call, the Nuisance Detector computes the nuisance of the

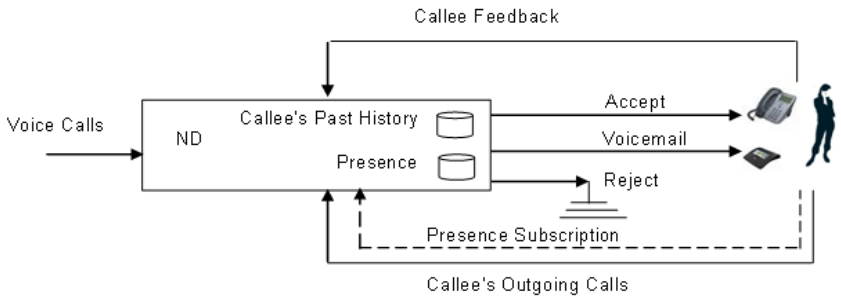

FIGURE 5.1. Architecture of Nuisance Detector (ND). The ND infers the nuisance of incoming calls based on the past call history (incoming and outgoing calls) and the feedback given by the callee to those calls from that caller

call based on the callee's past history with the caller (Callee's Past History) and the previous outgoing calls from the callee to the caller (Callee's Outgoing Calls). Both these histories are maintained by the ND by logging the call specific information for every call received and made by the callee. The computed nuisance is checked with a pre-configured threshold value to make a decision as to whether to forward the call or reject it. When the analysis results in

(1) *Accept*: The call is forwarded to the callee. The callee's phone rings and the callee answers the call. The callee can then press a configured "unwanted-call" button on the IP phone to provide the feedback (the spam button discussed in Chapter 4 can be the unwanted cll button) to the ND about the

nuisance of the call. The ND then logs this feedback to use when inferring the nuisance value for subsequent calls from that caller.

(2) *Forward to voice-mail*: A decision is made to directly forward the call to voice-mail of the callee if the computed nuisance of the call is greater than the threshold configured by the callee. The callee would have an added flexibility of giving a feedback about the nuisance of the calls in his voice-mail box. Similar to the feedback about accepted calls, the ND logs the feedback information to use for subsequent calls from that caller.

(3) *Reject*: A decision is made to reject the call when the caller is in the blacklist of people configured by the callee. In this case, all the calls from the caller are directly blocked unless the callee removes the caller from the blacklist.

The above model of callee giving a feedback to the ND about the nuisance of every call for distinguishing wanted callers from unwanted callers is already demonstrated in Chapter 4. However, it is not always imperative that the callee answers every call and gives a positive or negative feedback. The callee's presence (mood or state of mind) plays a major role in deciding whether to answer the call. For example, to an incoming call, an executive in the middle of a meeting may ignore the call, activate silent mode or forcefully disconnect it. The executive might still like to receive such calls when his presence changes e.g., he is ready to take calls as the meeting has finished. In this case, if the executive doesn't answer the call, the ND should not increase the degree of nuisance for next calls from the caller. Alternatively, possibly the executive doesn't answer the call because it is truly unwanted. We, therefore, require a mechanism that distinguishes the feedback in such ambiguous cases. To eliminate this ambiguity, callee's presence information can be integrated with the ND to reduce untimely calls (presence subscription in Figure 5.1).

5.4. Nuisance Detection Framework

To receive calls just from wanted people at a preferred time, static filtering techniques have to be integrated with behavior learning models. These models should incorporate mechanisms for inferring the callee's presence and tolerance towards the caller, the caller's previous behavior, and, finally, the caller's social closeness to the callee. However, social closeness depends on the other three constructs of callee's tolerance and presence along with the caller's previous behavior.

5.4.1. Presence, Tolerance and Behavior

In our daily life, when we answer a phone call, we always try to estimate the importance of the call before answering it. We base this estimation on

- *Presence*: Our mood or state of mind based on context (situational, spatial, temporal). For example, when we are getting ready for a meeting, we would like to only receive calls of relevance to the meeting. A call from a friend or even from a family member is unwanted at that instant.
- *Caller behavior*: While we would normally receive a friend's call, if that friend makes repeated calls, we usually tend to ignore some of those calls. The caller's behavior influences our decision whether to answer the call or reject it even if the caller is closely related to us.
- *Tolerance*: Every one of us inherently assigns a level of tolerance to every caller we expect to receive calls. This tolerance dictates the extent to which we acknowledge the caller. This level of tolerance is more for our own family members and friends compared to other people like neighbors or distant relatives, i.e. we have more tolerance to behavioral fluctuations to calls from close people such as family members and friends compared to others.

All the above three parameters of caller behavior, the callee's presence and tolerance dictate the social closeness of the caller to the callee. Quantitative measurement of social closeness is a complex and challenging task. There are many works in

the literature in social networks for measurement of social ties ([30], [49]). In addition to presence, expected behavior and tolerance, other parameters such as physical proximity between caller and callee, multiplexity (interaction in multiple contexts), and reputation of caller to the callee could be very useful for further investigations and characterization of social closeness. However we cannot consider all these parameters mainly because they cannot be directly extracted form the calling patterns.

5.4.2. Social Closeness

We in our social life talk to different people at different instances. These people constitute our social network. However, most of the time, we are connected with a small group of individuals within our social network such as family members, friends, neighbors and colleagues. Based on this social connectivity, we can divide our social network members into four broad categories.

- *Socially Close Members*: These are the people with whom we maintain the highest socially connectivity. Most of the calls we receive on a PSTN network (Public switched telephone network) or a cellular network come from individuals within this category. We receive more calls from them and we tend to talk with them for longer periods. We even tolerate abnormal changes in behavior (in the pattern of calling) from this group of people. In essence, these people exhibit least fluctuations in behavior and we have maximum tolerance for them. e.g., family members, friends and colleagues.
- *Socially Near Members*: People in this category are not as highly connected as family members and friends, but when we connect to them, we talk to them for considerably longer periods. Mostly, we observe intermittent frequency of calls from these people. Although we have near to maximum tolerance while we talk to socially near individuals, we seldom acknowledge abnormal changes in behavior from them e.g., neighbors and distant relatives.

- *Opt-in's*: These individuals have less connection with our social life. These people call us with less frequency. We acknowledge them rarely. Among these would be, for example, a newsletter group or a private organization with whom we have previously subscribed. We still receive calls but we do not acknowledge them often and have less tolerance towards their behavior e.g., discussion groups.
- *Opt-out's*: We are least connected to the people belonging to this group. These individuals have no previous interaction or communication with us. We have the least tolerance for calls from them. We find any abnormal behavior from these people extremely unacceptable. e.g., strangers, telemarketers, fund raisers

The classification of social network members into these four categories helps us understand the closeness of those people. However, the closeness of people changes depending upon our presence. Figure 5.2 presents the relation between social closeness, tolerance, and behavior for two temporal presence factors (personal time, professional time).

As shown in Figure 5.2, during non-work hours (personal presence), we have more tolerance and high acceptance in behavior for closely related social network people such as family members and friends. We have high tolerance but a relatively less behavior acceptance for socially near members such as neighbors and acquaintances. Alternatively, we have low tolerance for behavior of other people belonging to the group of Opt-ins and Opt-outs. However, during work hours (professional presence), we prefer to receive calls from secretaries, boss, clients, and customers and may resent unnecessary calls from family members or friends. Clearly, the social closeness and tolerance we have with the people change based on their behavior and our presence. ND can use the presence information of the callee to limit unwanted calls. One way of integrating the presence information of the callee is to synchronize the ND with callee's calendar (scheduled events). This would enable the ND to allow calls from

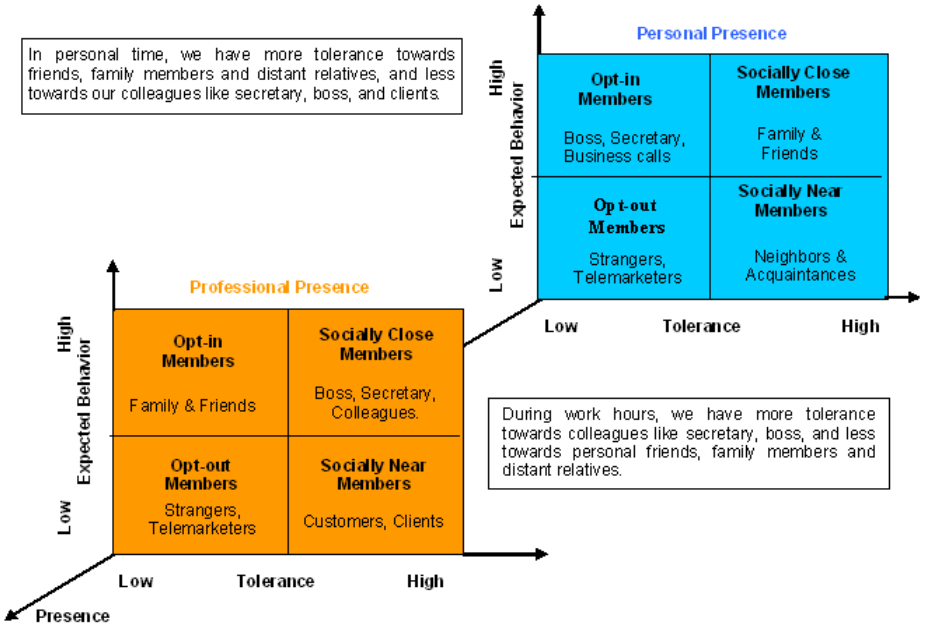

FIGURE 5.2. Relationship between Tolerance, Behavior, Closeness and Presence. Our closeness, tolerance and acceptance in behavior change for different presence factors

family members or friends and give less preference to calls from other people when the calendar indicates personal time. Similarly, when the calendar indicates work hours, ND gives more preference to calls from secretaries, colleagues and less preference for calls from family members and friends. However, in both the cases, ND infers all the factors of behavior, tolerance, closeness and presence to compute nuisance value of incoming calls. In this chapter, we considered one single presence factor of personal time (inferred using personal cellular calling patterns) for computing the nuisance value. For testing this model, we collected data from real-life individuals as discussed in Section 5.4.2.1. In Section 5.4.2.2, we present a model for inferring social closeness of all the callers based on their behavior (frequency of calls to and from the callee) and callee's tolerance (talk-time with the caller). The collected data from real life

individuals helped us validate the relation between the behavior, tolerance and closeness shown in Figure 5.2. In Section 5.4.3, we present a nuisance computation model that takes into account the inferred social closeness and other behavioral patterns between the caller and the callee.

5.4.2.1. *Real-life data sets.* Every day calls on the cellular network include calls from different sections of our social life. We receive calls from family members, friends, supervisors, neighbors, and strangers. Every person exhibits a unique traffic pattern. Calls to our home from neighbors and business associates may not be as frequent as those from family members and friends. Similarly, we talk for longer periods to family members and friends compared to neighbors and distant relatives. These traffic patterns can be analyzed for inferring the closeness to the callee. This closeness represents the social closeness of the callee with the caller on the cellular communication network.

To study closeness the people have with their callers, we collected the calling patterns of 20 individuals at our university. The details of the survey are given in Dantu et al [19]. We found it difficult to collect the data sets because many people are unwilling to give their calling patterns due to privacy issues. Nevertheless, the collected datasets include people with different types of calling patterns and call distributions.

As part of the survey, each individual downloaded two months of telephone call records from his online accounts on the cellular service provider's website. Each call record in the dataset had the 5-tuple information as shown below:

Call record: (date, start time, type, caller id, talk-time) where

date: date of communication

start time: the start time of the communication

type: type of call i.e., Incoming" or Outgoing"

caller id: the caller identifier

talk-time: amount of time spend by caller and the individual during the call

We then used the collected data for deriving the traffic profiles for each caller who called the individuals. To derive the profile, we inferred the frequency (number of calls in unit time interval) and talk-time (total time of communication during that time interval). Figure 5.3 shows an example traffic profile (frequency and talk-time patterns) for one individual communicating with his family member, friend and supervisor. The patterns are plotted with a 1-day time interval for a total of 61 time intervals (2 months).

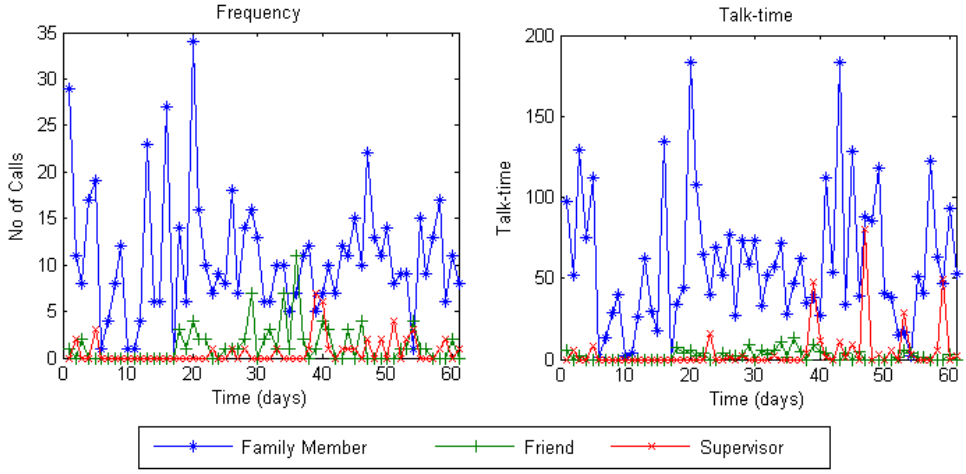

FIGURE 5.3. Calling patterns for 61 days. The calling patterns represent the frequency and talk-time patterns between the callers and one of the 20 individuals who participated in the survey.

Once we derived the traffic profiles of all the callers for each individual, we used these profiles for inferring the closeness of the callers calling them. In this chapter, to discuss the model for inferring social closeness, we present the calling patterns of three individuals that belong to these three types of people who had varied connectivity (call distribution) with their callers.

(1) Least-connected: These people have the least distributed calls from their callers. The majority of their communication occurs with a small number of

people from their closely connected social network members (family members and friends). Their amount of communication with other people is highly limited.

(2) Moderately-connected: These people have moderate call distribution with others on the cellular network. They have reasonable distribution of calls from socially closest members such as family members and friends and not so minimal communication with neighbors and acquaintances.

(3) Highly-connected: These people are highly connected with different social network members on the cellular network. Nearly all the calls to and from these people are highly distributed across types of callers defined before in this section. They have similar distribution of calls towards people such as family members, friends, neighbors and business associates.

To infer the closeness of callers to each individual, we analyzed the incoming and outgoing calling patterns separately. Analyzing incoming and outgoing calling patterns helped us understand the closeness of callers exhibiting different behavioral patterns.

5.4.2.2. *Inferring social closeness from calling patterns.* For inferring the social closeness of the callers based on their calling patterns, we define the parameters such as behavior and tolerance that form the basis for calculating the nuisance of the incoming call. These parameters of behavior and tolerance can be effectively described using the frequency and talk time of calls between the caller and the callee. We can reasonably assume that the higher the frequency of accepted calls from a particular caller, the more likely that the caller is among the callee's closely-connected people. Similarly, lower the frequency of accepted calls from a particular caller, more likely the caller is not among the callee's closely-connected people. However, as we noted earlier, it is also quite possible that some of our socially closest people have least communication with us on the cellular network because they use other modes of communication such as e-mail and face-to-face interactions. Therefore, to effectively

categorize the people in our social network, we computed the expected behavior of a caller based on the coefficient of variation in call frequency normalized by the maximum value (other normalization methods such as z-score have been considered but results were not satisfactory). In addition, we computed the tolerance based on the normalized talk-time spent during the communication. Irrespective of the frequency of calls from callers, a change in call frequency of socially close members is lower (and, therefore, high expected behavior) when compared to socially distant members. Similarly, we normally have high tolerance to family and nearest friends. This can be explained by examining the duration of calls from our telephone bills. We usually expend more talk time for calls with callers belonging to the socially closest and socially near groups. Comparatively, we have lower talk-time (and therefore least tolerance) for phone calls with callers belonging to Opt-in and Opt-out groups.

Closeness of least-connected individual:

Figure 5.4 represents frequency and talk-time patterns for a least connected individual from 3 types of callers in his social network for a total time period of 61 days.

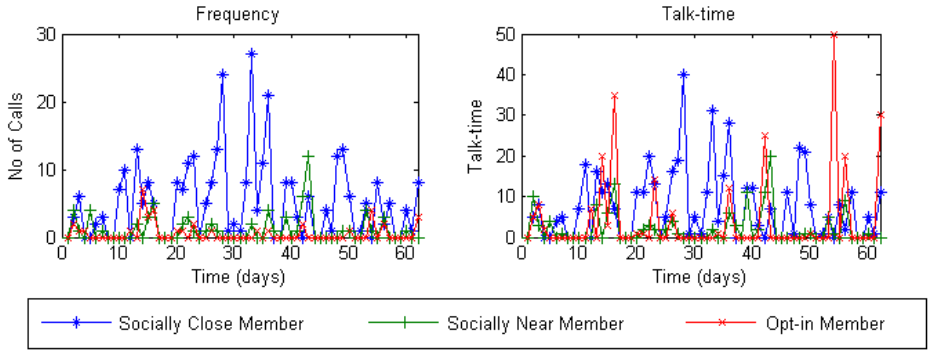

FIGURE 5.4. Calling patterns for a least-connected individual from three types of callers. The individual has considerably more calls from the socially closest member compared to others

From Figure 5.4, the least-connected individual has more calls from a caller belonging to the family and friends group compared to other callers such as neighbors, and acquaintances. By analyzing the complete calling patterns, we inferred that the majority of calls to the least connected individual are from a group of family members and friends. We observed that the individual had relatively less number of calls from other people. Using the above frequency and talk-time patterns separately for incoming and outgoing calls, we computed the expected behavior and tolerance values for every caller. The computed values are then used for positioning those callers into different quadrants. Figure 5.5 describes the grouping of the callers for the least-connected individual based on his incoming" calling patterns. In Figure 5.5, the space position on the 2D plot is based on the computed values of expected behavior and tolerance, and they are colored based on the individual's feedback. From Figure 5.5, it can be inferred that the least-connected individual has few callers with high expected behavior and high tolerance. In addition, the above mechanism for grouping callers into four quadrants can be used for quantizing the social closeness of all callers to the least-connected individual based on incoming calling patterns. We quantitatively define the closeness of the caller to be equal to the distance from the origin to the caller's position on the 2D plot.

$$Social\ Closeness(SC) = Distance\ from\ origin$$

$$Therefore, SC = \sqrt{Tolerance^2 + Expected\ Behavior^2} \tag{12}$$

Similar to the grouping of callers based on incoming calls, we can group the people receiving calls from the least-connected individual i.e., using the outgoing calling patterns of the individual.

Figure 5.6 represents the classification of people receiving calls from the least-connected individual. It can be observed that few people receive calls regularly

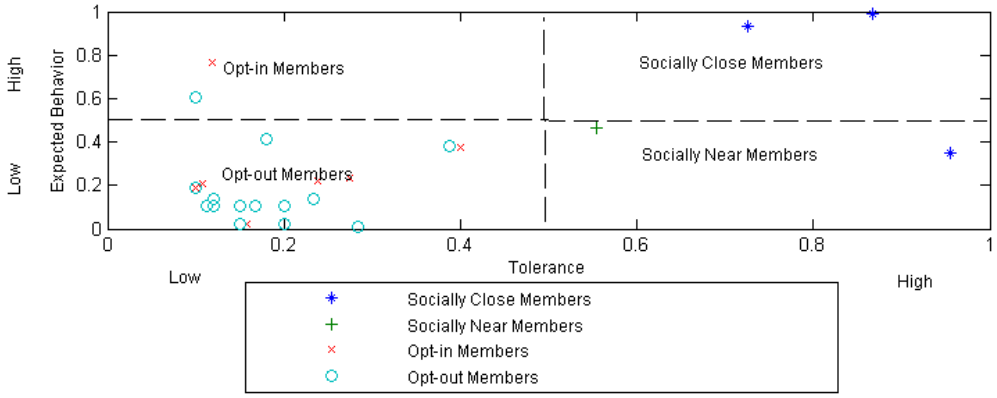

FIGURE 5.5. Grouping of callers for the least-connected individual based on his incoming calling patterns. The individual accepted calls regularly from two callers. Calls to the individual from other people are not regular

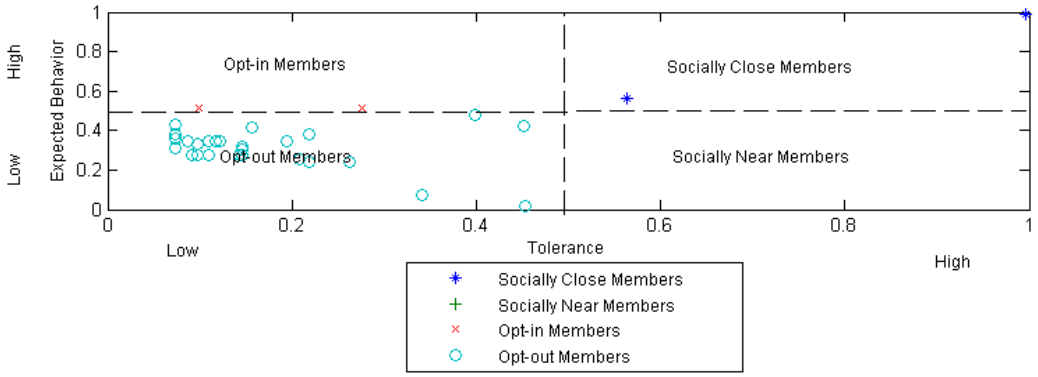

FIGURE 5.6. Grouping of callers for least connected individual based on his outgoing calling patterns. The individual made calls regularly to few people and calls to other people are not regular.

(high expected behavior) and maximum talk-time (high tolerance) from the least-connected individual. Similar to the quantitative measurement for social closeness

based on the incoming call patterns, Figure 5.6 allows us to estimate the social closeness of the people receiving calls from the least-connected individual.

The classification of people based on incoming and outgoing calling patterns can be analyzed for establishing a final classification. This final classification represents the actual closeness in principle between the caller and the callee on the cellular network. We use a simple preference-based selection (as shown in Figure 5.7) that takes into account the nearest social closeness classification among the incoming and outgoing classification mechanisms.

The selection process can be defined as follows:

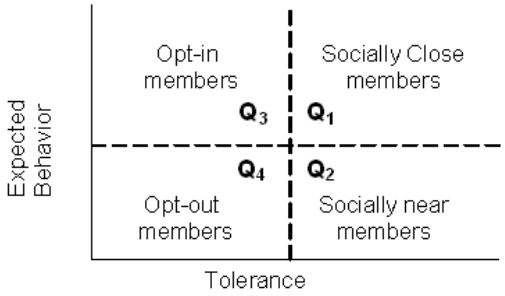

FIGURE 5.7. Grouping of callers for least connected individual based on his outgoing calling patterns. The individual made calls regularly to few people and calls to other people are not regular.

Preference based selection: For all callers C_i for i=1..n to a given callee, if $C_i \in Q_s$ based on incoming classification and $C_i \in Q_t$ based on outgoing classification for s,t = 1.. 4 in Figure 5.7, then $C_i \in Q_{min(s,t)}$ in the final classification.

We used the above selection mechanism for grouping of people communicating with the least-connected individual. Figure 5.8 represents the final classification of the callers into four different quadrants for the least-connected individual.

To this point we have focused on the classification of a least-connected individual. To demonstrate how this classification mechanism allows us to generate

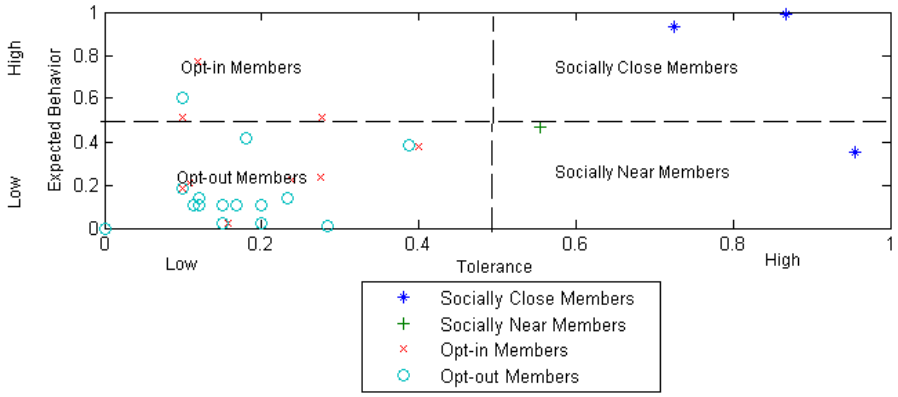

FIGURE 5.8. The classification of callers for the least-connected individual based on his incoming and outgoing calling patterns.

effective classifications, we briefly present the final classification for a moderately-connected and highly-connected individual.

Closeness of a moderately connected individual:

Figure 5.9(a) presents the frequency and talk-time plots for an individual who is moderately-connected with his social network members. An analysis of the calling patterns of this individual determined that he had moderate call distribution with his family members, friends and other socially connected people like neighbors, acquaintances and opt-in callers. However, this individual had relatively more number of calls distributed across different groups compared to that of the least-connected individual. We validate this call distribution across different groups by plotting the expected behavior with tolerance and use the feedback given by the actual individual.

Figure 5.9(b) represents the grouping of the people making and/or receiving calls to/from the moderately-connected individual. Note that the maximum distribution of calls and talk time is from his family members and this individual has the least tolerance for Opt-out callers (like telemarketers). Unlike the least-connected individual, it can be observed that the moderately-connected individual had more

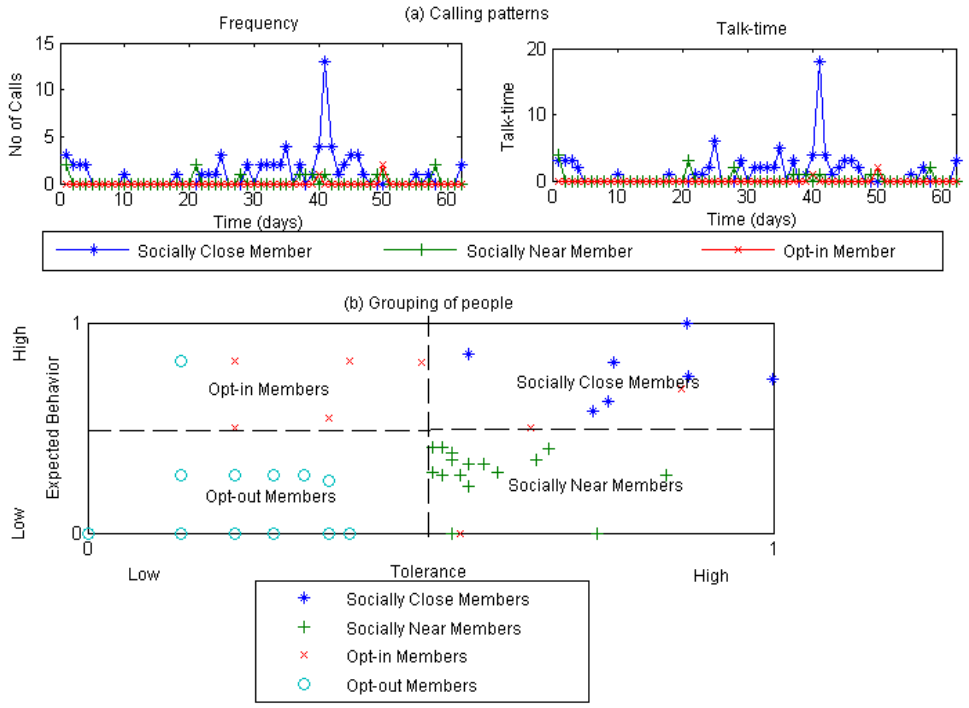

FIGURE 5.9. The classification of people communicating with the moderately-connected individual based on his incoming and outgoing calling patterns.

people distributed across the four quadrants.

Closeness of a highly connected individual:

Figure 5.10(a) represents the frequency and talk-time plots of a highly-connected individual for three of his callers belonging to different groups. It can be observed that the frequency of calling from a caller belonging to the group of neighbors and acquaintances is comparable to the frequency of calling from a family member. Figure 5.10(b) represents the final classification of the people making and/or receiving

calls to/from the highly-connected individual. Clearly, this individual is socially well-connected on the cellular network with considerable number of people with similar calling distribution across four quadrants.

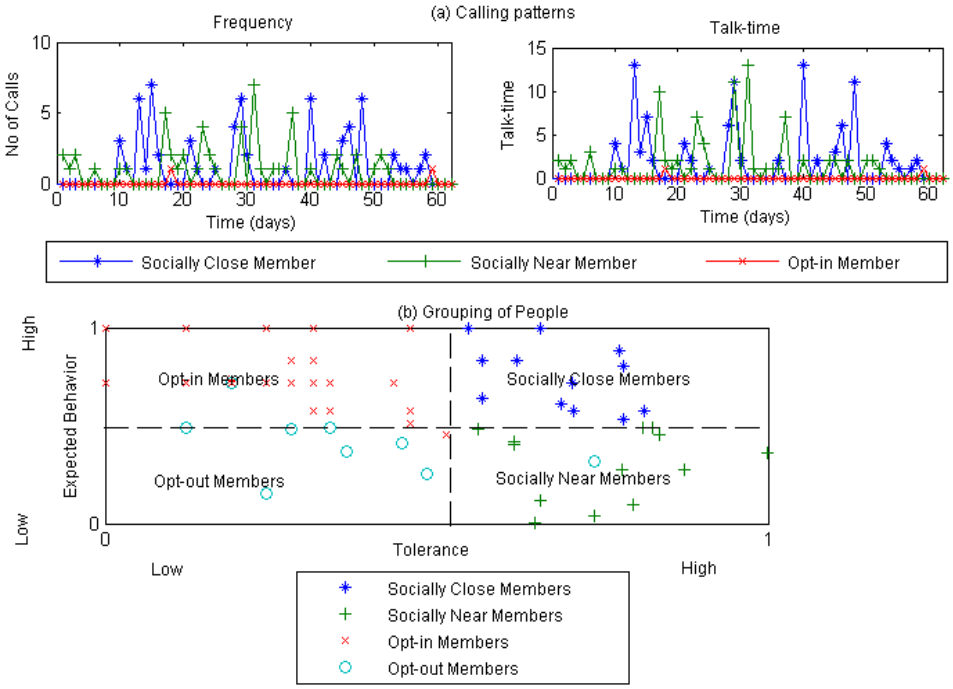

FIGURE 5.10. The final classification of callers for the highly connected individual based on his incoming and outgoing calling patterns.

We validated the inferred social closeness of all the people using the feedback from the individuals.

Validation of results: We interviewed the individuals about their social association with the people they communicate on the cellular network. During the interview process, they identified the family members, friends, neighbors, acquaintances, and business associates. Here, we present the comparison of our inferred social closeness and perceived closeness (based on the feedback) of the three individuals described in Section 5.4.2.2. Table 5.1 compares results from our analysis with their feedback. For

example, our analysis for the highly-connected individual resulted in a classification of 12 socially closest members, 15 socially near members, 31 socially distant members, and 7 socially farthest members. But, the highly-connected individual responded with a feedback that he talked to 12 socially closest members, 13 socially near members, 32 Opt-in members, and 8 Opt-out members.

We noticed some errors in our analysis, but we attribute these errors to two factors:

TABLE 5.1. Validating computed closeness with perceived closeness of the individuals.

	Least Connected Individual		Moderately Connected Individual		Highly Connected Individual	
Group	Our Analysis	Individual's Feedback	Our Analysis	Individual's Feedback	Our Analysis	Individual's Feedback%
Socially Close	2	3	8	7	12	13
Socially Near	2	1	19	17	15	13
Opt-ins	4	9	7	10	31	32
Opt-outs	44	39	24	24	7	8
Error(%)	9.6		5.0		3.0	

(1) *Use of other communication means*: Personally close members might not always be the socially closest on the cellular network. They may be communicating using other modes such as face-to-face or e-mail. We have restricted our classification to the calling patterns observed on the cellular network. It does not necessarily depict the actual personal closeness the individuals may have with their callers.

(2) *Unavailability of sufficient calling patterns*: Our analysis is based on the calling patterns for a period of 2 months. However, using calling patterns for an extended period of time considerably increases the social network inference.

Accuracy in inferring Social Closeness is more with calling patterns for an extended period of time:

With the availability of calling patterns for an extended period of time, our model for inferring social closeness based on frequency and talk-time patterns showed a marked improvement in accuracy. To determine the improvement in accuracy, we requested the least-connected individual to provide us with an additional month of phone records (for a total period of 3 months). With this additional information, we again analyzed the least-connected individual's calling patterns. Then, we plotted the final classification of all the people communicating with the least-connected individual as shown in Figure 5.11.

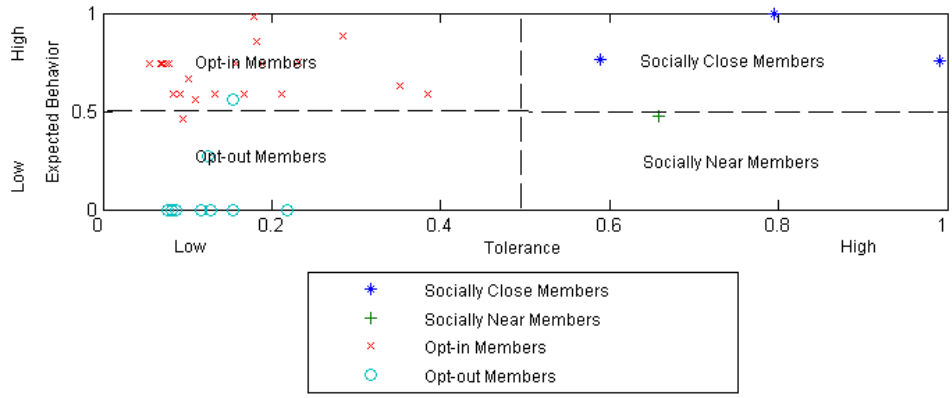

FIGURE 5.11. Grouping of callers for the least-connected individual after the end of 3-month period. The social closeness inferred using three months of data showed a marked improvement compared to the social closeness inferred using only 2 months of calling data.

To validate the results, we again interviewed the individual about the social closeness of all the callers after the end of third month. Table 5.2 presents a comparison of the inferred social closeness with the perceived closeness of the least-connected individual for a total time of 2 and 3 months respectively. It can be observed that

the error in the classification based on the computed social closeness using calling patterns has reduced from 9.6% to 3.8% when we have more available data. It can also be observed that the number of people in the groups has changed from the end of second month to the end of the third month. This is possible because callers exhibit different frequency patterns. There might be callers who called infrequently during the first two months but the number of their calls increased in the third month and vice versa. We have validated the movement of callers among different groups with the least-connected individual. At any instant, the snapshot of closeness based on frequency and talk-time analysis represents the closeness callers have with the individual on the communication network.

TABLE 5.2. The accuracy in inferring the social closeness is more with available calling patterns for an extended period of time. The accuracy of inferring social closeness improved for least-connected individual when we had calling patterns for extended time.

	2 months of calling patterns		3 months of calling patterns	
Group	Our Analysis	Individual's Feedback	Our Analysis	Individual's Feedback
Socially Close	2	3	3	3
Socially Near	2	1	1	1
Opt-in's	4	9	20	21
Opt-out's	44	39	28	27
Error(%)	9.6		3.8	

Our model for inferring the social closeness of people making and/or receiving calls using the frequency (intensity) and talk-time (time spent in a relationship) can be used for computing the nuisance value of incoming calls. As proposed in Granovetter et al [30], a relationship's strength depends on intensity, time spent in the

relationship, the intimacy and reciprocal services. In this chapter we define a nuisance model that takes into account the frequency and talk-time patterns, social closeness, periodicity, and reciprocity (reciprocal services shown by the callee for previous calls) with the caller.

5.4.3. Nuisance Computation

Nuisance can be defined as an unwanted activity. In case of voice calls this refers to receiving unwanted calls. As described in Section 5.4.1, nuisance depends on behavior, tolerance, presence, and closeness. These parameters of behavior, tolerance, presence and closeness can be used to infer different behavioral parameters that affect the nuisance. These behavioral parameters, defined next, are: (i) Unwantedness (U), (ii) Wantedness (W), (iii) Presence (PRES), (iv) Periodicity (PER), and (v) Reciprocity (REC). Therefore, nuisance can be defined as a function F of these parameters (η=F(U,W,PRES,PER,REC)).

a) The unwantedness or unwillingness (U) based on the "incoming" call patterns. Nuisance is directly proportional to the unwantedness shown by the callee in receiving voice calls; i.e., nuisance $\eta \propto U$ where U represents the unwantedness. This unwantedness in receiving voice calls is inversely proportional to

   (i) Expected Behavior based on "incoming" calling patterns (EB_I). This expected behavior is inferred using the frequency patterns as discussed in Section 5.4.2.2.

(13) $$U \propto \frac{1}{EB_I}$$

   (ii) Tolerance based on "incoming" calling patterns (T_I). The tolerance level for the caller can be inferred using incoming talk-time patterns of the callee as discussed in Section 5.4.2.2.

(14) $$U \propto \frac{1}{T_I}$$

(iii) Social Closeness based on "incoming" calling patterns (SC_I). The higher the social closeness, the lower is the unwantedness and vice versa.

(15) $$U \propto \frac{1}{SC_I}$$

However, $SC_I = \sqrt{T_I^2 + EB_I^2}$ (see Section 5.4.2.2) where EB_I and T_I represent the expected behavior and tolerance on the "incoming" calling patterns respectively.

(16) $$U \propto \frac{1}{\sqrt{T_I^2 + EB_I^2}}$$

From Equations (13), (14), and (16), we have

$U \propto \frac{1}{T_I EB_I \sqrt{T_I^2+EB_I^2}}$

(17) $$\eta \propto \frac{1}{T_I EB_I \sqrt{T_I^2 + EB_I^2}}$$

b) The wantedness (W) to communicate based on the "outgoing" call patterns. The higher the wantedness shown by the callee in communicating to the caller, the lower is the nuisance and vice versa i.e., $\eta \propto -W$. This wantedness in receiving voice calls is directly proportional to

(i) Expected Behavior based on "outgoing" calling patterns (EB_O). The higher the expected behavior, the higher is wantedness and vice versa.

(18) $$W \propto EB_O$$

(ii) Tolerance based on "outgoing" calling patterns (T_O). The higher the tolerance, the higher is the wantedness and vice versa.

(19) $$W \propto T_O$$

(iii) Social closeness based on "outgoing" call patterns (SC_O). The higher the social closeness on the outgoing calling patterns, the higher is the wantedness and lower the nuisance and vice versa. Therefore $W \propto SC_O$

Similar to the equation deduction shown in step 1, $SC_O = \sqrt{T_O^2 + EB_O^2}$

where EB_O and T_O represent the expected behavior and tolerance on the "outgoing" calling patterns respectively.

(20) $$Therefore\ W \propto \sqrt{T_O^2 + EB_O^2}$$

From Equations (18), (19), and (20), we have

$W \propto T_O EB_O \sqrt{T_O^2 + EB_O^2}$

(21) $$\eta \propto -T_O EB_O \sqrt{T_O^2 + EB_O^2}$$

c) Presence (PRES): Presence depends on the callee's immediate context. One aspect of presence (mood information) can be extracted directly from the calling patterns. This mood information can be inferred by analyzing the calling patterns to see the number of calls accepted by the callee during a given time period. Nuisance is inversely proportional to elapsed time since the last call i.e., the smaller the time since last call, the higher is the nuisance and vice versa. The nuisance in this case is a function of both the factors of time since the last call from the actual caller (τ_1) and the time since the last call from any caller (τ_2). Every day, we base our decision to answer a call on the above two time factors.

(22) $$PRES_1 \propto \frac{1}{\tau_1}\ and\ PRES_2 \propto \frac{1}{\tau_2}$$

d) Periodicity of the caller(PER): Periodicity is the regularity in which the callee has accepted calls from a caller. We can derive the periodicity information from the calling patterns by analyzing the accepted calls by the callee in a given time interval (such as 15 minutes, 1 hour or 1 day). By inferring the periodicity during the time interval of the present call, the nuisance of the call is updated. Nuisance decreases with increasing periodicity. The higher the periodicity, the lower is the nuisance and vice versa.

(23) $$PER \propto P(t_P)$$

where t_P is the time interval for which the periodicity is sought.

e) Reciprocity shown by the callee to the calls from the caller(REC: Reciprocity signifies the callees eagerness in responding to the previous calls initiated by that caller. Nuisance decreases with increasing reciprocity. The higher the reciprocity, the lower is the nuisance and vice versa. This reciprocity information can be deduced from the average or the most probable (median) response time to a call from the caller in the past. Reciprocity is inversely proportional to the response time i.e., the longer the time for response from the callee to the caller, the lower is the reciprocity and vice versa.

$$REC \propto \frac{1}{\lambda(t_R)} \tag{24}$$

where λ is the average or median response time by the callee to a call from the caller in the time window t_R.

From Equations (17), (21), (22), (23), and (24) we can infer the final nuisance equation to be

$$\eta = W_I \frac{1}{T_I EB_I \sqrt{T_I^2 + EB_I^2}} - W_O T_O EB_O \sqrt{T_O^2 + EB_O^2} + W_{P,1} \frac{1}{\tau_1} + W_{P,2} \frac{1}{\tau_2} - W_{PE} P(t_P) - W_R \frac{1}{\lambda(t_R)}$$

where $W_I, W_O, W_{P,1}, W_{P,2}, W_{PE}, \textit{ and } W_R$ are the weight constants for Unwantedness, Wantedness, Presence based on last call from the caller, Presence based on last call from any caller, Periodicity and Reciprocity respectively. This equation allows us to plot the nuisance value over a specified time period. To present the accuracy of the nuisance computation model, we chose the least-connected individual (discussed in Section 5.4.2.2) to be a representative of the all the individuals. Figure 5.12 represents the nuisance value for one caller in each of the quadrants Q1, Q2 and Q3 for the least-connected individual. The nuisance is plotted for a total period of 61 days based on frequency and talk-time patterns.

It can be observed from Figure 5.12 that the nuisance value for a call from a closely-connected caller is lower than the nuisance posed by a caller who is

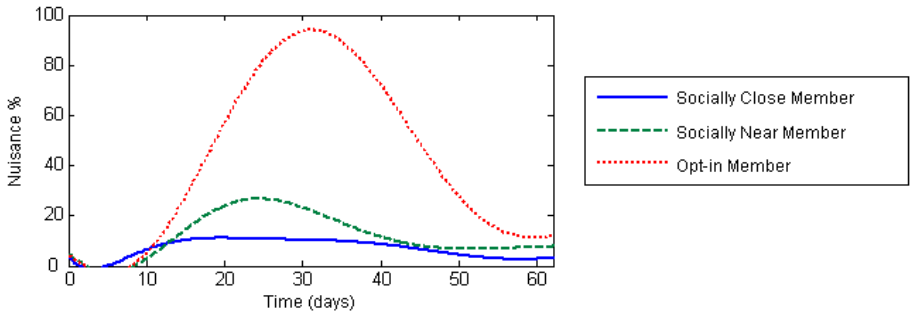

FIGURE 5.12. Computed nuisance for three different types of callers for the least connected individual during the two month period.

relatively distant. In addition to the nuisance value for three callers belonging to three different quadrants, we present nuisance plots for three socially close callers of the least-connected individual as shown in Figure 5.13. The plots represent the nuisance for a period of 61 days (2 months).

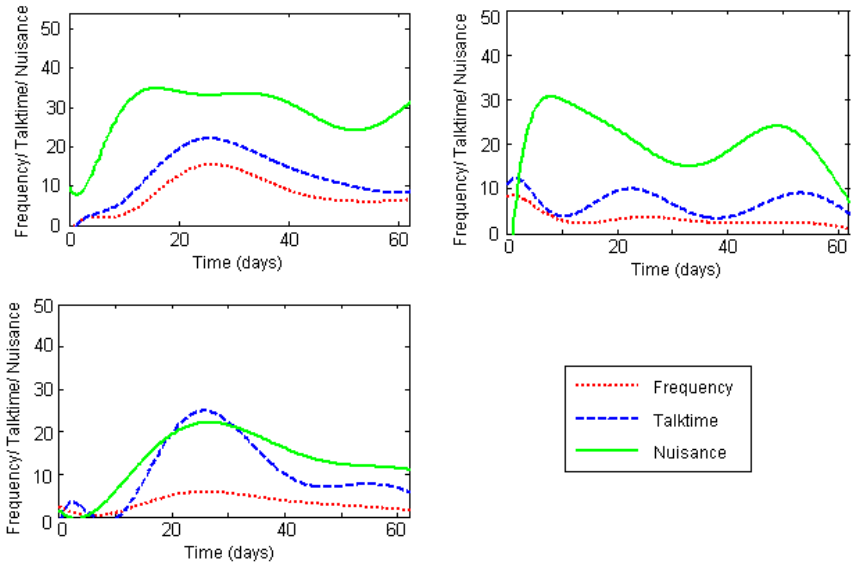

FIGURE 5.13. Nuisance for calls from three socially close callers to the least-connected individual.

We used the above model for inferring the nuisance of calls to all the individuals who gave us the data. To validate the nuisance plots, we showed the frequency and talk-time patterns from three of their socially close callers to the individuals and asked them to rate their perceived nuisance on a scale of Low, Medium Low, Medium, Medium High, and High. We also gave them an option to rate the nuisance to be "Not Sure.". The individuals were asked to rate the nuisance for an interval period each of 1 week. Based on the above directions, the individuals rated the nuisance values on the above scale over a period of 2 months (about 9 weeks). Here we present the validation of our computed nuisance for three individuals (least-connected, moderately-connected, and highly-connected individuals discussed in Section 5.4.2.2) based on their feedback in Table 5.3.

TABLE 5.3. Validating computed nuisance with perceived nuisance of individuals. For a period of 9 weeks, three individuals (least-connected, moderately-connected and highly connected) rated their nuisance for three of their socially closest callers. "Hit" represents a match of our computed nuisance with their perceived nuisance and a "Fail" represents a mismatch between our computed nuisance and their perceived nuisance

	Least Connected Individual			Moderately Connected Individual			Highly Connected Individual		
Caller	Hit	Fail	Unsure	Hit	Fail	Unsure	Hit	Fail	Unsure
1	7	1	1	8	1	10	8	1	10
2	7	2	0	9	0	0	8	1	10
3	8	0	1	8	1	0	9	0	0
Error(%)	11.1			7.4			7.4		

We have examined the feedback from all the individuals about their perceived nuisance for their callers. Based on their feedback, we observed the minimum

and maximum accuracy of our nuisance model to be at 88% and 95% respectively.

We have shown the application of the nuisance model with real-life voice call data on the cellular network. We believe that the real advantage of the nuisance model can be taken for reducing unwanted voice/video calls on the open Internet.

5.4.4. Applicability to Voice over IP

Using a VoIP service (lot cheaper than landline and wireless cellular services), a person can call any other person having any of the three voice services (landline, cellular, VoIP). This advantage of cheaper call rates and flexibility in use brings forward lot more issues and challenges that have to be addressed before complete deployment of VoIP systems. One major issue that hinders a successful deployment of VoIP is VoIP spam commonly referred to as SPIT (Spam over Internet Telephony). Internet being open and free, it is easier to create and transmit SPIT traffic towards the VoIP infrastructure. An analysis [71] indicates that IP-based SPIT is roughly three orders of magnitude cheaper to send than traditional circuit-based telemarketer calls. Therefore, computing nuisance for voice calls using VoIP is far more necessary compared to other voice communication medium such as cellular, landline. For realizing this objective, the calculated spam level of the call (discussed in Chapter 4) can be integrated with the nuisance computation model described in this chapter. As shown in Figure 5.14, VSD analyzes the incoming call for its trustworthiness based on the callee's preferences. While the untrusted calls are dropped, the trusted calls are analyzed by the Nuisance Detector to infer the nuisance. In addition, the initial three spam calls required to learn the behavior of a spammer by VSD could be directly filtered by the Nuisance Detector thereby reducing the false negatives. In brief, the social authenticity of the caller is established by the trust and reputation relationships as specified in Chapter 4, and proactive behavior prediction (using closeness and nuisance computations) is achieved based on the previous history of calling patterns with the caller using the Nuisance Detector. This approach would provide a complete solution for deciding whether to allow a call to be forwarded to the callee. By using

factors such as previous behavior, tolerance, and presence, this pro-active approach results in the phone learning over a period of time the wantedness of calls from a caller to a callee. In essence, the solution would enable the phone to ring only when the callee would have liked to receive the call.

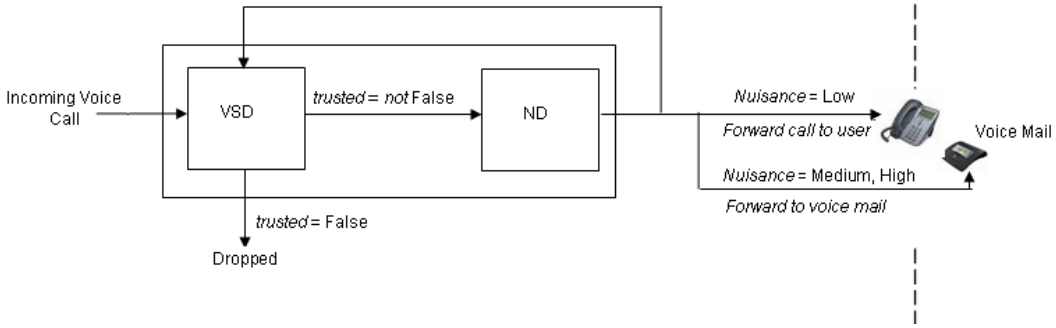

FIGURE 5.14. A VoIP specific solution for integrating Nuisance Detector with a VoIP spam filter for filtering unwanted calls in addition to proactively inferring the imminent need to forward the call to the callee based on past mutual behavior and present context of the callee

Integrating VSD and ND into a single filtering process would limit more number of unwanted calls reaching the callee than VSD or ND could attain individually. The VSD and ND can compute the probability of the call to be spam and unwanted respectively. The computed probabilities are compared to predetermined threshold values to make a decision whether to forward the call to the callee or quarantine it. Frequently it is observed that the filter administrators configure these threshold values who have minimum knowledge about the callees' preferences. The configured thresholds play an important role in the decision making process, and have to be dynamically updated for fast and adaptive learning of caller and callee behavior.

CHAPTER 6

AUTOMATIC CALIBRATION USING RECEIVER OPERATING CHARACTERISTICS CURVES

6.1. Introduction

With increasing number of applications providing value added services, system administrators find it increasingly difficult to maintain a network infrastructure's inherent security. In response, developers are defining security defenses to protect applications and the infrastructure providing those application services. One such line of defense is an application-level filter. Application filters are used for quarantining unwanted traffic or requests in types of traffic such as HTTP (Hypertext Transfer Protocol), SMTP (Simple Mail Transfer protocol), and SIP (Session Initiation Protocol) traffic. Depending on the type of application, the respective filter needs to employ different filtering techniques for quarantining unwanted traffic. These filtering techniques can be broadly classified into two types.

- Signature-based filtering: These filtering mechanisms implement static checking of incoming traffic for patterns that do not change over a specific time period. Therefore, the filter just needs a static collection of signatures it can use for detecting anomalies. For example, an HTTP application filter can use a compilation of virus signatures to check for traces of virus in the incoming HTTP traffic. The challenge in such filtering techniques is that the

filter requires constant updating with the signatures of new viruses. However, with the number of viruses or worms increasing every day, these filtering techniques are inadequate.

- Dynamic Behavior-based filtering: With these filtering mechanisms, the filter observes abnormal changes in behavior over a period of time and makes a decision to forward or quarantine the request. This mechanism is particularly useful when behavioral changes exist in one or more entities that participate in the traffic. For example, an e-mail filter checks for the spam nature associated with an incoming e-mail message by analyzing the behavior of entities such as the e-mail sender and the e-mail forwarding SMTP proxies. When the cumulative spam behavior of all the participating entities exceeds a predetermined threshold, the filter blocks the e-mail, sending it to the intended recipient's junk folder.

Many of the existent application level filters use these dynamic behavior filtering mechanisms. The filter administrators configure the tolerance limits that are generally referred to as threshold values. The filter uses these thresholds to infer the end user's tolerance in accepting the unwanted behavior exhibited by different participating entities. Many a times, incompetent filter administrators configure the filter with inappropriate threshold values because of a multitude of reasons. These reasons range from the administrators being unaware of optimum threshold values to constantly changing end user preferences. This results in a decrease in the accuracy of the filter. However, most filter administrators conduct due diligence for inferring the optimum threshold value using various techniques. One such technique widely used by the filter administrators to analyze filter accuracy and determine optimum threshold is ROC (Receiver Operating Characteristics).

Traditionally, ROC has been used to analyze filter performance. Filter accuracy is determined by comparing false positives (messages mistakenly classified as unwanted) with true positives (genuinely unwanted messages). ROC uses both the

filter classification and the true Classification feedback provided by the end user (or the user personalized database) to determine which classifications are false positives. ROC then analyzes these false positives and true positives to calculate an optimum threshold value that the filter administrator can use to configure the filter. The optimum threshold may be used for some time (usually until the accuracy decreases) before an administrator repeats the manual threshold update. During this interval, when the threshold is fixed, it is highly possible that many entities' behavior changes from wanted to unwanted and vice versa. Therefore, even though the filter administrators configure the threshold value at regular intervals, problems with mistaken classification remain because of dynamic behavior changes.

We believe that because mistaken classification decreases a filter's accuracy significantly, we require a filter that can learn and adjust quickly to the participating entities' changing behaviors. To achieve this objective, we propose a model for automatic calibration, using ROC and end-user feedback to converge quickly on the optimum threshold.

6.2. Background

A number of researchers have studied Receiver Operating Characteristics curves [23][94][5][31]. Fawcett [23] details the basic understanding of ROC curves. In addition, Fawcett lists performance measuring algorithms that can be used for determining a classifier's accuracy by plotting the ROC curves. The classifier analysis provided by the proposed algorithms can also be used to configure application parameters for optimum performance. Wright[94] justifies use of ROC in finding a filter's optimal threshold in SDT (Signal Detection Theory). This paper suggests that ROC offers a principle graphical device for analyzing filter accuracy. The paper further supports the assertion with a brief discussion of ROC curves and how the optimum decision criterion can be reached. Bradley[5] investigates the area beneath the ROC curves as a performance measure for machine-learning algorithms using real-world medical diagnostic data. The paper compares the results of this study with that of other

conventional methods to show the effectiveness of ROC analysis. Hanley [31] explores the possible relation between the areas under the ROC curve with application performance in the field of medical diagnosis. The paper specifies that the area under the ROC curves can be an effective mechanism for measuring system accuracy. The paper supports this by comparing the analysis using area under ROC with a well-established non-parametric Wilcoxon statistic [[84], [92]]. Each of these methods provides a solution for non real-time evaluation of system performance using the ROC curves. We believe that the non real-time solution for analyzing system accuracy can be automated for real-time feedback. This real-time feedback can then be used to set filter parameters so that the filter learns quickly and adjusts itself accordingly.

6.3. Receiver Operating Characteristics Curves

ROC curves use true classification based on a real end user's feedback to estimate false and true positives during classification. The inferred accuracy helps an administrator to set a suitable threshold value that minimizes false alarms. Figure 6.1 shows a basic example of a non real-time ROC analysis process, highlighting the sequential transactions that take place when computing the optimum threshold value using the non real-time ROC.

(1) The application filter receives a request for service. The filter processes the request using available knowledge based on previous learning by computing a score value (S) for determining the request to be unwanted. Based on the score value and previously configured threshold value, the filter decides whether to acknowledge the request or not.

(2) The filter then informs the end user about the classification decision (for an e-mail application, the spam filter forwards genuine messages to the Inbox and spam messages to the Junk folder).

(3) The application filter logs the score value in a predefined database.

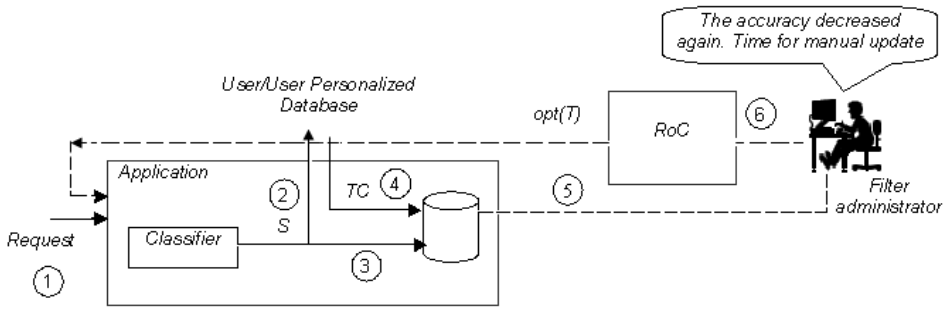

FIGURE 6.1. Non real-time analysis using ROC curves for an application classifier. Analyzing the incoming request, the classifier computes the score (S) for the request. Using a pre-configured threshold for the score (S) to signal if the message is unwanted, the filter decides whether to forward or reject the request and makes the end user aware of the decision. The end user gives feedback (True Classification (TC)) about the request which the filter logs in addition to its classification. When the filter performance falls below the desired accuracy, the administrator runs the threshold update algorithm for optimum threshold ($opt(T)$) to increase the filter's the accuracy.

(4) The end user then gives feedback (either explicitly by letting the filter knows about the true classification or implicitly when the user is satisfied with the filter classification). The application logs the true classification information with the score value generated by the filter.

(5) The filter administrator analyzes the filter's accuracy at regular intervals.

(6) If satisfied with the accuracy, the administrator does not change the filter threshold. However, when the administrator finds an unacceptable decrease in accuracy when using the preconfigured threshold, the administrator invokes ROC to compute a new optimum threshold.

(7) The administrator manually feeds the optimum threshold to the filter.

This series of transactions occurs every time the administrator updates the filter with a new optimum threshold. This analysis of filter accuracy using ROC does reduce the number of false alarms. However, the reduction is not considerable when participating entities change their behavior frequently. Also, manually updating the thresholds using ROC proves to be cumbersome, particularly when it has to be done frequently. Therefore, we require an automated calibration mechanism that adjusts to participating entities' changing behavior in real time.

6.3.1. Need for Automatic Calibration

Different filters use different criterion to optimize. This criterion depends on the underlying application's sensitivity, the end-user's preferences, such as mood, and on presence (situational, temporal, and spatial). For example, a non real-time e-mail filter analyzes the incoming e-mail traffic for spam messages and attempts to minimize false alarms. Even if the filter fails to correctly classify an e-mail message, the nuisance created because of misclassification is negligible. However, if the underlying application is a voice-call filter, the analysis for unwanted calls must be conducted in real time. Unlike with e-mail messages, nuisance because of improperly classified voice calls is high. Consider, for example, the nuisance felt by an executive in a meeting (false negative) who receives a junk voice call, or a husband waiting to hear from his hospitalized wife who never receives the call (false positive) because the filter has quarantined it. Therefore, in these cases, we need an adaptive voice-application filter that adjusts automatically to the caller's changing behavior and end user's preferences. Although some adaptive filters do analyze the service-requesting user's behavior, we believe that addition of automatic calibration of a filter's threshold can be used to integrate the end user's threshold preferences, resulting in a more accurate filter.

6.3.2. Automatic Threshold Calibration Complements an Adaptive Filter

Many adaptive filters learn and update themselves based on the behavior of service-requesting entities. But, none advocate automatic learning and update of

thresholds based on end user's preferences. These adaptive filters are usually configured with a static threshold for identifying unwanted behavior. This static threshold is updated only when classification accuracy decreases. However, using an automatic-calibration mechanism, we can adaptively calibrate thresholds to accurately reflect the end users' tolerance, presence, and other preferences. Therefore, the automatic calibration process can be used as a complementary solution to make a filter more adaptive. This adaptive learning of the end user's preferences can be coupled with the learning about the service-requesting entities (as normal adaptive filters do) to result in a complete solution for limiting unwanted requests.

6.4. Automatic Calibration using RoC

An automatic-calibration mechanism that infers the optimum threshold value using ROC analysis after every request is shown in Figure 6.2. The figure represents the sequential transactions that take place when the filter uses adaptive calibration mechanism.

(1) The application filter receives a request for service. Using available knowledge, the filter computes a score value S for the request to be unwanted. Based on a predetermined threshold value and computed score, the filter makes a decision whether to acknowledge the request or not.
(2) The filter then informs the end user about the classification decision.
(3) The application filter logs the score value in a predefined database.
(4) The end user gives feedback about the validity of the request. The application logs the true classification information with the score value generated by the filter.
(5) Using the end user' feedback, the application's integrated ROC module re-computes the optimum threshold. This re-computation results in a slight

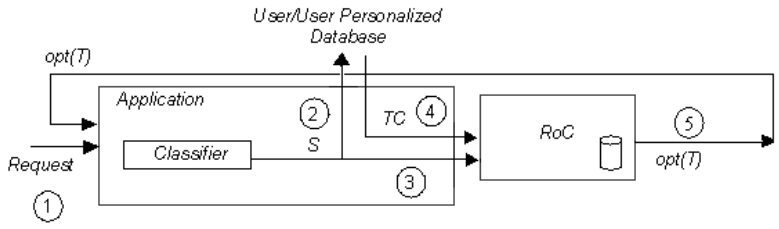

FIGURE 6.2. Automated calibration of filter threshold using real-time analysis of filter performance using ROC curves. The inbuilt ROC module in the application re-computes the optimum threshold value after the completion of every request, based on end user's feedback. The computed optimum threshold is used as the filter threshold for the next incoming request. This eliminates the manual updated process by the administrators as the filter adjusts quickly and optimizes its performance.

threshold increase when a just completed request is classified as a false positive. Alternatively, the re-computation results in a slight threshold decrease when the just completed request is classified as a false negative. Once computed, the new threshold value is fed back to the filter. The application uses the new threshold to categorize the next incoming request as either wanted or unwanted. Unlike the manual update of the filter's threshold, the automatic calibration enables the filter to be updated with the optimum threshold for every service request.

With considerable learning, the filter converges on an optimum threshold irrespective of the initially configured threshold, i.e., the filter accuracy is least dependent on the initial threshold value. Thus, this real-time analysis of ROC after the end of every request enables the filter for fast and adaptive learning during dynamic behavior changes. We tested the automatic-calibration mechanism with a VoIP spam filter (VSD discussed in Chapter 4)and describe our results in Section 6.5.

6.5. VoIP Spam Filtering using Automatic Calibration

To demonstrate the applicability of automatic calibration in a real-time scenario, we integrated the scenario it VSD. The interface diagram for this intgration is shown in figure 6.3.

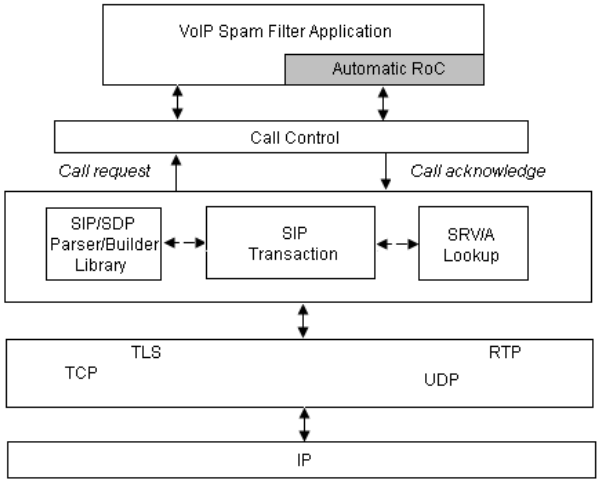

FIGURE 6.3. The interface diagram for integrating the automatic ROC calibration mechanism into a SIP application (a VoIP spam filter) for filtering voice calls.

6.5.1. Automatic Threshold Calibration is Not a Bottleneck for VoIP Spam Filter Performance

The threshold-update process using the automatic-calibration mechanism is conducted real-time using the end user's feedback either during the call or after the call is terminated. Therefore, there is nor setup delay during the call generation process.

6.5.2. VoIP Spam Filtering is Threshold Dependent

VSD computes the probability of the incoming call to be spam, and compares the probability with a threshold value configured by the VSD administrator. The configured threshold represents the VSD's sensitivity. This threshold is to be appropriately configured when analyzing real-time voice calls. In context of dynamically

changing callers' behavior and the end user's preferences, it becomes more necessary to use dynamically changing thresholds for incorporating those behavioral changes. For achieving this, we can integrate VSD with the automatic-calibration mechanism to optimize the accuracy (i.e, number of false positives and false negatives). However, we always make a tradeoff between the false positives and false negatives for a given threshold.

6.5.3. Tradeoff between False Positives and False Negatives for a Given Threshold

All spam filters aim at minimizing false alarms during the classification process. However, it is difficult to optimize the false positives and false negatives at the same time. It is understood that there is always a tradeoff between false positives and false negatives for a given threshold [[78],[95]]. Scott[78] explains the tradeoff between false positives and false negatives and suggests that a proper choice of threshold depends on which among the false positives or false negatives we optimize. Yih[95] presents the above tradeoff for an e-mail spam filter and attempts to reduce false negatives, keeping the false positives to minimum. This relation between the false positives, false negatives, and threshold is particularly true with spam behavior associated with a SPIT call. The computed spam score S for a SPIT call is compared to the preset threshold value (T_P) to make a decision whether to allow or reject the call. For a threshold ($T > T_P$), the call participants have an extra tolerance of ($T - T_P$) for the call to be forwarded to the end user. This results in more false negatives as real spam calls have extra tolerance to get filtered. Because of this extra tolerance, the filter classifies fewer valid calls as spam i.e., fewer are false positives. This, in turn, results in fewer spam calls being filtered. Similarly, for a threshold ($T < T_P$), i.e., for a lower end user tolerance value, more calls are filtered. However, a majority of the filtered calls constitute falsely classified valid calls (false positives). Therefore, the threshold value that will be configured must be carefully chosen to either limit false positives or false negatives.

Taking the above tradeoff into consideration, we studied the performance of VSD

by integrating the calibration mechanism. For this, we present the performance of the spam filter in three cases: false positives with $T < opt(T)$, false negatives with $T > opt(T)$ and filtered calls with $T > opt(T)$ where $opt(T)$ represents the optimum threshold. The other three cases of false negatives with $T < opt(T)$, filtered calls $T < opt(T)$ and false positives with $T > opt(T)$ are trivial. The reason is as follows: For a configured threshold of $T < opt(T)$, there would be least false negatives because of lesser tolerance (lower threshold). For a configured threshold $T < opt(T)$, the number of filtered calls are very high . However, lot of them would be false positives as the threshold used is less than the optimum threshold. Similarly, for a configured threshold of $T > opt(T)$, there would be least false positives as the tolerance is very high. The call participants can get through the filter even with more spam behavior resulting in less false positives.

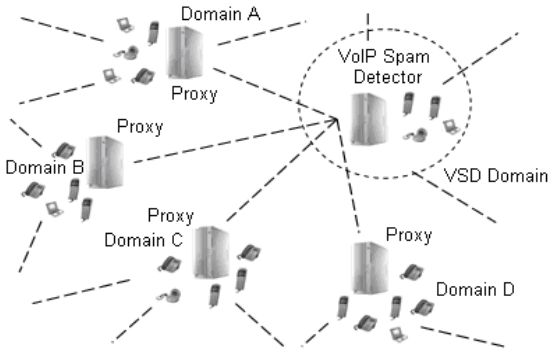

FIGURE 6.4. Simulation model for testing improvement in the VSD's accuracy using the automatic calibration. Calls are generated among the users in different domains and VSD domain. The VSD analyzes for spam calls from the spam entities (some end users, hosts, and domains that are initially configured to be spam).

For analyzing VSD's performance using the automatic-calibration mechanism, we constructed a simulation model for generating random voice calls. Figure 6.4 presents a simulation model that consists of call generating users, the call receiving

users and the VoIP spam detector (VSD). As shown in Figure 6.4, the VoIP spam Detector acts as a spam call filtering device for the users behind it (users inside the VSD domain). Calls are generated randomly (using a random user-id and a random IP address) among the users outside and inside the VSD domain. The call-generation process strictly follows a Bernoulli distribution. Calls are generated at an average rate of 8 calls per minute and all the experiments were run for a total of 2000 calls.

Upon receiving a call to a user inside the VSD domain, the VSD analyzes the spam behavior associated with that call as described in Chapter 4. By comparing the computed score to a predetermined threshold value, the VSD then decides to forward the call to the user if the score is less than the given threshold or to the user's voice mail if the score is greater than the configured threshold. Users within the VSD domain are equipped with spam recognition capabilities. When these users receive the call forwarded by the VSD (either to the voice phone or voicemail), they give the feedback about the validity of the call to the VSD. The VSD then updates the call participants' history based on this feedback. This history is later used for computing the spam score for later calls from those call participants.

Using the described simulation setup, we conducted experiments with varying thresholds and examined the filter performance (false positives, false negatives, filter calls, etc.) with and without automatic calibration. We conducted a non real-time analysis using ROC after 2000 calls (i.e. without using the automatic calibration mechanism) and observed that the filter delivered optimum performance at a threshold of 0.03 (*opt(T)*). Therefore, we present the benefit of using the automatic calibration mechanism in improving the accuracy of the filter for two different thresholds of $0.01(< opt(T))$ and $0.05(> opt(T))$.

Figure 6.5 presents the improvement in false positives when the VSD used the automatic calibration of filter threshold. In this experiment, we used an initial threshold of T = 0.01. It can be clearly observed that the total number of false positives with automatic calibration resulted in far fewer false positives as the VSD quickly

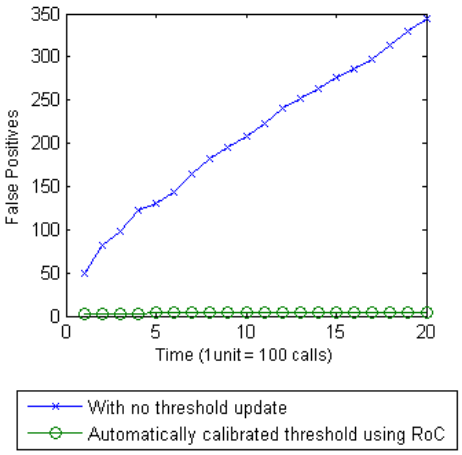

FIGURE 6.5. Improvement in false positives due to automatic calibration for a threshold T = 0.01 ($< opt(T)$). Because of a static threshold lower than the optimum threshold, the filter classified many valid calls as spam, i.e. false positives. However, using the automatic calibration, the filter quickly converged to optimum threshold and showed near optimal performance.

converged to the optimum threshold value of 0.03. Without the automatic calibration of threshold, the false positives were high because of lower (static) threshold than the optimum value. In this case, we noticed that every call from call participants having a low spam score was filtered. This is possible because legitimate call participants inherit spam behavior by appearing with spam call participants. Therefore, when the cumulative spam score of a call from these call participants exceeds the threshold T, the call is filtered. However, the end user reports the call as non-spam in feedback to the VSD which then categorizes it as false positive. While we observed a considerable increase in the filter's performance in minimizing false positives for a lower threshold $T < opt(T)$, we also observed an increase in the filter's performance in minimizing false negatives for a threshold $T > opt(T)$ as shown in Figure 6.6. From Figure 6.6,

it can be observed that the number of false negatives because of automatic calibration is lower when compared to the number of false negatives when VSD did not use automatic calibration.

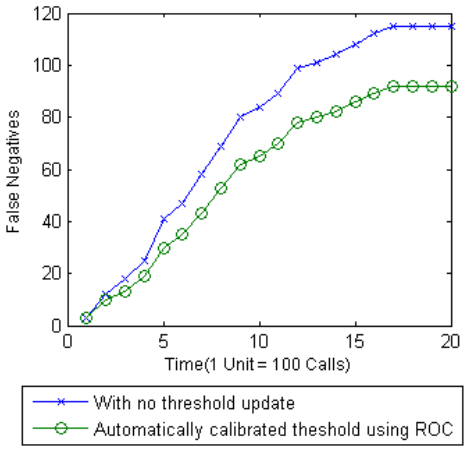

FIGURE 6.6. Improvement in false negatives due to automatic calibration for an initial threshold $T = 0.05 (> opt(T))$

When VSD is integrated with the automatic-calibration mechanism, the filter threshold automatically decreases due to the recomputation of the optimum threshold. As a result, the spam filter forwards fewer spam calls, i.e., fewer are false negatives. As mentioned previously, the number of false positives and false negatives is always a tradeoff with a given threshold value. When using automatic calibration, the filter showed better performance in false positives for an initial threshold ($T = 0.01 < opt(T)$ as shown in Figure 6.5) and in false negatives for an initial threshold ($T = 0.05 > opt(T)$ as shown in Figure 6.6). However, in both the cases, performance increased with regard to the total number of false alarms when we used the automatic-calibration mechanism as shown in Figure 6.7.

From Figure 6.7, we show that the number of false alarms when VSD uses automatic-calibration mechanism is lower than the number of false alarms without automatic calibration. We ran these experiments with thresholds lower than and higher

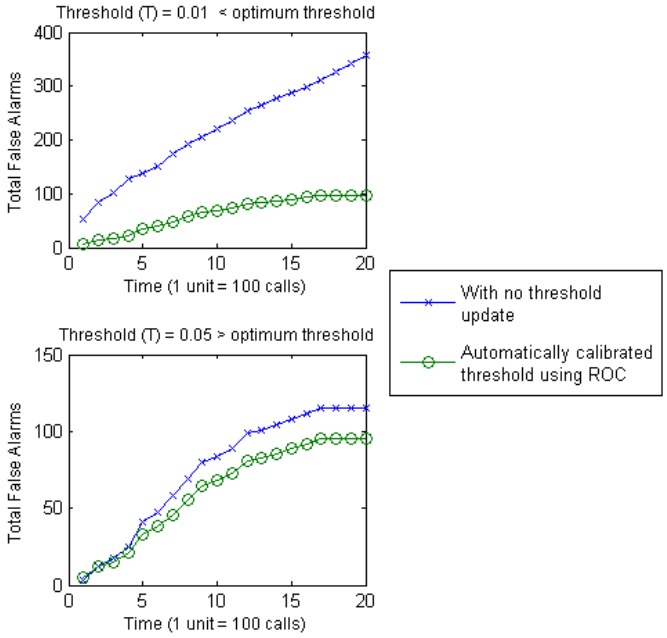

FIGURE 6.7. Fewer false alarms occur when the filter uses automatic calibration.

than the optimum threshold. In both cases, the number of false alarms calculated by the VSD is lower when VSD uses the automatic calibration. Irrespective of its initially configured threshold, the VSD quickly converges to the optimum threshold and, therefore, the number of false alarms is less. In addition, VSD showed considerable improvement in filtering spam calls for an initial threshold greater than the optimum threshold value as shown in Figure 6.8.

Figure 6.8 presents the improvement in filtered spam calls by automatically calibrating the threshold for an initial threshold of T = 0.05 which is more than the optimum threshold of 0.03. Without automatic calibration, VSD uses a statically configured threshold to filter calls. It is lot harder to filter calls with a higher threshold. However, with a spam filter integrated with an automatic-calibration mechanism, for every incoming call to the spam filter, the filter is updated with the optimum threshold value. In this case, the recalculated optimum threshold results in a decrease in the threshold value; therefore, more calls are filtered. Using the automatic-calibration

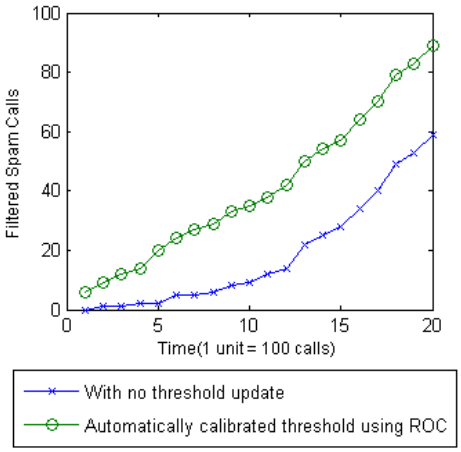

FIGURE 6.8. The improvement in filtered spam calls due to automatic calibration for a threshold $T = 0.05 (> opt(T))$.

mechanism, the filter showed an improvement as high as 100 % in filtering actual spam calls. This is due the filter converging with the optimum threshold from the initially configured higher threshold value.

The plots presented so far ($Figure\ 6.5 - Figure\ 6.8$) reflect improvement in VSD's performance when we implemented automatic calibration. In addition automatic calibration showed greater accuracy in classification compared to a manual threshold-update procedure. This relation between the false positives and the update period can be inferred from Figure 6.9. The figure represents the plot for false positives when the automatic-calibration mechanism is integrated into the VoIP spam filter and run at specific regular intervals (similar to a manual threshold update by filter administrators when they find a decrease in the filter's accuracy). The spam filter showed increase in accuracy when it is quickly updated with the optimum threshold value.

The filter showed the best performance when we integrated the automatic-calibration mechanism as integration involves real-time threshold value updates. Table 6.1 shows the spam filter's accuracy with the interval update process.

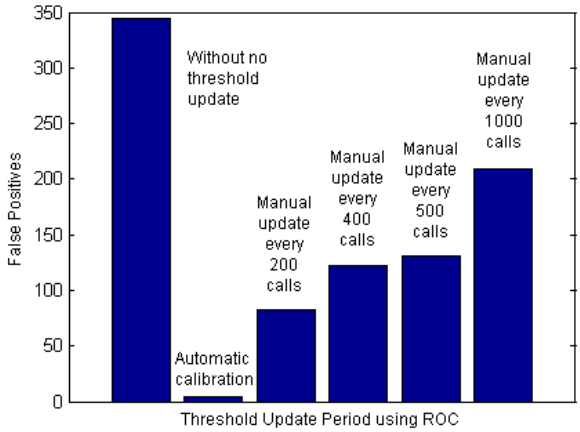

FIGURE 6.9. A lower time period for the update process results in a lower number of false positives. The automatic-calibration mechanism shows the highest performance as it involves real-time update for every call.

TABLE 6.1. Accuracy due to updating the filter with optimum threshold values at differing intervals.

Threshold Update	Accuracy %	Std. Error
Static	37.2	0.05
Automatically calibrated using ROC	97.2	0.04
Manual update every 200 calls	95.8	0.005
Manual Update every 400 calls	94.4	0.06
Manual Update every 500 calls	94.1	0.06
Manual Update every 1000 calls	92	0.08

Actual precision of classification for updating optimum threshold values at regular intervals is shown in Figure 6.10. From Figure 6.9, Table 6.1 and Fig. 6.10, it can be observed that the VSD's performance with false positives is better when we calibrate the threshold automatically after every call. The filter showed the highest precision when it used the automatic calibration. The classification precision is also

greater for shorter intervals when compared to longer intervals. In addition, frequently recompiling and updating the optimum threshold at short intervals when compared to longer intervals results in faster convergence.

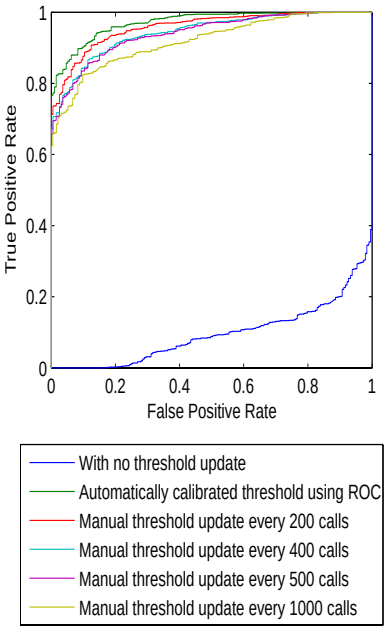

FIGURE 6.10. The classification precision due to updating the filter with optimum threshold values at differing intervals.

6.5.4. Automatic Threshold Calibration for Multiple Thresholds

Many adaptive filters use multiple thresholds for inferring unwanted behavior of a service-requesting entity. Updating multiple thresholds manually and frequently for optimizing filter performance is a time intensive, and sometimes, very frustrating process. The time required to manually update thresholds for optimum performance would be a polynomial increase in the order of the number of thresholds. We believe that the automatic-calibration mechanism can be effectively used with filters having multiple thresholds. To investigate this, we experimented with a variant of VSD where it independently assesses the spam behavior associated with individual call

participants. For simplicity, we configured the VSD to analyze 3 call participants: the calling user, the call host, and the call-originating domain. We used 3 initial threshold values: T_1, T_2 and T_3. For reference, a nonreal-time analysis resulted in optimum threshold values of 0.25(*opt(T_1)-user)*, 0.5(*opt(T_2)*-host) and 0.5(*opt(T_3)*-domain). Similar to the analysis described in earlier experiments, the total number of false positives, false negatives, and filtered spam calls is plotted for the filter using multiple thresholds. Figure 6.11 represents the improvement in false positives, false negatives, and filtered spam calls when the 3 threshold values are independently and automatically calibrated using the automatic-calibration mechanism. With a multiple-threshold setting, the filter showed better performance by automatic calibrating of the multiple threshold values after the end of every call the spam filter processed than it achieved without automatic calibration. As the multiple thresholds are independently calibrated after the end of every call, the maximum increase in time required for updating the threshold would be a linear increase in the order of number of thresholds.

The VSD and ND can benefit by using the automatic calibration mechanism for dynamically updating their threshold values. This dynamic update of thresholds will enable the VSD and ND to deliver optimum performance. In addition to using the automatic calibration mechanism for learning the communication behavior of callers and callees, the VSD and ND can benefit from the set of social communication relations between the callers and the callees. These relations define the possible communication patterns, and therefore can provide further evidence for limiting unwanted calls. For defining these communication patterns, we discuss a formalism of communication behavior between callers and callees in next chapter.

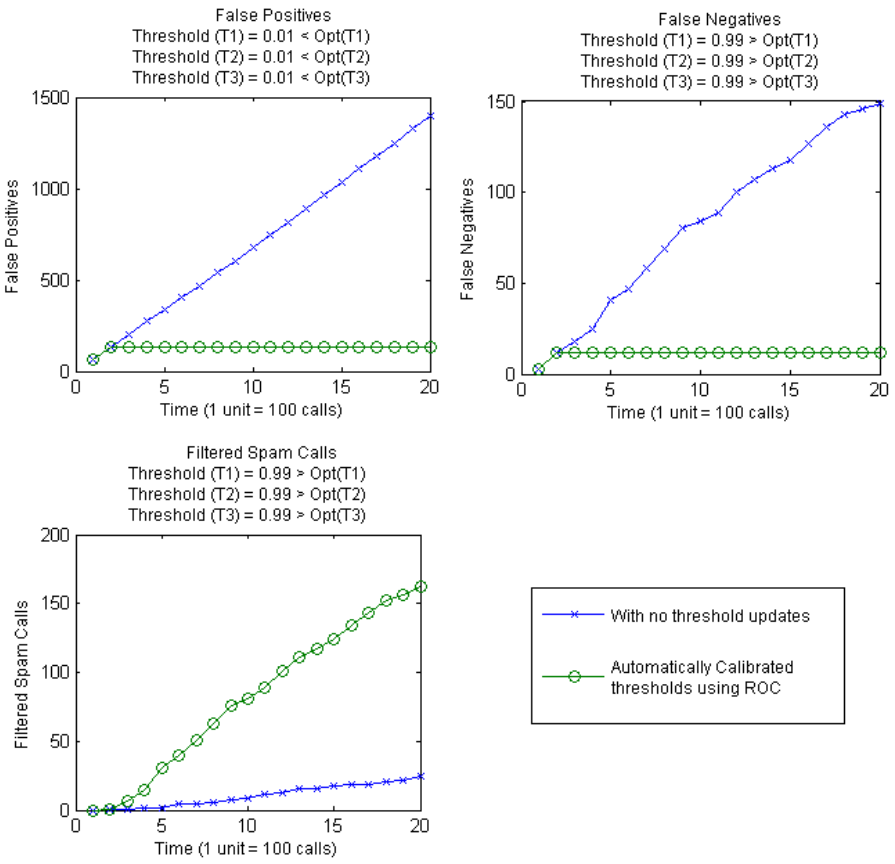

FIGURE 6.11. The spam filter's performance improvement configured with 3 threshold values for inferring the spam behavior of the calling user, call host and call-originating domain separately.

CHAPTER 7

CALL ALGEBRA

7.1. Introduction

The people individuals communicate with on communication networks such as PSTN and on cellular networks range from close or well-known associates such as family members to strangers who may even be spammers and telemarketers. Each individual callee employs differing types of connectivity and call distribution with each type. Based on our investigation of recent research and our own work, we believe that a human-social dynamics exists in the ways humans make and receive calls between individuals and groups. This dynamics depends mostly on the type of individuals communicating with each other. We believe that this human-social dynamics can be used for solving multitude of problems related to identification of wanted calls and solicited callers.

In this chapter, we discuss this dynamics in three sections. In Section 7.2, we enumerate normal calling patterns that exist between a specific callee and those that callee calls on the communication network. Here, we discuss the calling patterns between the callees and all the individuals they communicate on the voice communication network. These calling patterns dictate the callee's normal communication behavior in a day-to-day life. We use these calling patterns to establish algebraic rules that define the communication behavior between the individuals. Based on the communication behavior between the callees and the individuals they communicate, we can construct matrices based on fundamental calling patterns such as frequency and talk-time. Using the constructed matrices, we derive operations in Section 7.3

to define meaningful calling constructs that can be used to judge the legitimacy of the communicating parties. Finally, in Section 7.4, we discuss how these communication patterns and operations can be grouped for solutions to some of the existing IP telephony problems.

7.2. Communication Patterns

A social and human dynamics exists in the way callers and callees exchange real-time voice calls. This dynamics, of course, differs depending on the type of person we are communicating with at any specific time on the voice network. For example, we spend considerable time communicating with family members, friends, and distant relatives. Alternatively, we have an insignificant amount of communication with strangers such as telemarketers and fund-raisers. However, within this broad range of activity, we can draw some general conclusions about the types of calls that occur. In this section, we present a formalism that represents the social and human dynamics that occur when people make and receive calls from individuals who communicate with them from time to time. To present this formalism, we first define a basic call operation.

Definition 1: A call is a real-time operation between communicating parties.
A simple call (C) between two communicating parties A and B can be represented as an operation (Φ) between two parties as shown below

$$C = \Phi(A, B)$$

Definition 2: A conference call is a multimedia communication between two or more communicating parties. A conference call can be represented as

$$ConferenceCall = \Phi(A, B, ...K)$$

where A,B,...K are the communicating parties involved in the call.
Next, we present how the calls are generated and handled between the communicating

parties.

Definition 3: A unicast call is a call which involves only two communicating parties.

In this call, one party generates the call. The second party receives it. This call can be shown as follows

$$UnicastCall = \Phi(A, B) = A \rightarrow B$$

Definition 4: A broadcast call is call from a caller to all the individuals within a callee's community.

Such calls frequently are telemarketer or spam calls. However, we note that a broadcast call is equivalent to the caller generating a unicast all to each community member. For "M" number of callee's community members, a broadcast call from caller P can be represented as follows

$$P \rightarrow A$$
$$P \rightarrow B$$
$$.$$
$$.$$
$$P \rightarrow M$$

i.e. $(P \rightarrow A) \cup (P \rightarrow B) \cup ..... \cup (P \rightarrow M)$ where A,B,..M are all of the callee's community members.

Definition 5: A multicast call is a call from a caller to a multicast group who are members of community.

When a caller makes a conference call to subscribed individuals in the callee's community, then it is termed as a multicast call. Similarly to a broadcast call, a multicast may be the equivalent to the caller making a unicast call to each individual in the multicast group. This multicast call can be represented as

$$P \rightarrow A$$
$$P \rightarrow B$$
$$.$$
$$.$$
$$P \rightarrow G$$

i.e. $(P \rightarrow A) \cup (P \rightarrow B) \cup ..... \cup (P \rightarrow G)$ where A,B,..G are a set of callee's community members.

Although call distributions (Def. 3 - 5) signify the type of calls from the callers to one or more of the callee's community members, other calling patterns exist that represent how a callee handles a call from a caller.

Definition 6: A forwarded call is a call from a caller forwarded by a callee to another callee.

Forwarded calls are frequently observed when the caller fulfills his communication by communicating with a callee who was referred by another callee. For example, a company employs a receptionist (A) who receives all incoming calls. The receptionist, however, forwards each call to its intended callee (B) within the company. A simple call forward (divided into two legs) can be represented as follows

Leg 1: $P \rightarrow A$

Leg 2: $A \rightarrow B$

Thus the communication between the communicating parties can be represented as

$$\Phi(P, A, B) = P \rightarrow A \rightarrow B$$

Definition 7: A returned-call is a responding call by a callee to previous call from a caller.

Such returned calls frequently are in response to previous conversations/calls in voice mail boxes. For example, a callee (A) was unable to receive a call from caller P. P leaves a recorded message in A's voice mail box. At a later time, caller A returns P's call. A simple returned-call can be shown as follows

Leg 1: $P \rightarrow A$

Leg 2: $A \rightarrow P$

This communication between the communicating parties can be represented as

$$\Phi(P, A) = P \rightarrow A \rightarrow P$$

Definition 8: A conference join is a service where in an individual can join in a conference or the members already in the conference can join the individual.

Group communication is often preferred when multiple individuals exchange similar information. Therefore, communicating parties organize conference calls and participants join and leave the conference call based on their needs and interest. A conference join can happen in two ways:

An individual joins an already progressing conference

$$P \rightarrow (A, B, ...K)$$

A member of an ongoing conference invites an individual to join in the conference

$$(A, B, ...K) \rightarrow P$$

We now have defined how the calls are created, distributed, and handled between the communicating parties. Using this call-distribution and call-handling formalism, we can derive algebraic rules that represent the possible communication patterns between the callees and types of individuals they communicate on a voice network.

7.2.1. Algebraic Rules

Using the call definitions discussed before, we can derive algebraic rules that represent possible communication patterns between callees and types of individuals they communicate on the VoIP network. We discuss the communication patterns based on the type of individuals the callee communicates on the voice network. As we observed in Chapter 5, we can divide the individuals communicating with a callee into four broad categories.

(1) Socially Close Members: These are the people with whom the callee maintains the highest socially connectivity. Most of the calls the callee receives come from individuals within this category. The callee receives more calls from them and spends more time talking to them e.g., family members, friends, and colleagues

(2) Socially Near Members: People in this category are not as highly connected as family members and friends, but when the callee connects to them, the callee talks to them for considerably longer periods e.g., neighbors and distant relatives.

(3) Opt-ins: These individuals have less connection with the callee's social life. The callee acknowledges the calls from these individuals rarely e.g., discussion groups and newsletters.

(4) Opt-outs: These people are least connected people with the callee on the communication network and the number of calls the callee has accepted from these individuals is bare minimum e.g., strangers, telemarketers, fund raisers.

For each type, we derived algebraic relations that describe the communication the type's callers have with a callee or the callee's community members. We enumerated the communication patterns for such relationships as associative, distributive, and commutative. Consider a community with m members, and each member having a maximum of n people in each of the four groups defined before. The individuals communicating with a callee R_i for i=1..m are represented by S_{ijk} (caller k belonging

to group number j of community member R_i for i=1..m, j=1..4 and k=1..n). Using the above categorization, we can derive algebraic rules for people belonging to a specific type of caller.

7.2.1.1. *Socially Close members.* The socially closest members of a callee's community are people with the most connectivity to a callee. The callee frequently communicates with and may spend significant time interacting with these people. Using our Algebraic rules, we describe the connectivity a callee has with members of the socially close Community.

(1) Calls from socially close members are distributive.

Calls from a member of a socially close community to a callee and to a callee's community members are distributive. When a socially close caller intends to convey information to a set of callees, the caller can create a conference $[S_{i1k} \rightarrow (R_1, R_2, ...R_l)$ to communicate with them, or the caller can make unicast calls to the callees to convey the information individually. This distributive relation can be represented as

$$\begin{aligned} & & S_{i1k} \rightarrow (R_1) \\ & & S_{i1k} \rightarrow (R_2) \\ S_{i1k} \rightarrow (R_1, R_2, ...R_l) & = & . \\ & & . \\ & & S_{i1k} \rightarrow (R_l) \end{aligned}$$

In either case, the callee will have similar willingness/interest for completing the call. Therefore, we have

$S_{i1k} \rightarrow (R_1, R_2, ...R_l) = (S_{i1k} \rightarrow R_1) \cup (S_{i1k} \rightarrow R_2) \cup ..... \cup (S_{i1k} \rightarrow R_l)$ for $i,l \in 1..m$ & $k \in 1..n$.

(2) Calls from socially close members are commutative.

Calls between a callee and any socially-close members are commutative, i.e., a bidirectional call-generation process and reception pattern is observed. We can represent this commutative relation as follows

$$S_{i1k} \rightarrow R_l = R_l \rightarrow S_{i1k}$$

$$\text{i.e. } \Phi(S_{i1k}, R_l) = \Phi(R_l, S_{i1k}) \text{ for } i,l \in 1..m \ \& \ k \in 1..n$$

(3) Calls from socially close members are associative.

When a socially close member A intends to communicate with two callees B and C, then A places a call to one callee and then invites the second callee to join the conference. For example, a socially close caller S_{i1k} can communicate with two callers R_l and R_K in both the ways as shown below

(a) Caller S_{i1k} makes a call to R_l first and then to R_s. This can be represented by $(S_{i1k} \rightarrow R_l) \rightarrow R_s$ for $i,l,s \in 1..m \ \& \ k \in 1..n$

or

(b) Callee R_l makes a call to R_s and then S_{i1k} joins the conference $S_{i1k} \rightarrow (R_l \rightarrow R_s)$for $i,l,s \in 1..m \ \& \ k \in 1..n$

In either way, the willingness/interest of all the three participants will be same. Therefore, we have

$(S_{i1k} \rightarrow R_l) \rightarrow R_s = S_{i1k} \rightarrow (R_l \rightarrow R_s)$ for $i,l,s \in 1..m \ \& \ k \in 1..n$ i.e. the calls are associative.

7.2.1.2. *Socially Near members.* Socially near members of a callee's community are the people, with whom the callee has a relatively low connectivity but has high tolerance to calls from them, i.e., the callee usually spends considerable time with people belonging to this group. The connectivity the callee has with the people in

this group is relatively less than that of the people in the socially close group.

(1) Calls from socially near members are distributive.

Whenever, a socially near member of a callee's community intends to communicate with a callee or some of the callee's community members (e.g., callee's socially close members), the socially near member can create a conference to communicate with them $[S_{i2k} \rightarrow (R_1, R_2, ...R_l)]$ or make calls to them individually and communicate with them. Therefore,

$$S_{i2k} \rightarrow (R_1, R_2, ...R_l) = \begin{cases} S_{i2k} \rightarrow (R_1) \\ S_{i2k} \rightarrow (R_2) \\ . \\ . \\ S_{i2k} \rightarrow (R_l) \end{cases}$$

In either case, the callee will have similar willingness/interest for completing the call. Therefore, we have

$S_{i2k} \rightarrow (R_1, R_2, ...R_l) = (S_{i2k} \rightarrow R_1) \cup (S_{i2k} \rightarrow R_2) \cup ..... \cup (S_{i2k} \rightarrow R_l)$ for $i,l \in 1..m$ & $k \in 1..n$.

(2) Calls from socially near members are commutative.

There is always a mutual call-generation behavior between a callee and socially near members, i.e., the socially near members make calls to the callee and the callee also makes calls to them. Therefore, based on this mutual calling behavior, we can show a commutative relation as follows

$$S_{i2k} \rightarrow R_l = R_l \rightarrow S_{i2k}$$

$$\text{i.e. } \Phi(S_{i2k}, R_l) = \Phi(R_l, S_{i2k}) \text{ for } i,l \in 1..m \ \& \ k \in 1..n$$

(3) Calls from socially near members are associative.

Whenever a socially near member intends to communicate with two callees, he can

either call one callee and invite the other $[(S_{i2k} \rightarrow R_l) \rightarrow R_s]$, or join in a conference call that is already in progress in between the two callees $[S_{i2k} \rightarrow (R_l \rightarrow R_s)]$. In either case, the socially near member can convey his information to the intended callees.

Therefore, $(S_{i2k} \rightarrow R_l) \rightarrow R_s = S_{i2k} \rightarrow (R_l \rightarrow R_s)$ for $i,l,s \in 1..m$ & $k \in 1..n$ i.e. the calls are associative.

7.2.1.3. *Opt-ins.* Opt-ins are people with whom a callee has minimal connectivity. A callee rarely acknowledges calls from the people belonging to this group.

(1) Calls from opt-in callers are not distributive.

Opt-in callers are callers from whom a callee may occasionally solicit information. At the time of the solicitation, the callee considers the calls from opt-ins reasonable. Depending on the callee who has acknowledged, the opt-in callers make calls to them individually All the calls from them to each callee are simple one-to-one calls. Therefore, the calls from opt-in callers are not distributive.

(2) Calls from opt-in callers are limited commutative.

Callees do not often acknowledge calls from opt-in callers. However, when calls from opt-in callers are of situational interest, callees communicate with the opt-in callers. Compared to the socially close and socially near members, the callees show least communication behavior with opt-in callers. Therefore, calls from opt-in callers are limited commutative. This communication behavior can be shown as follows

$$S_{i3k} \rightarrow R_l = R_l \xrightarrow{l} S_{i3k}$$

i.e. $\Phi(S_{i3k}, R_l) \stackrel{l}{=} \Phi(R_l, S_{i3k})$ for $i,l \in 1..m$ & $k \in 1..n$ i.e., calls from opt-in callers are limited commutative.

(3) Calls from opt-in callers are not associative.

As nearly all the calls from the opt-in callers are unicast calls, only rarely will opt-in callers intend to communicate with two callees at the same time. No opt-in callers would organize or join conferences. Therefore, the calls from the opt-in callers are not associative.

7.2.1.4. *Opt-outs.* Opt-outs are callers with whom a callee has the least connectivity. The callee never acknowledges communication with members of this group. As a result of this behavior, opt-outs have a call distribution as described below.

1. Calls from opt-out callers are distributive.

When an Opt-Out caller intends to communicate with all the members of a callee's community, the caller generates a conference call (easily possible with in VoIP networks by building a VoIP broadcast client) $[S_{i4k} \rightarrow (R_1, R_2, ...R_l)]$, or the Opt-Out caller can make a unicast call to each callee, described as $[S_{i4k} \rightarrow (R_l)]$ for $i,l \in 1..m$ & $k \in 1..n$. Therefore,

$$S_{i4k} \rightarrow (R_1, R_2, ...R_l) = \begin{array}{l} S_{i4k} \rightarrow (R_1) \\ S_{i4k} \rightarrow (R_2) \\ . \\ . \\ S_{i4k} \rightarrow (R_l) \end{array}$$

i.e. $S_{i4k} \rightarrow (R_1, R_2, ...R_l) = (S_{i4k} \rightarrow R_1) \cup (S_{i4k} \rightarrow R_2) \cup ..... \cup (S_{i4k} \rightarrow R_l)$ for $i,l \in 1..m$ & $k \in 1..n$.

(2) Calls from opt-outs are not commutative.

As a rule, callees do not want to receive the types of calls distributed by opt-out members. Virtually no one calls back opt-out callers. Therefore, this communication

is never commutative.

(3) Calls from opt-out callers is not associative.

As discussed above, calls from opt-out members are least sought. The callees do not get together to accept calls from opt-out callers. Therefore, these calls are not associative. However, it has been observed that individual callees might be interested in some services an opt-out member may provide. In this case, if an individual callee opts in for a specific service, the opt-out caller would then jump into the opt-in group of that callee. Therefore, at any instant, the snapshot of members in the opt-out group are the people who are of negligible interest to both the callee and members of the callee's calling community. None of the callees shows an interest in communicating with the opt-out community.

All the above rules describe the calling patterns callees have with their social network members. We can analyze these calling patterns with some operations to generate meaningful call-constructs that can be used to provide solutions to a number of current telephony problems.

7.3. Operations based on Calling Patterns

Current research problems in telephony applications deal with identifying the legitimacy of the incoming calls and the callers making those calls. This is necessary because unwanted callers such as spammers and phishers can use real-time voice communication for spamming users and extracting confidential information for their personal and organizational benefit. In this section, we describe operations on the caller-callee calling patterns. All these operations provide filtering characteristics that can be used for identifying the legitimacy of incoming calls and the callers making those calls.

The fundamental parameters that we can extract based on communication between a caller and a callee are frequency, duration, time-of-arrival, and inter-arrival

time. Based on these communication parameters, we can define caller-callee matrices that describe the communication between multiple callers and callees (e.g., the callee and the callee's community members). For m callees (R_i for $i = 1..m$) inside the callee's community and p callers ($\{S_l\}_{l=1..p} = \{S_{ijk}\}_{i=1..m,j=1..4,k=1..n}$ i.e., all callers to all callees) making calls to the callees in the community, a matrix for parameter X based on the communication between the callers and the callees can be shown as in Figure 7.1

The matrix SR_X represents the communication matrix of parameter X between the

$$SR_X = \begin{array}{c} \\ S_1 \\ S_2 \\ \vdots \\ \\ S_p \end{array} \begin{array}{c} \begin{array}{cccc} R_1 & R_2 & \cdots & R_m \end{array} \\ \begin{bmatrix} (S_1R_1)_X & (S_1R_2)_X & \cdots & (S_1R_m)_X \\ (S_2R_1)_X & (S_2R_2)_X & \cdots & (S_2R_m)_X \\ \vdots & & & \\ \\ \vdots & & & \\ (S_pR_1)_X & (S_pR_2)_X & \cdots & (S_pR_m)_X \end{bmatrix} \end{array}$$

FIGURE 7.1. A parameter matrix based on calls generated from callers to callees.

callers and the callees where X can be any of frequency, duration, time of arrival, inter-arrival time etc. Each element S_iR_j in the matrix represents a normalized value of communication parameter X from caller S_i to callee R_j for i=1..p and j=1..m. Based on the above representation, we can construct parameter matrices such as frequency (SR_F), duration (SR_D), time of arrival (SR_T), and inter-arrival time (SR_I) parameters. However, it is also possible that the callee's community members generate calls to the callers (outgoing calls). Based on this, we can define outgoing communication parameter matrices such as RS_F, RS_D , RS_T, and RS_I. Using these incoming and outgoing matrices, we derived operations for detecting wanted calls between the communicating parties.

7.3.1. Connectivity

Connectivity represents the amount of communication the parties have in a voice network. This amount of communication can be measured based on the extent of frequency and duration of calls between the parties. We argue that higher the connectivity, higher will be the trust between the caller and the callee. Similarly, the higher the connectivity of callers towards a specific group of callees, the higher will be the trust of those callers with respect to those callees. The frequency and duration matrices such as SR_F, SR_D, RS_F, and RS_D can be used for deriving the amount of connectivity between the callees and the callers.

7.3.1.1. *Connectivity based on Incoming Calls.* The matrices SR_F and SR_D can be used for determining the connectivity of the callers to the callees based on incoming calls. For example, consider the multiplication operation between the frequency matrix and the transpose of the duration matrix.

$$SR_F * SR_D^T =$$

$$\begin{bmatrix} (S_1R_1)_F & (S_1R_2)_F & . & . & . & (S_1R_m)_F \\ (S_2R_1)_F & (S_2R_2)_F & & & & \\ . & . & & & & \\ . & . & & & & \\ . & . & & & & \\ (S_pR_1)_F & (S_pR_2)_F & . & . & . & (S_pR_m)_F \end{bmatrix} * \begin{bmatrix} (S_1R_1)_D & (S_2R_1)_D & . & . & . & (S_pR_1)_D \\ (S_1R_2)_D & (S_2R_2)_D & & & & \\ . & . & & & & \\ . & . & & & & \\ . & . & & & & \\ (S_1R_m)_D & (S_2R_m)_D & . & . & . & (S_pR_m)_D \end{bmatrix}$$

The result of the above matrix operation is a matrix C^S such that each element of the matrix is equal to

$C^S_{i,j} = (S_iR_1)_F * (S_jR_1)_D + (S_iR_2)_F * (S_jR_2)_D + ...(S_iR_m)_F * (S_jR_m)_D$

i.e. $C^S_{i,j} = \sum_{k=1}^{m}[(S_iR_k)_F * (S_jR_k)_D]$ for $i,j \in 1..p$. The leading diagonal elements of the resultant matrix C^S (i.e. elements $C^S_{i,j}$ such that i=j for $i,j \in 1..p$) has p number of elements since the matrix C^S is a *pxp* matrix. Each element of the diagonal ($C^S_{i,j}$)

represents the connectivity of the caller S_i with respect to all the callees. We can infer that, based on the frequency and duration patterns, the higher the value of the diagonal element, the higher is the connectivity of the respective caller towards all the callees.

7.3.1.2. *Connectivity based on Outgoing Calls.* Similar to the operation shown for incoming calls, we can use the matrices RS_F and RS_D to establish the connectivity of callees based on the outgoing calls. Consider a matrix C^R that is equal to the product of RS_F and transpose of RS_D. Each element in the resultant matrix can be represented by $C^R_{i,j} = \sum_{k=1}^{p}[(R_iS_k)_F * (R_jS_k)_D]$ for $i,j \in 1..m$. The diagonal elements in the matrix C^R (i.e. elements $C^R_{i,j}$ such that i=j for $i,j \in 1..m$) has m number of elements as the size of the matrix C^R is mxm. Each element of the diagonal represents the connectivity of a callee with respect to all the callers. And, higher the connectivity of a callee, higher is the trust of the callee with respect to all the callers.

While the connectivity gives information about the direct trust based on past communication, we can derive more legitimacy information based on forwarded calls by deducing the reputation of callers and callees.

7.3.2. Reputation

Reputation represents social status. It is derived based on recommendations from trusted peers. However, with respect to calling patterns, we believe that the reputation can be derived based on the preference (addressed using call forwarding) of calls from the callers and the callees.

7.3.2.1. *Reputation based on Incoming Calls.* The reputation of the callers can be derived using multiplication operation between the SR_F and RS_F matrices. For the matrix $D^R = SR_F * RS_F$, each element of the matrix is represented as $D^R_{i,j} = (S_iR_1)_F * (R_1S_j)_F + (S_iR_2)_F * (R_2S_j)_F + ...(S_iR_m)_F * (R_mS_j)_F = \sum_{k=1}^{m}[(S_iR_k)_F * (R_kS_j)_F]$.

From the above equation, we can observe that every call from a caller S_i to a callee

R_k for $k \in 1..m$, the callee forwards the call to another caller S_j for $i, j \in 1..p$. It can be observed that all the non-diagonal elements represent this call forward relation as shown in Figure 7.2.

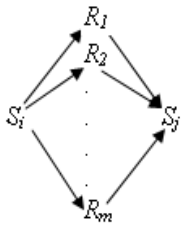

FIGURE 7.2. A call forward relation for calls from callers.

We believe this is call-forward function represents the reputation of the caller. Therefore, $D^R_{i,j}$ of matrix D^R gives the reputation of caller S_i for $i,j \in 1..p$ & i $\neq$ j.

7.3.2.2. *Reputation based on Outgoing Calls.* We can derive reputation of callees using the same RS_F and SR_F matrices by performing the multiplication operation $RS_F * SR_F$. The result of this multiplication operation is a matrix D^S such that each element $D^S_{i,j}$ can be represented as $D^S_{i,j} = \sum_{k=1}^{p}[(R_iS_k)_F * (S_kR_j)_F]$. Each non-diagonal element of the matrix D^S ($D^S_{i,j}$ such that i $\neq$ j for $i,j \in 1..m$) represents the call-forward function for all calls from R_i to R_j by S_k i.e. $D^S_{i,j}$ represents the reputation of R_i for $i,j \in 1..m$ & $k \in 1..p$.

The connectivity and reputation information provides a measure of legitimacy of calls. In addition to these measures, we can also derive functions for parameters such as reciprocity and periodicity. Integrating these functions with operations such as connectivity and reputation can result in separating wanted calls and wanted callers.

7.3.3. Reciprocity

Reciprocity represents the response shown by one party to calls from another party. This reciprocity can be established using the SR_F and RS_F matrices.

7.3.3.1. *Reciprocity based on incoming calls.* The reciprocity shown by callees can be determined using a multiplication operation between the SR_F and RS_F matrices. For the matrix $D^R = SR_F * RS_F$ shown in 7.3.2.1, each element of the matrix is represented by $D^R_{i,j} = (S_iR_1)_F*(R_1S_j)_F+(S_iR_2)_F*(R_2S_j)_F+...(S_iR_m)_F*(R_mS_j)_F = \sum_{k=1}^{m}[(S_iR_k)_F * (R_kS_j)_F]$. In matrix D^R, the diagonal elements are of the form $D^R_{i,j} = \sum_{k=1}^{m}[(S_iR_k)_F * (R_kS_i)_F]$ for $i \in 1..p$ (since i=j). We note that when we extract the diagonal elements, the i^{th} diagonal element represents the call-back function to the caller i.e., the value of the i^{th} diagonal element represents the reciprocity shown by the callees to calls from caller S_i for $i \in 1..p$ as shown in Figure 7.3

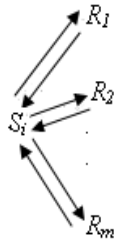

FIGURE 7.3. A call back relation by callees for calls from callers.

7.3.3.2. *Reciprocity based on Outgoing Calls.* Similar to the derivation shown for reciprocity shown by callees, each diagonal element $D^S_{i,j}$ of matrix $D^S = RS_F * SR_F$ gives the reciprocity shown by callers to calls from callees.

Although the parties show eagerness to return the unanswered calls, they often tend to acknowledge calls mostly during some defined intervals. These intervals can be described by computing periodicity of the incoming or outgoing calls.

7.3.4. Periodicity

Periodicity represents the regularity in which the callees accept/make calls from/to the callers. It is of common observation that people prefer defined intervals for communicating with other parties (e.g., after-work hours) and resent calls during other intervals (e.g., work hours). Therefore, the periodicity information can be used for understanding the callees' eagerness in receiving calls at different times. One way of defining a periodicity matrix is as follows: Define matrix M such that each element in the matrix is defined by

M_{ijk} = Number of calls received by callee j from caller i at time unit k for $i \in 1..p$, $j \in 1..m$, $k \in 1..nt$ where nt is the number of time units in a given day. The value of nt may depend on the resolution required by the application e.g., nt= 24 time units (1 hour interval) or nt=96 time units (15 min interval). Using the above matrix M, we can derive a periodicity matrix P^t at a given time unit t. Each element in the periodicity matrix i.e. $P^t_{i,j}$ represents the periodicity of the calls from caller S_i to callee R_j for $i \in 1..p$, $j \in 1..m$ at time interval t and is defined as follows

$$P^t_{i,j} = \frac{M_{ijt}}{\sum_{k=1}^{nt} M_{ijk}}$$

We can extend the matrix P^t and define a matrix $P^{'}$ such that each element of the matrix $P^{'}$ is defined as follows

$P^{'}_{i,j}$ = Periodicity of caller S_i at time unit T_j for $i \in 1..p$, $j \in 1..nt$. Each element of the matrix can be expressed as $per_i^{T_j}$ i.e. periodicity of caller S_i at time unit T_j for $i \in 1..$p, $j \in 1..nt$. The matrix $P^{'}$ can be shown as in Figure 7.4

Even though we can define a caller's periodicity at a given time unit, we have observed that callers have significant call distribution only during specific time periods. We can derive the most probable times during which a caller or a group of callers have a high call distribution using the $P^{'}$ matrix. For this, however, we

$$P' = \begin{array}{c} \\ S_1 \\ S_2 \\ . \\ . \\ . \\ S_p \end{array} \begin{array}{c} \begin{array}{cccccc} T_1 & T_2 & . & . & . & T_n \end{array} \\ \begin{bmatrix} per_1^{T_1} & per_1^{T_2} & . & . & . & per_1^{T_n} \\ per_2^{T_1} & per_2^{T_2} & . & . & . & per_2^{T_n} \\ . & . & & & & \\ . & . & & & & \\ . & . & & & & \\ per_p^{T_1} & per_p^{T_2} & . & . & . & per_p^{T_n} \end{bmatrix} \end{array}$$

FIGURE 7.4. A periodicity matrix based on calls from callers to callees.

needed to formulate a technique to reduce the number of columns of the matrix but still create a sub-matrix that preserves the information represented in the P' matrix. For this reduction, we apply the Eigen principles to construct Eigen vectors of the matrix P' [60] and define sub-matrices of the form

$$\begin{array}{c} \\ S_a \\ S_b \\ . \\ . \\ S_l \end{array} \begin{array}{c} \begin{array}{ccccc} T_d & T_f & . & . & T_g \end{array} \\ \begin{bmatrix} . & . & . & . & . \\ . & . & & & \\ . & & . & & \\ . & & & . & \\ . & & & & . \end{bmatrix} \end{array}$$

FIGURE 7.5. Inferring most-probable-times during which callers generate calls.

From the resulting matrix, we can infer that the set of callers $\{S_a, S_b, ..S_l\}$ where $a, b, ..l \in 1..p$ have significant calling distribution during the time intervals $\{T_d, T_f, ..T_g\}$ for $d, f, ..g \in 1..nt$. Using the same procedure, we can define sub matrices and infer the most probable times during which the callers have significant calling patterns.

Understanding the eagerness using reciprocity and periodicity computation

helps in inferring the legitimacy of communicating parties. In addition to these constructs, we can derive call constructs based on call distribution. Examining the distribution of calls from callers to callees provides additional information during filtering processes. For example, it is highly unlikely that legitimate callers make broadcast calls to all the members in the callee's community. Further, the calling patterns from a legitimate caller to a callee rarely coincide with the calling patterns from the same caller to other callees in the community. Alternatively, it is frequently observed that spammers make broadcast calls, and their calling patterns to community wide callees are similar (e.g., by sharing callee lists). Therefore, here we discuss some call constructs such as similarity and correlation that account for call distribution information. We can integrate these constructs along with legitimacy providing parameters such as connectivity, reputation, reciprocity, and periodicity for making high accuracy decisions.

7.3.5. Similarity

Similarity represents the commonality in call distribution. Two callers are said to be identical if they have equal calling distribution across a set of callees in the callee's community. We can determine similarity in call distribution using "rank" of SR_F and RS_F matrices. We believe that a case scenario for the "rank" of a matrix provides information about the similarity of the people in making calls.

Rank is defined as the number of independent rows or columns in the given matrix. Additionally, Row Rank represents the number of independent rows and Column Rank represents the number of independent columns in a given matrix. Dependency in the rows or columns occurs if one or more of the rows or columns is or are a multiple (dependent) on other rows or columns, respectively.

Consider as a case scenario where a row is equal (multiple by 1) to another row in a simple SR_F matrix. If two callers S_i and S_j have equal values to all callees (i.e. the two rows for the callers are same in the SR_F matrix), then it represents that the two callers have equal calling distribution to all the callees. In this case, the two

callers are similar in call generation. As the row rank of the above matrix gives the number of independent rows, then (*p - row rank*) callers are similar to at most (*p-row rank*) callers in the given matrix where p is the total number of callers.

Note: Similar to the above discussion, we can determine callees having similar calling distributions using the RS_F matrix and its row-rank.

7.3.6. Correlation

Callers are not always strictly similar. However, two or more callers might have a similar pattern in their call distribution. We can find the set of callers having this similar pattern by defining the amount of correlation between the callers. Consider an SR_F matrix. Using this matrix, we can find the correlation (for example, using Pearson correlation coefficient) between two callers S_i and S_j for $i,j \in 1..p$ based on their call distribution to m callees R_j for $j \in 1..m$. If the call distribution by two callers towards all the callees represent two random variables X_I and X_J, then the Pearson correlation coefficient can be computed as follows

$$\text{Pearson's Coefficient } (\sigma) = \frac{X_{IJ}}{\sqrt{X_{II} * X_{JJ}}}$$

$$\text{where } X_{II} = \sum_{k=1}^{m}(S_iR_k)^2 - \frac{(\sum_{k=1}^{m}(S_iR_k))^2}{m}$$

$$X_{JJ} = \sum_{k=1}^{m}(S_jR_k)^2 - \frac{(\sum_{k=1}^{m}(S_jR_k))^2}{m}$$

$$X_{IJ} = \sum_{k=1}^{m}(S_iR_k)(S_jR_k) - \frac{(\sum_{k=1}^{m}(S_iR_k))(\sum_{k=1}^{m}(S_jR_k))}{m}$$

The Pearson correlation always lies in the range [-1 1]. A value of 1 represents a perfect correlation (both the variables change in the same way) and a value of -1 represents a perfect negative correlation (both the variables change in the opposite way). A value of near about 0 for the Pearson coefficient represents minimal correlation. Therefore, this coefficient value represents the amount of similarity the

two callers have in their call distribution.

The similarity and correlation information is useful for determining the extent to which the callers have similar call distribution. These call distributions between the callers and the callees can be a uniform or a random distribution. It is generally observed that callees acknowledge calls from caller who have uniformly distributed calls than the caller who calls randomly. In this scenario, it can be helpful to understand the randomness in call distribution so that more appropriate customized filtering policies can be formulated. Here, we discuss a measure for determining the randomness in calling patterns using entropy computations.

7.3.7. Entropy

Entropy is used for measuring randomness [21]. Using the SR_F matrix, we can infer the entropy based on the frequency communication patterns. Entropy as described by Claude Shannon [81] is given as

$$H(x) = -\sum_{i=1}^{n} p(i) log_2 p(i)$$

Each *p(i)* in the above equation can be computed by dividing a given time period (e.g., 1 day) into n intervals (e.g. 24*60 = 1440 intervals with an interval size of 1 minute). By observing the frequency over an extended period of time (e.g. 6 months), each *p(i)* for i=1..1440 can be computed as follows

$$p(i) = \frac{No\ of\ calls\ from\ caller\ at\ time\ i}{Total\ number\ of\ calls\ from\ caller\ at\ all\ time\ intervals}$$

All the computed *p(i)* for i=1..n can be substituted in the Shannon entropy equation to result in an entropy value. The higher the entropy value, the higher is the randomness. Similarly, the lower the entropy value, the lower is the randomness. Therefore, using the computed entropy, we can segregate callers with non-periodic behavior from callers with periodic behavior.

So far, we have discussed algebraic rules that represent the calling patterns

between the callers and callees in 7.2. In 7.3, we discussed matrix operations for deriving useful information about calling patterns. We can use the algebraic rules and matrix operations for solving important problems such as detecting spam and botnets.

7.4. Applications

One of the important problems in IP telephony research is the task of identifying unwanted calls. For detecting these unwanted calls, solutions have to be devised for a number of problems such as determining the legitimacy of the callers, and proactively estimating the callee's interest in receiving calls from different callers. In this section, we discuss some problems and derive solutions to those problems using the rules and operations we have discussed before. The solutions to the problems can be collectively applied to result in a comprehensive solution for identifying wanted calls from legitimate callers.

(1) *Spam Filtering*: Incoming calls from spammers and phishers are considered to be spam and are unwanted to the callee. Therefore, it is highly desirable to filter these unwanted calls even before they reach the end callee. Here we present steps for designing a real-time spam filtering application that filters unwanted spam calls. The procedure outlined here is an example, but a detailed solution can be found in chapter 4.

   (a) Determine callers'/callees' connectivity using frequency and duration communication patterns based on incoming/outgoing calls between the callers and callees as shown in 7.3.1.

   (b) Define a trust matrix F such that each element F_{ij} represents the perceived trust of caller S_i by callee R_j and computed using the connectivity of callers to callees based on incoming and outgoing calls.

(c) Define a reputation matrix M such that each element M_{ij} represents the reputation of caller S_i with respect to callee R_j as shown in Section 7.3.2.1.

(d) Define a spam probability matrix L such that each element L_{ij} represents the spam probability of caller S_i as perceived by callee R_j. Each L_{ij} is computed using the trust and reputation values. One way of computing the spam probability is using a correlation function between the trust and reputation i.e. $L_{ij} = 1 - (\alpha_F F_{ij} + (1 - \alpha_F) M_{ij})$

(e) Define thresholds for the inferred spam probabilities.

(f) Categorize callers as spammers or legitimate callers using the spam probability values and the determined threshold values.

(2) Identifying Broadcast spammers or broadcast callers: A kind of spammers (e.g., telemarketers) broadcasts calls to all the members in the callee's community. These kinds of spammers attain their objective (personal or organizational gain) by attempting to communicate with every member in the community. In this context, it can be useful if the incoming calls are identified to be generated from broadcast spammers so that they can be proactively filtered before they reach the callees. Identifying broadcast spammers can help quarantine unwanted calls destined to all callees in the community. We can derive the set of broadcast callers using the SR_F matrix. When a caller S_i has calls equally distributed across the callees $(S_i R_j) = (S_i R_k)\ \forall\ j,\ k \in 1..m\ \&\ i \in 1..p$, then the caller S_i is a broadcast caller. Additionally, if a group of callers are similar (i.e. they have same call distribution across the callees as shown in Section 7.3.5), and each of them are broadcast callers, then the group of callers share the same callee list and most likely broadcasting from the same domain. The group of broadcast callers can be callers that are close to the callee in the callee's social network or farther away. In

this case, the connectivity and the reputation of the callers can be inferred to check if they are legitimate or spam callers.

(3) Defining Social Groups: We enumerate steps for classifying callers into four social groups as defined in Section 7.2.1. The procedure outlined here is an example, but a detailed solution can be found in Chapter 5.

   (a) Derive closeness of each caller to the callee based on the incoming calls. For deriving this closeness, construct a matrix G^I such that each element $G^I_{ij} = \sqrt{(S_iR_j)^2_F + (S_iR_j)^2_D}$ for $i \in 1..p$ & $j \in 1..m$. Each element G^I_{ij} represents the closeness of caller S_i to callee R_j for $i \in 1..p$ & $j \in 1..m$ based on "incoming" calls to callee.

   (b) Derive closeness of each caller to the callee based on the outgoing calls. For deriving this closeness, construct a matrix G^O such that each element $G^O_{ij} = \sqrt{(R_iS_j)^2_F + (R_iS_j)^2_D}$ for $i \in 1..m$ & $j \in 1..p$. Each element G^O_{ij} represents the closeness of callee R_j to caller S_i for $i \in 1..m$ & $j \in 1..p$.

   (c) A caller's closeness to a callee depends on the closeness based on incoming and outgoing calls from and to the callee. This dependency can be a simple correlation function such as $\alpha_G CI + (1 - \alpha_G)CO$ where CI represents Closeness based on incoming calls and CO represents closeness based on outgoing calls. The factor α_G can be customized based on callee's preferences.

   (d) Define thresholds for closeness (e.g., 0.9-1.0: socially close, 0.7-0.9: socially near, 0.5-0.7: opt-in, 0-0.5: opt-out.)

   (e) For each callee R_j, sort all the callers based on their closeness and group them into different groups using the thresholds defined above to result in social groups.

(4) *Nuisance Computation*: Every real-time call is associated with certain amount of nuisance. This nuisance is less for calls from close people such as family members and friends compared to calls from people such as strangers and

telemarketers. Integrating nuisance computation in unwanted call filtering processes helps in pro-actively identifying the callee's eagerness in receiving incoming voice calls. The nuisance for an incoming voice call can be computed as follows (procedure outlined here is an example, but a detailed solution can be found in Chapter 5)

(a) Define a matrix W that provides information about the callees' eagerness in receiving calls. Each element of the matrix W_{ij} for $i \in 1..p$ & $j \in 1..m$ represents the callee R_j's eagerness in receiving voice calls from caller S_i. For computing W_{ij}, determine the connectivity of caller S_i towards callee R_j based on incoming and outgoing calls. The eagerness of callee R_j to receive calls from caller S_i depends on the amount of connectivity the caller has with the callee.

(b) Define a reciprocity matrix D^R such that each diagonal element D^R_{ij} represents the reciprocity shown by callee R_j for calls from caller S_i for $i \in 1..p$ & $j \in 1..m$ as shown in Section 7.3.3.

(c) Define a Periodicity matrix P such that each element P_{ijk} represents the periodicity of calls from caller S_i to callee R_j for $i \in 1..p$, $j \in 1..m$, $k \in 1..nt$.

(d) Using the matrices W, D^R, and P, define a nuisance matrix N. Each element of matrix N_{ij} represents the nuisance for callee R_j because of calls from caller S_i for $i \in 1..p$ & $j \in 1..m$. One example of nuisance computation is $N_{ij} = \frac{1}{W_{ij}+D^R_{ij}+P_{ij}}$

(e) Define thresholds for nuisance values of different categories of people such as socially close, socially near to customize the filter for receiving selective calls.

(5) *Botnets identification*:Botnet is a term used for collection of compromised systems (known as "bots") that are used as a starting point for generating attacks. VoIP spammers can use bots as a starting point for generating

spam calls. We can design an application that identifies bots using time series analysis. The procedure outlined here is an example, but a detailed solution can be found in [61].

(a) Define a matrix B which records the communication parameters for the calls originating from all the hosts (used by the callers to generate calls). Assume that there are a total number of q distinct hosts that are used for generating calls. Each element of matrix B (i.e., B_{ij}) represents the normalized value of communication parameter j for $j \in 1..np$ (np is the number of parameters where a parameter can be time-of-arrival, frequency, duration etc.) of host i.

(b) Derive Eigen vectors of matrix B. Define a set of Eigen vectors $[e_1^{'}, e_2^{'}, ..e_k^{'}]$ i.e., the first k prominent Eigen vectors that describe the data values in matrix B [60].

(c) Extract the sub matrix H from matrix B that constitutes the normalized values of k parameters that the above Eigen vectors correspond. The matrix H would be then a qxk matrix with each element H_{ij} representing the normalized value of parameter j from host i.

(d) Compute the correlation matrix (X) using the above Eigen matrix. Each element of matrix X (i.e., X_{ij}) represents the correlation of host i with host j for $i \in 1..q$, $j \in 1..k$ i.e., correlation of two hosts with respect to k parameters. The correlation can be computed using the Pearson's correlation coefficient discussed in Section 7.3.6.

(e) Using the correlation values in matrix X, we can use existing clustering techniques such as K-means, Fuzzy C-means, and Hierarchical clustering to cluster hosts with high correlation. Using the classification based on above techniques, all botnets are clustered into a group [61].

(6) *Prediction*: Prediction, one critical research, deals with estimating a future pattern using learning based on past patterns. We believe that prediction

can be researched in context of real-time calls too. It would be very helpful if an application can be designed that predicts whether a particular caller of a given type (e.g., spam, legitimate, socially close, socially near) would call at a given time with a maximum probability and minimum amount of error. Knowing well in advance that a caller would be making a call to him, the callee can make appropriate policies for granting or denying access. Here, we discuss a prediction model that can be used to determine future communication patterns.

(a) Define a matrix A such that each element of the matrix (A_{ij}) represents the entropy of caller S_i with respect to callee R_j for i $\in 1..p$, j $\in 1..m$ using the frequency matrix SR_F. Using the entropy values, we can find the callers that are more predictable.

(b) Similarly, define a matrix T such that each element of the matrix T_{ij} represents the entropy of caller S_i with respect to callee R_j using the inter-arrival time patterns given in the SR_I matrix. Using this, we can predict the time at which a caller or callers is going to call.

Integrating the VSD and ND with the formalism discussed in this chapter will help the filters increase their overall performance. The filters can benefit from the enumeration of communication patterns between the callees and the callers calling them as discussed in 7.2. In addition, the operations presented in 7.3 and probable solutions for some applications presented in 7.4 can be applied for solving some of the existing IP telephony research problems.

All the above discussed solutions account for determining the possibility of the incoming call to be unwanted. In addition to unwanted calls to callees in a VoIP network, there can be unwanted (malicious) VoIP traffic coming into a network that attempts to compromise VoIP network devices. Many articles explain how intruders break into systems [9] [36]. For adequate security management, there is a need to frequently monitor the risk levels of critical network infrastructure and take remedial

actions for preserving the security of the network [20] [98] [72]. We need adaptive risk computation processes as the risk management at one part of the network influences the chances of exploits at other parts of the network. For example, patching a firewall by closing access to port ssh (port 22) limits the possibility of exploiting an internal ssh server. In lieu of this, we present a risk management technique for dynamically updating risk levels of network infrastructure devices given evidence (e.g., evidence of an attack or that of an applied patch).

CHAPTER 8

NETWORK RISK MANAGEMENT

8.1. Introduction

The increase in the size of an enterprise VoIP network is an ever-growing process. With the increase in number of hosts connected to the network, there is always a mounting risk of protecting computers from the outside attacks. Intelligent hackers can create malicious VoIP traffic for compromising VoIP network devices. Improperly configured VoIP network hosts result in increased host vulnerabilities, making more hosts susceptible to outside attacks. Accurate vulnerability analysis requires a deep understanding of both failure and attack modes and their impact on each network component, as well as the knowledge of how components interact during normal and attack modes of operation. For managing the security of a network, security administrators identify security holes by probing the network hosts; assess the risk associated with the vulnerabilities on the computer hosts, and fix host vulnerabilities using patches released by the vendors.

We see frequent release of patches from product vendors to reduce the effect of vulnerability once it is reported. The product vendors, for vulnerability assessment, focus on active prevention methodologies to close vulnerabilities before they are exploited. Patching network hosts supplies a short-term solution for avoiding attacks, but this solution requires closing vulnerabilities in the network hosts and its components. Unfortunately, patching end hosts involves a great deal of human

intervention, time, and money. It requires the administrative staff to frequently monitor end systems using a set of monitoring tools to identify and prevent intrusion. However, capabilities for preventing attacks worsen when state-of-the-art monitoring tools cannot effectively identify new vulnerabilities.

Clearly, security administrators need an effective method to manage the risk presented by the identified vulnerabilities so that they can determine which vulnerabilities to address first. Risk management refers to the process of making decisions that minimize the effects of vulnerabilities on the network hosts. However, in the context of high-exploit probability, risk management is a nightmare to plan with. And also, identifying new exploits and vulnerabilities is difficult. For many years, security engineers have performed risk analysis using economic models for the design and operation of risk-prone, technological systems using attack profiles [20][35][66]. Based on the type of attacker identified, security administrators formulate effective risk management policies for a network. Simultaneously, a great deal of psychological and criminological research has been devoted to the subject; but the security engineers do not use these studies. Many articles explain how intruders break into systems [9][36]. Companies like Psynapse, Amenaza, and Esecurity have built products using the behavior of intruders. To our knowledge, no work has been reported on integrating behavior-based profiles with sequences of network actions to compute resource vulnerability. Thus, the overall goal of this research is to estimate the risk to a critical resource based on attacker behavior and a set of vulnerabilities that the attacker can exploit. This approach implies a more fine-grained repertoire of risk mitigation strategies tailored to the threat rather than using blanket blocking of network activity as their sole response.

8.2. Background

Considerable work has been reported on attacker profiles and risk management. Jackson[35] introduces the notion of behavioral assessment to determine the intent

behind the attack. The proposed Checkmate intrusion detection system distinguishes legitimate use from misuse through behavior intent. But, the paper does not discuss in detail the vulnerable device identification based on the assessed behavior. Rowley[72] views risk analysis as involving threat identification, risk assessment, and steps to be taken for risk mitigation. The potential threats are identified and an estimate of the damage each threat could pose is calculated. Although these studies individually discuss about the behavior and risk, none of them attempt to integrate risk analysis with attacker behavior and the sequence of network actions that can be performed by the attacker. We believe that there is a need for integrating risk mitigation strategies with attacker (e.g., hacker) behavior to help reduce the possibility of impending attacks.

The psychological and criminological research on hacker community defines categories of hackers such as novices, crackers, and criminals based on their intent, skill, and attack proficiency. We believe this research can provide insights into attacker profiling that we can use to improve the risk analysis of network systems. Rogers[69] proposes categorizations of a hacker community and advises hacker profiles derived from intruder behavior. Yuill[98] profiles detection of an on-going attack by developing a profile of the attacker using the information the attacker reveals during attacks. Kleen[40] developed a framework for analysis of hackers by reviewing existing hacking methods, classifications and profiles, with the goal of better understanding their values, skills and approaches to hacking. In addition to the above studies, several works in the literature focus on hacker profiles [41][66][68], but none of them tie the profiles to exploits in the network. All the theories proposed account for the hacker behavior but do not attempt to relate the hacker behavior with exploits and vulnerabilities.

In addition to research that has attempted to categorize network attackers, researchers have been examining the feasibility of using graphing techniques to represent network actions. An attack graph represents all the possible network actions

that can be used to compromise a network component, or exploit a given vulnerability. Each node in the attack graph represents a network action and a sequence of network actions from the root to the leaf node represents a successful attack. Attack graphs are beginning to be used to formalize the risk for a given network topology and exploits. Sheyner[82] models a network by constructing an attack graph using symbolic-model-checking algorithms. Moore[54] documents attacks on enterprises in the form of attack trees, where each path from the root to the end node documents how an attacker could realize his desire of exploiting the host and network. However, current research [54][82][86] does not combine the attacker's behavior with these graph transitions. Loper[47] and McQuade[51] indicate that mapping network actions to social motives is sustained by the available data. Our mechanism marries profiling with chain of exploits, and detects highly vulnerable resources in the network. In addition, behavior profiles are used for calculating the trust of a given attack path. Our work uses theory from criminology, statistical analysis, behavioral-based security, and attack graphs to develop a model for risk management.

8.3. Risk Management

Increasingly, attack graphs (or attack trees) are being formalized to provide models for representing system security. An attack graph can be created using network topology, interconnection between hosts, and various vulnerabilities of each host [54][82][86]. The attack graphs represent a sequence of network actions for exploiting each network resource and, ultimately, the whole network. We used these attack graphs to calculate the vulnerability level and risk to a critical resource in a given VoIP network for a number of attacker profiles. Our procedure consists of five steps which we repeatedly execute until we achieve an optimum security. Our hypothesis is that a relationship exists between network actions and social behavior attributes of the attackers.

8.3.1. Step 1: Creation of an Attacker Profile

The profile of an attacker gives the attacker's expendable resources. Among these resources can be cost, computer and hacking skills, time, attitude, tenacity, perseverance, and motives (such as revenge or reputation) that the attacker would expend when exploiting a vulnerability. Each attack profile has distinct behavioral attribute values for attacker resources. For example, a corporate espionage agent has more money compared to a script kiddie who hacks systems for fun with little or no money. In another example, a corporate insider has more knowledge regarding the enterprise network topology - a valuable resource - whereas external hackers may have only a basic understanding of the topology they are attacking. Once we identify an attacker's resources, we can assign attribute values to those resources. For example, one way of assigning relative attributes for a profile who has low level of skill (e.g., 0.2), medium level of attitude (e.g., 0.6) and high level of time (e.g., 0.8).

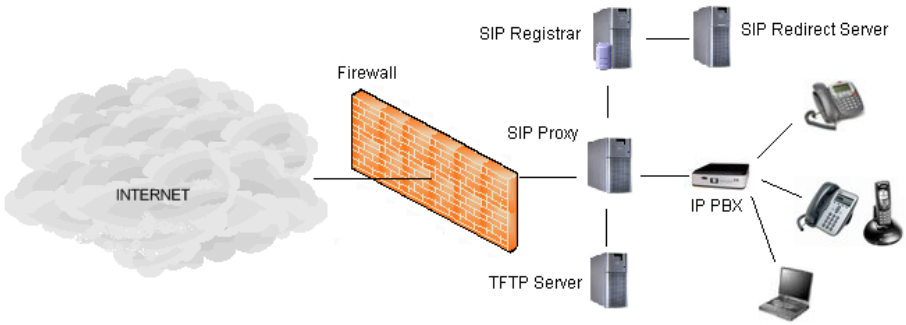

FIGURE 8.1. Network Diagram

8.3.2. Step 2: Creation of Attack Graphs

Once we created the attacker profiles, we develop attack graphs to aid in our analysis of the risk. For a given network topology as shown in Figure 8.1, we can derive the sequence of network actions that can be executed to exploit various network host vulnerabilities or attacks.

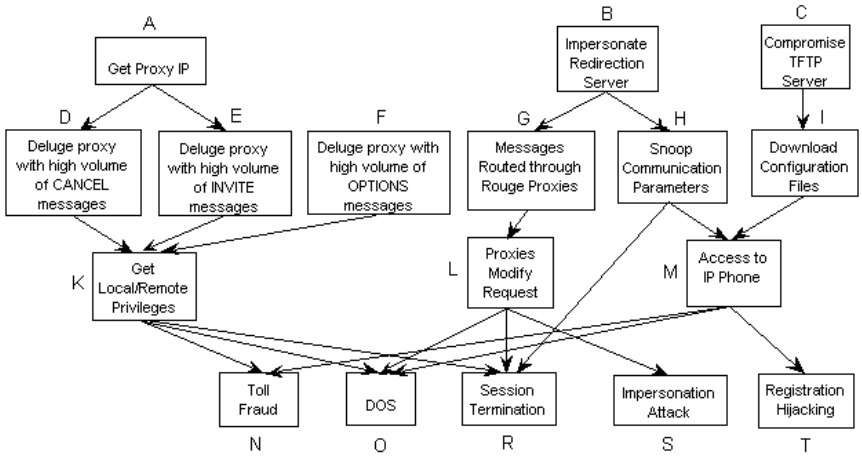

FIGURE 8.2. An example attack graph with a chain of exploits

Figure 8.2 is an attack graph that represents the network actions that can be executed to perform different attacks such as Toll Fraud and Dos. Each node in the graph represents a network action and a path from root to leaf represents a successful attack. Using this graph, we can learn how intruders culminate a sequence of state transitions to successfully achieve an attack. For example, an attack path (see Figure 8.2) can be a sequence of events such as: impersonate the redirection server, eavesdrop the communication parameters, and finally terminate the session causing a session termination attack.

8.3.3. Step 3: Assigning Behavior Attributes to Attack Graph Nodes

For a given attacker profile, we can assign the attack graph nodes using a set of behavior attributes such as computer and hacking skills, time, attitude, tenacity, cost of attack, and techniques for avoiding detection. For understanding these attributes, we conducted an online survey that helped us identify the values for attributes such as skill, time, and attitude for people with varied behavior (Appendix E). Using the assigned behavior attributes, we construct individual profile attack graphs by documenting attack paths that different profiles can execute. For example, Figure 8.3 represents attack graphs constructed for two example profiles A and B respectively

for three attributes of skill, time and attitude. A measure of these attribute values gives the amount of risk associated with the profile.

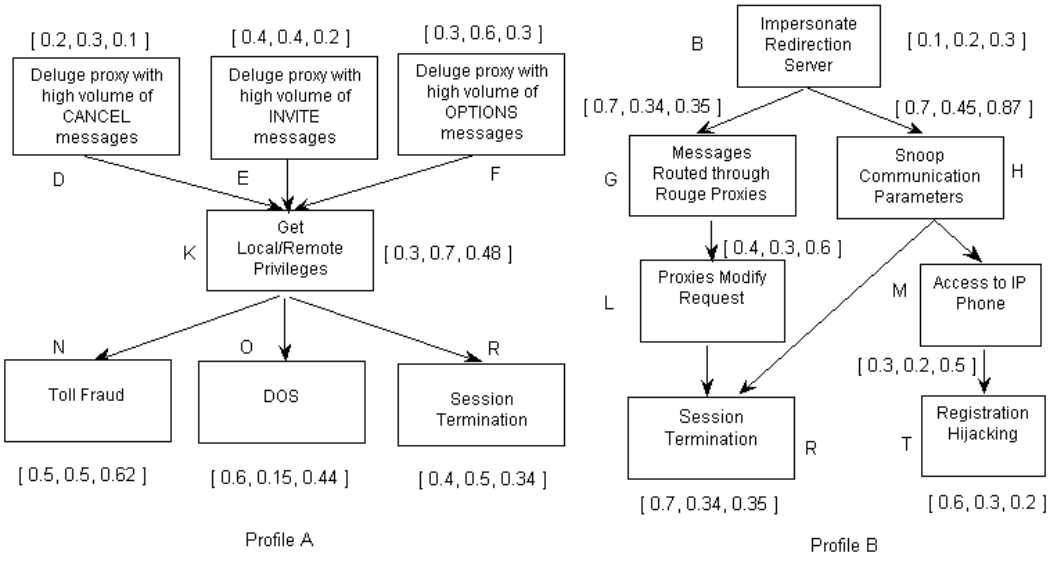

FIGURE 8.3. Attack paths from Figure 8.2 for two example profiles (3 Tuple skill, time, attitude).

8.3.4. Step 4: Risk Computation

In this step, using the set of paths, attributes, and attacker profile, we calculate a risk level for critical resources. For this, we discuss a Bayesian networks-based estimation technique to calculate a resource's aggregated risk value. We mark a resource as attack prone if this risk value exceeds a predetermined threshold value.

8.3.4.1. *Bayesian Networks for Risk Inference.* A Bayesian network is a graphical model for showing probabilistic relationships among a set of variables (representing nodes) in a graph. Each node is represented by a random variable that is associated with a set of Probability Distribution Functions. Therefore, the attack graphs can be modelled as Bayesian Networks by reducing them to causal graphs and associate the graph nodes with probabilities. Using monitoring or intrusion detection systems,

protocol state machines and traffic patterns observed between states in the state machine, the initial subjective beliefs can be formulated. Any deviation from normal behavior gives the evidence for inferring the posterior probabilities using Bayesian inference techniques. Using Bayesian statistics, we can quantify the available prior probabilities or knowledge based on the evidence collected at any node. The evidence thus collected updates the subjective belief about all other random variables' (nodes') probability distributions. The posterior probability distributions of the graph nodes represent the updated subjective beliefs of the ability of the attacker to compromise respective network actions. These updated probability values represent the updated risk levels associated with the network resources. These posterior probability calculations are done before and after the exploits are patched to estimate the new risk level of the critical resources.

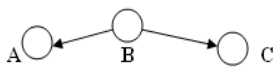

FIGURE 8.4. Representing Conditional Probability.

Figure 8.4 is a simple Bayesian network with three nodes. For this network, the joint probability distribution function can be shown to be

$P(A, B, C) = P(A|B)P(B)P(C|B)$

Therefore, for a set of variables in $X = X_1, X_2, ...X_n$, the probability distribution would be

$$P(X = X_1, X_2, ...X_n) = \prod_{i=1}^{n} P(X_i|parents(X_i))$$

8.3.4.2. *Inference Based on attacker profiles.* We can initialize a given profile's attack graph using different methods such as expert knowledge, past observations. In this dissertation, we discuss an initialization technique using a survey (Appendix E). This initialization helped us assign behavior attribute values. However, in this chapter, we use an example assignment of values of behavioral resources for different

profiles. For example, consider the attack-graph of profile A in Figure 8.3 for a quantifying variable "attitude" required to carry out the Toll Fraud, Dos, and Session Termination attacks (represented by nodes N, O, and R).

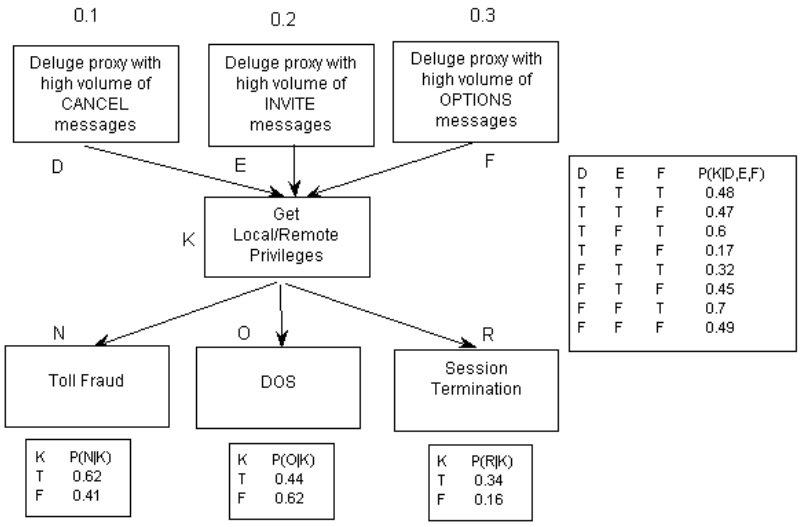

FIGURE 8.5. A small Bayesian Causal graph.

Assume each of the nodes of the profile attack graph to be associated with two states "yes" or "no," and the initialized probability values to the nodes be as shown in Figure 8.5. With available initial knowledge, we compute the posterior probability for observed evidence. If an attacker generates an high volume of CANCEL messages represented by node D, then we can calculate the probability that the attacker can carry out the Toll Fraud attack at node N using $P(N|D)$ as given below.

(25)
$$P(N|D) = P(N, K|D) + P(N, \sim K|D) = P(N|K)P(K|D) + P(N| \sim K)P(\sim K|D)$$

$$where P(K|D) = P(K, E|D) + P(K, \sim E|D)$$

(26)
$$i.e. P(K|D) = P(K|D, E)P(E) + P(K|D, \sim E)P(\sim E)$$

$$P(K|D,E) = P(K,F|D,E) + P(K,\sim F|D,E)$$

$$i.e. P(K|D,E) = P(K|D,E,F)P(F) + P(K|D,E,\sim F)P(\sim F)$$

(27) $$(0.48)(0.3) + (0.47)(0.7) = 0.473$$

$$and P(K|D,\sim E) = P(K,F|D,\sim E) + P(K,\sim F|D,\sim E)$$

$$i.e. P(K|D,\sim E) = P(K|D,\sim E,F)P(F) + P(K|D,\sim E,\sim F)P(\sim F)$$

(28) $$(0.6)(0.3) + (0.17)(0.7) = 0.299$$

(29) $$Therefore P(K|D) = (0.473)(0.2) + (0.299)(0.8) = 0.3338$$

$$Similarly P(\sim K|D) = P(\sim K,E|D) + P(\sim K,\sim E|D)$$

(30) $$i.e P(\sim K|D) = P(\sim K|D,E)P(E) + P(\sim K|D,\sim E)P(\sim E)$$

$$where P(\sim K|D,E) = P(\sim K,F|D,E) + P(\sim K,\sim F|D,E)$$

$$i.e. P(\sim K|D,E) = P(\sim K|D,E,F)P(F) + P(\sim K|D,E,\sim F)P(\sim F)$$

(31) $$= (0.52)(0.3) + (0.53)(0.7) = 0.527$$

$$and P(\sim K|D, \sim E) = P(\sim K, F|D, \sim E) + P(\sim K, \sim F|D, \sim E)$$

$$i.e. P(\sim K|D, \sim E) = P(\sim K|D, \sim E, F)P(F) + P(\sim K|D, \sim E, \sim F)$$

(32) $$= (0.4)(0.3) + (0.83)(0.7) = 0.701$$

(33) $$Therefore P(\sim K|D) = (0.527)(0.2) + (0.701)(0.8) = 0.6662$$

Therefore, finally by using Equations (25), (29) and (33), we get

(34) $$P(N|D) = (0.62)(0.3338) + (0.41)(0.6662) = 0.48$$

i.e., given that the attacker carries out the network action represented by node D, the probability the attacker will eventually carry out the Toll Fraud attack represented by node N is equal to 0.48.

For patch-management purposes, it would be useful to know an exploit's actual cause so that it can be patched to prevent the attack in the first place. We can estimate this by calculating the probability of the occurrence of a given cause based on observed evidence. For example, in Figure 8.5, if we have evidence regarding a Toll Fraud attack represented by node N, then we can compute the probability that the attacker generated CANCEL messages to deluge the proxy represented by node D using $P(D|N)$. From Bayes theorem we have

(35) $$P(D|N) = \frac{P(D)P(N|D)}{P(N)}$$

We already have $P(N|D) = 0.48$ from (8)

$$and P(N) = P(N, K) + P(N, \sim K) = P(N|K)P(K) + P(N| \sim K)P(\sim K)$$

(36) $$where P(K) = P(K, D) + P(K, \sim D) = P(K|D)P(D) + P(K| \sim D)P(\sim D)$$

We have $P(K|D) = 0.3338$ using (2) and (5)

$$and P(K| \sim D) = P(K, E| \sim D) + P(K, \sim E| \sim D)$$

(37) $$i.e. P(K| \sim D) = P(K| \sim D, E)P(E) + P(K| \sim D, \sim E)P(\sim E)$$

$$where P(K| \sim D, E) = P(K, F| \sim D, E) + P(K, \sim F| \sim D, E)$$

$$i.e. P(K| \sim D, E) = P(K| \sim D, E, F)P(F) + P(K| \sim D, E, \sim F)P(\sim F)$$

(38) $$= (0.32)(0.3) + (0.45)(0.7) = 0.411$$

$$P(K| \sim D, \sim E) = P(K, F| \sim D, \sim E) + P(K, \sim F| \sim D, \sim E)$$

$$i.e. P(K| \sim D, \sim E) = P(K| \sim D, \sim E, F)P(F) + P(K| \sim D, \sim E, \sim F)P(\sim F)$$

(39) $$= (0.7)(0.3) + (0.49)(0.7) = 0.553$$

Therefore, $P(K| \sim D) = (0.411)(0.2) + (0.553)(0.8) = 0.5246$
and $P(K) = (0.3338)(0.1) + (0.5246)(0.9) = 0.50552 \implies P(\sim K) = 1 - P(K) = 0.49448$

Therefore, P(N) = (0.62)(0.50552) + (0.41)(0.49448) = 0.5161

Therefore, $P(D|N) = \frac{0.1*0.48}{0.5161} = 0.09300$ i.e., given the leaf node N in Figure 8.5 has been carried out, the probability that the root node D has been used is 0.093. We perform this analysis for each node in the attack graph, documenting the posterior probabilities of all nodes obtained using our analytical model. We validated these estimated (posterior probability) values using HUGIN DEMO [33]. HUGIN employs Bayesian inference techniques to create and test belief networks. Depending on the graph structure and initial values for each node encoded in CPT (Conditional Probability Tables), HUGIN initializes the graph. For given evidence at a node, HUGIN re-computes and updates the probability values of all nodes. At any given time, the new probability values portray the nodes' current vulnerability level. Our analytical model takes into account any value of new evidence in the range [0 1]. We do this by using available prior probabilities depicted in the CPT tables as shown in Figure 8.5 and Bayesian inference along with the conditional probability analysis. Here we show the proof of our analytical model for extreme values and validate using HUGIN for observed evidence. Table 8.1 gives the posterior probabilities of all other nodes of the attack graph represented in Figure 8.5 given evidence at the leaf node N using our analytical model and HUGIN.

For given evidence that the network action of compromising network resources represented by leaf nodes in the graph is observed, we document the posterior probabilities of the root nodes of the graph along all the attack paths.

Table 8.2 presents the computed posterior probability values for a given profile and attack path DN of Figure 8.5. This procedure is performed for all attack paths and profiles that can achieve an effective attack. Hence, for a given resource, we can infer all the probable attack paths that can lead to successful exploitation.

Similar to the graph shown in Figure 8.5 and using our analytical model, we can periodically update the risk estimate for the entire attack graph shown in Figure 8.2.

TABLE 8.1. Computed probabilities of nodes in Figure 8.5 using our analytical model and HUGIN for given evidence of an attack at node N.

Node	Our Analytical Model	Hugin
D	0.093	0.095
E	0.1948	0.196
F	0.297	0.306
K	0.412	0.425
O	0.55	0.543
R	0.17	0.199

TABLE 8.2. Bayesian probability at the root node of attack path given evidence at the leaf.

Path	Skill	Time	.	.	.
1	0.182	0.33	0.093		
2					
...					

Consider that the conditional probability tables for the nodes depicted in Figure 8.2 are as shown in tables given in appendix F. With these values in the CPT tables, the graph can be initialized based on the parent child relationships. The probability values based on this initialization is given in Table 8.3.

Now, with the initial values established, we can compute the posterior probability values of all the nodes given evidence at any part of the graph. Example, for observed evidence that the network actions represented by one or more root nodes have been compromised, the probability with which the attacker can perform the network action represented by node P is given in Table 8.4.

TABLE 8.3. Initialized values using initial values in CPT tables.

Node	Prob.	Node	Prob.	Node	Prob.
A	0.4	B	0.63	C	0.34
D	0.424	E	0.51	F	0.3180
G	0.4192	H	0.3537	I	0.4826
K	0.5058	L	0.3594	M	0.5232
N	0.3971	O	0.3409	R	0.5474
S	0.5203	T	0.5477		

TABLE 8.4. Probability of exploitation of node N.

Node	Our Analytical Model
A	0.6571
B	0.3407
C	0.3408
A&B	0.3427
B&C	3406
A&C	003428
A&B&C	0.3426

8.3.4.3. *Relating Risk, Behavior and Penetration.* As we described before, we believe that sequence of network actions carried out by an attacker relates to social behavior. We attempt to derive the relation between vulnerability of a given resource and the network penetration an attacker can achieve in exploiting the network. The penetration represents the depth of the graph the attacker can achieve using his available resources. This can be computed by initializing the probability of each node in each attack path and inferring the posterior probability given evidence at a node, usually the leaf node i.e. the node representing the final network action for a successful attack. Figure 8.6 is a part of an attack graph of Figure 8.2 and the probabilities

of the nodes are represented using Conditional Probability Tables (CPT). The CPT tables give the probability of nodes given the value of its parents. Assume each node to be in one of two states, either yes or no.

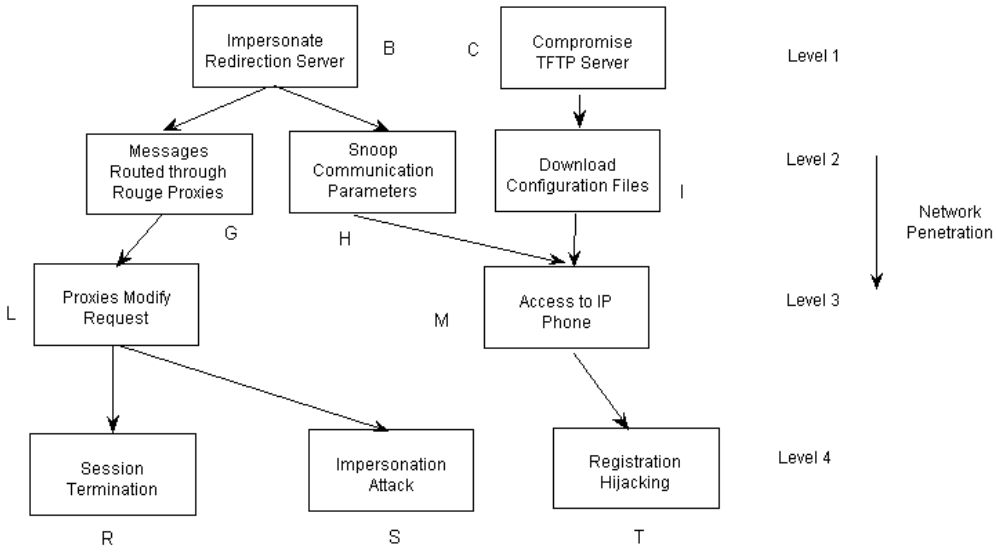

FIGURE 8.6. Example sub-attack graph from Figure 8.2

For inferring network penetration, we consider five nodes (B, G, L, R and S) and three profiles of attack behavior. These profiles are Opportunists-people with criminal behavior (e.g., Corporate Insiders- attackers having access to network resources, Corporate Espionage - spies), Hackers-people with hacking behavior, and Explorers-people who are liberal minded and believe in open doors. For the five nodes in analysis, assume that the CPT tables would be as shown in Table 8.5 and Table 8.6 for the chosen three profiles. In reality, initialisation of CPT tables is carried out by analyzing a statistical data from an interview or a survey (Appendix E).

The values reported within the CPT tables describe the behavior of each profile. For example, those who fall within the profile of opportunists have more profound knowledge of the corporate network topology than others who belong to the other two profiles. Thus, the risk opportunists pose is greater than the risk posed

TABLE 8.5. Probability of nodes B, G, L of Figure 8.6 given their parents (B does not have a parent).

Profile	P(B)	P(G) given B=yes	P(G) given B=no	P(L) given G=yes	P(L) given G=no
Opportunities	0.8	0.75	0.82	0.85	0.7
Hackers	0.6	0.7	0.31	0.51	0.46
Explorers	0.4	0.52	0.36	0.48	0.32

TABLE 8.6. Probability of nodes R and S of Figure 8.6 given their parents.

Profile	P(R) given L=yes	P(R) given L=no	P(S) given L=yes	P(S) given L=no
Opportunities	0.71	0.83	0.69	0.77
Hackers	0.3	0.57	0.52	0.63
Explorers	0.62	0.41	0.44	0.62

by the other profiled attackers. The probabilities of hackers being a risk are understandingly less compared to opportunists because of their limited resources. Liberals possess the least skill, time, and attitude (Appendix E); hence, are of least risk to the network. Thus, the probability that they will be successful is considerably lower than those of the other categories. In Figure 8.6, for inferring the network penetration, we compute the posterior probabilities of the nodes for evidence that an attacker would be able to reach the leaf node and successfully executes an attack. These posterior probabilities of the nodes represent a measure of the attacker's ability to compromise the network action represented by them. Using these posterior probabilities, the network penetration can be directly inferred. For example, if evidence regarding the happening of the network action represented by node S is observed, then nodes B, G, L, and R are directly affected. The inferred posterior probabilities

of the nodes directly affected by the observed evidence using our analytical model (see Section 8.3.4.2) are given in Table 8.7.

TABLE 8.7. Bayesian Inference for directly affected nodes due to evidence at node S.

Profile	P(B)	P(G)	P(L)	P(R)
Opportunities	0.8002	0.7609	0.7975	0.7343
Hackers	0.5991	0.5416	0.4395	0.4513
Explorers	0.3980	0.4112	0.3102	0.4751

The risk values shown in Table 8.7 can be plotted against network penetration (level number of nodes B, G, L, and R from Figure 8.6) of the attacker.

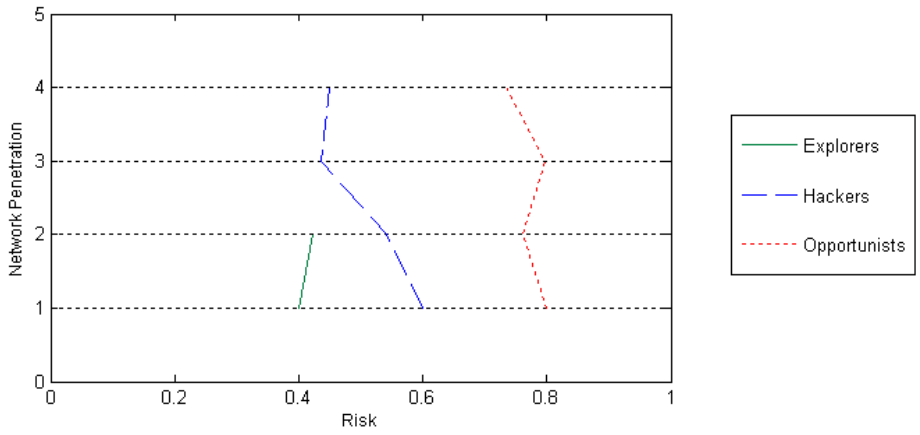

FIGURE 8.7. Relating risk, behavior and penetration for an attribute of all profiles.

Figure 8.7 represents the relationship between risk, behavior, and network penetration for the three profiles. In reality, the CPT values will be a range instead of a single value [47]. Assume that the conditional probability tables representing a range of values for the five nodes in analysis (B, G, L, R and S) of Figure 8.6 are as given in Table 8.8 and Table 8.9 (α/β for each node represents the range of probabilities in which the values of all attributes of the profile extend).

TABLE 8.8. Given three profiles, range of probabilities of nodes B, G and L of Figure 8.6 given their parents.

Profile	P(B)	P(G) given B=yes	P(G) given B=no	P(L) given G=yes	P(L) given G=no
Opportunities	0.8/0.92	0.75/0.87	0.82/0.94	0.85/0.97	0.7/0.82
Hackers	0.6/0.75	0.7/0.85	0.31/0.46	0.51/0.66	0.46/0.61
Explorers	0.4/0.65	0.52/0.71	0.36/0.56	0.48/0.73	0.32/0.67

TABLE 8.9. For given three profiles, range of probabilities of nodes R, S of Figure 8.6, given their parents.

Profile	P(R) given L=yes	P(R) given L=no	P(S) given L=yes	P(S) given L=no
Opportunities	0.71/0.83	0.83/0.94	0.69/0.82	0.77/0.89
Hackers	0.3/0.45	0.57/0.72	0.52/0.67	0.63/0.78
Explorers	0.62/0.84	0.41/0.63	0.44/0.68	0.62/0.81

For the values of the nodes given in the above conditional probability tables, we can infer the posterior probability range of all the profiles. Table 8.10 represents the inferred probability range for nodes B, G, L, and R for observed evidence at node S.

TABLE 8.10. Inferred probability range of the three profiles for directly affected nodes B, G, L, and R.

Profile	P(B)	P(G)	P(L)	P(R)
Opportunities	0.8/0.92	0.76/0.874	0.7975/0.974	0.7343/0.835
Hackers	0.5991/0.75	0.57/0.75	0.45/0.61	0.43/0.55
Explorers	0.3980/0.649	0.4112/0.655	0.3102/672	0.4751/0.771

From Table 8.10, we infer that for all attributes, the probability of node B falls in the range [0.8, 0.92] for the opportunists and within the range [0.398, 0.649] for the explorers. For the given profiles, the relation between the risk, behavior and penetration looks as in Figure 8.8.

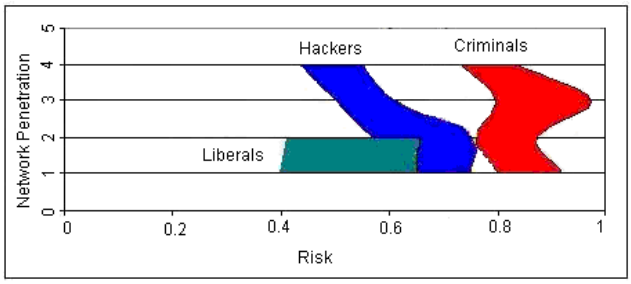

FIGURE 8.8. Relation between risk and network penetration

We can extrapolate the relationship depicted in Figure 8.8 for each of the three profiles and attributes, as shown in Figure 8.9. The relationships shown between behavior, risk, and network penetration suggest that certain behaviors overlap regardless of the type of threat.

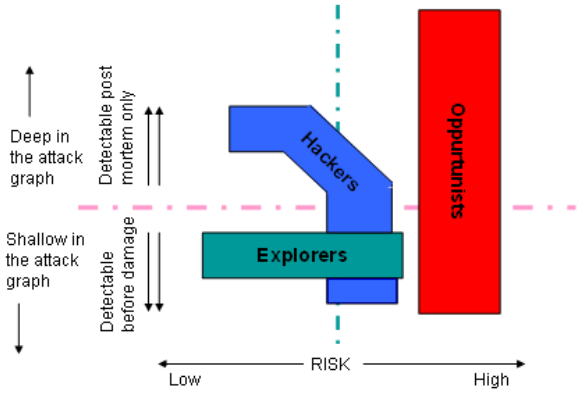

FIGURE 8.9. Relation between risk, behavior and network penetration

8.3.5. Step 5: Optimizing the Risk Level

Risk management is frequently conducted for limiting risk of network devices. This risk management involves processes such as patching exploits or vulnerabilities, changing network configurations (e.g., moving the firewall to another location in the topology, changing the firewall rules, or deploying intrusion detection systems). Managing risk (e.g., patching a host vulnerability) at one part of a network affects the amount of risk at other parts. Our risk computation scheme (steps outlined in Section 8.3.1 - Section 8.3.4) can be performed repeatedly to obtain an optimum risk value.

Updating Risk after patching: Consider the attack graph represented in Figure 8.5. From Section 8.3.4.2, we have used Bayesian inference techniques to infer the risk associated with a given profile. Given that the root node D is exploited, we inferred the Toll Fraud attack represented by node N will be carried out with a probability of 0.48 (Equation (34)). And also, we inferred that, given that the leaf node N was exploited, the probability that the root node D was used as a starting point for the attack path is 0.093. Now, consider that we patch the proxy so that an attacker would not be able to carry out the Toll Fraud attack (node N). By doing this, the attribute values {skill, time, and attitude} are considerably reduced for node N. After reducing the attribute values at node N, consider that the initial values of the nodes for the same attribute value "attitude" are as shown in Figure 8.10.

Now, with these values, we compute the new probability of Node N using Bayesian inference techniques. Therefore, using Equations (25)- (34), we can derive the new probability that the Toll Fraud attack at node N can be exploited given that the attacker has exploited the root node D is given to be 0.0066. In addition, with the above values of nodes and Equations (35)- (39), we can derive the probability that the root node D was used given that the leaf node N was exploited is given to

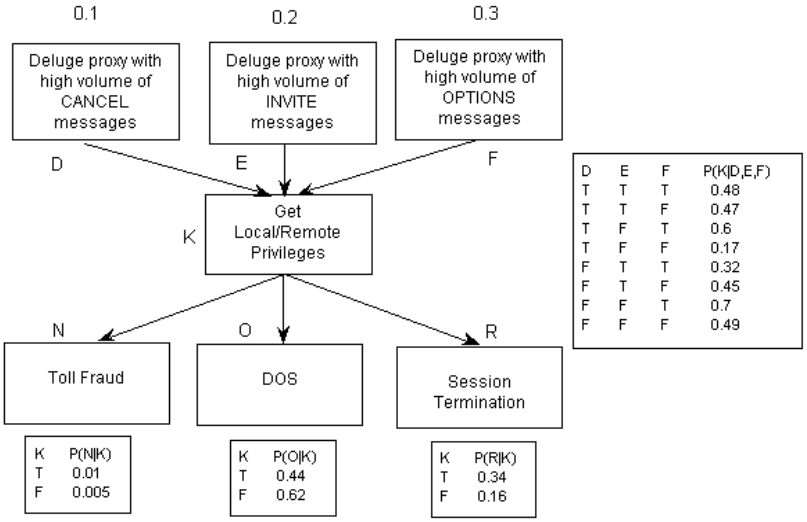

FIGURE 8.10. Modified attack graph of Figure 8.6 after patching N.

be 0.088. In this way we dynamically infer the risk and the new values represent the updated risk levels for the observed evidence.

Inferring Network Penetration after patching: Network penetration also changes as a result of patching. Consider we patch node R so that the attacker would not be able to carry out the session termination attack. When a node is patched, the probability values of nodes B, G, L and R for opportunists (such as corporate insiders and espionage agents) would be same as shown in Table 8.5 and Table 8.6 because patching does not affect their ability to penetrate. However, the values for two profiles of hackers and explorers change. Assume that the changed probability values for the two profiles Hackers and explorers are as given in Table 8.11.

Now with the above values after patching node R, the posterior probabilities of nodes can be re-computed to result in updated values. These updated values are different for the two profiles of hackers and explorers as shown in Table 8.12.

From Table 8.12, we can observe that the posterior probabilities of the profiles of hackers and explorers are reduced, thus, decreasing the risk posed by them. We can plot the above values of risk with the network penetration (level number of

TABLE 8.11. Probability of node R (Figure 8.6) given its parents after patching local over flow vulnerability for the two profiles that are affected by the patching process.

Profile	P(R) given L=yes	P(R) given L=no
Opportunists	0.71	0.83
Hackers	0.19	0.24
Explorers	0.01	0.005

TABLE 8.12. Updated node values due to evidence at node R after patching local over flow vulnerability at node Q.

	Before Patching				After Patching			
Profile	P(B)	P(G)	P(L)	P(R)	P(B)	P(G)	P(L)	P(R)
Opportunists	0.8	0.76	0.79	0.73	0.8	0.76	0.79	0.73
Hackers	0.599	0.54	0.43	0.45	0.599	0.54	0.43	0.218
Explorers	0.398	0.41	0.31	0.475	0.398	0.41	0.31	0.022

nodes B, G, L, and R from Figure 8.6) the profiles can attain. Figure 8.11 presents the attainable network penetration of the three profiles after patching the local overflow vulnerability at node Q.

It can be clearly seen that a marked difference is observed in the case of profile of a hacker. This profile attacker is limited to a network penetration of level 3 when we patched node R, whereas when there was no patching involved, hackers had a network penetration until level 4 (see Figure 8.7). The network penetration of the other profile, Opportunists, is unaffected by patching in this instance (e.g. Corporate Insiders have inside access even after patching). System administrators can, however, limit network penetration possible by the people having opportunists' profile by formulating policy rules that restrict access to individuals who have displayed

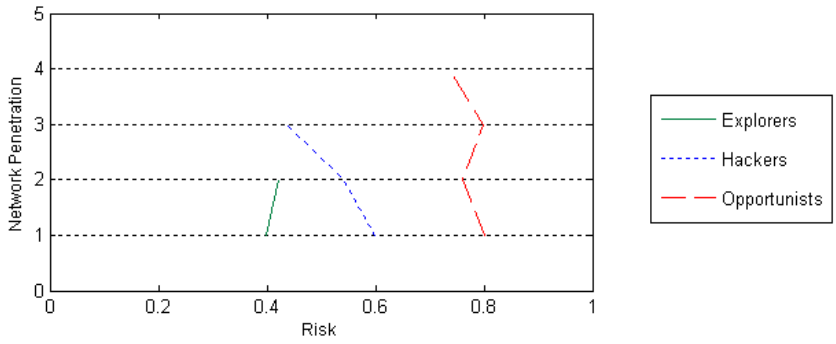

FIGURE 8.11. Network penetration of all the profiles after patching the local overflow vulnerability

attributes similar to a opportunist. For example, a policy rule advocating an extra line of defense may limit the access to the network by the opportunists and, thereby, limit how deeply they can penetrate the network and, thus, reduce risk presented by individuals who fit this profile.

CHAPTER 9

VOIP CALL WEB INTERFACE

The VSD discussed in Chapter 4 is a passive SIP server that analyzes incoming VoIP calls for spam behavior. If the call is identified to be spam, the call is blocked and is not forwarded to the callee. Alternatively, if the call is designated as legitimate, it is forwarded to the callee. It can be useful for the callee if an interface can be provided that the callee can use for keeping track of the previous calls. To realize this objective, we have built a web-interface to the VSD. Using this interface, the callee would be able to

- 1. Keep track of all the allowed, filtered, blocked calls.
- 2. Configure set of legitimate and spam callers.
- 3. Configure customized filter options

Figure 9.1 represents the login screen for the web-interface. Here, a registered user with VSD can use his username and password for logging into the interface.

Figure 9.2 represents the call-lists provided by the interface. The call lists that are shown are the filtered calls (calls that have been designated as spam using adaptive learning), blocked calls (calls that have been blocked based on the callee's blacklist), and allowed calls (calls that have designated as legitimate and forwarded to the callee). The interface allows easy navigation and updating of call list records if in case the callee intends to delete them.

Figure 9.3 represent the list of callers encoded into white and black list. Here, the callee can configure his own set of wanted and unwanted people from whom they would like to receive and resent calls respectively. The callers in the blacklist are directly blocked at VSD and the call records are updated in the blocked calls list.

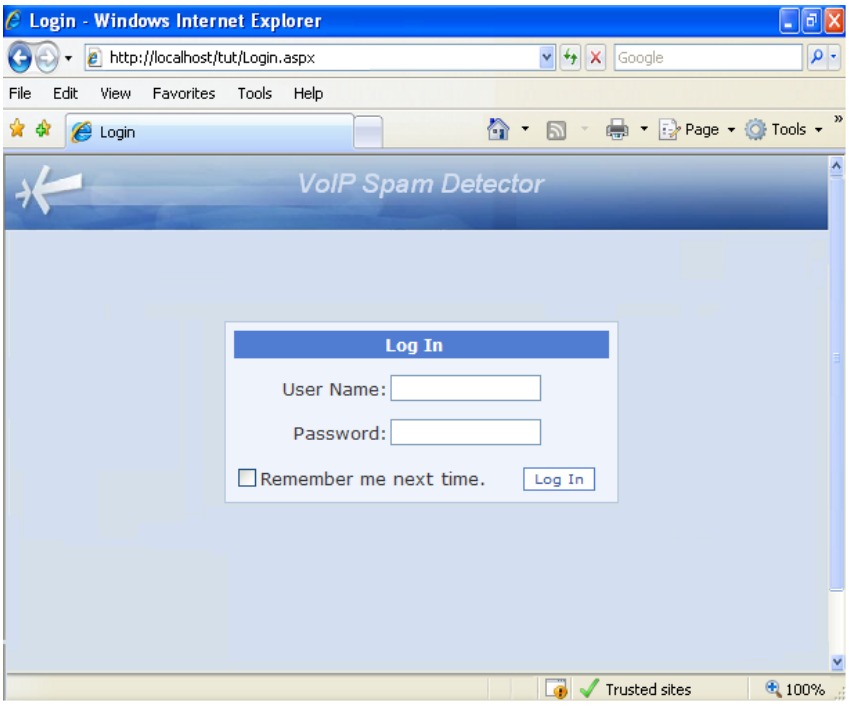

FIGURE 9.1. Login screen for the web-interface

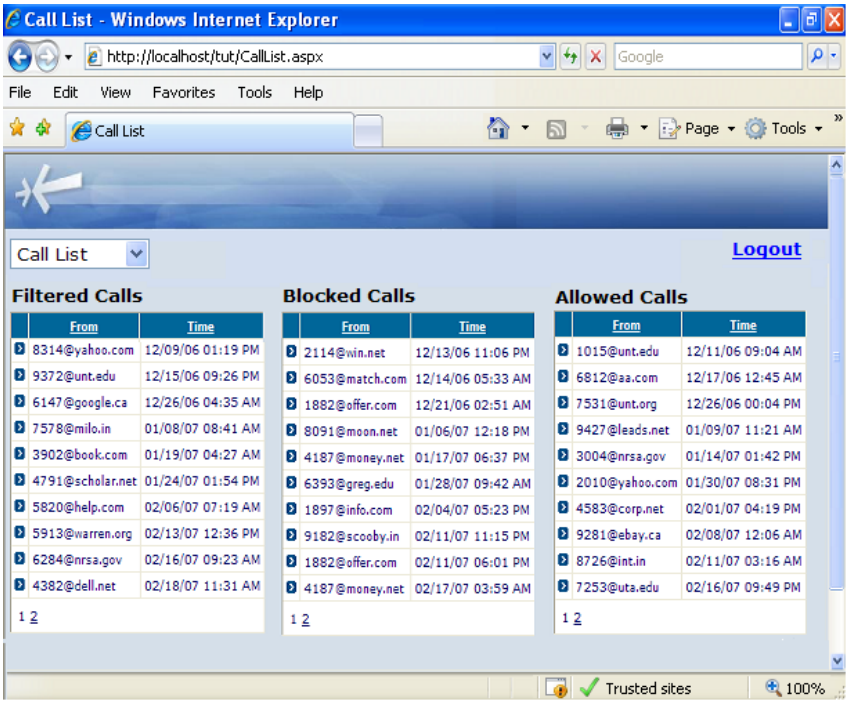

FIGURE 9.2. Previous call list of callee

Alternatively, the callers in the whitelist are directly forwarded to the callee and the call records in the allowed calls list is updated.

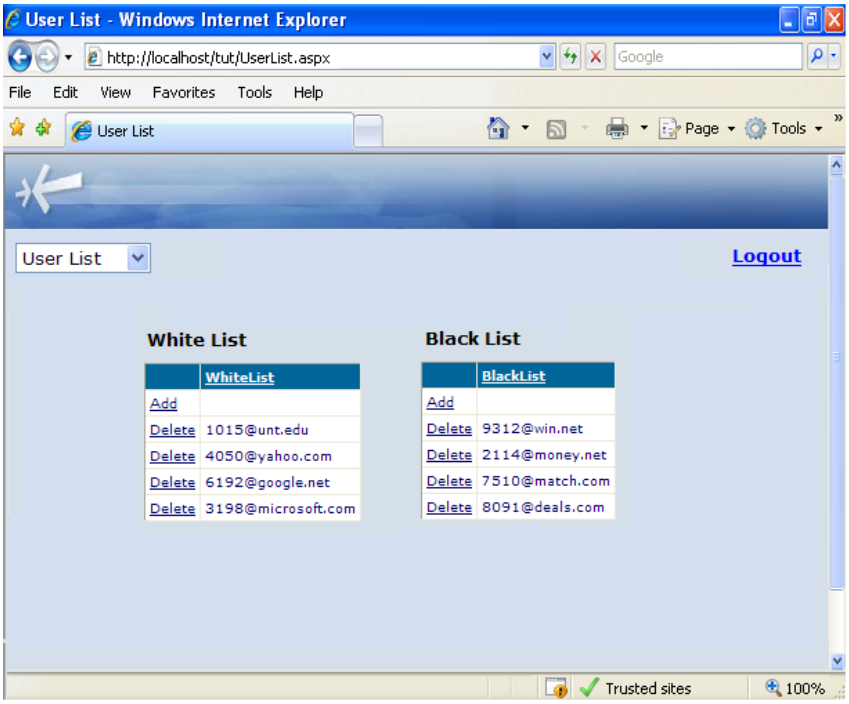

FIGURE 9.3. Callee's callers List

Figure 9.4 represents a list of parameters the callee or the filter administrator will be able to configure. These list of parameters include threshold values, set of spam users, hosts, and domains. Depending upon these customized parameter values, the VSD updates the spam and legitimate behavior of the call participants.

Therefore, using the interface, the callee can configure his own customized filtering preferences. Depending upon the preferences, VSD learns the behavior of the callee himself and the callers calling the callee.

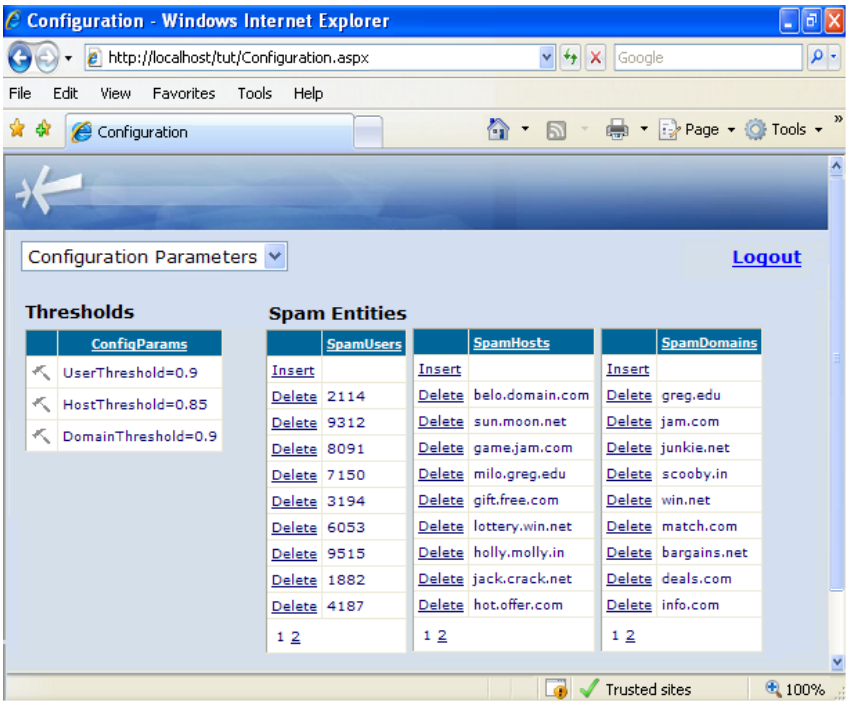

FIGURE 9.4. Configuration parameters

CHAPTER 10

CONCLUSION

The research presented in this dissertation discusses models for identifying unwanted calls coming into a VoIP network. In this chapter, we first summarize all the solutions presented in the previous chapters. Next, we discuss the applicability of the solution in limiting unwanted voice calls. In addition, we present limitations of the presented solution and suggestions to overcome them. Finally, we end the chapter by describing the problems we faced during the course of this research work.

10.1. Summary

VoIP has become a key enabling technology for transmitting voice packets on the open IP network. In addition, the Internet being an open network virtually eliminates geographic limitations for placing calls. However, one challenge that the VoIP networks face is the problem of unwanted calls. These calls can be SPIT calls from spammers (unauthorized callers) or unwanted calls from legitimate callers who have interacted with the callees in the past. For limiting these unwanted calls, we discussed an unwanted call detection framework that analyzes incoming voice calls. This framework involves a VoIP spam detector for detecting SPIT calls, a nuisance detector for pro-actively inferring the nuisance of incoming calls, and an automatic threshold calibration mechanism for fast and adaptive learning of the mutual behavior between the caller and the callee.

The VoIP spam detector includes a multi-stage adaptive model based on the social notions of trust and reputation of the caller with respect to the callee. The trust model involves a Bayesian learning mechanism for updating history and recomputing the new trust values based on past transactions. The reputation model involves a Bayesian Network mechanism for adaptively inferring the reputation of the

caller domain. For this inference, the model takes into account the network topology and draws inferences based on the parent child relationships that exist between the nodes of the topology graph. VSD also takes into account the domain level trust of the caller with respect to all the callee's community members. Using these trust and reputation models, the authenticity of the caller is established, and the VSD can make a decision whether to forward the call or quarantine it.

The calls that are forwarded by the VSD are analyzed by the Nuisance Detector for inferring the nuisance associated with them. This nuisance estimation is based on callers behavior, the callee's tolerance and presence, and the social closeness between the caller and callee. We presented a statistical model for inferring the caller's behavior and the callee's tolerance. In addition, we discussed a model for inferring the social closeness of different callers calling a callee. Using the inferred social closeness, we divided the callers making calls to the callee into four groups

- Socially close members
- Socially near members
- Opt-ins
- Opt-outs

The nuisance detector uses the inferred social closeness of the callers along with other behavioral patterns (such as periodicity of the caller and reciprocity of the callee) to compute the nuisance of voice calls from those callers. Inferring the nuisance of incoming voice calls helps in limiting unwanted calls from reaching the callee.

VSD and ND analyze and learn the behavior of callers over a period of time. For their optimum performance, it is imperative that the filters also take into account the callees' preferences for quarantining incoming calls. However, due to rapidly changing preferences of callees', it is seldom observed that the filters' performance is not optimal. To overcome this limitation, we discussed an automatic calibration mechanism using Receiver operating Characteristics Curves that can be used for dynamically updating the thresholds of the VSD and ND for every call processed by them. The

automatic calibration mechanism aids the VSD and ND to quickly converge to their optimum threshold values thereby enabling both of them to deliver an optimum performance.

The VSD and ND learn and update knowledge about spam and legitimate callers from time to time. During this learning period, it is possible that the VSD and ND might block calls from genuine callers and forward calls from spammers. For reducing these false alarms, the detectors can benefit by integrating knowledge about general calling patterns of the callees inside the callee-domain. To present these calling patterns, we discussed a human/social dynamics that exist between communicating individuals in the way calls are generated, handled, and received. We believe that this dynamics can be used for detecting and filtering unwanted calls. To present this dynamics, we first enumerated normal calling patterns that is observed between callees and callers. Next, we discussed deriving some call-constructs (using operations on caller-callee communication matrices) that can be used for determining the legitimacy of calls. Finally, we discussed how the calling patterns and operations can be grouped to provide solutions for some existing IP telephony problems. The VSD and ND can integrate this dynamics for fast and adaptive learning, and therefore, result in minimal false alarms.

The VSD and ND are used for filtering unwanted calls to the end callee. In addition to being unwanted, these calls can be a threat to network infrastructure. Attackers can target network components by sending malicious calls in an attempt to compromise those components. In this context, it would be imperative to guard against malicious calls and frequently conduct vulnerability and risk analysis of network devices. For this analysis, we discussed a network level risk management technique of prioritizing end host vulnerabilities by computing risk levels associated with the network devices. The prioritization scheme uses a Bayesian network probabilistic model for inferring the updated risk levels given evidence (e.g., evidence of an attack,

evidence of an applied patch). Using the computed risk levels, suitable security policies can be formulated for minimizing the risk and increasing the security of network devices.

10.2. Applicability of the Presented Solution

The solutions we discussed in this dissertation collectively infer the possibility of the call to be unwanted. While the VSD and ND are used for limiting unwanted calls to an end callee, the risk management mechanism can be used for minimizing the risk due to those unwanted calls. The spam detection framework is primarily used for detecting calls from spam and legitimate callers, thus, establishing the callers authenticity. The calls from authenticated callers are then processed by the ND for pro-actively determining the eagerness of the callees in receiving calls from that caller. The detection accuracy of VSD and ND can be optimized using automatic calibration of filter parameters, and knowledge integration of individuals communication behavior. Integrating these models into filtering processes helps reduce the number of incoming spam calls.

The VSD and ND can be deployed at any level of the network e.g. at the end IP phone level or in the central call processing devices such as proxy servers and IP PBXs. At any level, the VSD and ND learn and update their knowledge for limiting unwanted calls to callees inside their influence-domain. However, it is often observed that central VoIP devices process a large volume of call traffic when they are deployed at higher network level e.g., a service-provider. Here, it is a challenge not to burden the central devices with resource intensive operations for determining the legitimacy of the incoming calls during time-restrictive transactions e.g., call-setup. In this case, one way of easy integration of our proposed solutions is to perform filter operations either during the call or after the call is terminated. As majority of our filter operations are driven by callees feedback, the computations can be performed and the results be logged for examining the next incoming calls from that caller.

All the solutions discussed in the dissertation present models for identifying

unwanted calls on a VoIP network. However, we found some limitations in the applicability of the discussed solutions.

10.3. Limitations

We have observed some limitations in different models we have discussed in this dissertation. In this section, we present these limitations and make some suggestions that can improve the accuracy of filtering processes.

1. *VoIP Spam Detection* Here are the limitations to the VoIP spam detection framework presented in chapter 4.

- *Identity*: The VSD presented in Chapter 4 computes the probability of the incoming call to be spam based on previous history between the caller and the callee. For computing this spam level, VSD infers the previous history of the calling source (calling user, calling host, and calling domain). The callers with more previous spam history have a higher chance of getting filtered at the VSD. However, if the spammers frequently change their identity, it would be difficult for the VSD to identify the spammers, and therefore, some of the spam calls get forwarded to the callees until the VSD has sufficient knowledge to filter calls from them. For reducing the impact due to frequently identity changing spammers, the VSD needs to be integrated with identity establishment algorithms. For example, the VSD can be integrated with the identity establishment mechanism discussed in [80] for maintaining the spam history.
- *Bayesian Networks*: During reputation analysis, VSD constructs a domain proxy graph and derives the reputation of the call-source using Bayesian Inference techniques. In this dissertation, we have provided an example proxy graph with 4 nodes and derived Bayesian equations for inferring the reputation values, However, if the VSD is deployed at a level where the

domain proxy graph constitutes large number of nodes (more number of domain proxies), the derivation of reputation inference equations can be a complex task. In this context, our framework can use the Bayesian inference computation methodology defined in [8].

2. *Pro-active Nuisance Computation*

   - *Presence*: In this chapter, we discuss integrating presence information for pro-actively inferring the callees eagerness in receiving the incoming voice call. To present the applicability of presence information, we discussed an example of Personal Presence information using cell-phone calling patterns. While the personal presence information provides considerable context information, ND can largely benefit by incorporating additional information from more presence factors such as location and time. More fine classification or different types of presence information can lead to an increase in filter performance.

10.4. Problems Faced

During our research, we encountered few problems that I would like to mention. Majority of the problems were during the data collection and analysis stages.

Real-time Data: VoIP is an emerging technology and is not yet vastly deployed. There are only a few service providers that have started to provide VoIP services to its customers. As VoIP is primarily used as a technology for voice communication, the underlying data always constituted private and confidential information. It was an impossible task for us to convince the service providers and get real-time data for performance evaluation.

For some experiments related to nuisance computation, we approached different individuals at our university for real-time cellular calling patterns. Most individuals were reluctant to provide us with call-detail records that give us temporal and spatial information. Finally, we could succeed to convince few of them to give us real-time

cellular calling patterns only after elaborate discussions and showing proof of certification from human rights organizations.

Result Validation: For few experiments (e.g., in chapter 5) discussed in this dissertations, there are no available benchmarks for performance comparison. This is mainly due to fact that the work is an entirely new area of research. In this case, the only available option for us was to meet with each individual and get his feedback. As most of our research involves filtering unwanted calls, the notion of an "unwanted" call depended on individual perception too. In some circumstances, it was difficult to explain the individual the motivation behind the research and get "appropriate" feedback e.g., difference between personal relationships and communication relationships.

For few other experiments (e.g., chapter 8) related to hacking behavior, risk, and penetration, it was always hard to get a sincere feedback from the individuals. Here, the individuals were not forthright and always had ethical answers for hacking related questions. In this case, we had to tweak the questions and get appropriate feedback.

APPENDIX A

VSD TERMINOLOGY

- A caller/calling party is a person/entity generating a call.
- A callee/called party is a person/entity receiving a call.
- A call participant can be user, host, domain, proxy in the path, etc.
- Spaminess: Amount of spam behavior or the associated spam history. This is given by the number of past spam calls.
- Legitimateness: Amount of valid behavior or the associated non-spam history. This is given by number of past legitimate calls.
- n: Number of call participants in an incoming call. $n \in N$ (set of natural numbers).
- i: Refers to the i^{th} call participant.
- N_S: Total number of spam calls processed by VSD. $N_S \in N$.
- N_V: Total number of legitimate calls processed by VSD. $N_V \in N$.
- $N_{i,s}$: Spaminess of a call participant i. $N_{i,s} \in N$.
- $N_{i,v}$: Legitimateness of a call participant i. $N_{i,v} \in N$.
- C_i: A call set of participant i. The call set C_i for participant i is given by $\{N_{i,s}, Ni, v\}$.
- C: Set of call sets of all participants. $C = \{C_1, C_2...C_n\}$.
- $N_{i,B}$: Total calls from a participant i. $N_{i,B}$ is the summation of spaminess and legitimateness of the participant, i.e., $N_{i,B} = N_{i,s} + N_{i,v}$. $N_{i,B} \in N$.
- m: Number of callee community members. $m \in N$.
- u_k: k^{th} member in the callee's community such that $1 \leq k \leq m$ and $k \in N$.
- $N^d_{i,s}$: Spaminess (total spam calls) of a call participant i observed by callee's domain. The superscript d represents the domain-wide scope of i. $N^d_{i,s} \in N$.
- $N^d_{i,v}$:Legitimateness (total valid calls) of a call participant i observed by callee's domain. $N^d_{i,v} \in N$.
- T_i: Trust level of a call participant i. $T_i \in [0\ 1]$.
- $T_{i,t}$: Trust level of a call participant i at time t . $T_{i,t} \in [0\ 1]$.
- T: Trust of an incoming call. $T \in [0\ 1]$.

- D: The distrust of an incoming call . $D \in [0\ 1]$.
- D_i:The distrust level of a call participant i. $D_i \in [0\ 1]$.
- D^d:The distrust of an incoming call observed by callee's community. $D^d \in [0\ 1]$.
- N_{CF}:Number of spam or valid calls that can move current distrust level D_C to a final distrust level D_F . $N_{CF} \in$ N.
- R: Reputation of the calling party. $R \in [0\ 1]$.
- p_c:Spam probability of an incoming call based on trust and reputation. $p_c \in [0\ 1]$.
- Y: The callee's tolerance (threshold) towards the spam behavior of the incoming call. $Y \in [0\ 1]$.
- F: Forward or Filter decision. This decision F can be simple Boolean function resulting in True (filter the call) or False (forward the call). $F \in \{$True, False$\}$.

APPENDIX B

NAIVE BAYESIAN PROBABILISTIC MODEL FOR COMPUTING SPAM PROBABILITY OF CALL

Given an instance M for the incoming call and values of $n_1, n_2, n_3...n_n$ to the feature variables, the probability of the instance M to have a classification CL can be computed using Bayes theorem.

$P(CL = cl_k|M = n) = \frac{P(CL=cl_k)P(M=n|CL=cl_k)}{P(M=n)}$

Using a Nave Bayesian classifier (Good[29], SAHAMI et al [74]) that assumes that every feature is conditionally independent of all other features, we have

$$P(M = n|CL = cl_k) = \prod_{i=1}^{n} P(M_i = n_i|CL = cl_k)$$

Therefore,

$$P(CL = cl_k|M = n) = \frac{P(CL=cl_k)\prod_{i=1}^{n} P(M_i = n_i|CL = cl_k)}{P(M=n)}$$

From Bayes theorem, we also have

P(M=n) $= \sum_k P(M = n|CL = cl_k)P(CL = cl_k)$ for k different classifications of the instance M.

Therefore, $P(CL = cl_k|M = n) = \frac{P(CL=cl_k)\prod_{i=1}^{n} P(M_i = n_i|CL = cl_k)}{\sum_k P(M = n|CL = cl_k)P(CL = cl_k)}$

We use the above Nave Bayesian Probabilistic model for inferring spam behavior of incoming calls. The model is used for classifying the incoming call instance into two different classifications of call being spam or the call being legitimate.

For this, we define the features of call instance to be the calling user, calling host and call-generating domain. Each of the above call features is independent to others.

From the above equation, the probability of the call being classified as spam is given by

$$P(CL = spam|M = n) = \frac{P(CL=spam)\prod_{i=1}^{n} P(M_i = n_i|CL = spam)}{\sum_{c_k \in \{spam, legitimate\}} P(M = n|CL = cl_k)P(CL = cl_k)}$$

(40)

$$P(CL = spam|M = n) = \frac{P(CL=spam)\prod_{i=1}^{n} P(M_i = n_i|CL = spam)}{P(CL=spam)\prod_{i=1}^{n} P(M_i = n_i|CL = spam) + P(CL = legitimate)\prod_{i=1}^{n} P(M_i = n_i|CL = legitimate)}$$

Each term in the above equation can be computed by logging the feedback from all the called parties for all the forwarded calls to them. In Equation (40), P(CL=spam) represents the probability of spam calls processed by the VSD and it can be estimated from the logged history.
i.e. P(CL=spam) = $\frac{TotalSpamCalls}{TotalSpamCalls+TotallegitimateCalls} = \frac{N_S}{N_S+N_V}$ where N_S and N_V represent the total number of spam and legitimate calls seen before respectively.

Similarly, P(CL=legitimate) = $\frac{TotalLegitimateCalls}{TotalSpamCalls+TotallegitimateCalls} = \frac{N_V}{N_S+N_V}$
However, for negligible N_S compared to N_V (i.e., when the number of observed previous spam calls is negligible compared to legitimate calls), we have $P(CL = legitimate) \rightarrow 1$. In this case, every incoming call would be initialized to a high legitimate probability when substituted in Equation (40). Therefore, in this context, few spam calls go unnoticed resulting in false negatives.

Similarly, for a negligible N_V compared to N_S, (i.e., when the number of observed previous legitimate calls is negligible compared to spam calls), we have $P(CL = spam) \rightarrow 1$. In this case, every incoming call would be initialized to a high spam probability when substituted in Equation (40). In this context, legitimate calls get filtered at the spam filter.

Therefore to reduce the influence of above two terms in call classification, we compute those terms based only on the histories of the participating entities of the

call such as the calling user, calling host and call-generating domain i.e.

$$\text{P(CL=spam)} = \frac{TotalSpamCallsfromcallparticipants}{TotalSpamCallsfromcallparticipants+TotallegitimateCallsfromcallparticipants}$$

$$\text{and P(CL=legitimate)} = \frac{TotalLegitimateCallsfromcallparticipants}{TotalSpamCallsfromcallparticipants+TotallegitimateCallsfromcallparticipants}$$

$$\text{Therefore, P(CL=spam)} = \frac{\sum_{i=1}^{n} N_{i,s}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})} and P(CL = legitimate) = \frac{\sum_{i=1}^{n} N_{i,v}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})}$$

Also, probability for a call participant n_i to be spam $= P(M_i = n_i|CL = spam) = \frac{N_{i,s}}{N_{i,s}+N_v}$ and probability for a call participant n_i to be legitimate $= P(M_i = n_i|CL = legitimate) = \frac{N_{i,v}}{N_{i,s}+N_v}$

Therefore, we have

$$P(CL = cl_k|M = n) = \frac{(\frac{\sum_{i=1}^{n} N_{i,S}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,S}}{N_{i,S}+N_{i,V}})}{(\frac{\sum_{i=1}^{n} N_{i,S}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,S}}{N_{i,S}+N_{i,V}}) + (\frac{\sum_{i=1}^{n} N_{i,V}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,V}}{N_{i,S}+N_{i,V}})}$$

APPENDIX C

BAYESIAN NETWORK PROBABILISTIC MODEL FOR DERIVING REPUTATION INFORMATION

A Bayesian network is a graphical model for showing probabilistic relationships among a set of variables in a Directed Acyclic Graph (DAG). A Directed Acylic Graph contains a set of nodes and directed links between them where each node is a variable and the links connecting two nodes in a DAG are the dependencies existing between those two variables. For an n node graph represented by n random variables $V_1, V_2, V_3..., V_n$, the probability distribution function would be equal to

$P(V_1, V_2, V_3..., V_n) = P(V_1)P(V_2|V_1).....P(V_n|V_1...V_{n-1})$

However, the values of a node are conditioned only on its parents

Therefore, $P(V_1, V_2, V_3..., V_n) = \prod_{i=1}^{n} P(V_i|parents(V_i))$

where every joint probability $P(V_i|V_j)$ can be expanded into sum of two joint probabilities to include the parent of a variable and can be shown as

$P(V_i|V_j) = P(V_i, V_k|V_j) + P(V_i, \sim V_k|V_j)$ for parent node V_k of V_i and $i,j,k \in 1..n$.

A conditionalized version of chain rule is given by

$P(V_i, V_k|V_j) = P(V_i|V_k, V_j)P(V_k|V_j)$ for $i,j,k \in 1..n$.

The above chain rule can be used for deducing the posterior probability of a random variable for observed evidence. If$P(V_i)$ for i =1..n represents the previous subjective belief of a variable V_i, then $P(V_i|V_j)$ represents the posterior probability of V_i for an observed evidence at V_j i.e., the observed evidence at node represented by variable V_j can be propagated throughout the graph by computing $P(V_i|V_j)$ for $i \neq j$ and $i,j \in 1..n$.

For a given directed graph represented in Figure C.1, assume that the nodes

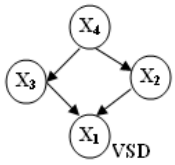

FIGURE C.1. Network Topology graph of domain proxies

represent the proxy nodes of domains that help in routing the call. Calls are generated

from different users in different domains to an end user in the domain X_1 for which VSD acts as a spam filtering device. Assume $P(X_i)$ represents the reputation (initial subjective belief) of domain X_i for i=2..4 with respect to the VSD. Each value $P(X_i)$ for $i = 2..4$ ranges between 0 and 1.

Assumptions: The random variables associated with these proxy nodes are independent. The reputation of a given domain X_i for $i = 1..4$ does not depend on the reputation of other domain X_j for $j = 1..4$ and $i \neq j$. For observed evidence that a call from domain X_4 to X_1 is spam (when the end user gives a feedback to the VSD that the received call is spam), the updated reputation of domain X_4 with respect to VSD can be inferred by $P(X_1|X_4)$. Using the conditionalized version of chain rule,

$$P(X_1|X_4) = P(X_1, X_2|X_4) + P(X_1, \sim X_2|X_4) = P(X_1|X_2)P(X_2|X_4) + P(X_1| \sim X_2)P(\sim X_2|X_4)$$

where

$$P(X_1|X_2) = P(X_1, X_3|X_2) + P(X_1, \sim X_3|X_2) = P(X_1|X_2, X_3)P(X_3) + P(X_1|X_2, \sim X_3)P(\sim X_3)$$

and $P(X_1| \sim X_2) = P(X_1, X_3| \sim X_2) + P(X_1, \sim X_3| \sim X_2) = P(X_1| \sim X_2, X_3)P(X_3)$

+

$P(X_1| \sim X_2, \sim X_3)P(\sim X_3)$

and $P(X_3) = P(X_3, X_4) + P(X_3, \sim X_4) = P(X_3|X_4)P(X_4) + P(X_3| \sim X_4)P(\sim X_4)$

APPENDIX D

ANALYTICAL MODEL FOR DERIVING RELATION BETWEEN AMOUNT OF SPAM AND SPAM DETECTION CAPABILITY

To determine the relationship between the spam detection capability for varying amounts of spam, assume that the filter processes x% and y% of spam in a given time t. Assume that in both the cases the filter processes a total of t calls. Assume also that $x < y$.

Now, for a call with n call participants, the distrust D is given by

$$D = \frac{(\frac{\sum_{i=1}^{n} N_{i,S}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,S}}{N_{i,S}+N_{i,V}})}{(\frac{\sum_{i=1}^{n} N_{i,S}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,S}}{N_{i,S}+N_{i,V}}) + (\frac{\sum_{i=1}^{n} N_{i,V}}{\sum_{i=1}^{n}(N_{i,S}+N_{i,V})})(\prod_{i=1}^{n}\frac{N_{i,V}}{N_{i,S}+N_{i,V}})}$$ from Equation (1) in Chapter 4.

For both the amounts of spam, the factors $\prod_{i=1}^{n}\frac{N_{i,s}}{N_{i,s}+N_{i,v}}$ and $\prod_{i=1}^{n}\frac{N_{i,v}}{N_{i,s}+N_{i,v}}$ are constant as these two factors depend upon the spaminess and legitimateness of individual call participants rather than the number of spam and legitimate calls processed by the spam filter.

Therefore for x% of spam, distrust D is given by $\frac{xa}{xa+(t-x)b}$ where $\frac{\sum_{i=1}^{n} N_{i,s}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})}$ = x (i.e, x% of calls are spam calls among the total calls), and a, b are assumed to $\prod_{i=1}^{n}\frac{N_{i,s}}{N_{i,s}+N_{i,v}}$ and $\prod_{i=1}^{n}\frac{N_{i,v}}{N_{i,s}+N_{i,v}}$ respectively. Similarly for y% of spam, the distrust is given by $\frac{ya}{xa+(t-y)b}$ since $\frac{\sum_{i=1}^{n} N_{i,s}}{\sum_{i=1}^{n}(N_{i,s}+N_{i,v})}$ = y (i.e., y% of calls are spam calls), and a,b are same as above.

Now, $\frac{xa}{xa+(t-x)b} < \frac{ya}{ya+(t-y)b}$

$\implies xya + xb(t-y) < xya + yb(t-x)$

$\implies x(t-y) = y(t-x)$

$\implies x < y$

Conversely, if $x < y$, then $\frac{xa}{xa+(t-x)b} < \frac{ya}{ya+(t-y)b}$ i.e., the distrust for a call for less previously observed spam(x%) by the VSD is less than distrust for the call for more previously observed spam (y%). It is easy to filter out spam calls with more spam behavior or distrust than the one's with less spam behavior. As $x < y$, therefore, it can be understood that more the amount of spam, easier to detect it.

APPENDIX E

SURVEY OF BEHAVIOR PROFILES

E.5. Introduction

To better understand different types of attack behavior and to estimate the behavioral resources, we conducted an online survey. The details of the survey can be found at [19]. The prime objectives of conducting the survey are to:

(1) *Assign attributes for nodes in the attack graph*: The nodes of the attack graph given in Figure 2 of Chapter 8 represent different network actions for exploiting a given vulnerability. Different profiles have different amounts of resources for exploiting the network action. Establishing attribute values for resources of different profiles enables us to infer profile capabilities.

(2) *Analyze the relationship between behavior and network actions*: Different behavior profiles have capabilities of exploiting different network actions based on their available resources. Studying the relationship between behavior profiles and network actions helps in reducing profile based attacks.

(3) *Understand the relationship between risk, network penetration and user profiles*: Network penetration represents the depth an attacker can achieve with his available resources in the attack graph. Different behavior profiles have varied network penetration, and thus cause different types of risk to the underlying network. Understanding the relation between the amount of risk and network penetration for different behavior profiles aids in risk management (e.g., penetration testing and patch management).

E.6. Participants

The survey of behavior profiles was organized among a set of participants with computer knowledge ranging from novices to experts. A total of 59 participants took

the survey.

E.7. Survey

All the participants took a survey with questions divided into two parts. The responses to the questions were used to infer different profiles and their capabilities (or available resources) to exploit network actions. The two parts are described in detail as follows.

E.7.1. Network Actions (Part I)

The first part of the survey consists of 14 questions for the participants to answer. The set of survey questions are as follows

(1) What is your knowledge of scanning open ports?
(2) How often you change system files?
(3) How often you change read/write permission to files?
(4) How often you use ping to find out remote host is alive or not?
(5) How often you find IP address through DNS queries?
(6) How often you finger remote machine and gather useful information (e.g., list of tasks)?
(7) How often you kill a task?
(8) How often you use rlogin, ftp?
(9) How often you install tool kits down loaded from the web?
(10) How often you login as root or admin user?
(11) Do you use anonymous FTP any time?
(12) How often do you use anonymous usernames and passwords?
(13) Do you know there are some protocols/services that have default passwords?
(14) Do you know how to guess on what operating system the remote machine is running?

The 14 questions represent network actions that are concerned with day-to-day operations for computer network penetration. The responses to these network actions can be used for inferring the resources that are required to carry out the network action. We identified three resources: skill, attitude and time-spent to be associated with each of the 14 network actions. Skill defines the attacker's ability to carry out the network action. Attitude represents the attackers attack intent (or inclination) in carrying out the network action. Finally time represents the amount of time the attacker has for carrying out the attack.

For assigning attribute values for skill, attitude, and time for the survey participant, we analyzed the responses to the survey. Each option for a given question is assigned a score for skill, attitude and time. The sum of the scores of the selected options by the participant gives the amount of skill, attitude, and time available with the participant.

E.7.2. Behavior Profiles (Part II)

The second part of the survey consists of 32 questions. The responses to these questions can be used to infer the behavior of the survey participant. The questions the participants had to answer in Part II are as follows

(1) I would rather chat online than in person.
(2) Computers help me to feel in control
(3) I enjoy experimenting with computers
(4) I am more interested in how the Internet works than what can be found on it.
(5) I enjoy exploring with others on computers
(6) I enjoy competing with others on computers
(7) If people don't want others to get access to their computer or computer systems, they should have better security.
(8) If your computer gets hit, you deserve it.

(9) Information on computers should be free to anyone.

(10) It is morally okay to view information, even if you don't have legitimate access, as long as you don't harm it.

(11) I would never do anything that is against the law on a computer.

(12) Computer-related laws are intended primarily to protect the rich and powerful.

(13) Technology innovation depends on free and unlimited access to information.

(14) I enjoy flaming people in chat or on Email lists.

(15) It is okay to publicize computer security flaws to the public

(16) The best way to publicize a computer security flaw is to exploit it.

(17) It is okay to download computer security exploits and experiment with them on your own machine.

(18) It is okay to download computer security exploits and experiment with them on other people's machines.

(19) It is okay to download computer security exploits and experiment with them on other people's machines, if you could be sure not to get caught.

(20) If you had obtained illegal access, it would be morally okay to publicize income levels within a corporation.

(21) If you had obtained illegal access, it would be morally okay to publicize income levels within a corporation, if you could be sure not to get caught.

(22) If you had obtained illegal access, it would be morally okay to publicize confidential communications if they described unethical or illegal behavior.

(23) If you had obtained illegal access, it would be morally okay to publicize confidential communications if they described unethical or illegal behavior, if you could be sure not to get caught.

(24) If you had obtained illegal access, it would be a good idea to try to personally profit from confidential information.

(25) If you had obtained illegal access, it would be a good idea to try to personally profit from confidential information, if you could be sure not to get caught.

(26) It is okay to harass someone who behaves poorly online.

(27) It is okay to harass someone who behaves poorly online, if you could be sure not to get caught.

(28) It is okay to threaten someone who behaves poorly online.

(29) It is okay to threaten someone who behaves poorly online, if you could be sure not to get caught.

(30) It is okay to use computer service without paying for it.

(31) It is okay to use computer service without paying for it, if you could be sure not to get caught.

(32) It is okay to use computer service without paying for it because it is not being used otherwise.

The extracted behavioral information from the responses of survey participants can be used for constructing profiles of people. In this survey, we assumed that there are three kinds of people who attempt to penetrate or compromise resources like networks and computers. These are people with hacker-behavior, opportunist-behavior, and explorer-behavior. People differ in the mindset for attack behavior. For example, a person with opportunist behavior may intend to be isolated and hidden, whereas a person with explorer behavior is someone who believes in open door principles.

For classifying the participant into one of the three profiles, we assigned a score to each option of a question in Part II of the survey. The sum of the selected option scores by the participant to all the 32 questions is used to classify the participant into one of the three profiles.

E.8. Survey Results

Using the responses given in Part II, we divided the participants into three groups: hacker, opportunist, and explorer-behavior. We analyzed the values of skill, attitude, and time for the people in the three groups based on Part I of the survey. For inferring these values, the median (or most probable responses) of all the people classified into one group are taken into consideration. A normalized set of values in the range of 1-10 for the people in three groups are given in Table I.

TABLE E.1. Attribute values of network actions for the three behavior profiles.

	Hacker Behavior			Opportunist Behavior			Explorer Behavior		
Question	Skill	Attitude	Time	Skill	Attitude	Time	Skill	Attitude	Time
1	9.198	8.431	9.043	10	8.796	10	8.221	8.686	7.913
1	9.198	8.431	9.043	10.000	8.796	10.000	8.221	8.686	7.913
2	8.346	7.628	7.913	10.000	8.796	10.000	5.414	6.058	4.957
3	9.198	8.431	9.043	10.000	8.796	10.000	6.817	7.372	6.435
4	8.346	7.628	7.913	10.000	8.796	10.000	4.010	4.745	3.478
5	7.494	6.825	6.783	6.917	7.263	6.087	5.414	6.058	4.957
6	7.494	6.825	6.783	10.000	8.796	10.000	4.010	4.745	3.478
7	7.494	6.825	6.783	7.945	7.774	7.391	5.414	6.058	4.957
8	7.494	6.825	6.783	10.000	8.796	10.000	4.010	4.745	3.478
9	7.494	6.825	6.783	6.917	7.263	6.087	5.414	6.058	4.957
10	9.198	8.431	9.043	7.945	7.774	7.391	9.624	10.000	9.391
11	7.494	6.825	6.783	6.917	7.263	6.087	4.010	4.745	3.478
12	7.494	6.825	6.783	6.917	7.263	6.087	4.010	4.745	3.478
13	6.642	6.022	5.652	6.917	7.263	6.087	4.010	4.745	3.478
14	6.642	6.022	5.652	5.890	6.752	4.783	4.010	4.745	3.478

From the computed score, we observed the following

- Participants classified into the opportunist-behavior profile have higher attitude, skill and time compared to participants belonging to other profiles.
- Participants with hacking behavior had intermediate values of skill, attitude, and time among all the participants.
- Participants classified into the explorer-behavior profile have the least amount of attitude among the participants of all the three profiles.

These observations can be clearly seen in Figure E.1 - Figure E.3. The x-axis represents the network actions that we have outlined in Part I of the survey.

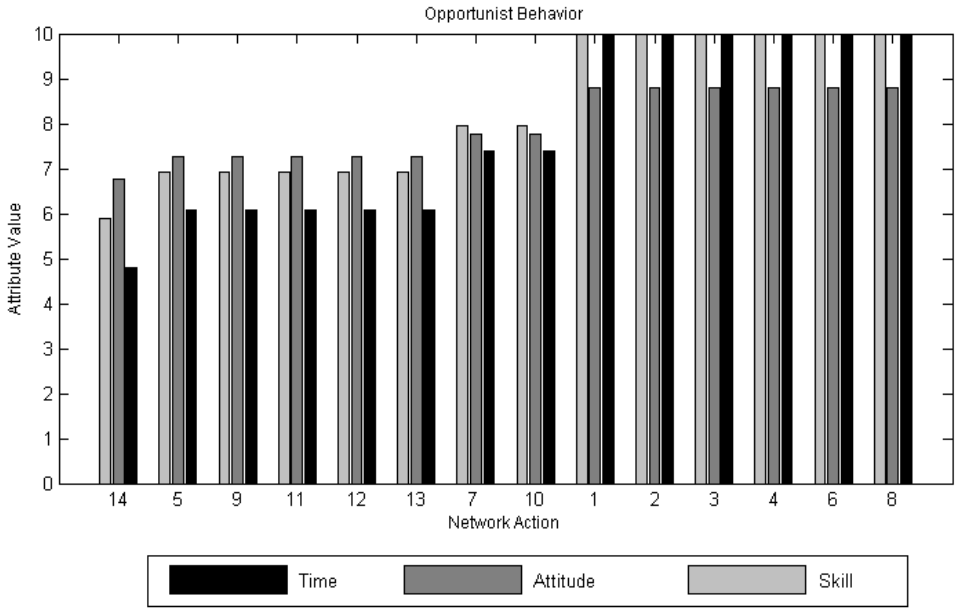

FIGURE E.1. Behavior attributes for Opportunist-behavior individuals

The attribute values of skill and attitude are higher for opportunists followed by hackers then explorers. However, it can be observed that explorers have high values of attributes for question #10, which inquires about the frequency with which the participants logs into a system as root or admin user. More explorers use root user to login compared to opportunists and hackers as they tend to believe in

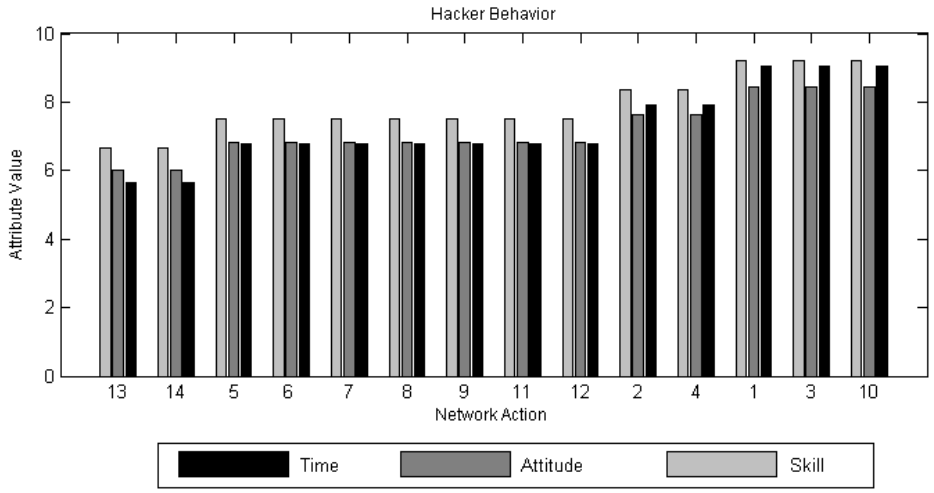

FIGURE E.2. Behavior attributes for Hacker-behavior individuals

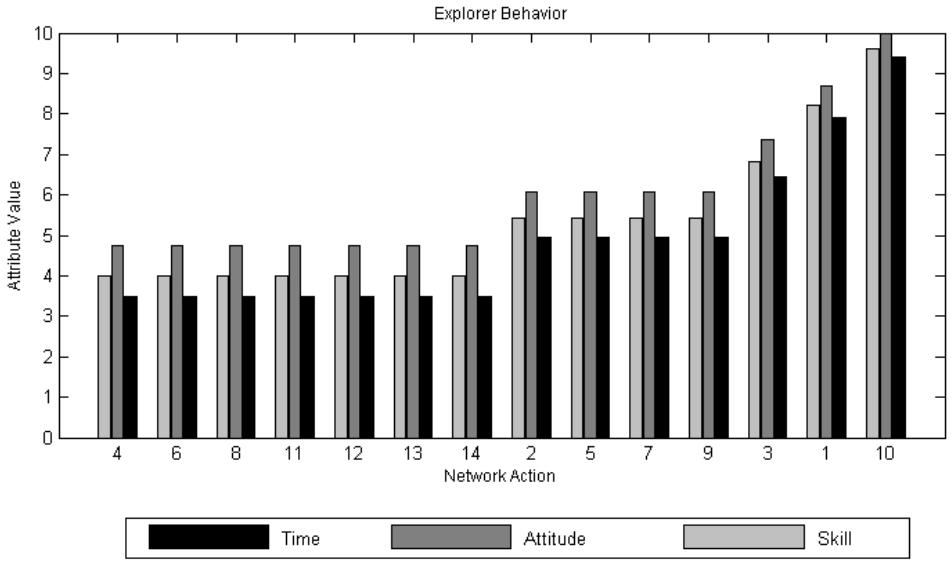

FIGURE E.3. Behavior attributes for explorer-behavior individuals

open door policies.

In Table II, we sorted the sum of scores for attribute values in a descending order of attitude, time (if any other participant have same value of attitude), and then by skill (if there are any participants with same skill and attitude). Based on

the above sort order, we observed that all the higher order participants are the people with opportunist-behavior followed by people with hacking and explorer behavior. This order justifies the classification that high attitude are the ones with opportunist behavior and the ones with explorer behavior have lower values of attitude.

TABLE E.2. Attribute values and behavior classification of survey participants.

Participant ID	Attitude	Skill	Time	Behavior Classification
13	10.000	10.000	9.559	Opportunist
27	9.855	9.405	10.000	Opportunist
45	9.420	8.333	8.824	Opportunist
50	8.696	8.452	9.412	Opportunist
32	8.406	8.929	7.794	Opportunist
26	8.261	8.690	8.235	Opportunist
4	8.261	7.381	6.618	Opportunist
43	7.971	8.452	7.500	Opportunist
10	7.971	7.857	6.765	Opportunist
57	7.826	8.571	7.647	Opportunist
7	7.536	7.738	8.088	Opportunist
25	7.536	8.333	7.353	Opportunist
30	7.391	8.214	8.088	Opportunist
29	7.246	8.214	8.235	Opportunist
33	7.246	7.738	6.912	Opportunist
6	7.246	7.619	6.618	Opportunist
49	7.101	7.857	7.794	Opportunist
20	6.957	7.619	8.235	Opportunist
1	6.957	7.500	6.029	Opportunist
52	6.667	7.500	8.382	Opportunist

TABLE E.3. Attribute values and behavior classification of survey participants (Contd).

Participant ID	Attitude	Skill	Time	Behavior Classification
18	6.667	7.262	6.765	Opportunist
21	6.522	7.738	8.529	Opportunist
19	6.522	6.190	6.324	Opportunist
37	6.522	6.905	5.882	Opportunist
14	6.377	6.905	7.353	Opportunist
11	6.377	7.500	7.206	Opportunist
17	6.377	7.500	6.324	Opportunist
47	6.232	7.500	8.235	Hacker
8	6.232	6.429	7.647	Hacker
16	6.232	6.071	7.059	Hacker
24	6.232	7.262	6.618	Hacker
3	6.232	7.738	6.324	Hacker
34	6.087	7.381	7.941	Hacker
40	6.087	7.024	7.353	Hacker
53	6.087	7.143	5.882	Hacker
23	5.942	7.024	9.706	Hacker
58	5.942	6.667	7.647	Hacker
39	5.942	6.548	6.029	Hacker
48	5.652	6.905	6.765	Hacker
46	5.507	6.905	7.500	Hacker

In, conclusion, we hope our research will help in better understanding the relationship between the attributes (such as skills, time, and attitude) required for network penetration and the profiles (such as hackers, opportunists, and Explorer behavior) of people that do the penetration.

TABLE E.4. Attribute values and behavior classification of survey participants(Contd).

Participant ID	Attitude	Skill	Time	Behavior Classification
59	5.507	6.190	6.029	Hacker
38	5.507	6.071	5.735	Explorer
51	5.362	6.548	7.941	Hacker
56	5.362	6.310	6.765	Hacker
42	5.362	6.310	6.471	Hacker
2	5.072	7.024	6.765	Hacker
12	5.072	6.071	6.324	Explorer
5	4.928	5.595	6.471	Explorer
28	4.928	6.190	5.441	Hacker
9	4.783	5.833	7.500	Explorer
54	4.783	4.762	5.294	Explorer
31	4.638	5.714	6.176	Explorer
55	4.638	5.238	5.588	Explorer
15	4.348	5.833	5.588	Explorer
36	4.348	5.357	5.000	Explorer
41	4.058	5.357	5.294	Explorer
35	4.058	5.714	5.147	Explorer
44	3.768	4.762	4.559	Explorer
22	3.478	3.810	4.706	Explorer

APPENDIX F

CONDITIONAL PROBABILITY TABLES FOR NODES OF ATTACK GRAPH

Probabilities of nodes A, B, C

$\mathrm{P}(A) = 0.4$

$\mathrm{P}(B) = 0.63$

$\mathrm{P}(C) = 0.34$

TABLE F.1. Probabilities of node D

A	$P(D\|A)$
T	0.52
F	0.36

TABLE F.2. Probabilities of node E

A	$P(E\|A)$
T	0.72
F	0.37

TABLE F.3. Probabilities of node F

A	$P(F\|A)$
T	0.27
F	0.35

TABLE F.4. Probabilities of node G

B	$P(G\|B)$
T	0.36
F	0.52

TABLE F.5. Probabilities of node H

B	$P(H\|B)$
T	0.35
F	0.36

TABLE F.6. Probabilities of node I

C	$P(I\|C)$
T	0.41
F	0.52

TABLE F.7. Probabilities of node L

G	$P(L\|G)$
T	0.22
F	0.46

TABLE F.8. Probabilities of node K

D	E	F	$P(K\|D,E,F)$
T	T	T	0.48
T	T	F	0.47
T	F	T	0.6
T	F	F	0.17
F	T	T	0.32
F	T	F	0.45
F	F	T	0.7
F	F	F	0.49

TABLE F.9. Probabilities of node M

H	I	$P(M\|H,I)$
T	T	0.35
T	F	0.83
F	T	0.72
F	F	0.26

TABLE F.10. Probabilities of node N

K	M	$P(N\|K,M)$
T	T	0.48
T	F	0.37
F	T	0.59
F	F	0.12

TABLE F.11. Probabilities of node O

K	L	M	$P(O\|K,L,M)$
T	T	T	0.22
T	T	F	0.28
T	F	T	0.33
T	F	F	0.7
F	T	T	0.54
F	T	F	0.28
F	F	T	0.32
F	F	F	0.41

TABLE F.12. Probabilities of node R

K	L	H	$P(R\|K, L, H)$
T	T	T	0.34
T	T	F	0.16
T	F	T	0.7
T	F	F	0.83
F	T	T	0.29
F	T	F	0.49
F	F	T	0.24
F	F	F	0.74

TABLE F.13. Probabilities of node S

L	$P(S\|L)$
T	0.2
F	0.7

TABLE F.14. Probabilities of node T

M	$P(T\|M)$
T	0.5
F	0.6

BIBLIOGRAPHY

[1] C. Biever, "Move over spam, make way for "spit"," *http://www.newscientist.com/article.ns?id=dn6445*, 2004 [Mar 2005]

[2] G. Blackwell, "FoIP (Fraud over IP)," *VoIP Planet*, http://www.voipplanet.com/trends/article.php/ 3616771, June 2006 [Nov 2006]

[3] P.O. Boykin, V. RoyChowdary, "Personal Email Networks: An Effective Anti-spam Tool," *Preprint, http://www.arxiv.org/abs/cond-mat/0402143*, 2004 [Aug 2005]

[4] P.O. Boykin, V. RoyChowdary, "The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms," *Pattern Recognition*, 30(7), 1145-1159, 1997 [Mar 2006]

[5] A.P. Bradley, "The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms," *Pattern Recognition*, 30(7), 1145-1159, 1997 [Mar 2006]

[6] V. Cahill, B. Shand, E. Gray, N. Dimmock, A. Twigg, J. Bacon, C. English, W. Wagealla, S. Terzis, P. Noxon, C. Bryce, G.M. Serugendo, J.M. Seigneurl, M. Carbone, K. Krukow, C. Jenson, Y. Chen, M. Nielsen, "Using Trust for Secure Collaboration in Uncertain Environments," *IEEE Pervasive Computing*, Volume 2, Issue 3, Page(s):52 - 61, 2003 [Sep 2006]

[7] J. Canny, "Is an HCI Revolution Just Around the Corner?" *ACM Queue*, Vol. 4, no.6, 2006 [Jan 2007]

[8] E. Castillo, J. M. Gutierrez, A.S. Hadi, "Sensitivity Analysis in Discrete Bayesian Networks" *IEEE Transactions on SYSTEMS, MAN, and CYBERNATICS*, vol 27, No. 4, July 1997 [Nov 2004].

[9] A. Chandler, "Changing Definition of Hackers in Popular Discourse," *International Journal of Sociology and Law*, 24(2), 229-252, 1996 [Oct 2003].

[10] W.W. Cohen, "Learning Rules that Classify e-mail," *In Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access*, 1996 [Apr 2004]

[11] E. Damiani, S.D.C. Vimercati, S. Paraboschi, P. Samarati, "P2P-Based Collaborative Spam Detection and Filtering," *In Proceedings of 4th IEEE Conference on Peer-to-Peer Computing (P2P '04)*, Zurich, Switzerland, 2004 [Oct 2006]

[12] R. Dantu, "Intrusion Detection using Attacker Profiles,", Work in Progress *19th Annual Computer Security Applications Conference(ACSAC)*, Las Vegas, 2003

[13] R. Dantu, P. Kolan, "Preventing Voice Spamming," *In IEEE GlobeComm Workshop on VoIP Security*, 2004 [Dec 2004]

[14] R. Dantu, K. Loper, P. Kolan, "Risk Management Using Behavior Based Attack Graphs," *IEEE International Conference on Information Technology (ITCC)*, April 2004 [Apr 2004]

[15] R. Dantu, P. Kolan, "Survey of Behavior Profiles," *University of North Texas Internal Survey*, http://secnet.csci.unt.edu/risk/, 2004 [Aug 2004]

[16] R. Dantu, J. Cangussu, A. Yelimeli, "Dynamic Control of Worm Propagation," *IEEE International Conference on Information Technology (ITCC)*, 2004 [Apr 2004]

[17] R. Dantu, P. Kolan, "Detecting Spam in VoIP Networks," *In Proceedings of USENIX, SRUTI(Steps for Reducing Unwanted Traffic on the Internet) workshop*, 2005 [Mar 2005]

[18] R. Dantu, P. Kolan, "Risk Management using Behavior Based Bayesian Networks," *Lecture Notes in Computer Science (LNCS)*, Springer Verilag, , 2005 [May 2005]

[19] R. Dantu, P. Kolan, "Survey of Calling Patterns," *University of North Texas Internal Survey*,http://secnet.csci.unt.edu/nuisance/, 2006 [Aug 2006]

[20] J. Desmond, "Checkmate IDS Tries to Anticipate Hackers Actions," *www.esecurityplanet.com/prodser*, 12th June, 2003 [Oct 2003]

[21] N. Eagle, *"Machine Perception and Learning of Complex Social Systems,"* PhD Dissertation, Massachusetts Institute of Technology, 2005 [Sep 2006]

[22] D. Evett, "Spam Statistics 2006," *http://spam-filter-review.toptenreviews.com/spam-statistics.html*, 2006 [Dec 2006]

[23] T. Fawcett, *"ROC graphs: Notes and Practical Considerations for Researchers,"* Technical Report, HP Lab, Palo Alto, CA, 2003 [Jul 2006]

[24] N. Foukia, L. Zhou, C. Neuman, "Multilateral Decisions for Collaborative Defense Against Unsolicited Bulk E-mail,"*International Conference on Trust Management*, 2006 [Dec 2006]

[25] "Frost and Sullivan," *www.frost.com* [Mar 2007]

[26] J. Goecks, E.D. Mynatt, "Enabling Privacy Management in Ubiquitous Computing Environments through Trust and Reputation Systems," *Proceedings of CSCW(Workshop on Privacy in Digital Environments: Empowering Users)*, 2002 [Sep 2005]

[27] J. Golbeck, J. Hendler, "Reputation Network Analysis for Email Filtering," *IEEE conference on Email and Anti Spam*, 2004 [Feb 2005]

[28] A. Gonsalves, "Phishers Snare Victims with VoIP," *TechWeb Technology*, http://www.techweb.com/wire/security/186701001, April 2006 [Mar 2007]

[29] I.J. Good, *"The Estimation of Probabilities: An Essay on Modern Bayesian Methods,"* M.I.T Press, 1965 [Jan 2003]

[30] M.S., Granovetter, "The strength of Weak Ties," *In American Journal of Sociology*, 78:1360-80, 1973 [Nov 2006]

[31] J.A. Hanley, B.J. McNeil, "The Meaning and the Use of the Area under a Receiver Operating Characteristic (ROC) Curve," *Radiology*, 143, 29-36, 1982 [Jun 2006]

[32] M. Hepburn, D. Wright, *"Execution Contexts For Determining Trust In A Higher-Order* Π *Calculus,"* Technical Report, School of Computing, University of Tasmania, 2003 [Oct 2006]

[33] "Hugin Demo," *http://www.HUGIN.com/* [Jul 2003]

[34] "ISS Finds Flaws in Cisco VoIP," *VoIP Magazine*,http://www.voip-magazine.com/index.php?option=com_content&task=view&id=272, July 2005 [Sep 2006]

[35] G. Jackson, "Checkmate Intrusion Protection System: Evolution or Revolution," *Psynapse Technologies*, 2003 [Dec 2003]

[36] S. Jasanoff, "A sociology of Hackers," *The Sociological Review*, 46(4), 757-780, 1998 [Dec 2003]

[37] A. Josang, R. Ismail, C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, 2006 [Oct 2006]

[38] A. Khalil, K.H. Conelly, "Improving Cell Phone Awareness by Using Calendar Information," *In Proceedings of International Conference on Human-Computer Interaction (INTERACT)*, 2005 [Dec 2006]

[39] A. Khalil, K.H. Conelly, "Context-aware Telephony: Privacy Preferences and Sharing Patterns," *In Proceedings of Computer Supported Collaborative Work (CSCW)*, 2006 [Dec 2006]

[40] L. Kleen, *"Malicious Hackers: A Framework for Analysis and Case Study,"* Ph.D. Thesis, Air Force Institute of Technology, Ohio, 2001 [Oct 2003]

[41] "Know Your Enemy: Motives," *Honeynet Project*, 27th June, 2000 [Oct 2003]

[42] K. Krukow, M. Nielsen, "From Simulations to Theorems: A Position Paper on Research in the Field of Computational Trust," *In Proceedings of Formal Aspects in Security and Trust*, 2006 [Oct 2006]

[43] S. Lacy, "Is Your VoIP Phone Vulnerable?," *http://www.businessweek.com/technology/content/jun2006/tc20060613_799282.htm*, 2006 [Feb 2007]

[44] K. Lancaster, "Resilient Packet Ring: Enabling VoIP Delivery," *Internet Telephony*, 2003 [Feb 2007]

[45] H. Lei, G.C. Shoja, "A Distributed Trust Model for E-Commerce Applications," *IEEE International Conference on e-Technology, e-Commerce and e-Service*, 2005 [May 2006]

[46] J. Leyden, "Cisco VoIP Kit Open to Snooping Attacks," *http://www.theregister.co.uk/2004/02/20/cisco_voip_kit_open/*, Feb 2004 [Sep 2006]

[47] K. Loper, *"The Criminology of Computer Hackers: A Qualitative and Quantitative Analysis,"* Ph.D. Thesis, Michigan State University, 2000 [Oct 2003]

[48] R. Macintosh, D. Vinukurov, "Detection and Mitigation of Spam in IP Telephony Networks using Signaling Protocol Analysis," *IEEE symposium on Advances in Wired and Wireless Communication*, Page(s):49 - 52, 2005 [Oct 2006]

[49] P.V. Marsden, K.E. Campbell, "Measuring Tie Strength," *Social Forces*, Vol. 63, No. 2, pp. 482-501, 1984 [Aug 2006]

[50] S. Marsh, *"Formalizing Trust as a Computational Concept,"* Ph.D. Thesis, University of Stirling, 2006 [Oct 2006]

[51] S. McQuade, K. Loper, "A Qualitative Examination of the Hacker Subculture Through Content Analysis of Hacker Communication," *American Society of Criminology*, November 2002 [Oct 2003]

[52] E. Mier, R. Birdsall, R. Thayer, "Breaking Through IP Telephony," *Network World Lab Alliance, Network World*, http://www.nwfusion.com/reviews/2004/0524voipsecurity.html, May 2004 [Feb 2007]

[53] T.M. Mitchell, *"Machine Learning,"* McGraw-Hill, 1997 [Sep 2003]

[54] A.P. Moore, R.J.Ellison, R.C. Linger, *"Attack Modeling for Information Security and Survivalility,"* Technical Note, CMU/SE1-2001-TN-001, March 2001 [Oct 2003]

[55] L. Mui, M. Mohtashemi, A. Halberstadt, "A Computational Model of Trust and Reputation," *In Proceedings of the 35th Hawaii International Conference on System Science*, 280-287, 2002 [Nov 2004]

[56] S. Niccoloni, S. Tartarelli, M. Stiemrling, S. Srivastava, "SIP Extensions for SPIT identification," *IETF SIP draft*, draft-niccolini-sipping-feedback-spit-02, 2006 [Mar 2007]

[57] K. Ono, H. Schulzrinne, "Trust Path Discovery," *IETF Internet Draft*, 2005 [Nov 2005]

[58] P. Orbaek, J. Palsberg, "Trust in the λ calculus," *Functional Programming*, vol. 7, no. 6, pp. 557–591, 1997 [Oct 2006]

[59] S. Palla, R. Dantu, "Detecting Phishing in Emails," *Spam Conference*, Massachusetts Institute of Technology, 2006 [Jan 2007]

[60] S. Palla, *"A Multi-Variate Analysis of SMTP Paths and Relays to Restrict Spam and Phishing Attacks in Emails,"* Masters Thesis, University of North Texas, 2006 [Jan 2007]

[61] S. Palla, H. Husna, R. Dantu, J.W. Cangussu, "Behavior Patterns of Spam Botnets," *Conference on Email and Anti-Spam (CEAS)*, 2007 [Aug 2007]

[62] S. Rago, "VoIP Spells Equipment Oppurtunities Now," *Networking and Optical Communications Topical Report*, Isuppli, 2006 [Mar 2007]

[63] A. A. Rahman, S. Hailes, "A Distributed Trust Model," *Proceedings of New Security Paradigms Workshop*, ACM Press, pp. 4860, 1998 [Nov 2004]

[64] I. Ray, S. Chakraborty, "A Vector Model of Trust for Developing Trustworthy Systems," *In Proceedings of 9th European Symposium on Research in Computer Security (ESORICS'04)*, Sophia Antipolis, France, 2004 [Sep 2005]

[65] Y. Rebahi, D. Sisalem, "SIP Service Providers and the Spam Problem," *Proceedings of Voice over IP Secuity Workshop*, Washington, USA, 2005 [Jan 2006]

[66] G. Richarte, "Modern Intrusion Practices," *CORE security technologies*, http://www1.corest.com/common/showdoc.php?idx=360&idxseccion=13&idxmen, July 2003 [Oct 2003]

[67] I. Rigoutsos, T. Huynh, "Chung-Kwei: A Pattern Discovery based System for the Automatic Identification of Unsolicited E-mail messages," *In Proceedings of the first conference on E-mail and Anti-Spam*, 2004 [Jan 2005]

[68] M. Rogers, *"Running Head: Theories of Crime and Hacking,"* MS Thesis, University of Manitoba, 2003 [Nov 2003]

[69] M. Rogers, *"A New Hackers Taxonomy,"* University of Manitoba, 2003 [Nov 2003]

[70] J. Rosenberg, H. Schulzrinne, G. Camerillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "Session Initiation Protocol," *RFC 3261*, 2002 [Apr 2004]

[71] J. Rosenberg, C. Jennings, J. Peterson, "The Session Initiation Protocol (SIP) and Spam," *Spam Draft*, draft-ietf-sipping-spam-02.txt, 2006 [Dec 2006]

[72] I. Rowley, "Managing In An Uncertain World: Risk Analysis And The Bottom Line," *IEEE Colloquium on Systems Engineering Contribution to Increased Profitability*, Oct 1989 [Oct 2003]

[73] J. Sabater, C. Sierra, "Review on Computational Trust and Reputation Models" *Artificial intelligence review*, 24:33-60, 2005 [Apr 2004]

[74] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz, "A Bayesian Approach to Filtering Junk E-Mail. Learning for Text Categorization," *Papers from the AAAI Workshop*, pages 5562, Madison Wisconsin. AAAI Technical Report WS-98-05, 1998 [Nov 2003]

[75] G. Sakkis, I. Androutpoulos, G. Paliouras, V. Karkaletsis, C.D. Spyropoulos, P. Stamatopoulos, "A Memory based Approach to Anti-spam Filtering for Mailing Lists," *Information Retrieval*, 2003 [Nov 2003]

[76] G. Salton, M.J. McGill, *"Introduction to Modern Information Retrieval,"* McGraw- Hill, 1998 [Mar 2004]

[77] B. Scheiner, "Attack Trees: Modeling Security Threats," *Dr. Dobbs Journal*, Dec 1999 [Nov 2003]

[78] C. Scott, J. Branch, B. Szymanski, E. Breimer, "Intrusion Detection: A Bioinformatics Approach," *Proceedings of the 19th Annual Computer Security Applications Conference*, p.24, December 08-12, 2003 [Nov 2003]

[79] "Big Security Flaws Found In Asterix PBX, IAX VoIP Client," *Network computing*, http://www.networkcomputing.com/channels/networkinfrastructure/showArticle.jhtml?articleID =189400851, Jun3 2006 [Oct 2006]

[80] J.M. Seigneur, N. Dimmock, C. Bryce, C.D. Jensen, "Combating Spam with TEA (Trustworthy Email Addresses)," *In Proceedings of the Second Annual Conference on Privacy, Security and Trust (PST04)*, pages 4758, Fredericton, New Brunswick, Canada, 2004

[81] C.E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379-423 and 623-656, July and October 1948 [Apr 2007]

[82] O. Sheyner, H.J. Joshua, S. Jha, R. Lipmann, J.M. Wing, "Automated Generation and Analysis of Attack Graphs," *IEEE Symposium on Security and Privacy*, 2002 [Oct 2003]

[83] D. Shin, C. Shim, "Voice Spam Control with Gray Leveling," *In Proceedings of 2nd VoIP Security Workshop*, Washington DC, 2005 [May 2005]

[84] S. Siegel, *"Non-Parametric Statistics for the Behavioral Sciences,"* McGraw-Hill, 1956 [May 2007]

[85] N. Soonthornphisaj, K. Chaikulseriwat, P. Tang-on, "Anti-Spam Filtering: A Centroid Based Classification Approach," *IEEE proceedings ICSP*, 2002 [Apr 2004]

[86] L.P. Swiler, C. Phillips, D. Ellis, S. Chakerian, "Computer-Attack Graph Generation Tool," *IEEE Symposium on Security and Privacy*, 2001 [Nov 2003]

[87] K. M. Teal, "VoIP Network Security: How a Hacker Took Advantage of Vulnerabilities," *New Telephony*, http://www.businessweek.com/technology/content/jul2006/tc20060710_811021.htm, June 2006 [Feb 2007]

[88] "VoIP Future Outlook: Revenue Forecast," *Telephony Online*, http://telephonyonline.com/voip/metrics/voip_revenue_forecast_021705/, June 2006 [Mar 2007]

[89] Y. Wang, J. Vassileva, "Bayesian Network-Based Trust Model," *In Proceedings of IEEE/WIC International Conference on Web Intelligence*, 2003 [Sep 2005]

[90] Y. Wang, J. Vassileva, "Bayesian Network-Based Trust Model in Peer-to-Peer Networks," *Proceedings Workshop on Deception, Fraud and Trust in Agent Societies at the Autonomous Agents and Multi Agent Systems(AAMAS-03)*, 2003 [Sep 2005]

[91] B. Wattson, "Beyond Identity: Addressing Problems that Persist in an Electronic Mail System with Reliable Sender Identification," *In Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, 2004 [Oct 2006]

[92] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics*, 1, 80-83, 1945 [Aug 2006]

[93] "WINBUGS" *http://www.mrc-bsu.cam.ac.uk/bugs* [Oct 2003]

[94] D.B. Wright, "Receiver Operating Charactristics Curves" *Encyclopedia of statistics in behavioral science*, volume 4, pp. 1718-1721, 2005 [Jun 2006]

[95] W. Yih, J. Goodman, G. Hulten, "Learning at Low False Positive Rates," *Conference on Email and Anti-Spam*, 2006 [Feb 2005]

[96] B. Yu, M.P. singh, "Towards a Probabilistic Model of Distributed Reputation Management," *Proceedings of 4th Workshop on Deception, Fraud and Trust in Agent Societies*, Montreal, Canada, 2001 [Jul 2004]

[97] B. Yu, M.P. singh, "An Evidential Model of Distributed Reputation Management," *In Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Volume 1, ACM Press 294-301, 2002 [Apr 2004]

[98] J. Yuill, S.F. Wu, F. Gong, H. Ming-Yuh, "Intrusion Detection for an On-going Attack," *RAID symposium*, 1999 [Oct 2003]

[99] G. Zacharia, P. Maes, "Trust Management through Reputation Mechanisms," *Applied Artificial Intelligence* , 14(9) 881-908, 2000 [Mar 2004]

[100] G. Zacharia, A. Moukas, P. Maes, "Collaborative Reputation Mechanisms in Electronic Marketplaces," *Proceedings of 32nd Hawaii International Conference on System Sciences*, 1999 [Mar 2004]

[101] P.R. Zimmerman, *"The Official PGP User's Guide,"* MIT Press, 1995 [Jun 2003]