Model-Checking DoS Amplification for VoIP Session Initiation

Ravinder Shankesi, Musab AlTurki, Ralf Sasse,
Carl A. Gunter and José Meseguer

University of Illinois at Urbana-Champaign, Urbana IL 61801, USA

Abstract. Current techniques for the formal modeling analysis of DoS attacks do not adequately deal with *amplification attacks* that may target a complex distributed system as a whole rather than a specific server. Such threats have emerged for important applications such as the VoIP Session Initiation Protocol (SIP). We demonstrate a model-checking technique for finding amplification threats using a strategy we call *measure checking* that checks for a quantitative assessment of attacker impact using term rewriting. We illustrate the effectiveness of this technique with a study of SIP. In particular, we show how to automatically find known attacks and verify that proposed patches for these attacks achieve their aim. Beyond this, we demonstrate a new amplification attack based on the compromise of one or more SIP proxies. We show how to address this threat with a protocol change and formally analyze the effectiveness of the new protocol against amplification attacks.

1 Introduction

Relatively speaking, formal modeling and analysis of protocols with respect to their availability properties—in particular the analyses of vulnerabilities and/or defense measures with respect to Denial of Service (DoS) attacks—is a subject considerably less developed than formal analysis of other security properties such as secrecy and authentication. Part of the challenge is that availability properties are intimately related to performance, and therefore have an inescapable quantitative nature that does not have an obvious formal model or analysis technique.

In spite of these challenges, a number of formal approaches [18,16,1] [24,17,10,11,2,4,5], have indeed been proposed and shown effective in analyzing various kinds of DoS attacks and defenses. However, none of these works addresses directly the formal modeling of *amplification attacks*, in which an attacker is able to convert a given level of resources into a larger level by enlisting the aid of other nodes, often on a *network wide* basis. A characteristic example of such a strategy is a smurf attack, in which LAN broadcast addresses enable a single packet to be 'amplified' into a packet from each of the hosts on the LAN. Methods like the Cost-Based Framework [18] and its successors [16,1] cannot be straight-forwardly applied to this type of global attack. Indeed, at any given point in an amplification attack, the cost inflicted on a specific targeted server may not be significantly higher than that incurred by the attacker. What is new

M. Backes and P. Ning (Eds.): ESORICS 2009, LNCS 5789, pp. 390–405, 2009.

in this kind of attack is that *the cost may be spread out to an entire networked system*, possibly including the objects that mediate the network communication.

Vulnerabilities to this kind of attack have become more common as increasingly complex distributed systems are being deployed, ones that may rely on resource limits for the system as a whole rather than just for specific servers. A good illustration of this trend is the discovery of DoS vulnerabilities for the Session Initiation Protocol (SIP) that sets up Voice over IP (VoIP) telephony sessions; such vulnerabilities have been noted in efforts by IETF [14] and in the academic literature [11]. VoIP is a broad term describing a set of technologies enabling audio communication, similar to a telephone conversation, but over a packet-switched IP network, instead of a circuit-based network. Call set-up similar to circuit-switching is done by SIP proxies that assure that calls find their destination and bulk communications are handled by the communicating VoIP clients. DoS attacks on this system are able to effectively turn SIP proxies against one another with exploding messaging amplifications. With the growing reliance of telephony on VoIP internationally, such attacks must be viewed as a major systemic threat so efforts are being made to design protocols that are resilient to amplification attacks.

The aim of this paper is to develop and illustrate a new approach to the formal analysis of amplification attacks based on *model checking*. Since model checking computes the *truth value* of some property such as an invariant or a temporal logic formula, at first sight it might not seem easily applicable to the analysis of the *quantitative properties* involved in DoS attacks in general and amplification attacks in particular. The key observation in this regard is that one can define various quantitative measures, including measures on the global state of an entire system and not just on the local states of a given attacker or targeted server in that system. Then we can use various comparisons between such measures, or between a measure and a chosen threshold, as the Boolean-valued property that we model-check. In particular, we can characterize an amplification attack by means of states where some measure comparisons hold true. We call this technique *measure checking*.

Our main focus is on demonstrating the usefulness in practice of measure checking. We validate the effectiveness of measure checking for analyzing amplification attacks in two studies. In the first study we show how measure checking can discover a known but serious amplification attack on the SIP protocol. We then show that the IETF RFC5393 revisions for SIP are effective in eliminating this threat. These are model-checking studies, so the first part proves a risk for a representative set of initial configurations, and the second proves that risk is eliminated for that set. This does not prove that RFC5393 is always effective for any configuration, but such model-checking can be an effective tool to find flaws. In the second study we entertain the possibility that one or more SIP proxies are compromised. Typical security analysis techniques usually do include some type of study of what happens if, say, a session key is compromised, so such an investigation of defense-in-depth is of value. Moreover, the compromise of SIP proxies is somewhat likely given the nature of how these proxies are emerging

in practice, so such a concern is real. We use measure checking to find a new amplification attack that succeeds even for SIP augmented with the RFC5393 protections if a SIP proxy is compromised. It is not obvious how to address this insider threat, but we describe a technique that burdens attackers significantly at plausible cost to valid nodes. We again use measure checking to show that this technique is effective (for a non-trivial collection of configurations).

The paper is organized as follows. Section 2 gives some background on the SIP protocol and a short introduction to rewriting logic with particular emphasis on its use to model and analyze network protocols. Section 3 describes amplification attacks and the formal analysis framework for finding amplification attacks using measure checking. It shows that the original SIP protocol is vulnerable to amplification attacks, whereas SIP patched with RFC5393 is not. Section 4 shows that an amplification attack is still possible under the assumption of an insider proxy. Section 5 describes a new defense mechanism that we propose and gives an analytical bound on the amplification that an attacker can achieve under the modified protocol. It also gives a formal analysis that confirms the analytical bound. Section 6 gives a brief overview of related work on formal modeling and analysis of DoS. Section 7 concludes with a discussion of future directions.

2 Background

In this section we present the required background. We start with an overview of the SIP protocol and continue with a brief explanation of how rewriting logic and its Maude implementation are used to model and analyze the SIP protocol.

Session Initiation Protocol. Voice over IP (VoIP) consists of a set of protocols and related tools that deliver voice (and sometimes other media) over the Internet. There are different protocol suites, such as Skype, that support this functionality. The open protocol-suite by IETF is what we refer to as the VoIP protocol in this document.

The protocol suite consists of various protocols such as the SIP protocol used for initiating sessions between two users, the Session Description Protocol (SDP) used for exchanging session parameters, the Real-Time Transport Protocol (RTP) used for transfer of data once the session is established, and others. In this document we focus on the Session Initiation Protocol.

SIP is used for establishing a session between two parties who support VoIP. The session setup functionality in SIP is handled by various architectural components. User-Agent Clients (**UAC**) and User-Agent Servers (**UAS**) are the hardware or software components that initiate and respond to the end users requests respectively. A **Proxy** within a given domain handles the requests on behalf of user-agents belonging to that domain. It may require authentication from the client before it forwards any such requests. A User-Agent will typically register itself with a **Registrar** within its domain, and the agents actual IP addresses are stored with a **Location Server**. Note that these architecture components are logical in nature and in reality one or more of these components

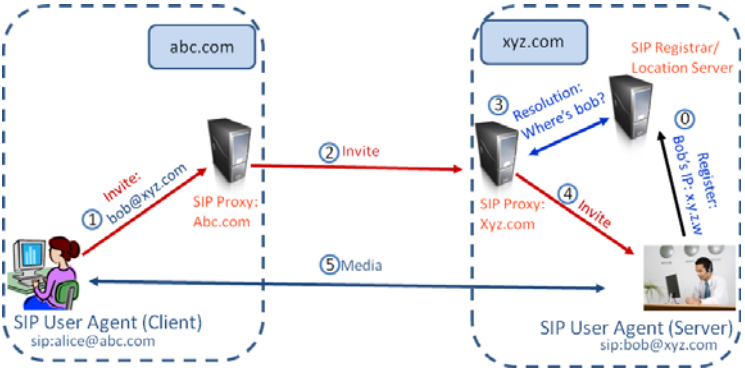

Fig. 1. SIP Protocol: Call setup when alice@abc.com invites bob@xyz.com

may be merged into a single piece of software or hardware. For instance, the proxy server usually performs the job of a registrar/location server.

One particular run of the SIP protocol is given in Figure 1 where the user Alice (at domain `abc.com`) is attempting to invite another user Bob (at domain `xyz.com`). Here, the client initiating the protocol, i.e., the UAC corresponding to `alice@abc.com`, sends a SIP Invite message addressed to Bob's logical SIP address `bob@xyz.com` (Step 1). The SIP invite message from Alice will be addressed to its proxy, which in turn forwards it to the proxy corresponding to the domain `xyz.com` (Step 2). The proxy at `xyz.com` tries to resolve Bob's actual IP address from it's location database (Step 3) and forwards the invite there (Step 4). Once the software server on the receiver, i.e., Bob's UAS, receives and accepts this message, the two parties can start exchanging voice-data using other media-transmission protocols such as RTP (Step 5).

Note that we have not shown some protocol messages for the scenario such as acknowledgement messages from the proxies (ACK, OK, TRYING) which inform the UAC when it is waiting for the response from Bob's UAS. Also, the outbound SIP proxy at domain `abc.com` may ask Alice for authentication (REAUTH) before it agrees to forward the Invite request on its behalf (Step 2).

Besides locating the actual address corresponding to a SIP address, proxies also perform various other functions. For instance, they also handle authentication, registration of the users, accounting the transactions and redirecting call invites to other locations (to support mobility of users).

A feature of particular importance for our analysis is the forking of invite messages. This feature allows proxies to forward a single invite to an address, say `sip:help@domain.org`, to multiple addresses. In effect this allows calls placed to one particular address to be handled by any one of the various users. This feature makes SIP vulnerable to various forms of amplification attacks known since [13], and as we further explain in this paper.

Protocol Analysis in Rewriting Logic and Maude. Rewriting logic [19] can model very naturally network protocols and, more generally, distributed systems [20]. A network protocol $\mathcal{P}$ is specified as a rewrite theory $\mathcal{P}{=}(\Sigma_{\mathcal{P}}, E_{\mathcal{P}}, R_{\mathcal{P}})$, where $(\Sigma_{\mathcal{P}}, E_{\mathcal{P}})$ is an *equational theory*, with typed function symbols $\Sigma_{\mathcal{P}}$ and equations $E_{\mathcal{P}}$ that specify the set of *states* of $\mathcal{P}$ as an algebraic data type, and $R_{\mathcal{P}}$ is a set of *rewrite rules* that specify the protocol's *concurrent transitions*. The rewrite theory $\mathcal{P}$ then provides both a *mathematical model* of the protocol (its initial model [19]), and an *executable semantics* for it by term rewriting, which can be used for both simulation and model checking.

We can illustrate all this by explaining how the SIP protocol is specified as a rewrite theory $SIP = (\Sigma_{SIP}, E_{SIP}, R_{SIP})$. A protocol state is modeled as a *configuration*, that is, a multiset of objects and messages built up by an empty-syntax (juxtaposition) union operator $\_\_ : Conf\ Conf \longrightarrow Conf$, where *Conf* is the type of configurations, and where the multiset union operator $\_\_$ is *associative*, *commutative*, and has the empty multiset $\emptyset$ as its *identity* element. Therefore, E_{SIP} contains the equations $(x\,y)\,z = x\,(y\,z)$, $x\,y = y\,x$, and $x\,\emptyset = x$. Instead, the rules R_{SIP} describe SIP's protocol transitions. For example, the acceptance of a call invitation by the addressee user is modeled by the rule

$$user(addr, addrSet)\ invite(addr', addr) \longrightarrow user(add, addrSet\ \ addr')$$

where the caller's address $addr'$ is added to the set of addresses in the callee's state. Note that rewriting with R_{SIP} takes place *modulo* the equations E_{SIP}, i.e., modulo the associativity, commutativity and identity axioms for $\_\_$.

The executability of rewriting logic specifications, such as the one described above for SIP, is supported by the Maude rewriting logic language [6]. Furthermore, Maude also supports *model checking* formal analysis, both for verifying reachability properties using its breadth first search command (`search`), and for verifying linear temporal logic properties [6].

Since our analysis of SIP amplification attacks uses breadth first search, we give a short summary of this type of model checking. As we show in this paper, the `search` command can be used for analyzing various quantitative measures on a selected set of system states specified in some way, e.g., by a predicate, or by selecting only terminating states with the `=>!` search mode. The measures can be performed on the selected states or on selected objects (e.g., an attacker) within such states. We can then use the `search` command to compare different measures on the selected states. For example, we can verify whether in any terminating state a measure `M1` is greater than a measure `M2` by giving the command

```
search init =>! X:Conf such that M1(X:Conf) > M2(X:Conf) .
```

Since breadth first search explores all reachable states, it is a *semidecision procedure* for finding states with the specified property in the infinite-state case, and becomes a decision procedure for finite-state systems.

3 Finding SIP Amplification Attack Vulnerabilities

In this section we first describe the kind of attacks that we are interested in, called *amplification attacks*. We then present our formal model of SIP. We describe how

we find the known attack for the SIP protocol version given in the RFC 3261 [12] in our formal model. We also specify and analyze the version of SIP with the patch according to the IETF standard [14], which we call SIP+5393, and find it to be safe by itself, i.e., the patch works as desired in our model.

Amplification Attack Description. A common form of DoS attacks is to ensure that the server spends a lot of its time (or other resources) servicing requests from the attacker. This makes it difficult for the server to handle requests from legitimate clients. The attacker can achieve this in different ways. Firstly, it can simply bombard the server with a large number of requests (a flooding DoS attack). Secondly, if the protocol allows it, the attacker can send requests that take disproportionately large amount of time for the server to process (compared to the effort spent by the attacker). These costly actions might include, for instance, generation of cryptographic keys.

Some analysis of this form can be done by using the Cost-Based Framework by Meadows [18]. In a cost-based analysis, every action (acceptance of a message, generation of a key, sending of a message, etc.) of either server or user is associated with a cost. The protocol is then considered secure with regards to a DoS attack if, at every accepting event in a run of the protocol, the cost of the server is within some factor of the cost of the attacker. In general, we want the cost of the server to be less than the cost of the attacker, with some threshold given by a tolerance relation in the framework.

In this work, we want to focus on a slightly different form of DoS attack. Instead of observing whether the protocol allows the server to have a higher cost (as compared to an attacker or user), we analyze if the protocol allows a configuration where with minimal starting cost the attacker can achieve a multiplying effort from the system in general. More specifically, we analyze if we can get a configuration where the number of messages on the network can amplify to essentially an arbitrary large number, starting from a very small number of messages, without requiring further work by the attacker.

It is obvious that, for such configurations, if we looked at any given protocol step, the cost of one server is not necessarily much more than the cost of the attacker (unless we associate a very large cost with sending a message, which would be impractical). Therefore this type of amplification attack will not be straightforwardly detected in general using the cost-based framework. We describe later in this section how to detect such an attack.

Note that this type of attack of course implies that the total cost of all honest proxy servers together (by, say, using the number of messages sent and received as the measure) is much larger than that of the attacker, which in the best case only needs to send some very few initial messages to create what could best be described as a perpetual motion machine for the proxy servers to deal with.

Formal Analysis. Now we describe our formal analysis framework for the SIP part of the VoIP protocol. We focus on amplification attacks, as explained in the prior paragraphs. We develop a formal model of the SIP protocol in the rewriting-logic based engine Maude. It models the sending and receiving of invite

messages between proxies and users on a global shared channel as sketched in Section 2. Each proxy or user belongs to a domain, and either consumes the invite (presumably starting actual communication), or forwards the invite to another participant, or forks the invite to multiple recipients.

We use our model of the SIP protocol to analyze its behavior not just for one hand-picked starting state, but for a whole family of starting states. This family of starting states depends on three parameters: the number of proxies, the number of users, and the number of forking redirects that we consider. We define rewrite rules in our model that, depending on those parameters, non-deterministically create different initial configurations by adding users to proxies and connecting them. Each connection here states that a message for user u is to be forwarded, or forked, to the list of users $u_1, \ldots, u_n$, given by their respective domain and user name. Using breadth-first search model checking we then examine all possible initial configurations and the runs of the protocol starting from them. Note that we make sure to create as few isomorphic initial configurations as possible. An example of isomorphic initial configurations is with 2 domains, one case where the first domain has 2 users and the second domain has 1 user, and the other case where the first domain has 1 user and the second domain has 2 users. These are substantially the same, but would both be generated by a naive exhaustive state space generation. Note that each initial configuration represents a model of a number of proxies and users participating in the SIP protocol with the connections as specified. There is only one initial invite message on the network, and the network is modeled as a shared channel.

We apply our measure checking by means of breadth-first search in Maude, which explores all possibilities under the given non-determinism for the generation of initial configurations. Actually, the same command then also searches through all possible interactions of each model with the one given initial message. This of course requires enough memory in the system running the experiment, but we have had no issues with that as the attacks are reachable for fairly small numbers of proxies and users already.

Amplification Attack on the Original SIP Protocol. Measure checking breadth-first search finds the well-known amplification attack ([13]) of the SIP protocol in our model of SIP, based on RFC3261 [12]. The reason this attack is feasible is the availability of forking proxies, see Section 2. A forking proxy forwards an invite message it receives to more than one other proxy or user. If that invite comes back to this proxy in some way, e.g., through a loop, then it will be forked again. On each iteration of the loop at least one extra message will be generated. This results in an amplification attack by the extra messages and furthermore creates additional work for the proxies that are part of the loop.

We create the initial state space configuration with exactly one invite message to start as part of the search command. We are searching for states in which a number of messages exceeding a defined threshold (the simplest form of a measure) is on the network, where all the messages are in response to the one initial message. In that case we are interested in the initial configuration for which this is possible. That initial configuration shows how to set up the connections and

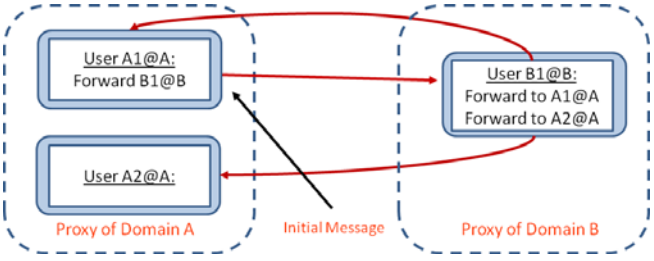

Fig. 2. SIP amplification attack configuration with 3 users and 2 domains

forking for users allowing the amplification attack to unfold. In our model we find the configuration that was already suggested in Section 2 for the attack.

We used the threshold of 50 messages related to the original one for the goal of the search, but it is obvious in this model that if 50 such messages can be created, an arbitrary number of messages could be created by further execution. As expected, the amplification attack is easily found in the model, for just 2 proxies and 3 users with forking to at most 2 other participants at each proxy. One configuration that can cause this attack is shown in Figure 2.

The search command that lets Maude find this attack in our model is:

```
search in SIP :
  createEnvironment(2,3,2) protocolSteps(100)     =>!
  X:Config such that amplification(50,X:Config).
```

The initial configuration is created by invoking `createEnvironment` with 2 proxies, 3 users and allowing forking of at most 2 for each forwarding entry, and one initial invite message. We also limit the total number of steps of the protocol to be executed to 100, by `protocolSteps`. We search for those *final* states, (specified with `=>!`), which we can do because we limit the total number of steps, in which there are at least 50 extra messages, which the predicate `amplification` tests, given that final configuration and the number of messages to check for.

Analysis of SIP+5393. We have also formally analyzed the effect of the proposed patch to the SIP protocol as described in IETF RFC5393 [14] by adapting our model of the original SIP protocol to accommodate the changes suggested by the patch and calling the result SIP+5393. The patch adds a so-called `via` field to each message, which keeps track of which proxies have been visited by this message so far. When a proxy receives a given message that has its own identifier in that `via` field already (and it is further recognized that no other parameter of the message has changed), it will drop the message.

We do not model the *max breadth* suggestion of SIP+5393, since that feature only spreads the attack out over time, but does not reduce the actual traffic that is being generated. It gives observers, like system administrators, more time to detect and stop the attack in ways that are not part of the protocol specification. As such, it is not central to whether an amplification attack is possible or not.

We find that there is no attack for the SIP+5393 protocol directly, and show that below. However, with an intruder, namely a single malicious proxy, a similar attack exists again, as shown in Section 4.

No Amplification Attack in SIP+5393. With the changes for SIP+5393 included in the model we can run the exact same search command that finds the attack for the original version, to see if it is still possible. In the SIP+5393 version of the protocol that attack is no longer found for the same parameters. We also investigated what happens when different parameters are changed, in particular allowing for more proxies and users, which gives more possibilities for the attacker. Our analyses showed that the attack is not possible even after those parameter changes. Looking at the attack in the old model, it is indeed quite clear that that attack is infeasible in the new model of the patched version.

4 A New Insider Threat

Amplification Attack with Intruder. It is important to note that for the IETF loop patch described and modeled as SIP+5393 to work, it is implicitly assumed that all SIP proxies are trusted to behave according to the protocol. In practice, most of the SIP-based VoIP solutions that are currently available assume fairly high levels of trust on intermediate proxies. This is primarily because of the fact that providing end-to-end security for SIP signaling, while maintaining simplicity and efficiency of the protocol, can be a very challenging task by the very nature of the protocol [22]. In particular, the protocol expects intermediate proxies to process SIP messages by accessing their headers and updating them (e.g., appending values to the `via` field). Therefore, means for protecting the integrity and confidentiality of SIP messages, like S/MIME as suggested in SIP [12], cannot be used to lift or relax trust assumptions on SIP proxies.

While these trust assumptions can be reasonable for proxy servers that are under direct control of the VoIP service provider, it is unfortunately too optimistic for a user or a service provider to assume that all SIP proxies are trustworthy. In fact, the possibility of a single *malicious proxy* along a SIP signaling path is actually quite practical, as an attacker can easily run his or her own proxy server from any given machine. Furthermore, an attacker can ensure that he/she keeps receiving the SIP messages by using the Record-Route option which points to its own address. A Record-Route option is usually inserted by a proxy to ensure that it is kept in the signaling path (typically to enable accounting). This malicious proxy can then remove all contents of the `via` field whenever a message passes through it, which, as we explain below, may re-introduce the same amplification attack as for the original protocol.

Definition (Intruder). An *intruder* is a malicious user that registers itself possibly at multiple proxies and sets up its forwarding preferences so as to create a forking loop, along which it assumes control of a forwarding proxy (referred to as the *malicious proxy*) that is capable of manipulating values of the `via` fields of incoming invite messages.

To see that the malicious proxy is not going to get overloaded by this DoS attack itself, it is important to notice that only a very small percentage of the messages created needs to go through it. The malicious proxy essentially needs to be on a single loop, which at each step creates *extra messages that are not part of the loop the malicious proxy is in*. The fraction of the network traffic which impacts that one machine depends on the length of that loop and the amount of forking along it. Effectively, the attacker can increase its bandwidth by a factor of around 60, which is the maximum allowed forking.

Formal Analysis of the Insider Threat. We have further extended the model for the patched protocol SIP+5393 with the possibility of an intruder. Specifically, we extend the model with a malicious proxy capable of dropping the `via` fields of invite messages. With this extension, we can show that an amplification attack entirely similar to the original one in Section 3 can be found by running the same search command for SIP+5393 but now with one intruder.

```
search in SIP+5393+Intruder :
  createEnvironment(2,3,2) protocolSteps(100)  withIntruder    =>!
  X:Config such that amplification(50,X:Config).
```

The intruder is non-deterministically associated with one proxy in the configuration using the operator `withIntruder` to enable the search command to explore all possible intruder assignments. The resulting attack is still of the form depicted in Figure 2, but now the intruder pays a small price on every loop. Thus, this is not an attack of the form of a perpetual motion machine and instead requires the attacker to do some work, but it still gives the attacker an amplification attack on the honest participants of the protocol with a lot of leverage in the form of a large multiplication factor for its capabilities.

5 A Tit-for-Tat Defense in Depth Mechanism

To harden the SIP protocol against the insider threat presented and analyzed in Section 4, we propose a slight modification of the SIP protocol with the IETF patch, denoted *SIP+5393+t4t*, that alleviates such a malicious proxy amplification-based DoS attack. The idea is to force such an intruder to expend a cost proportional to the number of messages generated and processed as a result of forking. The gain by the attacker should indeed be significantly lower than the 60-fold advantage in cost it can achieve over honest participants as noted in Section 4. Specifically, the proposed modification allows a message amplification attack to be mounted by an insider I only if I is willing to spend some message generation and processing effort that is at best (for the attacker) four times smaller than the total effort forced by the attacker on all honest parties.

SIP+5393+t4t Description. The proposed modification to the protocol is as follows. When a forking proxy P receives an invite message m that is to be forked to k nodes, the following steps are taken:

1. P sends a verification message to Q, the originating proxy of m.
2. If Q does not recognize the session of m, Q replies back to P with an "invalid session" response, which causes P to drop m.
3. Otherwise, if Q recognizes the session of m, Q sequentially performs k re-authentications with the user node in its domain that initiated m. For each one of the k re-authentication requests,
   (a) the user is simply re-authenticated according to the protocol.
   (b) If the re-authentication request succeeds, Q sends to P a success message, then P forwards a single copy of m to one of its remaining destinations.
   (c) Otherwise, if the re-authentication request is unsuccessful, a failure message is sent to P, which causes P to drop m altogether.

By the time P receives all k successful re-authentication responses from Q, P will have completed the process of forking the message m.

This modified protocol does not require any changes on the part of the end user device, which is potentially a phone hand set which cannot be updated easily, but only on the part of the proxies, which have to be changed anyway. We now define the cost of participating in the protocol.

Definition (Cost). The *cost of engaging in a protocol* is the total number of messages sent and received (processed) as a result of running the protocol.

Note that in the prior sections we did not need to consider the original invite sender in any detail, since all its cost was a single message. However, with the change to SIP proposed here, the initial sender needs to pay a cost whenever forking happens, in the form of the re-authentication messages. The attacker is the one setting up all the redirects and forks, and the one sending the initial message. Thus, it is reasonable to associate the costs of both the initial invite sender and the intruder-controlled proxy to the attacker.

Note, also, that when calculating the message-processing cost a naive cost calculation would associate a large cost when a single invite message is simply passed along a long chain of SIP proxies without forking (i.e., redirected from one proxy to the next) and consumed or discarded at the end. Clearly, this is not an amplification attack as we have described it. This does not create a DoS attack either on the network or on a given server as, at any given time, there is only one invite message in the system. We ensure that we do not consider such configurations as leading to an amplification attack by specifying the invariant (`amplification`) to include a measure on the number of active messages in the system. In the case of the scenario above, where a long chain of proxies simply forward the message to the next one, the number of active messages in the system at any given time will only be 1.

The multiplication factor the attacker can gain is the quotient of the cost of the legitimate participants of the protocol and the attacker's cost. As we shall see below, our modified version of the SIP protocol bounds, by a factor of at most four, the leverage that is available to the attacker for an amplification attack.

Amplification Bound. We now compute a bound on the proportional cost of amplification to legitimate proxies (or the environment) compared to the cost

incurred by the intruder, where the cost measure is as defined above (the cost of dropping values of the `via` field is assumed to be negligible and is included in the cost of forwarding a message). Let I be the intruder initiating the invite message m. We first note that the signaling path for m can, in general, be a graph with one or more cycles (at least one of which was carefully planned by I). The intruder proxy can be virtually anywhere within the graph as long as it lies on one of these loops. However, for I to maximize the effectiveness of his/her attack, I would need to minimize the amount of effort exerted by the intruder proxy. In particular, the originating (domain) proxy or a forking proxy are not the optimal choices for I. This is because the forking proxy not just forwards one message, but forks multiple messages and thus has a much higher cost associated with it than just that of forwarding messages. For the originating (domain) proxy it is even worse, as any forking with factor k will require it do k re-authentication steps with the user (which an intruder could just ignore doing) but also requires k successful re-authentication messages to the forking proxy (which even the intruder has to do), while the cost to the forking proxy is to receive k successful re-authentication messages and then forks k messages.

Theorem 1 (Tit-for-Tat Defense). *Using* SIP+5393+t4t*, and in the presence of an intruder, the cost of engaging in the protocol for legitimate proxies is at most four times the cost for the intruder.*

Proof. Suppose n is the total number of forking proxies along the signaling path of m. Suppose also that the average forking factor for m is k. Obviously, the worst case occurs when all n forking proxies are located on the signaling cycle created by I.

In every iteration of the loop, each of the n proxies in the signaling loop receives a message and replies back to the originating proxy, adding cost $2n$ to the forking proxies and adding cost n to the originating proxy. For each one of n messages the originating proxy sends k re-authentication messages to the originating user (adding cost nk). The originating user receives and replies with re-authentication responses (adding cost $2nk$ to the originating user). For each one of these re-authentication responses, the originating proxy forwards its reply to the forking proxies (adding cost $2nk$) and the forking proxies in turn forward the invite to the intended destination users (adding cost $2nk$ for receiving and sending the messages).

To summarize, the costs of processing m for the environment *env* (forking and originating proxies) and the attacker *att* (user I) are:

$$\begin{aligned} \text{cost}(env) &= 2n + 2nk + && \text{(received and sent by forking proxies)} \\ &\quad\ n + 2nk + nk && \text{(received and sent by originating proxy)} \\ &= 3n + 5nk \\ \text{cost}(att) &= 2nk \end{aligned}$$

Thus, we have

$$\frac{\text{cost}(env)}{\text{cost}(att)} = \frac{3n + 5nk}{2nk} = \frac{1.5}{k} + 2.5 \leq 4 \text{ for any } k \geq 1.$$

Formal Analysis. To verify correctness of SIP+5393+t4t, we extended the formal model we have developed so far by specifying the new behaviors for SIP proxies. With this modification, we can now verify for our running example of Section 4 by measure checking that in the presence of an intruder, the cost of an attempted amplification attack will always respect the bound given by Theorem 1 for SIP+5393 patched with our tit-for-tat defense mechanism.

```
search in SIP+5393+Intruder+t4t :
  protocolSteps(100) createEnvironment(2, 4, 3)
  withIntruder environmentCost(0) attackerCost(0)
  =>! X:Config environmentCost(N:Nat) attackerCost(M:Nat)
  such that amplification(50, X:Config) /\ N:Nat > 4 * M:Nat .
```

The operators `environmentCost` and `attackerCost` record, respectively, the costs for legitimate proxies and for the intruder. The query checks for a state where the attacker cost is less than a quarter of that for the environment, and fails for all the parametrically generated initial configurations, as expected.

6 Related Work

There have been several attempts to formally characterize DoS attacks in the literature. One of the early and influential such attempts was Meadows's framework [18]. Her framework implements a generic, cost-based approach in which actions in a protocol are identified and assigned costs, for example computational costs, that can then be combined and compared to the costs incurred by an attacker as a result of participating in the protocol. A DoS attack is then characterized by having legitimate participants expend more effort than a given threshold, specified by a tolerance relation in the framework. Meadows's work has later inspired other cost-based approaches to analysis of DoS, including some process-algebraic techniques such as information-flow based static analysis [16], and dynamic analysis using behavioral equivalence [1]. Another approach to analyze DoS defense is the game-based analysis proposed in [17]. Here the authors analyzed a modified version of a key-exchange protocol (JFKr) using client-puzzles, where the interaction between the attacker and the server is modeled as a two-player strategic game. The protocol is verified for fairness towards clients and the attacker with respect to their solving of the client-puzzles. A systematic study of various vulnerabilities in the VoIP stack, including amplification- and reflection-based DoS attacks, and the formal analysis of some of them were presented in [11]. Other formal approaches and extensions to deal with DoS attacks and defense mechanisms have also been developed in, e.g., [24,10].

Another approach is the use of general term rewriting formalisms, such as rewriting logic, which is the method we employ in this work. In addition to analysis of traditional security properties of protocols, e.g. the work in [7,9,8], rewriting techniques have been successfully applied to the analysis of availability properties against DoS threats. Examples of this in the literature include the analysis of TCP SYN floods-based DoS attacks [2], and verification of some of the properties of the adaptive selective verification (ASV) protocol against DoS

attacks [4], both within the shared channel model. One interesting feature of the analyses of DoS vulnerabilities in [2,4], also shared by a similar rewriting-logic based analysis of QoS requirements in [15], is the use of *statistical model checking* [21,23] in conjunction with a quantitative temporal logic like QuaTEx [3] to analyze quantitative, performance-related aspects of DoS attacks and defenses. Furthermore, a modular approach using generic cookie wrappers, also based on rewriting logic, was given in [5] for DoS protection specification in communication protocols while preserving their safety properties.

7 Discussion and Conclusions

We have presented a new model checking technique, called measure checking, to analyze amplification attacks on network protocols. The technique is based on the idea of defining cost measures not only on individual objects, such as an attacker or a targeted server, but also on the entire network system. Model checking then analyzes whether certain measure comparisons characterizing an amplification attack are possible or not. Our technique is entirely general and can be used within many different formal frameworks and with different model checking tools. We have illustrated its effectiveness in detail for the case of the SIP protocol of the VoIP protocol suite using rewriting logic and Maude as our formal modeling framework and tool. Specifically, we have shown that our technique can: (i) find the original amplification attack on SIP, (ii) verify the effectiveness of the SIP+5393 patch against it, (iii) find a new amplification attack on SIP+5393 in the presence of a malicious proxy, and (iv) verify the effectiveness of a new tit-for-tat defense mechanism against this insider attack.

We view our new DoS analysis technique as complementary to the statistical model checking approach in [2,4]. Indeed, both are based on a rewriting logic model of a protocol. It may in fact be useful to combine both types of analysis on a network protocol model. For example, statistical model checking can be used to explore in greater depth the impact of DoS attacks and defenses on performance measures such as latency. Furthermore, statistical model checking is easily parallelizable, and is therefore more scalable, so that it can be used to search for a wider range of attack scenarios than those that can be feasibly explored with standard model checking techniques.

Note, that the technique presented in this work is specific for analyzing amplification attacks and similar attacks characterizable by cost measure comparisons. However, it is not a general method to analyze all DoS attacks possible. For instance, the attacker might simply send a large number of packets that take up all available output buffers within a proxy. The attacker could launch reflection attacks by spoofing the source IP address of the intended victim to a large number of proxies (thereby causing the proxies to reply back to the victim in large numbers). There are also DoS attacks possible by either spoofing a connection termination messages or by inserting spurious via fields. See [11] for a discussion on some of these attacks in the VoIP protocol.

There are several directions in which this work can be extended. Our plan is to use SIP and VoIP as a testing ground for new extensions of our techniques. One interesting possibility is to develop new techniques to formally characterize other kinds of DoS attacks such as *reflection attacks* and *smurf attacks* and verify them on SIP, which has been known to be vulnerable to such attacks [11]. Moreover, it would be interesting to evaluate the effectiveness and practicality of the intruder model assumed in our analysis by deploying (perhaps a modified version of) the SIP protocol on an appropriate test-bed using some open-source, standards-compliant implementation of the protocol, such as sipX (`http://www.sipfoundry.org/sipX`).

Acknowledgements. This work was supported in part by NSF CNS 07-16626, NSF CNS 07-16421, NSF CNS 05-24695, ONR N00014-08-1-0248, NSF CNS 05-24516, NSF CNS 05-24695, DHS 2006-CS-001-000001, NSF CNS 07-16638, and grants from the MacArthur Foundation and Boeing Corporation. The authors would also like to thank the anonymous reviewers for their suggestions. The views expressed are those of the authors only.

References

1. Abadi, M., Blanchet, B., Fournet, C.: Just fast keying in the pi calculus. In: Schmidt, D. (ed.) ESOP 2004. LNCS, vol. 2986, pp. 340–354. Springer, Heidelberg (2004)
2. Agha, G., Gunter, C.A., Greenwald, M., Khanna, S., Meseguer, J., Sen, K., Thati, P.: Formal modeling and analysis of DoS using probabilistic rewrite theories. In: International Workshop on Foundations of Computer Security, FCS 2005 (2005)
3. Agha, G., Meseguer, J., Sen, K.: PMaude: Rewrite-based specification language for probabilistic object systems. Electronic Notes in Theoretical Computer Science 153(2), 213–239 (2006)
4. AlTurki, M., Meseguer, J., Gunter, C.A.: Probabilistic modeling and analysis of DoS protection for the ASV protocol. Electron. Notes Theor. Comput. Sci. 234, 3–18 (2009)
5. Chadha, R., Gunter, C.A., Meseguer, J., Shankesi, R., Viswanathan, M.: Modular preservation of safety properties by cookie-based DoS-protection wrappers. In: Formal Methods for Open Object-Based Distributed Systems, pp. 39–58 (2008)
6. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.: All About Maude - A High-Performance Logical Framework: How to Specify, Program, and Verify Systems in Rewriting Logic. LNCS. Springer, Heidelberg (2007)
7. Denker, G., Meseguer, J., Talcott, C.L.: Protocol specification and analysis in Maude. In: Proc. of Workshop on Formal Methods and Security Protocols (1998)
8. Durgin, N., Lincoln, P., Mitchell, J., Scedrov, A.: Multiset rewriting and the complexity of bounded security protocols. J. Comput. Secur. 12(2), 247–311 (2004)
9. Escobar, S., Meadows, C., Meseguer, J.: A rewriting-based inference system for the NRL protocol analyzer and its meta-logical properties. Theor. Comput. Sci. 367(1), 162–202 (2006)
10. Goodloe, A.E.: A Foundation for Tunnel-Complex Protocols. PhD thesis, University of Pennsylvania (2008)

11. Gupta, P., Shmatikov, V.: Security analysis of voice-over-ip protocols. In: 20th IEEE Computer Security Foundations Symposium, Venice, Italy, pp. 49–63. IEEE Computer Society Press, Los Alamitos (2007)
12. IETF. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), Updated by RFCs 3265, 3853, 4320, 4916, 5393 (June 2002)
13. IETF. Addressing an Amplification Vulnerability in Forking Proxies draft-ietf-sip-fork-loop-fix-00. Internet-Draft (February 2006)
14. IETF. Addressing an Amplification Vulnerability in Session Initiation Protocol (SIP) Forking Proxies. RFC 5393 (Proposed Standard) (December 2008)
15. Kim, M.-Y., Stehr, M.-O., Talcott, C., Dutt, N., Venkatasubramanian, N.: A probabilistic formal analysis approach to cross layer optimization in distributed embedded systems. In: Bonsangue, M.M., Johnsen, E.B. (eds.) FMOODS 2007. LNCS, vol. 4468, pp. 285–300. Springer, Heidelberg (2007)
16. Lafrance, S., Mullins, J.: An information flow method to detect denial of service vulnerabilities. J. UCS 9(11), 1350–1369 (2003)
17. Mahimkar, A., Shmatikov, V.: Game-based analysis of denial-of-service prevention protocols. In: IEEE Computer Security Foundations Workshop (CSFW-18 2005). IEEE Computer Society Press, Los Alamitos (2005)
18. Meadows, C.: A formal framework and evaluation method for network denial of service. In: CSFW, pp. 4–13 (1999)
19. Meseguer, J.: Conditional rewriting logic as a unified model of concurrency. Theor. Comput. Sci. 96(1), 73–155 (1992)
20. Meseguer, J.: Rewriting logic and maude: a wide-spectrum semantic framework for object-based distributed systems. In: Smith, S.F., Talcott, C.L. (eds.) FMOODS. IFIP Conference Proceedings, vol. 177, pp. 89–117. Kluwer, Dordrecht (2000)
21. Sen, K., Viswanathan, M., Agha, G.A.: On Statistical Model Checking of Stochastic Systems. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 266–280. Springer, Heidelberg (2005)
22. Wang, X., Zhang, R., Yang, X., Jiang, X., Wijesekera, D.: Voice pharming attack and the trust of VoIP. In: SecureComm 2008: Proceedings of the 4th international conference on Security and privacy in communication netowrks, pp. 1–11. ACM Press, New York (2008)
23. Younes, H.L.S., Simmons, R.G.: Statistical probabilistic model checking with a focus on time-bounded properties. Inf. Comput. 204(9), 1368–1409 (2006)
24. Yu, C.-F., Gligor, V.D.: A specification and verification method for preventing denial of service. IEEE Trans. Softw. Eng. 16(6), 581–592 (1990)