Voip Honeypot Architecture

Mohamed Nassar, Radu State, Olivier Festor

VoIP Honeypot Architecture

Mohamed Nassar, Radu State, Olivier Festor
LORIA - INRIA Lorraine
615, rue du jardin botanique,
54602 Villers-Lès-Nancy, France
Email: nassar, state, festor@loria.fr

Abstract— **Voice Over IP (VoIP) or telephony services over Internet announces a new revolution in the telecommunication world for its management simplicity and cost reduction. VoIP security extends the existent risk range of IP protocols and infrastructures and introduces new attacks as well. Threats identification and standardization, secure signaling and media architectures, as well as intrusion detection and prevention mechanisms are currently under debate in the research community. We propose in this article a SIP (Session Initiation Protocol) specific honeypot. We describe its design and implementation. We detail the inference mechanism which classifies the received messages. We show how the model investigates about a received call and raises an appropriate conclusion.**

I. Introduction

Even if VoIP attacks are not in the headline news in security reviews yet they soon will be of major harmfulness for the Internet telephony services. We witness more and more cases like eavesdropping which harms the customers privacy, or fraudulent usage which impacts enterprises budget. Some of the attacks are inherited from the vulnerabilities of data networks over which the VoIP stack is operating. Others are possible because of the high performance and quality of service required by the application. Securing VoIP introduces new challenges in security management, and demands specific defense components such as firewalls, intrusion detection systems, and honeypots.

"A honeypot consists of an environment where vulnerabilities have been deliberately introduced in order to observe attacks and intrusions" [1]. This defense strategy is very successful even if sometimes it can be dangerous. We propose in this paper a phone playing such role. We aim in the first place to record the intruders activities, and understand their methodologies in order to enhance our protection systems.

In the next section, we go through an overview of the VoIP threats. We debate about the research challenges of the VoIP security in section III. Functional aspects of our honeyphone to record and study attack traces is depicted in section IV. In section V we describe the honeyphone architecture. The honeyphone agent which is the core of this architecture is described in section VI. Section VII is an overview of the inference engine with an example of unrolling. We mention the related works in section VIII and we conclude the article in section X.

II. VoIP threats

The most important attacks that have started to be reported in the VoIP community are summarized below. Some attacks explore the target domain and promote other attacks. We refer the reader who is looking for a complete taxonomy to [2].

A. Service disruption

Denial of Service (DoS) attacks aim to affect the availability of the service application. According to the employed strategy, they are of different types:

- exploiting vulnerabilities in the VoIP stack of a server and take it down using malformed packets;
- flooding the available bandwidth of a server by a distributed attack where several botnets are used in amplification;
- flooding the server with a large amount of requests with different `Call-Ids` and without completing the three steps handshaking. The server capacities (memory and CPU) will be overloaded;
- CANCEL and BYE SIP specific attacks might take place against the SIP call establishment procedure. If someone tries to call Bob, and Trudy sends a `CANCEL` to Bob, then Bob will be prevented from receiving calls. If Bob tries to make a call, and Trudy sends a `CANCEL` to the destination, then Bob will be prevented from making calls. Otherwise, Trudy could send a `BYE` to terminate a session after a few moments of its setup. Trudy could use proxy responses such as 4xx(client error), 5xx(server error) or 6xx(global error) to convince Bob that an error situation is preventing him from making calls.

B. Call tracking

The interception of SIP `INVITE` packets provide sensitive informations such as the source and the destination of the call (`To`, `From`, `Via` headers). The SIP body may contain Session Description Protocol (SDP) informations that allow eavesdropping of unencrypted traffic or encrypted traffic is the key negotiation was unencrypted.

C. Call hijacking

This kind of attack exploits the absence of authentication, integrity and non repudiation mechanisms. Even if SIP uses a strong authentication scheme similar to the HTTP digest, some security gaps still exist. A SIP user agent, proxy, redirect or registrar server might challenge a client user agent to

1-4244-0799-0/07/$25.00 ©2007 IEEE

authenticate. In addition, a user agent can challenge back a proxy server to be sure that it also knows the shared secret. Unfortunately, only the server authenticates the client in most VoIP implementation and the following attack is possible: Bob wants to call Alice. Bob's phone sends an `INVITE` message to the proxy server within its domain in order to route the call towards Alice's domain. Trudy impersonates the proxy and redirects the call to its own IP address. To do so, Trudy has to create a crafted SIP response and put its own IP address in the `Contact` header. Bob's phone does not require the authentication of the proxy, so it accepts the response and redirects the call to pass by Trudy's machine.

Missing the end to end authentication is another common case, and a man in the middle scenario is possible. A session established between Bob and Alice could be hijacked by Trudy. Indeed, Trudy who has intercepted the `INVITE` in the beginning of the session, copies the `Call-ID, To` and `From` tags to a new `INVITE` packet and increments the sequence number `CSeq`. Trudy steps in and sends the crafted `INVITE` to Bob or Alice. Since no authentication is required, Trudy ends up by hijacking the session. She can change both media and signaling characteristics, like for example changing media ports, adding or deleting media streams, changing the signaling path (`Via` headers) or denying signaling from any side to its benefit (`Contact` header).

D. Password cracking and user enumerating

A brute force algorithm which tries to guess the correct password of a SIP URI can be launched against registrars or proxy servers. These attacks could be preceded by a scanning for SIP devices and enumeration of existing user names. The enumeration consists in requesting the location server with a sequence of different destinations. Depending on the case (if a destination exists and is registered, if it exists and is not registered, or if it does not exist), then the response of the location server is different. In such a way, the existing addresses are filtered and input to the cracking component. The attacker may also find the user names of a domain belonging to an enterprise by exploring the home page of the enterprise or by watching the SIP traffic exchanged in the domain.

E. SPAM over Internet telephony (SPIT)

The unsolicited SPAM messages that threaten users can be amplified by a more annoying voice advertising. SPIT scenarios include call centers (human employees), calling bots, ringtone SPIT (playing the advertising audios as the phone is ringing by use of the `Alert-info` header) and combinations of them [3].

F. Host based intrusions

Media Gateway Controllers (MGCs) and voice mail servers are critical points in the VoIP architectures. The MGCs are responsible for bridging between different networks such as between SIP based signaling (Internet) and ISUP based signaling (PSTN). The translation between a SIP message and an ISUP message consists of mapping the corresponding parameters. Malicious management of this process is another source of threat which is caused by the lack of authentication and integrity mechanism in the SS7 network. A more comprehensive threat model of the integrated signaling between VoIP and PSTN is found in [4]. Call Detail Records (CDRs) and users confidential informations must be protected against host intrusions and remote code execution as buffer overflow attacks.

G. Media protocols related attacks

The suite of multimedia transport protocols such as Real Time Protocol (RTP) and Real Time Control Protocol (RTCP) are vulnerable to attack. A demonstrative tool of the RTP play out attack was presented in [5]. Eavesdropping is about capturing and reconstructing the media stream from the RTP flow. While encryption prevents this, it can often be bypassed in those cases where the key is negotiated in clear.

Another media related weakness is that SIP is out of band and does not monitor and control automatically the media traffic. Modifications of the QoS characteristics are possible by means of media protocols and are not transparent to SIP which harms a QoS-based tariffing approach.

H. Supporting protocols related attacks

Address Resolution Protocol (ARP) and Domain Name System (DNS) support the VoIP application. ARP manages the data link layer in a local area network, particularly that of an enterprise, while DNS offers translation services between domain names and IP addresses. In ARP poisoning between a phone and a server, the intruder tries to bind between its physical address and the IP address of the server in the phone, and on the other side, to bind between its physical address and the IP address of the phone in the server. DNS poisoning is very similar and consists on injecting falsified records in the DNS server. MAC and IP spoofing are among fundamental flaws in the Internet and have to be addressed from a global view of security architecture.

I. Firewall traversal

VoIP requires a new generation of firewalls with dynamic features to open and close media ports with respect to the session negotiation parameters. Attacking the firewalls by DoS or compromising them discloses the protected devices. One way is to exhaust the firewall ressources by many requests for port opening and closing. To traverse the firewall, malwares using backdoors could takeover a media port and transfer any kind of traffic over this channel.

J. Caller-ID spoofing

Caller-ID or Calling Number Delivery is a traditional telephony service which allows the called Customer Premises Equipment (CPE) to receive a calling party's directory number. Automatic Number Identification(ANI) is a system used by telephone companies to determine the number of the calling party. Traditionally, Caller-id / ANI spoofing was a complicated process requiring access to the operator company's

Private Branch eXchange (PBX) or switches. With VoIP, it is easy to spoof the caller identity (`From` SIP header) so the callee will receive it as the number or the name of the incoming call. Caller Identity spoofing promotes identity theft and harms automated or human verification systems such as voice mails.

K. Phishing attack

Phishing is about masquerading as a trustworthy third party in an attempt to acquire confidential information from victims. The phisher uses a link on an email as a lure, deceives the victim by a web page seeming to be its bank home page and asking him to fullfill some fields with sensitive informations. In a VoIP phishing scam, the victims are sent an official looking email asking them to call into the bank at a mentionned DID (Direct Inward Dialing) number. The victims click and call (or may call from the PSTN) and are directed to an automated voice system requesting that they enter account numbers and personal informations. The low-cost and ease of VoIP service providing promise to see such attacks in the future.

III. Research challenges

In front of this large vector of VoIP threats, the research community is facing of multiple challenges. We focus on two compromises that are essential in the security management planning:

- Defense without complication: In most deployments, encryption and authentication mechanisms are not incorporated because of the complexity of their application and their use by the customers. However, when a customer receives a call on his VoIP phone with the number of his bank agency posted on the screen, he has to ensure that he is talking to his bank adviser. What makes this challenge more difficult is that the voice communications have left the well centralized and secure switches and PBXs and they are now in a new environment where:
  - signaling and data are managed by independent protocols;
  - VoIP systems are distributed, heterogeneous and under different autonomous administrations;
  - Internet hackers and intruders have much experience in exploiting the battleground;
- Defense without isolation: It is common that people distribute personal cards, post personal addresses and phone numbers and include these kind of data in directories. Reachability is a fundamental condition in social and business lives which can be violated while dealing with SPIT prevention. SPAM techniques could not be simply imitated because SPIT calls are more annoying and disturbing. Solutions that consist in blocking all the callers who are not found in the address book are not practical. Block the disturbing call after its occurrence is not satisfying either. For these reasons, many researchers claim that SPAM, SPIT and SPIM (SPAM over Instant Messaging are basically sociological events and it is hard to prevent them.

Inside our VoIP honeypot architecture, we propose a verification scheme as a participation in the debate of the challenges mentionned above.

IV. Functional aspects of the honeyphone

In this section, we motivate the need for a VoIP honeypot by introducing functional scenarios where it can work and bring realistic benefits. The honeyphone works in an enterprise domain where multiple user agents are served by a VoIP PBX or a SIP proxy. We built our own testbed with both Asterisk PBX [1] and SIP Express Router (SER) [2] in place.

A Private Branch eXchange PBX is a device that is owned by a private business and that connects office telephones with the public telephone network. Users of a PBX share a certain number of outside lines for making external telephone calls. PBX is a term referring to the Public Switched Telephony Network (PSTN) and not to VoIP/SIP architectures. However, we have chosen the open source project Asterisk PBX which is growing fast and replacing in many cases specific SIP proxies and servers within Small Office/Home Office environments (SOHO). Asterisk supports different VoIP protocols (e.g. H323, IAX) as well as other telephony technologies (e.g. ISDN, PSTN). In addition to PBX switching functionality, Asterisk allows the use of various application services such as Voicemail, file playback and directory listing. It has a codec translator, a scheduler and an I/O manager. From the SIP point of view, rather than being a gateway with other protocols and standards, Asterisk is a SIP registrar, location server, and a back to back user agent.

SER is a high performance and open source SIP server that can be deployed in larger environments. As an advantage over Asterisk, SER is able to handle a high number of users and calls per second. SER can be a SIP proxy, registrar or redirect server. It provides accounting, monitoring and security services in addition to a user provisioning web interface. In many deployments, SER and Asterisk can be used simultaneously to setup a VoIP solution. We have chosen the deployment depicted in figure 1 for our testbed. Asterisk monitors and controls outgoing VoIP traffic and therefore accounting tasks while SER is responsible for the incoming traffic and load balancing when necessary. Our choice allows a multitude of scenarios where our honeyphone will be registered once at Asterisk and once at SER, and so on.

The honeyphone registers its IP address with a number of SIP URIs at the registrar sever imbedded in Asterisk or SER. The SIP URIs of the honeypot may be declared to the outside world as users of the domain, but because they do not represent real users they should theoretically never be called. The honeyphone allows to record and study most of VoIP attacks. In a user enumeration, while the attacker is trying to discover the SIP URIs of one domain, it can fall on the

[1] Asterisk: The Open Source PBX, `http://www.asterisk.org`
[2] iptel.org SIP Express Router, `http://www.iptel.org/ser/`

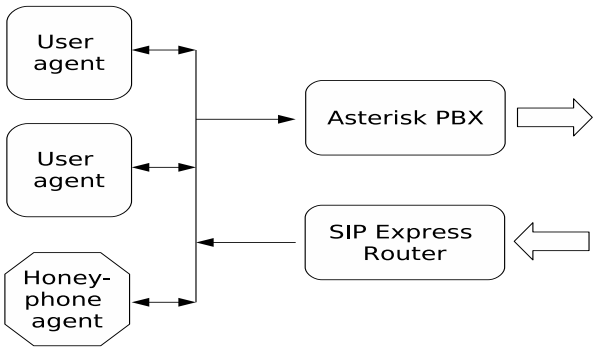


Fig. 1. Honeyphone environment

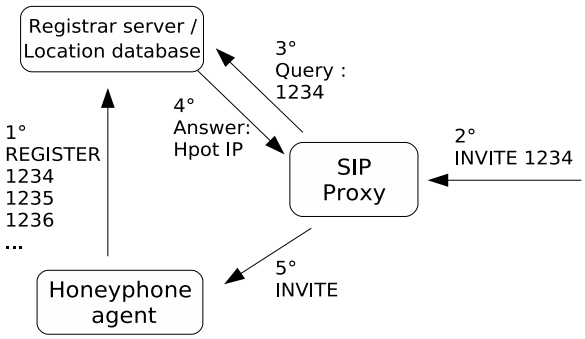


Fig. 2. functional scheme of reception of an INVITE

honeyphone address space. This could be a prestep for other forms of attack such as DoS. Similarly, the honeyphone could receive SPIT calls.

VoIP fingerprinting is a method used by an attacker to know the type of implementation of a SIP hardphone or softphone. In the second step, the attacker tries to exploit known vulnerabilities of the detected device. In passive fingerprinting, the attacker looks at the `User-Agent` header of a SIP message sent by the fingerprinted device. To deceive the attacker, we configure each SIP URI of the honeyphone to adopt one SIP user agent type and send its name in the `User-Agent` header. In such a way, we can study the attacks against one particular type of SIP user agent. Active fingerprinting is much more complicated [6] and we can not easily deceive the attacker. The honeyphone has to be implemented using the same stack of the SIP device desired to be under study. We do not address this difficulty yet, since active fingerprinting is still in a primary stage development and attacks still do not tailor the target device type. The honeyphone could masquerade also as a SIP proxy to record specific attacks against SIP servers.

Figure 2 reflects the functional scheme of the reception of an INVITE message by the honeyphone. When Asterisk or SER receives an INVITE to one of the honeyphone registered numbers, its location database will indicate the IP address of the honeyphone. The default behavior of a SIP proxy like SER is to add itself to the list of Via headers before it forwards the request, while Asterisk as a back to back user agent deletes all the Via headers and put itself instead. We prefer not to change the request when forwarding it and to not play the role of RTP proxy (in case of Asterisk). We intervene at the SER or Asterisk level, to configure them properly. We found that SER is more flexible for such requirements.

V. Honeyphone suite components

Starting from the tasks expected by the honey-pot, and admitting the importance of a high flexible and scalable design, we build our model around five main components as depicted in figure 3.

- The honey-pot agent: is the core of the honeypot architecture and the intelligent part of the application. It is responsible for accepting incoming calls and investigating possible attacks.
- The protocol stacks(SIP, SDP, RTP): are built based on the protocol standards, and are responsible for building and parsing the messages, as well as the transmission and reception of the packets over the transport layer.
- The honeypot profiles database: contains several configuration files and thus allows the administrator to choose one profile which is suitable to its needs rather than build it by itself. The benefit of this component is twofold, first, to set up the honeypot in its environment, and second, to control its behavior. We strive for application scalability and dynamism in the way we build up our profiles. Indeed, profiles in the database could be far from the honeypot area. They can be bots assessing the performance or the security of the domain.
- Reconnaissance tools: are used in the investigation procedure to check the received messages (e.g. service scanning: nmap [3], sip messages crafting: SIPSAK [4], passive OS finger printing tool: pOf [5]).
- The inference engine: is able to interpret automatically the results of investigation by means of special metrics and a Bayes model, we have chosen the Bayes model

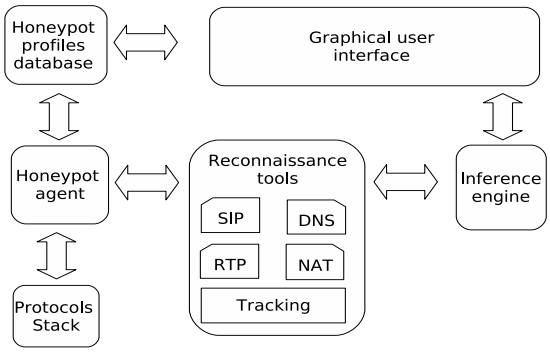


Fig. 3. Honeyphone architecture

[3]Nmap:Network Mapper, `http://www.insecure.org/nmap/`
[4]Sipsak: SIP Swiss Army Knife, `http://sipsak.org/`
[5]pOf: passive OS fingerprinting tool, `http://lcamtuf.coredump.cx/p0f.shtml`

due to its adaptive capability, scalability and real time performance. The belief output is on behalf of the domain administrator. More details are given in section VII.
- The graphical user interface: allows the administrator to choose and setup a honey-pot profile, as well as visualize traces, alerts and statistics.

VI. HONEYPHONE AGENT DESCRIPTION

In this section, we explain in more details the honey-pot agent components starting from the profile specifications and ending with the interaction with the reconnaissance tools.

A. Profile Configuration

A profile is formed by two independent sets of parameters. The "environment" parameters are used to set up the honeypot in its environment. They look like any soft-phone configuration parameters and use an `attribute:value` grammar. They determine two types of settings:

- host and honey-pot identity settings like for instance the IP address, protocol ports to be used, and the contact or list of contacts the honey-pot will register with.
- network dependencies like for instance the registrar server, the DNS server, SIP proxy and the VoiceMail server.

The "behavior" parameters allow to control the honey-pot behavior. The honey-pot can be in one of two functioning modes, either deterministic or randomized. Deterministic mode is based on fixed settings while randomized mode is based on mathematical random distributions. Each state of the state machine is coded in form of a clause:

```
State1 {
        Event1/Operator1[,
        Event2/Operator2 ...]
        }
```

To ease the presentation we make a short introduction to the state machine of a SIP user agent. The used model is taken from the open source VOCAL project documentation [7]. The base code provides a class for each state. Each state has a list of operators. Operators execute themselves at the entrance of the state, at the exit of the state or as response to occurred events which cause the machine to pass from a state to another. The events that drive the state machine of a SIP user agent are also coded in generic classes. The most important ones are `SipEvent` that occurs when a SIP message is received by the UA and `TimerEvent` that occurs when a period of time has elapsed. Our approach is to take the state machine of a SIP user agent respecting the SIP standard, and upgrade it by adding or modifying events and operators. In figure 4 we show a simplified state machine taken from [7] and how it can be written in our format. For example `Idle{onhook/OpOnHook}` means that in the `Idle` state the operator `OpOnHook` will take care of the event `onhook`.

To represent the operators that execute at the entry and the exit of a state, we add two keywords : `entry` is an event generated when logging in the state, and `exit` is an event generated when logging out the state. Let us now take an example of a honeyphone profile. Assume that we need to set up the honeyphone to respond automatically after let us say 5 seconds of receiving a call, the `Ringing` clause has to be changed as follow:

```
entry/OpStartRing(5),
ringtimeout/OpAnwerCall
```

After 5 seconds, the operator `OpStartRing` generates an instance of TimerEvent class. `ringtimeout` will be caught by the `OpAnswerCall` operator that drives the machine to the `InCall` state. To play back a speech file for 60 seconds after answering the call, a new line must be added in the `Incall` clause. In addition, we can record the conversation:

```
entry/OpRecord,
entry/OpPlayFile(speech0.mp3, 60)
```

If 2 operators catch the same event, they execute in parallel threads.

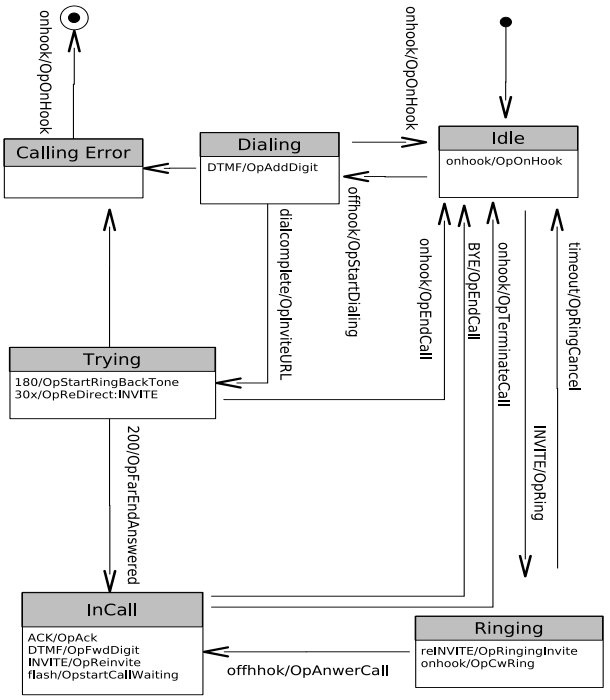
The advantage of such an approach is that instead of events initiated by human interaction such as `offhook` and `ophook`, a honeyphone is able to automate these decisions. The preparatory stage is the process of transforming the behavior configuration file into a runnable state machine and it is fully automated. All the operators and events are pre-coded in appropriate classes and documented so that they can easily be used. A more friendly way to set up the different behavior requirements can be through the graphical interface. The GUI prompts the administrator to fill up a review form and then provides a suitable file to the preparatory stage.

B. Interaction with the reconnaissance tools

The investigation procedure about a received INVITE message is added in the honey-pot machine at the `Idle` clause as a `INVITE/OpInvestigate` statement. In a parallel direction of the call progress, the operator `OpInvestigate` will release the work of the verification tools through the interface of a specialized library.

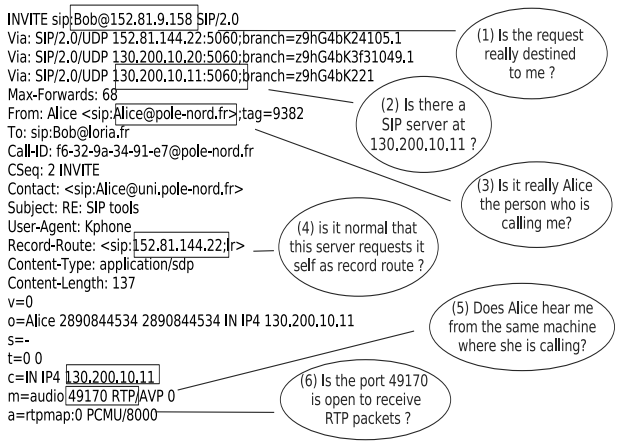
The IP addresses and protocol ports that participate in the session initiation form the list under interrogation. The concerned pieces of the INVITE message are the domain part in the `From` header, `Contact` and `Via` headers, `Record-Route` and `Route` if present, and from the SDP body `Connection(c)`, `Owner(o)` and `Media(m)` parameters. Some of the concerned pieces are underlined in the typical INVITE in figure 5. For each IP address or host name in the above cited list, the inquiry procedure aims to evaluate the following fields:

- `WHOIS` : ownership and real world information by means of Internet registries. In the honeyphone database, a WHOIS trust table assigns for each entry a trust coefficient between 0 and 1.
- Valid DNS : checking of DNS information by means of DNS, Reverse DNS, DNS SRV [8]or ENUM [9] requests according to the case;
- `AS` : the autonomous number where the IP or host resides. In the honeyphone database, an AS trust table assigns for each entry a trust coefficient between 0 and 1.
- `GL` : the geographic location of the IP or host. In the honeyphone database, a location trust table assigns for




```
Mode : Deterministic
Idle {
        onhook/OpOnHook,
        INVITE/OpRing,
        offhook/OpStarDialing
        }
Ringing {
        reINVITE/OpRingingInvite,
        onhook/OpCwRing,
        timeout/OpRingCancel,
        offhook/OpAnswerCall
Dialing {
        onhook/OpOnHook,
        DTMF/OpAddDigit,
        dialcomplete/OpInviteUrl,
        }
Trying{
        180/OpStartRingBackTone,
        30x/OpReDirect,
        onhook/OpEndCall,
        200/OpFarEndAnswered,
        }
Incall{
        onhook/OpTerminateCall,
        BYE/OpEndCall,
        DTMF/OpFwdDigit,
        INVITE/opReinvite,
        flash/OpstartCallWaiting,
        }
Calling Error{
        onhook/OpOnHook,
        }
```

Fig. 4. Behavior configuration of a simplified SIP UA state machine



To respond to question 1:
- check if i a m registered with the SIP URI: Bob

To respond to question 2 :
- send OPTIONS request to 130.200.10.11:5060
- if the response is OK so some SIP stack is running there
- check for some DNS SRV record in the DNS server

To respond to question 3 :
- hard to answer
- check if all the participating servers do not allow changing of the caller-ID

To respond to question 4 :
- check if the server belongs to the Via chain
- evaluate a trust in the server

To respond to question 5:
- check if 130.200.10.11 corresponds to uni.pole-nord.fr in the Contact header using DNS or Reverse DNS requests

To respond to question 6 :
- send UDP packet to 130.200.10.11:49170
- if no ICMP Destination Unreachable response is received so the port is open

Fig. 5. Typical INVITE message

each city a trust coefficient between 0 and 1.

- `Has Port` : if a port (SIP or RTP) proclaimed to be open is really open;
- `Fingerprint` : normally found in the `User-Agent` header, however we can actively fingerprint the caller device by sequence of OPTIONS requests as proposed in [6]. A fingerprint trust table assigns for each device a trust coefficient between 0 and 1;
- `Historical cache`: check in if and how frequently the IP or host is seen by the honeyphone. The maximum frequency is 1 and is reached if the IP or host was seen 10 times or more. An aging threat decreases the frequency and can delete the entry if it was not seen for a long time.

In addition, the inquiry procedure makes IP trace routes starting from the honeypot and arriving to the different hosts and proxies mentionned in the INVITE. The outputs of the inquiry procedure are brought towards the inference engine.

VII. The inference engine of the honeyphone

The mere fact of receiving a message at the honey-pot is subject of suspicion. In the same time, it could be the result of unintentional error from an innocent user. Our approach is not restricted to messages received by the honeyphone but we aim at a larger scope to distinguish between normal messages and rogue messages. Rogue messages are those produced by crafting tools where some SIP headers are spoofed but are also those normally built up but forming part of VoIP specified attacks such as SPIT, user enumeration or DoS.

The interpretation of the results issued by the network-based investigation functionalities could be assigned to a human operator who ranks an incoming message and notices

if something goes wrong. Nevertheless we choose to release the operator from this task and provide our honeyphone with an artificial intelligence engine. The networks of plausible inference proved high efficiency to materialize the human reasoning by a probabilistic formalism. For example, if a message arrives at the honeyphone which is publicly unknown, there is a high prior probability for a suspicious hypothesis. We give a brief introduction about the Bayes inference rules and then we present our detection model.

A. Introduction to the Bayes inference

Bayesian methods provide a formalism for reasoning about partial belief under conditions of uncertainty [10]. They are based on the empirically verifiable relationship between posterior (the belief we accord a hypothesis H upon obtaining evidence e) and prior (P(H)) probabilities:

$$P(H/e) = \frac{P(e/H)P(H)}{P(e)}$$

A Bayesian network is a directed acyclic graph whose arrows represent causal influences and each of its nodes represents certain knowledge. Bayesian trees are a subset of bayesian networks. In a Bayesian tree, each node might have several children and one parent. The propagation and fusion of the belief in a Bayesian tree are proceeded under the following rules:

- The likelihood (or diagnostic) messages λ are travelling upward the tree.
- The prior (or causal) messages π are travelling downward the tree.
- A child is linked to its parent by a conditional probability table (CPT) of which the elements are given by:

  $$CPT_{ij} = P(child = j/parent = i)$$

  Each row of the matrix is a discrete distribution over the child node states giving the parent node state and thus it sums to 1.
- The propagation of the prior messages is given by:

  $$\pi_{child} = \alpha\pi_{parent} \bullet CPT(child/parent) \qquad (1)$$

  where π is a row vector and α is a constant to normalize the distribution.
- The propagation of the likelihood messages is given by:

  $$\lambda_{child}^{to\_parent} = CPT(child/parent) \bullet \lambda_{child} \qquad (2)$$

  where λ is a column vector.
- The likelihood messages are fused together by an elementwise multiplication:

  $$L_{parent}(i) = \prod_{child \in children(parent)} \lambda_{child}^{to\_parent}(i) \qquad (3)$$

  λ_{parent} is obtained by normalizing the vector L_{parent} to the unit sum.
- Finally, the belief over the states at a node is obtained by an elementwise multiplication of λ_{parent} and π_{parent} and then normalizing the resulting vector by an appropriate constant β:

  $$BEL(i) = \beta\pi(i)\lambda(i) \qquad (4)$$

B. Bayes model to detect crafted and suspicious messages

We propose the model in figure 6. We are interested in three hypothesis at the root node:

- `Crafted`: where the incoming request is proved to be built up by an attack tool and some hosts and ports are verified to be not as claimed,
- `Suspicious`: where the hosts and ports are positively verified, but the request is part of a traffic attack such as SPIT, and
- `Normal`: where everything looks good and the request is just about a user fault.

The Bayes model can learn new hypothesis and we may have different degrees of suspicion or craftiness. We are satisfied with these three hypothesis in our first prototype. The leaf nodes are observable evidences drawn directly or in function of the investigation responses. The variables at the leaf nodes are assumed to be conditionally independent. The new variables are defined as follow:

- `To`: The first thing that we do when we receive a message is to check up if it is really sent for us. The honeyphone checks if the SIP URI in the `To` header is one of the SIP URIs that it is registered with at the registrar server. Under the Normal hypothesis, where receiving a message is due to a routing fault or a user fault, we assume that `To` will be false. Note that this reasoning is valid from the honeypot perspective, if the inference engine works with a user phone or group of phones, it is normal to receive messages with the right To header.
- `Dispersion of source points`: Let us take the honeyphone as reference and name it O. We consider also the signaling source (the host in the first Via header) as point A, the host in the Contact header as point B, the mediating source (i.e. the host in the Connection parameter) as point C , the host in the Owner parameter as point D, and the host returned by the location of the SIP URI in the From header as point E. For each of the 5 mentionned points, we measure the number of IP hops separating it from the honeyphone by means of traceroutes. Our traceroute consists of a sequence of OPTION requests with increased TTL IP header (A SIP layer traceroute is also possible using a sequence of OPTION requests with increased `Max-Forwards` SIP header). We have chosen the number of IP hops as a metric because it is more accurate than geographic location or AS number. Typically, the 5 mentionned points are very close and have to be at an equal number of hops from the honeyphone. A dispersed distribution of them reinforces the Crafted and Suspicious hypothesis. We adopt the dispersion of source points as the standard deviation of their number of hops:

  $$\sigma_{sources} = \sqrt{E(X^2) - (E(X))^2}$$

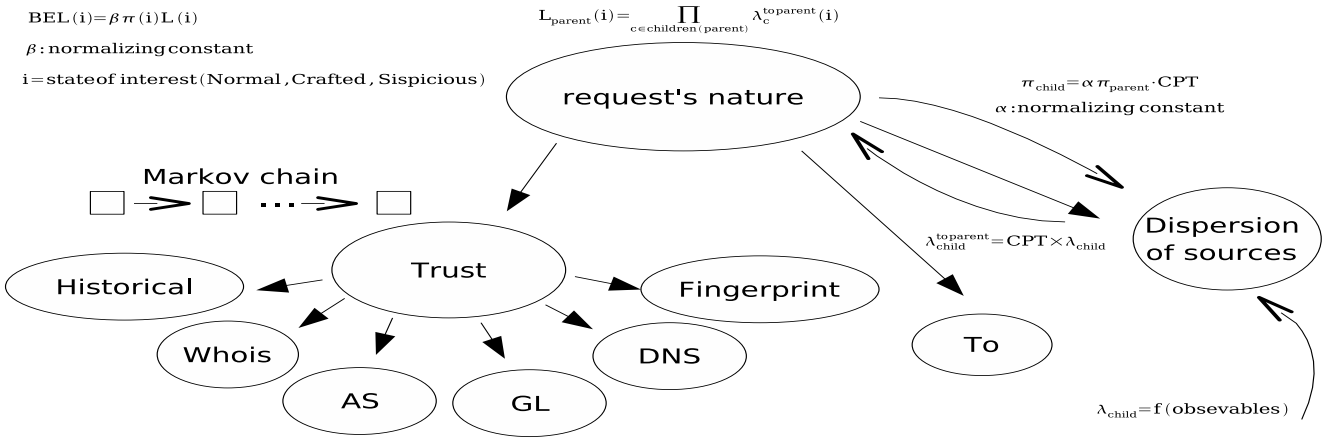


Fig. 6. Artificial reasoning about the nature of a SIP request

Where X is the discrete variable representing the number of hops of the 5 points and E is the expectation function.

- `Trust`: The trust on a message is composed of the trust in its source and the different proxies involved in its routing. As depicted in the figure 6, the trust variable is induced from its child nodes which are directly observed in the investigation results. The overall trust of the source and the proxies is calculated over multiple steps. In each step, we present the values of the pointed host and we release the inference work. A discrete Markov chain helps us to summarize the trust as steps proceed. This chain is designed to memorize a high degree of distrust if one proxy is declared to be distrusted, and to balance between different trust hypothesis otherwise.

C. Unrolling of a scenario

To clarify our approach, let us assume the following situation:

1) The INVITE message mentionned in figure 5 is received at the honeypot.
2) The request is totally correct, it is not crafted or part of an attack.
3) The honeypot was not registered with the SIP URI found in the To header.
4) The cause of receiving this invite is a configuration error at the registrar server.

We are interested in this scenario to show how the honeyphone will work upon receiving the message until concluding that it is kind of Normal behavior. We do not show all the calculation details but we just give an indication of the approach. Firstly, let us show in the next table the interesting headers and their values:

Header	Value
From	Alice@pole-nord.fr
Contact	Alice@uni.pole-nord.fr
Via_0	130.200.10.11:5060
Via_1	130.200.10.20:5060
Via_2	152.81.144.22:5060
Record-Route	152.81.144.22
Connection	130.200.10.11:49170
Owner	130.200.10.11

The table contains 3 IP addresses, one domain name and one host name. The results of the inquiry procedure are shown in the table I. To infer the trust on the list, we take its members one after the other. We present the likelihood messages at the leaf nodes of the sub tree corresponding to the root Trust, and we present the prior message at the root Trust. The likelihood messages come from the investigation results and the prior messages come from the Markov Chain. The CPT matrices are already fixed based on logical semantics and simulation adjustments. The field of each observable is divided in multiple intervals. In this unrolling, we use a small number of intervals for simplicity reasons. For the same reasons, only two hypothesis stand at the Trust node: T(trusted) and D(distrusted). Let "uni.pole-nord.fr" or shortly "uni" be the first member to process and take it as an example. The CPT matrix that relies between Trust and Historical nodes is:

$$CPT(Historical/Trust) = \begin{array}{c|ccc} & 0 & [1-2] & [3-\propto] \\ \hline T & 0.9 & 0.1 & 0.0 \\ D & 0.1 & 0.2 & 0.7 \end{array}$$

"uni" is never seen before, the likelihood message is:

$$\lambda_{Historical}\left|\begin{array}{l} 0 \\ [1-2] \\ [3-\infty] \end{array}\right. = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

The likelihood at the parent node is obtained from equation (2):

$$\lambda^{to\_parent}_{Historical}\left|\begin{array}{l} T \\ D \end{array}\right. = \begin{pmatrix} 0.9 \\ 0.1 \end{pmatrix}$$

TABLE I

RESULTS OF INVESTIGATION FOR THE INVITE MESSAGE

	pole-nord.fr	uni.pole-nord.fr	130.200.10.11	130.200.10.20	152.81.144.22
Frequency (Historical)	0 (not seen before)	0	0	0	0
WHOIS	Universite Pole nord	Universite Pole nord	Universite Pole nord	Universite Pole nord	LORIA-INRIA
AS	-	1200	1200	1200	1100
G.L.	Toulouse	Toulouse	Toulouse	Toulouse	Nancy
Are ports opened ?	-	yes	yes	yes	yes
SIP F.P.	-	Kphone	Kphone	SER	SER
DNS	true	147.210.9.155	uni.pole-nord.fr	proxy.pole-nord.fr	proxy.loria.fr

Similarly the other likelihood messages are initiated and calculated. The next step is to fuse them using equation(3) and normalize the result:

$$\lambda_{Trust}\left|\begin{array}{l} T \\ D \end{array}\right. = \left(\begin{array}{c} 0.91 \\ 0.09 \end{array}\right)$$

The prior vector is initiated by equal probabilities for T and D. The belief is calculated using equation (4):

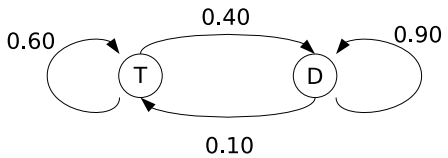
$$Bel_{Trust}\left|\begin{array}{l} T \\ D \end{array}\right. = \left(\begin{array}{c} 0.91 \\ 0.09 \end{array}\right)$$

The result is taken as post-belief and will be used to calculate the pre-belief for the subsequent inference:

$$PRE\_BEL_k = POST\_BEL_{k-1}M$$

The matrix M defines a Markov chain that translates the fact that a list tends to be distrustable if one of its member tends to be distrustable:

$$M = \begin{array}{c|cc} & T & D \\ \hline T & 0.60 & 0.40 \\ D & 0.10 & 0.90 \end{array}$$



The pre-belief is not different than the prior probability vector which will be used in the calculus of the subsequent member in the list.

At the end of the five inferences, we obtain the trust over the list:

$$\lambda_{Trust}\left|\begin{array}{l} T \\ D \end{array}\right. = \left(\begin{array}{c} 0.95 \\ 0.05 \end{array}\right)$$

The message seems to be trustable which indicates a Normal behavior hypothesis. Now we have to present the likelihood messages at the other children of the Nature node. The `To` header does not correspond to any of the URIs the honeyphone is registered with:

$$\lambda_{To}\left|\begin{array}{l} Yes \\ No \end{array}\right. = \left(\begin{array}{c} 0 \\ 1 \end{array}\right)$$

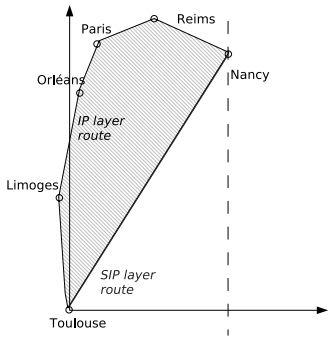


Fig. 7. SIP Vs. IP routes of the INVITE

The CPT matrix is already set to :

$$CPT(To/Nature) = \begin{array}{c|cc} & Yes & No \\ \hline Normal & 0 & 1 \\ Suspicious & 0.5 & 0.5 \\ Crafted & 0.5 & 0.5 \end{array}$$

After normalizing the likelihood at the parent is:

$$\lambda_{To}^{to\_parent}\left|\begin{array}{l} N \\ S \\ C \end{array}\right. = \left(\begin{array}{c} 0.5 \\ 0.25 \\ 0.25 \end{array}\right)$$

In our case, the 5 locations representing the source points are overcome. The dispersion is null ($\sigma_{sources=0}$) which indicates a Normal behavior hypothesis. The next step is to fuse all the likelihood messages at the Nature node using equation (3):

$$\lambda_{Nature}\left|\begin{array}{l} N \\ S \\ C \end{array}\right. = \left(\begin{array}{c} 0.9 \\ 0.05 \\ 0.05 \end{array}\right)$$

The prior vector is set to give more weight to the Suspicious and Crafted hypothesis regarding the nature of the honeyphone deployment.

$$\pi_{Nature} = \left(\begin{array}{ccc} 0.3 & 0.4 & 0.4 \end{array}\right)$$

Finally, our model produces the belief about the INVITE request basing on both prior and likelihood using equation

(4):

$$Bel_{Nature}\left|\begin{array}{l} N \\ S \\ C \end{array}\right. = \left(\begin{array}{c} 0.88 \\ 0.06 \\ 0.06 \end{array}\right)$$

The honeyphone concludes that the message is far from being a kind of naive (Crafted) or advanced (Suspicious) attacks, and that it is received by error (Normal). The SIP route formed by the succession of proxies, and the IP route traced back from the honeypot to the source are depicted in figure 7. A graphical interface provides the system administrator with such geographical view so he can detect abnormal routing situations (e.g.: proxies are abnormally distributed with respect to the IP or geographical routes). Automating this process is the subject of future works intention.

VIII. Related works

Many approaches to protect the emergent VoIP applications have been proposed in the research community. VoIP background, threats explanation and practical recommendations to configure properly, validate and monitor the security infrastructure are subject of recently published books as in [11]. Dynamic firewall design for IP telephony environments is evaluated and improved in [12]. Suitable intrusion detection and prevention architectures are proposed with prototype implementations as in [13].The authors of [4] present a threat model of the integrated signaling between PSTN and IP networks and propose a global solution to secure the gateways.

Our approach is new with respect to the previous works because it adds another layer of defense by using honeypots. Installing a honeypot allows to log and study attack traces and discover hackers methodologies ([14]). Our approach is also based on high reactivity properties of reconnaissance and investigation. We respond to the importance of inserting artificial intelligence in the analysis of data and the deduction of hypothesis. The theoretical background of our inference engine is inspired from [10]. Bayesian networks demonstrated real time feasibility and low fault rates when they were used in intrusion detection systems [15].

IX. Implementation and future works

The honeyphone software is partially implemented around the open source code of [7]. We hope to have the complete architecture integrated in the software to be inserted and tested in the VoIP test bed of the EMANICS[6] European project soon. Human users communicating across the project by a large set of IP phones will use different VoIP protocols and implementations which provide us with the necessary traces needed in the learning stage of the classifier. In addition, we are designing normal user bots according to normal user behavior in the PSTN telephony system to be launched in the test bed. In parallel, we are coding prototypes of VoIP attacks and SPIT bots to insert malicious attacks so the correctness of the classifier can be tested and re-adjusted. While the real time performance of the honeyphone to gather and deal with relevant informations about the attacker was considered from the beginning, optimizing this performance to put suitable decisions in action within the time constraints is matter of an extended study in the future.

[6] www.emanics.org

X. Conclusion

We presented in this paper an innovative approach to design a VoIP specific honeypot. The honeyphone is supplied by an application program interface which controls a rich set of network tools. Rather than receiving and dumping SIP packets to be studying as attack traces, the honeyphone is able to gather information in real time about the received messages. An important component in our architecture is the inference engine which can differ between a routing fault, a shooted crafted message and a distrustable call. Simulation unrolling of examples showed promising results about the effectiveness of the information gathering tools and the correctness of the inference engine deductions.

References

[1] F. Pouget and M. Dacier, "Honeypot-based forensics," in *AusCERT2004, AusCERT Asia Pacific Information technology Security Conference 2004, 23rd - 27th May 2004, Brisbane, Australia*, May 2004.

[2] VoIPSA, "Voip security and privacy threat taxonomy," Public Realease 1.0, Oct 2005, http://www.voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf.

[3] M. Hansen, M. Hansen, J. Möller, T. Rohwer, C. Tolkmit, and H. Waack, "Developing a legally compliant reachability management system as a countermeasure against spit," in *Third annual security workshop (VSW'06)*. ACM Press, June 2006.

[4] H. Sengar, R. Dantu, and D. Wijesekera, "Securing voip and pstn from integrated signaling network vulnerabilities," in *1st IEEE workshop on VoIP Management and Security (VoIP MaSe)*, Vancouver, Canada, April 2006.

[5] H. Abdelnur, V. Cridlig, J. Bourdellon, R. State, and O. Festor, "Voip security management," in *18th meeting of the Network Management Research Group (NMRG)*, Jul 2005.

[6] H. Yan, K. Sripanidkulchai, H. Zhang, Z.-Y. Shae, and D. Saha, "Incorporating active fingerprinting into spit prevention systems," in *Third annual security workshop (VSW'06)*. ACM Press, June 2006.

[7] L. Dang, C. Jennings, and D. Kelly, *Practical VoIP using Vocal*, 1st ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc., July 2002.

[8] "A dns rr for specifying the location of services (dns srv)," http://www.ietf.org/rfc/rfc2782.txt.

[9] "The e.164 to uniform resource identifiers (uri) dynamic delegation discovery system (ddds) application (enum)," http://www.ietf.org/rfc/rfc3761.txt.

[10] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[11] T. Porter, *Practical VoIP Security*. 800 Hingham Street, Rockland, MA 02370: Syngress Publishing, March 2006.

[12] U. Roedig, R. Ackermann, and R. Steinmetz, "Evaluating and improving firewalls for ip-telephony environments," in *1st IP telephony workshop*, Berlin, Germany, April 2000.

[13] Y. Wu, S. Bagchi, S. Garg, N. Singh, and T. K. Tsai, "Scidive: A stateful and cross protocol intrusion detection architecture for voice-over-ip environments." in *International Conference on Dependable Systems and Networks (DSN 2004)*. IEEE Computer Society, Jun 2004, pp. 433–442.

[14] T. Honeynet Project, *Know Your Enemy : Learning about Security Threats*, 2nd ed. Addison-Wesley Professional, May 2004.

[15] A. Valdes and K. Skinner, "Adaptive, model-based monitoring for cyber attack detection," in *RAID '00: Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*. London, UK: Springer-Verlag, 2000, pp. 80–92.