D(e|i)aling with VoIP: Robust Prevention of DIAL Attacks

Alexandros Kapravelos, Iasonas Polakis,
Elias Athanasopoulos, Sotiris Ioannidis, and Evangelos P. Markatos
email: {kapravel, polakis, elathan, sotiris, markatos}@ics.forth.gr

Institute of Computer Science,
Foundation for Research and Technology Hellas, Greece

Abstract. We carry out attacks using Internet services that aim to keep telephone devices busy, hindering legitimate callers from gaining access. We use the term *DIAL (Digitally Initiated Abuse of teLephones)*, or, in the simple form, *Dial attack*, to refer to this behavior. We develop a simulation environment for modeling a Dial attack in order to quantify its full potential and measure the effect of attack parameters. Based on the simulation's results we perform the attack in the real-world. By using a Voice over IP (VoIP) provider as the attack medium, we manage to hold an existing landline device busy for 85% of the attack duration by issuing only 3 calls per second and, thus, render the device unusable. The attack has zero financial cost, requires negligible computational resources and cannot be traced back to the attacker. Furthermore, the nature of the attack is such that anyone can launch a Dial attack towards any telephone device.

Our investigation of existing countermeasures in VoIP providers shows that they follow an *all-or-nothing* approach, but most importantly, that their anomaly detection systems react slowly against our attacks, as we managed to issue tens of thousands of calls before getting spotted. To cope with this, we propose a flexible anomaly detection system for VoIP calls, which promotes fairness for callers. With our system in place it is hard for an adversary to keep the device busy for more than 5% of the duration of the attack.

1 Introduction

The Internet is a complicated distributed system that interconnects different kinds of devices and interfaces with other types of networks. Traditionally, computer security deals with attacks that are launched from Internet hosts and target other Internet hosts. However, the penetration of Internet services in everyday life enables threats originating from Internet hosts and targeting non Internet infrastructures. Such a non Internet infrastructure is the telephony network, a vital commodity.

The ever increasing number of households that adopt Voice over IP technology as their primary telephony system, demonstrates our shifting towards a digitally interconnected community. According to estimations, IP communication subscribers will reach more than 1.8 billion worldwide by 2013 [2]. While this new technology coexists with the old technology, new methods for their interaction emerge. Today, an Internet user can place calls to anywhere in the world reaching anyone that has a telephone device and take advantage of all characteristics inherent in such digital technologies, thus introducing new threats against traditional telephony systems.

In this paper, we explore the feasibility of an attack using Internet services and targeting regular landline or cellular phones. We seek to characterize the parameter values that will make the attack effective and also the means to mitigate it. Our key contributions are the following:

Dial Attack. We develop an empirical simulation in order to explore the potential effectiveness of Dial attacks. Through the simulated environment we identify and quantify all of the attack's fundamental properties. Using experimental evaluation with existing telephone lines, we demonstrate that an *attacker manages to render an ordinary landline device unusable, holding it busy for 85% of the attack period by issuing only 3 calls per second* . The attack requires no financial resources, negligible computational resources and cannot be traced back to the attacker.

Defenses. We seek to reveal existing countermeasures through reverse engineering of real-world VoIP providers. Our findings suggest that current schemes are not efficient since they follow an *all-or-nothing* approach. We develop and analyze a server-side anomaly detection system for VoIP traffic, which significantly reduces the attack impact. With our defense system deployed, the attacker *can no longer hold the line busy for more than 5% of the attack period.*

This paper is organized as follows. We analyze our motivations in Section 2. In Section 3 we present a threat model and a potential attack in a simulated environment. We carry out the attack against a real landline device in Section 4. In Section 5 we present existing countermeasures and introduce our anomaly detection system, together with experiments that show how effectively our system mitigates the attack. Finally, we review prior work in Section 6 and conclude in Section 7.

2 Motivation

In this section we present the basic motivations that drove us to explore this area and led to the creation of this paper. We explore our motivation in terms of *goal* and *attack platform*.

Goal. The traditional communication through the telephone network has become an important commodity. Take into account, that over 300 billion domestic calls to landlines were served inside the US alone in 2005 according to the FCC[5]. Our argument is that *access to a telephone device is vital for humans*.

Considering the importance of the service, an adversary may target a telephone device in order to harm a user. Prohibiting users from accessing certain services has been done in the recent past. For example, a significant part of computer viruses disrupt Internet connectivity. The impact of such an attack can be enormous, either life threatening (targeting a fire-fighting station during a physical disaster), or financial (targeting a business, like pizza delivery, or hindering a bank to authenticate a transfer request [6]) or simple disturb someone.

Our research is composed by two complementary goals. The first goal is to find out if it is possible to render a telephone device unusable. We want to achieve this goal with no financial resources, negligible computational resources and without being traceable back to the attacker. The second goal is to design and build technologies for protecting users from attacks that target telephones. We want to achieve this goal with minimal deployment effort, minimal user interference and by using existing well-known technologies.

Attack platform. In order to achieve our goals we use VoIP providers as attack platforms against telephone devices. Our choice was driven by various reasons. First, we wanted an attack platform, which is affordable and easy to access. There are hundreds of free VoIP providers, which permit users to access any landline device with no cost at all or mobile devices with a minimal cost. Second, we wanted to be able to completely automate the attack and have enough flexibility in fine tuning the call placement. Most VoIP providers support the SIP protocol[19] which met our expectations. Third, we wanted to perform the attack anonymously. The very nature of VoIP technology allows a caller to hide his true identity. And finally, we wanted to launch the attack from a PC. The fact that our attack platform is already provided by the industry and anyone can use it to launch the attack motivates us highly to explore the area as a precaution from future exploitation of VoIP services.

One can argue, that parts of the attack described in this paper are well-known or can be carried out, manually, by performing an excessive amount of dialing. As far as the novelty is concerned, to the best of our knowledge, this paper is the first one to perform and evaluate a real automated attack *directly* targeting a telephone device. As far as manual dialing is concerned, we already enlisted the four reasons, which drove us to select VoIP as the attack platform. These four reasons, reveal characteristics of a platform far superior to humans performing manual dialing.

3 Attack Overview

In this section we present the fundamental properties of the attack we developed. We start by describing our attack in detail; we specify the threat model, the adversary's overall goal and list all the assumptions we have made. We develop a simulated environment in order to carry out the attack virtually. Based on our findings in this section, we proceed and develop the actual attack prototype in the next section.

3.1 Attack Description

The goal of the attacker is to render a telephone device unusable with zero financial cost. This can be achieved by injecting a significant number of *missed calls* towards a victim telephone device. A call is considered missed, if it is hanged up prior to the other end answering it. By placing the calls correctly in the network, the attacker can keep the target continuously busy and, thus, prevent other users from accessing the telephone device. Even though many VoIP providers allow calls to landlines free of charge, we designed our attack in a way to be able to attack cell phones even if such calls are not free. By hanging-up the placed calls on time, the adversary manages to launch the attack cost free. Even if the target telephone device is answered the attack does not degrade, but rather augments, taking into account that the resource is still in busy state.

The attack in principle consists of a resource R, an attack medium M and calling modules. The resource represents a telephone device and has two states; it can either be available or busy. The attack medium simulates the behavior of a VoIP provider; it receives requests and queries the resource in order to acquire it. A calling module places such requests to the attack medium at a configurable rate.

The proposed attack is based on some important assumptions. First, we assume that M is unreliable, meaning that communication messages

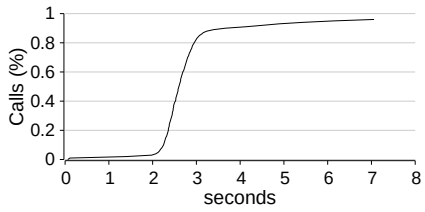

Fig. 1. Cumulative distribution of routing latency times for call placement.

may be lost, dropped or delayed. However, we assume that all faults in M are stabilized in the long run. Thus, we do not implement message faults for M in the simulated environment. Second, we assume that R does not support direct querying, or at least it supports it partially. There is no way to directly retrieve all states of R. However, it is possible to implement detection by analyzing parts of the communication messages. Third, we assume that, when no attack is taking place, the calling rate follows a Poisson distribution ($\lambda = 10$) [10]. When an attack is taking place, the calling rate significantly varies from the Poisson distribution. Finally, we make no assumptions about the routing latency for call placement, i.e. the time it takes for a request to reach R through M, or the release time of R after call termination. Instead, we perform real experiments to collect representative approximations of these quantities (see next section).

3.2 Simulation

Before starting experimenting with real calls in the wild, we performed a series of controlled experiments in a simulated environment. Based on the attack description we just presented, we developed a multi-threaded simulator where we instantiated a virtual calling module with an *aggressive behavior* to represent the attacker and one with a *non-aggressive behavior* to represent a legitimate caller. Then, we modeled the resource using a data structure that allows the module to obtain exclusive access with a certain probability. Each calling module behaves similar to a VoIP caller, *i.e.* it attempts to connect to the resource R through an attack medium M. In principle, M simulates a VoIP provider and R a telephone device.

In order to simulate the virtual calls in a realistic fashion we collected empirical values of durations from real call placement and hang-up operations. We issued 7,300 calls through a real VoIP provider over the time period of one week (see Figure 1). In this way, we collected representative

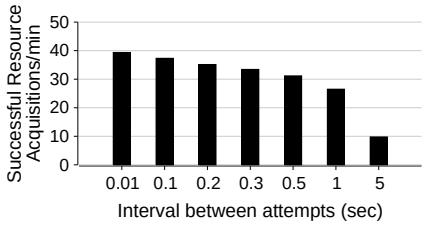

Fig. 2. Rate of successful resource acquisitions managed by an aggressive calling module in simulation environment.

routing latencies of call placements at various times and days of a typical week. The simulator maintains a pool with the 7,300 routing latencies and uses one, randomly, each time a virtual call attempt takes place. Unfortunately, we could not follow a similar approach for the hang-up operation, since it is hard to detect representative values for hang-up times of a real VoIP provider. However, we used the following approach, which we consider quite realistic. We injected pairs of call placements and hang-up operations in a real VoIP provider. We initially started injecting the pairs back-to-back. The result was that one of the two calls always reached the telephone device when it was in the busy state. In other words, the VoIP provider could not complete the hang-up operation of the first arrived call, before the second arrived. We started increasing the gap between the call pair, until we could measure that both calls had reached the telephone device in available state. We managed to successfully issue over 1,000 such call pairs with this property. The gap times ranged from 1 to 2 seconds. We consider this time window a realistic window for a hang-up operation. Thus, we modeled the virtual hang-up operation accordingly. Each virtual call hang-up operation takes from 1 to 2 seconds to restore R's state back to available.

Based on the above configuration we issued four 1-hour simulation runs, each one having an aggressive calling module placing virtual calls, with different intervals. We used intervals ranging from 0.01 to 5 seconds. Concurrently a legitimate module tried to acquire R following a Poisson distribution with $\lambda = 10$.

We examine the results of our experiments in terms of the aggressive calling module's success in acquiring resource R, the virtual call status distribution of the aggressive calling module and how many times the legitimate module succeeded in acquiring resource R. The rate of successful R acquisitions an aggressive thread managed to issue is presented in Figure 2. The best we could achieve was more than 39 acquisitions per

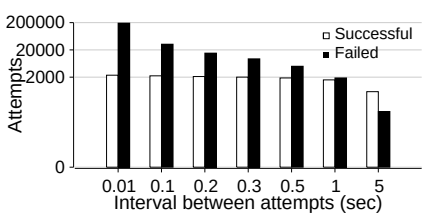

Fig. 3. Distribution of all acquire attempts issued by an aggressive calling module in simulation environment.

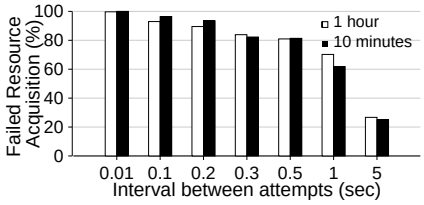

Fig. 4. Percentage of failed resource acquisitions for the legitimate module which models a legitimate caller in simulation environment.

minute. In the real-world, this result translates into more than 39 ringing calls per minute; a severe attack rate that would render the telephone device unusable. In Figure 3 we examine the call status distribution of all virtual call placements the aggressive calling module managed to issue. Observe that as the interval reduces, the amount of failures in acquiring R increases rapidly. Practically, there is no benefit in reducing the interval below 0.3 seconds.

The performance of the legitimate module is depicted in Figure 4. We also plot the results for the first 10 minutes of the experiment's duration in this case. First, observe that the legitimate module fails to acquire R for almost 85% of the simulation duration at the interval of 0.3 seconds, while the aggressive module issued less than 20,000 attempts. By reducing the intervals down to 0.01 seconds, the legitimate module is completely prevented from acquiring R, with the downside of requiring almost 200,000 more issued attempts to achieve just 15% more failed resource acquisitions compared to the interval of 0.3 seconds. Second, we can see that the first 10 minutes approximate the result of the full duration (1 hour) of the simulation, with tolerable error (from below 1% to 1.5%) in most cases. An exception occurs only in the case of the 1 second interval, where the difference is about 8%.

Our simulation experiments confirm our intuition for a potential threat against telephone devices. In addition to this, they highlight that only 3 calls per second are needed to render a device unusable and that the error between 1 hour and 10 minutes long experiments is tolerable.

4 Attack Evaluation

Based on the simulated studies we carried out in the previous section, we present an attack prototype. Our aim is to reach the performance we

achieved in the simulated environment, using an existing system which tries to acquire an actual telephone device.

4.1 Attack Prototype

Our attack prototype implementation uses VoipDiscount [7] as an attack medium, which uses the Session Initiation Protocol (SIP) [19] for remote communication. As SIP is the most common used protocol among VoIP service providers, our prototype implementation is not limited to VoipDiscount but could be applied using any different provider.

We implemented caller modules, which communicate with M, in our case the VoIP provider, using the SIP protocol and exchange invite and termination messages. We used the Python programming language and the `pjsip`[1] library which provides an implementation of the SIP protocol. We developed two types of callers: (a) an attacker caller and (b) a legitimate caller. The attacker caller places calls one after the other, trying to keep the telephone device busy continuously. The legitimate caller places calls following the Poisson distribution ($\lambda = 10$).

Recall, from section 3, that we assumed that the resource does not support querying, or it supports partial querying. Indeed, the telephone device does not support querying and thus there is no easy way to track down the status of the device, i.e. if it is in ringing or busy state. Although, SIP supports querying the status of a placed call, many providers do not implement this feature. The one we used is among them. Specifically, we can retrieve that the line is busy, using a SIP operation, but we can not retrieve a ringing status. To overcome this issue we implemented a detector module, based on a Fast Fourier Transformation of the incoming audio signal. This way we analyze the frequency of the audio signal and detect a ringing tone when we observe signals at 420 Hz. Notice that this approach successfully recognizes the ringing tone, since the tone has a constant frequency. Having immediate access to the ringing status is vital for the attack, since we want to achieve the attack with zero financial resources. We want to keep the telephone device busy by injecting short time lived calls (i.e. missed calls). For the generation of a missed call, the call has to be terminated immediately after the first ringing tone.

4.2 Real World Experiments

We conducted several real world experiments over the period of eight months using a landline device located in our lab as a victim. For the

[1] PJSIP, `http://www.pjsip.org/`.

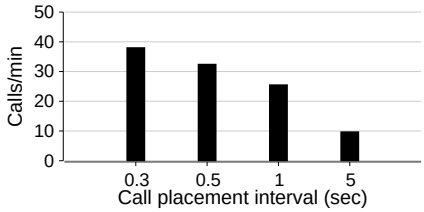

Fig. 5. Rate of ringing calls managed by an adversary.

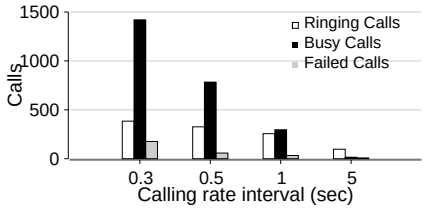

Fig. 6. Distribution of all calls issued by an adversary.

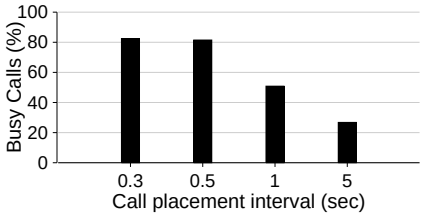

Fig. 7. Percentage of busy calls received by a legitimate caller, while the target telephone device was under attack.

presentation of this section we issued a series of runs over the duration of 1 week. This subset of runs is consistent with our overall experimental results. For each configuration we issued 6 runs each with a 10 minute duration with intervals ranging from 0.3 to 5 seconds. Recall from section 3 that the first ten minutes of each simulation run approximate the result of the full duration (1 hour) of the simulation, with tolerable error (from below 1% to 1.5%) in most cases. Thus we face the following trade-off: conducting more short-lived or less long-lived real world experiments. We chose the first approach, so as to be flexible enough to conduct a larger experimental base.

As was the case with the simulation, we are interested in three measurements: (i) the call rate of the attacker, (ii) the call status distribution of the attacker, and (iii) the probability for a legitimate user to acquire the resource, while an attack is taking place.

We present the call rate achieved by the attacker in Figure 5. Observe, that the results are highly consistent with the simulated ones (see Fig. 2). The adversary has managed to issue almost 40 ringing calls/minute for a calling placement interval of 0.3 seconds.

We present the distribution of all call attempts by the attacker in Figure 6. Again, the results are highly consistent with the simulated ones

(see Fig. 3). Note, that as the call placement interval reduces, the fraction of busy calls increases, having a negative impact on the attack. Another side-effect of shorter call placement intervals is failed calls. A call is considered failed, when no ringing or busy status is identified after 10 seconds from call placement.

Finally, in Figure 7 we present the percentage of busy calls received by a legitimate caller, while the target telephone device was under attack. Observe, that the adversary managed to hold the target landline device busy for 85% of the attack duration, preventing access, for most of the time, to the legitimate caller.

4.3 Attack Impact

In our real world experiments, we managed to hold an existing landline busy for 85% of the attack duration, by issuing only 3 calls per second. By aggressively issuing calls, an attacker targeting the telephone centers of critical infrastructures such as police or fire-fighting stations and hospitals can completely disrupt their operation and create life threatening situations. By issuing dozens or hundreds of calls an attacker can hinder legitimate users from accessing the call centers of critical services. Taking into account that these services are vital to our society, *any* threat against them must be seriously considered, and mechanisms for protection should be designed and employed.

4.4 Attacker's Anonymity

An important aspect of *Dial attacks* is that they cannot be traced back to the attacker. This is achieved by having two layers of anonymity. The first is provided by the part of traditional telephony network, where only the VoIP provider's Caller ID is revealed. In our experiments, the Caller ID was *Unknown*, requiring the use of law enforcement in order to find the source of the calls. The second layer is the communication with the VoIP provider. The only way that the VoIP provider can track the attacker is by her IP address. In order to remain anonymous, the call requests must be placed from a safe IP address, *e.g.* through an anonymization proxy or with the use of a botnet.

5 Countermeasures

In this section we investigate existing countermeasures currently employed by VoIP providers. We present a study about Skype, a leading

provider of VoIP services, VoipUser[8] and VoipDiscount[7], two representative VoIP providers. Based on our analysis, we propose and implement an anomaly detection system that promotes fairness to callers and is able to successfully mitigate the attack outlined in this paper.

5.1 Existing Countermeasures

Skype. Skype is a popular VoIP provider with more than 400 million user accounts and capable of serving 300,000 simultaneous calls without any service degradation [3, 1]. Skype internally uses an anomaly detection system, whose technical details are not publicly available. In order to reverse engineer part of its logic, we used four different user accounts and three different landline devices. We performed experiments with very aggressive call initialization rates against our landlines. Eventually, all four accounts were blocked permanently and all three victim landline devices were permanently banned from the system. This means, that the victim landlines were further inaccessible by *any* Skype user. We refer to this policy as *all-or-nothing*, meaning that the anomaly system either permits full access or no access at all to the service.

In Figure 8 we present the cumulative time of the call history of each blocked Skype account. Our initial intuition was that Skype blocks our account when we pass a specific call-rate threshold. However, each Skype account got blocked when it exceeded a totally different threshold, indicating non-deterministic detection based on heuristics or human inspection of call logs. With the first account we placed more than *one hundred thousand* calls before the anomaly detection system spotted us. The other accounts were blocked by making a large number of calls in a very short time period. This is shown in Figure 8 by an almost vertical increase of at least fifteen thousand calls.

In addition, Skype also blocked the landline telephone numbers which we used as victims. In Figure 9 we can see the call history of these numbers. The graphs terminate at the time the blocking actually happened. The Skype service permitted us to place more than 130,000 calls to the first line we used, before blocking it. The rest of the telephone lines we used were blocked as a result of more aggressive experiments.

We consider, that the *all-or-nothing* policy of Skype's anomaly detection is highly inefficient and, most importantly, enables further abuse. We proved that the slow reaction of the anomaly detection system allowed us to issue tens of thousands of calls. This would be catastrophic for any service that is based on telephone communication. We believe that the slow reaction is a fundamental result of the *all-or-nothing* approach. The

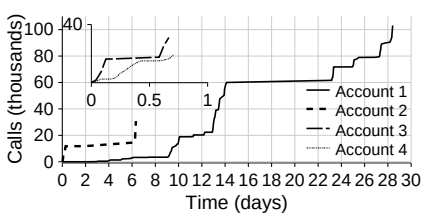

Fig. 8. Call history of each Skype account, until it is blocked.

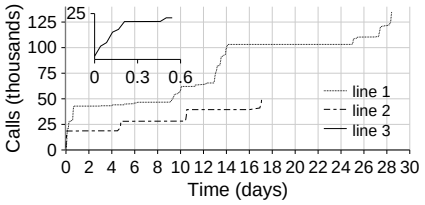

Fig. 9. Call history of each telephone line targeted through Skype, until it is blocked.

penalty is so high (i.e. permanent block), that the anomaly detection system is triggered only during occasions where there is severe abuse. An adversary, could still carry out the attack in a more stealthy fashion. We also showed that an adversary can intentionally block certain devices from Skype. All she needs is to issue a vast amount of missed calls towards the victim device for it to be completely banned from the system.

Voipdiscount. During our experiments with the Voipdiscount provider we have not observed any countermeasures. We have used their infrastructure for multiple experiments, issuing hundreds of thousands calls for over 8 months without being warned or banned.

Voipuser. We speculate that Voipuser relies on manual inspection which is not effective and cannot provide adequate defense against such attacks. After a series of initial experiments we conducted, they blocked the accounts used, as well as all other accounts we had created; note that these accounts had not been used in the attack experiments. Account bans based on the correlation of the domain of the email addresses we used for the account registrations suggest a manual process of log inspection. However, our accounts were banned after the experiments had ended, proving the inability of manual countermeasures for the early detection of such attacks.

5.2 Server Side Countermeasures

Our system is based on a detection module and a policy enforcement module. We decided to implement the detection module entirely in software, using the well-known Intrusion Detection System (IDS), Snort[18]. As far as the policy enforcement is concerned, we have two options. We can either implement it in software or in hardware. For the first case, we can use the built-in firewall functionality of Linux operating systems, `iptables`. However, this gives us poor flexibility in complex policies. On

Policy	INVITE behavior	Implementation
soft-mute	Drop	`iptables`
hard-mute	Drop	Click Router
hard-shape	Fixed rate	Click Router

Table 1. Policies supported by our anomaly detection system.

the contrary, the hardware solution gives as a range of functionalities employed by modern router devices. In order to easily perform an evaluation of various policies, we chose to use the Click router[16], which is a rich framework for testing router configurations. The Click router incorporates a wide range of elements for traffic shaping, dropping decisions and active queue management, which can also be found in most modern routers.

Detection Module. Snort is responsible for the detection. It handles user requests by monitoring all incoming traffic and flags flows that belong to hosts that initiate a large number of calls in a short amount of time. We further refer to this threshold as *abt (abuse behavior threshold)*, which is expressed in *invite*[2] requests per second per host. We have implemented, a Snort-rule similar to those for *port-scans* for detecting hosts that exceed *abt*. Whenever we have a Snort alert, the policy enforcement module is invoked, in order to mitigate the suspicious behavior.

Policy Enforcement Module. Policies are enforced over specific time windows. We refer to this quantity as *pew (policy enforcement window)*. Each policy applies an action to a host, that has been flagged suspicious by the detection module. We have implemented two different types of actions: mute and shape. The *mute action* drops all invitation messages and the *shape action* imposes a fixed rate of message delivery in a fashion that approximates a legitimate behavior.

We implemented the *mute* action using `iptables` in Linux, by using Snort's plugin SnortSam [4]. We provide a hypothetical hardware implementation of both *mute* and *shape* actions using the emulation environment provided by Click. In Table 1 we summarize the policies we support, along with their notation.

Evaluation In order to evaluate our anomaly detection system we performed our attack once again, but this time, both the attacker and legitimate caller were forced to pass their requests through our system. This was done at the network level, by rerouting all communication messages through a gateway that acts as an anomaly detection system.

[2] An *invite* request in SIP is associated with a call placement.

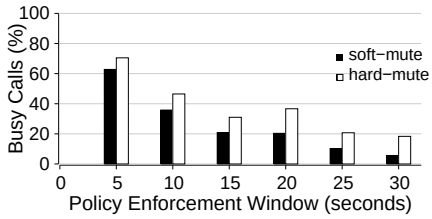

Fig. 10. Attack mitigation for soft-mute and hard-mute policies for various *pew*.

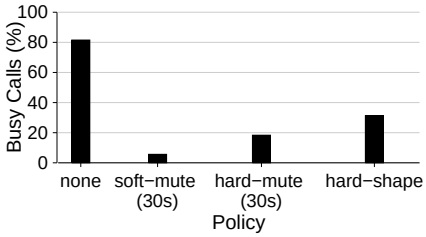

Fig. 11. A comparison of all policies along with the original attack.

In order to eliminate false positives we decided to use a more tolerating *abt* value, equal to 10 invitation messages per 30 seconds ($abt = 10msg/30secs$). Notice, that although this decision leaves us with no false positives (indeed, we have measured zero false positives in all experiments), relaxing *abt* is negative for the mitigation result. The attacker can become more aggressive and still remain under *abt*.

In Figure 10 we depict the effects on the attack's firepower when our policies are enabled. Each policy is applied for a time duration equal to *pew*. Notice, we do not provide results for hard-shape values for any *pew*, since the hard-shape policy is enforced for the whole attack duration. This is not explicitly forced by our detector, but it stems from the fact that the attacker does not adapt to the policy, and the *pew* is always extended.

In Figure 11 we provide a comparison of all policies along with the original attack. For each policy we state the *pew* used inside parenthesis. Observe that the attack's firepower can be reduced to 5% using *soft-mute* or up to 30% using a more relaxed policy, *hard-shape*. We consider the *shape* policy more relaxed than the *mute*, since the suspicious host is not muted and thus the policy is more tolerable in enforcing restraints on false positives.

5.3 Client Side Countermeasures

Telephone devices currently have no means of defense against a Dial attack. Our solution is based on *phone CAPTCHAs*. A CAPTCHA [22] challenge aims to distinguish between human initiated and automated actions. The core component of our platform is the Asterisk PBX, an open-source software implementation of a private branch exchange (PBX). It supports Interactive Voice Response (IVR) technology, can detect touch tones and respond with pre-recorded messages or dynamically created

sound files. To handle landlines, the host machine is equipped with specialized hardware that connects it to the telephony circuit.

When an incoming call is received, Asterisk places the call in a call queue. The caller then receives a phone CAPTCHA and has a limited time to respond by using the phone's dial pad. The phone CAPTCHA test requires the caller to spell a word randomly selected from a large pool[3]. If the caller provides the correct answer, Asterisk forwards the call to its destination. Otherwise the call is dropped. The use of words greatly increases the difficulty of phone CAPTCHAs being broken by speech recognition software, compared to traditional audio CAPTCHAs that only contain digits. We intend to further explore the use of CAPTCHAs as a client side countermeasure in future work.

6 Related Work

In this work we use VoIP technology as an attack medium. Given its low access cost and its wide deployment, VoIP services have attracted a lot of attention [15]. For example, extensive research has been recently conducted on VoIP security. Wang *et al.* exploit the anonymity of VoIP calls by uniquely watermarking the encrypted VoIP flow [23]. Wright *et al.* investigate whether it is possible to determine the language of an encrypted VoIP conversation by observing the length of encrypted VoIP packets [24]. Zhang *et al.* in [25] exploit the reliability and trustworthiness of the billing of VoIP systems that use SIP [19]. Spam over Internet Telephony has also gained significant attention [11, 17]. In this paper we explore new ways for abusing VoIP services as well as identifying possible defenses to this abuse.

Research for attacks to the telephony network has been carried out in the past, mostly targeting cellular networks. For example, it has been shown that a rate of only 165 SMS messages per second is capable of clogging both text and voice traffic across GSM networks in all of Manhattan [20, 21]. Countermeasures to alleviate this problem are based on using weighted queues before traffic reaches the air interface, and/or more strict provisioning and partitioning resources after traffic leaves this bottleneck [12, 13]. Enck *et al.* demonstrate the ability to deny voice service by just using a cable modem [12]. They claim that with the use of a medium-sized zombie network one could target the entire United States.

[3] Words have been widely used in the USA to help people memorize telephone numbers by "translating" numbers into letters.

Their work also included suggestions on how to counter SMS-based attacks. Specifically, they call for the separation of voice and data channels, increased resource provisioning, and rate limits of the on-air interfaces.

Last but not least, there are concerns in the research community about attacks that threaten the operation of emergency services. Aschenbruck *et al.* report that it is possible to peer VoIP calls to public service answering points (PSAP) [9]. This peering can have grave implications because it makes it possible to carry out DoS attacks against emergency call centers. In their work they monitored calls from a real PSAP of a fire department which serves about one million people. During emergencies the PSAP received approximately 1100 calls per 15 minutes. These calls overloaded the PSAP and the authors suggested that the high call-rate was the result of citizens constantly redialing until they got service. In their follow-up publication Fuchs *et al.* show that under heavy load at the same PSAP, up to half of the incoming calls were dropped [14].

7 Conclusion

In this paper we perform an extensive exploration of Dial attacks. Initiated by our theoretical findings, we implement a prototype and carry out the attack in the wild proving that an adversary can keep a telephone device in busy state for 85% of the attack duration by issuing only 3 calls per second. Our attack requires zero financial resources, negligible computational resources and cannot be traced back to the attacker. Considering the severity of such a threat, we explore already employed countermeasures and conclude that current VoIP infrastructures employ countermeasures based on an *all-or-nothing* approach, react slowly to possible abuse or offer no protection at all. As these defenses appear inefficient we propose an anomaly detection system for VoIP calls and demonstrate that it can mitigate Dial attacks and prevent an adversary from holding a telephone device busy for more than 5%.

Acknowledgments. We would like to thank the anonymous reviewers for their feedback and Professor Giovanni Vigna for his insightful comments. Alexandros Kapravelos, Iasonas Polakis, Elias Athanasopoulos, Sotiris Ioannidis and Evangelos P. Markatos are also with the University of Crete. Elias Athanasopoulos is funded by the Microsoft Research PhD Scholarship project, which is provided by Microsoft Research Cambridge. This work was supported in part from the FP7 project SysSec, funded by the European Commission under Grant agreement no: 257007 and by the Marie Curie Actions - Reintegration Grants project PASS.

References

1. Ebay Inc. FQ 2008 results. http://investor.ebay.com/results.cfm.
2. IDC Predicts more than 1.8 billion Worldwide Personal IP Communications Subscribers by 2013. http://www.idc.com/getdoc.jsp?containerId=219742.
3. Skype Fast Facts, Q4 2008. http://ebayinkblog.com/wp-content/uploads/2009/01/skype-fast-facts-q4-08.pdf.
4. Snortsam. http://www.snortsam.net.
5. Statistics of Communications Common Carriers 2005/2006 Edition. http://hraunfoss.fcc.gov/edocs_public/attachmatch/DOC-282813A1.pdf.
6. Thieves Flood Victims Phone With Calls to Loot Bank Accounts. http://www.wired.com/threatlevel/2010/05/telephony-dos/.
7. Voipdiscount. http://www.voipdiscount.com.
8. Voipuser.org. http://www.voipuser.org.
9. N. Aschenbruck, M. Frank, P. Martini, J. Tolle, R. Legat, and H. Richmann. Present and Future Challenges Concerning DoS-attacks against PSAPs in VoIP Networks. *Proceedings of International Workshop on Information Assurance*, 2006.
10. L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao. Statistical analysis of a telephone call center. *Journal of the American Statistical Association*, 100(469):36–50, 2005.
11. S. Dritsas, Y. Soupionis, M. Theoharidou, Y. Mallios, and D. Gritzalis. SPIT Identification Criteria Implementation: Effectiveness and Lessons Learned. In *Proceedings of The IFIP International Information Security Conference*. Springer, 2008.
12. W. Enck, P. Traynor, P. McDaniel, and T. L. Porta. Exploiting Open Functionality in SMS Capable Cellular Networks. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05), Alexandria, Virginia, USA*, 2005.
13. S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1993.
14. C. Fuchs, N. Aschenbruck, F. Leder, and P. Martini. Detecting VoIP based DoS attacks at the public safety answering point. *ASIACCS*, 2008.
15. A. D. Keromytis. A Look at VoIP Vulnerabilities. *USENIX ;login: Magazine*, 35(1), February 2010.
16. E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 2000.
17. B. Mathieu, Y. Gourhant, and Q. Loudier. SPIT mitigation by a network level Anti-SPIT entity. In *Proc. of the 3rd Annual VoIP Security Workshop*, 2006.
18. M. Roesch. Snort - lightweight intrusion detection for networks. In *LISA '99: Proceedings of the 13th USENIX conference on System administration*. USENIX Association, 1999.
19. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), 2002. Updated by RFCs 3265, 3853, 4320, 4916.
20. P. Traynor, W. Enck, P. McDaniel, and T. L. Porta. Mitigating Attacks on Open Functionality in SMS-Capable Cellular Networks. *12th annual international conference on Mobile computing and networking*, 2006.
21. P. Traynor, P. Mcdaniel, and T. L. Porta. On attack causality in internet-connected cellular networks. In *USENIX Security Symposium*, 2007.

22. L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. *CAPTCHA: Using Hard AI Problems for Security*. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003.
23. X. Wang, S. Chen, and S. Jajodia. Tracking anonymous peer-to-peer VoIP calls on the internet. In *CCS '05: Proceedings of the 12th ACM conference on Computer and Communications Security*, 2005.
24. C. V. Wright, L. Ballard, F. Monrose, and G. M. Masson. Language identification of encrypted VoIP traffic: Alejandra y Roberto or Alice and Bob? In *SS'07: Proceedings of the 16th USENIX Security Symposium*, Berkeley, CA, USA, 2007. USENIX Association.
25. R. Zhang, X. Wang, X. Yang, and X. Jiang. Billing attacks on SIP-based VoIP systems. In *WOOT '07: Proceedings of the first USENIX Workshop On Offensive Technologies*.