Efficient Directory Harvest Attacks and Countermeasures

Boldizsár Bencsáth and István Vajda
(Corresponding author: Boldizsár Bencsáth)

Laboratory of Cryptography and Systems Security (CrySyS)
Department of Telecommunications, Budapest University of Technology and Economics
BME Hiradastechnikai Tanszek 1521. Budapest Pf. 91., Hungary
(Email: boldi@crysys.hu)

(Received Dec. 5, 2005; revised and accepted Nov. 22, 2006)

Abstract

In this paper the E-mail Directory Harvest Attacks (DHA) are investigated. The goal of the DHA attacker is to identify valid e-mail addresses in a system, which addresses can be sold or used for spanming purposes. To achieve the goal the attacker tries numerous different addresses and selects valid addresses according to the response of the e-mail server. We elaborated a method for optimizing the wordlist size used by the attacker under limited resources. This optimization provides deeper insight into the capabilities of the DHA attacker, and yields firm ways upon which efficient protection can be developed. We analyzed the results and proved that our method is optimal. We present an efficient countermeasure against DHA. This is a network based method, where the possible attack events are collected by a trusted server (DHA RBL server). The DHA RBL server analyzes the data and builds up the list of attackers, which enables our prototype client module to filter out all emails coming from known attackers. The prototype implementation was examined in real-life systems, the results show that our approach is viable.

Keywords: Brute-force, dictionary attacks, directory harvest attacks, e-mail

1 Introduction

In the e-mail Directory Harvest Attack (DHA) the attacker's goal is to gain information about the e-mail addresses used in an internet domain. The collected e-mail addresses can be sold or used for sending spam. The attacker tries various e-mail addresses, and collects those addresses where the SMTP server of the attacked domain did not respond with an 'unknown user' error message. After a successful DHA the e-mail address gets inserted into a bulk e-mail address list and spam starts flooding the identified address.

The attack itself creates a high load on the attacked SMTP server. The high load can slow down the processing of the e-mail and degrades other services on the SMTP server, or the server can stop responding (DoS - Denial of Service). Therefore, we have to protect against DHA. We also have to protect the privacy of the users of the system and try to prevent unsolicited e-mails by disabling the possibility of collecting e-mail addresses.

The DHA attackers typically use some dictionary for the attack. Such a dictionary typically has a fixed size. The attack is launched against several systems with the same dictionary. One after the other words from the dictionary are tried on every targeted SMTP server, and the valid e-mail addresses are collected. The attackers often use multiple attacking computers, therefore it is a distributed attack. We show that this basic method of DHA is not optimal, and under resource constraint (when the number of trials is fixed) the attacker can increase the expected number of collected e-mail addresses by applying variable size of dictionary. We improve the attack by varying the size of the wordlist according to the expected number of users on the attacked SMTP server. We also prove that our method is optimal under certain constraints. The goal of such an optimization is to the develop a more efficient countermeasure. If we are able to protect against the worst attack, we surely can deploy protection against typical real-life attack.

We recommend a centralized protection method, where a realtime blacklisting server (DHA RBL server) gets information about every host that sends e-mails to unknown addresses. An e-mail SMTP server can query the DHA RBL server at the beginning of each incoming SMTP (e-mail) connection wether the DHA RBL server previously enlisted the sender as an attacker. If a sender is enlisted as a possible DHA attacker, the SMTP server can reject receiving e-mails from the sender. We present a prototype implementation of the proposed detection mechanism.

This paper is an extended version of our earlier conference presentation [3]. Moreover, the first version of our prototype system was used for DoS protection [2]. The present version contains more detailed descriptions about our prototype; furthermore, the mathematical and technical details were also improved with respect to the previous versions.

The structure of the paper is the following:

Related works are presented in Section 2. Section 3 contains the description of the DHA optimization problem, the definition of the algorithms, and the analysis. Here we prove the optimality of our attacking method. We also provide hints on the economical reasons for an attack. Our sample results for the effect of optimization is presented using simulation results.

Countermeasures against directory harvest attacks are described in Section 4. Our centralized protection system is is also presented in this section. Finally, Section 5 summarizes the paper.

2 Related Work

Dictionary attacks to crack passwords have been known for a long time. The oldest attacks were aimed at telnet and ftp accounts. Lately, attacks are deployed against SSH, POP3, RAS, Samba, and various HTTP based services.

These attacks show some similarity to DHA attacks, in the sense that the attacker's goal is to identify valid data and filter out the rest. Active countermeasure is possible for online systems: the invalid access / harvest trials can be detected and therefore the attackers can be rejected.

The typical countermeasure for brute force on-line password cracking is locking of the affected accounts, however this can result in a Denial of Service (DoS) attack against users of a particular system (see [12] for some details). The solution therefore can be extended e.g. by Captcha-based unlocking mechanism: The user cannot log into the locked account, but the account can be unlocked semi-automatically if the user proves that he is human and not an attacking program. A Captcha is an automated Turing test, a question that a human can answer, but an algorithm cannot. (e.g. recognition of characters in a picture [1])

Other possible protection is to insert some delay into the authentication process after a number of unsuccessful trials. Although this can deny the attack from a single host, it cannot solve the problem with a distributed attack. If the whole system slows down then a DoS attack is possible.

One possible protection against a DHA is via manipulation of SMTP error message (SMTP error 550 according to [11]), the message is distorted or withheld.

We think that this solution would be harmful, since this information is very important to the legitimate users of the internet. Some other solutions protect against DHA by filtering out hosts which send large volume of emails to non-existing (unknown) addresses. Unfortunately, this method does not protect the system from a highly distributed attack.

Many commercial solutions also provide countermeasures against harvest attacks. The Kerio MailServer [10] detects emails to unknown addresses and above a threshold the server begins to filter out possible attackers. This method can be inefficient against a distributed attack.

The Postini enterprise spam filtering manages e-mail services with DHA protection. Their white paper [14] gives details about the provided DHA protection method and about the DHA problem as well. Postini website also has important statistical data about current DHA activity they detected. No details about the principals of protection method are published.

The Secluda Inboxmaster offers customizable SMTP error message: If a spam is detected during the mail delivery (by other parts of the system), a bounce message is sent back to let the sender think that the address is not valid. The trial messages of DHA attackers often cannot be identified as spam, therefore the protection capabilities of this method is limited.

To the best of our knowledge, existing open source projects, like ProjectHoneypot [13] have not yet integrated protection against DHA. The ProjectHoneypot system tries to identify spammers by trap e-mail addresses. This can be extended by the identification of mass e-mail senders with many messages to unknown recipients. That was the initial idea described in this paper.

The producers of commercial products typically do not disclose detailed information about the protection method. We try to give a detailed description of the attack and our proposed protection.

3 Common Method of DHA and Our Proposed Enhancement

The main idea of the DHA attack is to test the validity of individual e-mail addresses. If an address is invalid, the target system (SMTP server) will respond with an error message thus enabling the attacker to identify the address as invalid.

During the SMTP protocol the client generally sends four basic protocol elements to the sender: a welcome message, the address of the sender, the address of the recipient, and the actual data of the e-mail (including header information). Each of the protocol steps is acknowledged by the server. After the client sends a recipient's address, and the address is not known by the server, the server generally responds to the client with an error message. In this way, an attacker can easily test the validity of e-mail addresses without even finishing all the protocol steps.

The goal of the attacker is to collect a list of valid e-mail addresses (also know as local parts) in a system. The attacker therefore tries lot of different possible local parts and according to the answers of the server, he selects those addresses which are not responded by an error message from the SMTP server. The attacker cannot do too much with a single collected e-mail address, but a list of thousands or millions of addresses can be very valuable for him. The collected email addresses can be sold for various purposes, mainly for delivering unsolicited e-mails. Usually an attacker attacks multiple systems and tries a huge number of possible e-mail address local parts.

In case of a distributed attack the attacker has control over multiple computers. With the usage of multiple computers the attacker can carry out a wider scale attack, attacking more SMTP systems and deploying more trials against a single SMTP server. In addition, the ability of the attacked systems to identify attackers can also be affected: Sometimes it is nearly impossible to distinguish between an attacker and a legitimate user mistyping an email address. If multiple attacking computers attack a single SMTP server, then a single attacker tries only a few addresses (5-100), which makes it more difficult to detect the attack. To protect the SMTP server it is not enough to filter out a single attacker, we have to filter out all attackers. Obviously, fighting against a distributed attack is considerably harder than fighting against a single, well-identifiable attacking computer.

The directory harvest attack can be categorized into two main categories:

- The attacker tries all possible valid character combinations with a limited number of characters. This can be enhanced such that only wordlike strings are used.
- The attacker uses a wordlist (or dictionary) of possible (frequent) user names (e-mail address local-parts). The wordlist is typically based on dictionary or generated using collected list of e-mail addresses.

The typical attacker cannot achieve a successful attack with ten-millions of e-mail trials to a single host, therefore we only analyze the problem of the more efficient wordlist based attacks.

3.1 Description of Common Attack Method

The actions of a DHA attacker can be described by the following steps:

1) The attacker selects a number of destination domains. The selection can be based on public information available about the domains. (E.g. number of expected users on the system)
2) The attacker gains control over innocent victim computers and turns them into zombies, computers which are remotely controllable by the attacker. This attack can be done by installing and running malicious code on the victim computer (by a trojan program [6] or e-mail worm [5], etc.)
3) The attacker carries out the attack by controlling the zombies.
4) The attacker collects the results from the zombies and analyzes it.

A network attack with a similar method is presented in [7].

We experienced that most harvest attacks aim hosts with immediate SMTP error reporting (if a mail is coming to an unknown address). Some of our hosts simply accept all incoming e-mails and pass them to another (protected) host (e.g. a firewall). On these systems a DHA attack is still possible, but the attacker should use and maintain a valid return address to get notified about the unknown users. We found that these hosts are almost never attacked by DHA. Using log analysis we found that the size of the dictionary of the attackers is around $100,000$ different words. Some attackers do not use a static dictionary but try to build up artificially generated words from syllables (eg. 'kizu', 'pugeriu').

3.2 Reasons to Attack: The Economy of the DHA

The attacker might earn profit from selling e-mail addresses for spamming purposes.

The attacker tries to maximize the profit from the attack. The net profit of the attacker is calculated by the following formula $V = -(C_0 - f_c(t)) + I$ where $I = \sum_{j=1}^{t} p_s(w_j) * \mu$.

Here C_0 os the initial cost of the attack, $f_c(t)$ is the cost depending on the number of trials, furthermore I is the income decomposed into a sum, where $p_s(w_j)$ is the success probability for the word tried in the j-th step, while μ is the value of an identified address.

We do not know the exact cost function, therefore in this section our goal is to optimize (maximize) the value of the income (I).

Figure 1 illustrates the cost and income curves: The intersection of them corresponds to the maximal profit of the attackers.

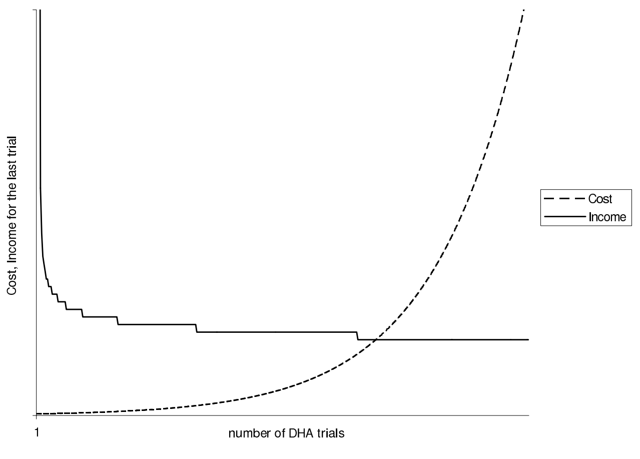

Figure 1: Cost and income functions of the attacker

3.3 The Algorithms

Let us introduce a few notations. An attacker controls zombie computers, $A = \{A_1, A_2, \cdots, A_{N_A}\}$, where N_A denotes the number of the zombies.

The target of the attack is the following set of domains: $D = \{D_1, D_2, \cdots, D_{N_D}\}$. Within domain j we have users $U^j = \{U_1^j, U_2^j, \cdots, U_{N_U^j}^j\}$. The dictionary (wordlist) of the known email address local parts (i.e. e-mail user names) is denoted by the sequence $W = W_1, W_2, \cdots, W_{N_W}$ therefore the size of the list is N_W.

For a geographic (cultural) region we can assume that the distribution of the local parts is very similar for every domain.

We assume that the probability distribution of the words $\{P(W_i)\}$ in every domain is known by the attacker. The attacker's dictionary is sorted according to the descending order of probabilities. $(P(W_i) \geq P(W_{i+1})$ for every $i)$

Using 10 million e-mail addresses collected from the internet (from public Web-sites, email traffic, downloadable address lists) we analyzed the distribution of local parts. Figure 2 shows the success probability for a DHA trial on systems with various user counts.

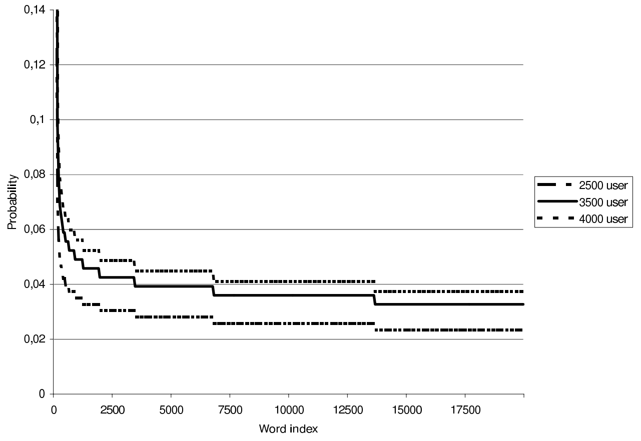

Figure 2: Probability of success for the wordlist elements in systems with different user numbers

The number of attack trials is limited by integer t. Now we describe the algorithms: Algorithm 1 is the experienced behavior of the DHA attackers. Algorithm 2 is our proposed algorithm.

Algorithm 1: The attacker tries the first $T = t/N_D$ items of the wordlist for each domain, which are the words with the highest probability.

Algorithm 2: For every trial the attacker determines the target domains and selects the domain where the probability of success is the greatest. The trial is carried out against the selected domain.

In pseudocode our proposed Algorithm 2 can be described as follows:

```
for all $d \leq N_D$ do
  $i[d] \Leftarrow 1$
end for
$trial \Leftarrow 1$
while $trial \leq t$ do
  $d \Leftarrow 1$, $p_{max} \Leftarrow 0$, $target \Leftarrow 0$
  while $d \leq N_D$ do
    if $p(W_{i[d]} \in U^d) > p_{max}$ then
      $p_{max} \Leftarrow p(W_{i[d]})$, $target \Leftarrow d$
    end if
  end while
  Try(word $W_{i[target]}$, on domain $target$)
  $i[target] \Leftarrow i[target] + 1$
  $trial \Leftarrow trial + 1$
end while
```

The wordlist elements are tried one after the other, $i[d]$ denotes the index of the next word in the wordlist for a given d domain.

In every step Algorithm 2 tries to find the index of the word where the probability of the success is maximal. This step is repeated for every trial until the limit of trials is reached.

3.4 Analysis of the Algorithms

Now we give formulae for the expected number of successfully found e-mail addresses.

Let random variable S_i denote the number of email addresses found by the attacker when he attacks domain D_i, $i = 1, \cdots, N_D$. Random variable S denotes the total number of successes, i.e. $S = \sum_{i=1}^{N_D} S_i$. Below we give an analysis for the expected value $E(S)$ for the Algorithm 1 and Algorithm 2 First consider Algorithm 1:

$$\begin{aligned} E(S_i) &= E\left[\sum_{j=1}^{T} \chi_{\{W_j \in U^i\}}\right] = \sum_{j=1}^{T} E\left(\chi_{\{W_j \in U^i\}}\right) \\ &= \sum_{j=1}^{T} N_U^i P(W_j) = N_U^i \sum_{j=1}^{T} P(W_j), \end{aligned}$$

where $T = t/N_D$ and χ_A is the indicator function for set A. Hence we get

$$E(S) = \sum_{i=1}^{N_D} N_U^i \sum_{j=1}^{T} P(W_j).$$

In case of Algorithm 2. let t_i denote the number of trials against domain D_i, $i = 1, \cdots, N_D$. Note that these numbers are deterministically determined by the probability distribution $P(W_j)$,$j = 1, 2 \cdots$, and by the sizes N_U^i, $i = 1, \cdots, N_D$, as it will be detailed below. Assuming the knowledge of t_i, $i = 1, \cdots, N_D$, for the expected number of email addresses found by the attacker applying Algorithm 2 we get:

$$E(S) = \sum_{i=1}^{N_D} N_U^i \sum_{j=1}^{t_i} P(W_j).$$

We introduce notations $h(t_k) = \sum_{i=1}^{t_k} P(W_i)$ and $n_i = N_U^i$. For the calculation of the number of trials, we have to maximize function H($\underline{t}$), where

$$H(\underline{t}) = E(S) = n_1 h(t_1) + \cdots . + n_k h(t_k),$$

under conditions $g(t) = t_1 + \cdots + t_k - t = 0,\ t_1 \geq 0, \cdots, t_k \geq 0$ where $\underline{t} = (t_1, \cdots, t_k),\ k = N_D$.

For solving this problem we use the Lagrange's multiplicator technique. We start from function

$$G(\underline{t}) = H(\underline{t}) + \lambda g(\underline{t}),$$

and we solve the following system of $k+1$ equations:

$$\begin{aligned} \frac{\partial G}{\partial t_i} &= n_i h'(t_i) + \lambda = 0,\ i = 1, \cdots, k \\ \frac{\partial G}{\partial \lambda} &= t_1 + \cdots + t_k - t = 0, \end{aligned}$$

where h' denotes the derivative of function h. Eliminating λ we arrive to the following system of k equation in unknowns t_i, $(i = 1, \cdots, N_D)$:

$$\begin{aligned} n_1 h'(t_1) - n_{i+1} h'(t_{i+1}) &= 0,\ i = 1, \cdots, k-1 \\ t_1 + \cdots + t_k - t &= 0. \end{aligned}$$

For instance, we show how to solve this system, when function h has the following form $h(x) = a/x^b$, $h'(x) = c/x^d$, $c = -ab$, $d = b + 1$. After some straightforward algebra we arrive to the following system of linear equations:

$$\begin{aligned} m_1 t_{i+1} - m_{i+1} t_1 &= 0,\ i = 1, \cdots, k-1 \\ t_1 + \cdots + t_k - t &= 0, \end{aligned}$$

where $m_i = n_i^{1/d}$. Whence we get the following result

$$t_i = t \cdot \frac{n_i^{1/d}}{\sum_{i=1}^{N_D} n_i^{1/d}},\ i = 1, \cdots, N_D.$$

It is easy to check that $t_1 \geq t_2 \geq \cdots \geq t_{N_D}$ and $t_1 + t_2 + \cdots + t_{N_D} = t$.

Theorem 1. *Algorithm 2 is optimal in the sense that the expected number of successfully identified e-mail addresses by the attacker is maximal.*

Proof. Assume that there exists a certain set of trials (X) that is better than the set (Y) produced by Algorithm 2. (both sets have the same size t) Therefore there should exist an element (word, domain pair) $x \in X$ for which $x \notin Y$. Since Algorithm 2 selects elements with highest possible success probabilities, therefore element x must have success probability less than or equal to the probability for any element from Y. Let's substitute the element x in the set X by an element y, $y \notin X$ and $y \in Y$. Let the modified set be denoted with X^*. The sum of success probabilities for X^* compared to that X will be higher or it remains the same. As we have shown above this sum of probabilities is actually the expected number of successfully identified e-mail addresses. In case when the summed probabilities over X^* is greater than that of over X we arrive to a contradiction. (X is not better than Y) In the other case when the two sums of probabilities are equal, we repeat the step of substitution. Exhausting all possible substitutions we either arrive to a contradiction or we'll see that the result of Algorithm 2 is not inferior to the optimal set X. □

3.5 Simulation

To support our results and to show the gain of our algorithm we carried out simulations. Two possible scenarios with different number of target domains and number of users were investigated. In the first scenario we set the number of users in 11 target systems as follows:

$$\{10, 20, 30, 40, 50, 80, 100, 200, 300, 400, 500\}.$$

In the second scenario we used a higher number of users on each of the six targeted systems:

$$\{1000, 2000, 3000, 4000, 5000, 8000\}.$$

We tried to use real-life data during the simulations. The user names (local parts) were chosen from the previously collected real-life e-mail address distribution. Moreover the simulated attacker's dictionary was collected from the log files of our systems attacked by real-life DHA attackers.

Both simulation scenarios were completed more than hundred times, the main results are summarized in Table 1.

Table 1: Simulation results

N_D	Total user	succ. Alg. 1. (std. dev.)	succ. Alg 2. (std. dev.)
11	1730	87 (20.6)	180 (17.1)
6	23000	5395 (146.7)	6754 (107.6)

The heading "Total user" denotes the total number of users in D, furthermore "succ. Alg i (std. dev.)" denotes the average of successfully identified addresses by Alg. i. (and the standard deviation) The simulation clearly shows the superiority of Algorithm 2.

4 Our Proposed Solution Against Directory Harvest Attacks

The goal of protection mechanisms in general is to detect the possible DHA attackers and to enforce effective countermeasures against them. These countermeasures can include the filtering of the traffic from the attackers.

We can separate the possible countermeasures, detection algorithms into two categories:

- Host based protection: An autonomous system has its own protection method, without relying on other parties.
- Network based protection: The system is cooperating with other parties to protect itself from the DHA. This method can be centralized: a server coordinates the protection.

4.1 Limitations of Host Based Methods

In order to identify valid usernames, a DHA attacker uses information collected about unknown users in a given domain. Therefore, some protection methods try to achieve protection against DHA by falsifying or denying information about unknown users. As we mentioned before, these protection methods rise new problems for the legitimate human users, therefore any protection method based on this behavior should be avoided.

Another solution is to filter based on error reports: When a computer is sending e-mail for an unknown user, we insert the address of the computer to a list and filter out all requests coming from computers on the list. To deal with the problem of false positives we only insert computers into the list if the number of e-mails to unknown recipients exceeds a threshold. The problem with such host-based filtering algorithm is that it is not resistant against distributed attacks. Experience shows that DHA attacks are highly distributed, the maximum of trials coming from an IP address can be as low as 1-2. Although the filtering algorithm can filter out attacking hosts, the attacker can utilize thousands of hosts which are not yet known by the target. From a global aspect, the attacking computers should be detected an all targeted SMTP server and should be filtered out by every individual filter routine to stop the attack.

The simple host based filtering can be extended:

- We can examine the address field of attacking e-mails. Some attackers sort the wordlist in alphabetical order. Similar suspicious properties can serve a starting point of a protection method.
- The attackers generally use the most frequent e-mail addresses during the attack. This list can be reconstructed by observing the trials of the attackers. An enhancement to the basic protection method can filter out all the IP addresses trying to send emails to unknown users where the particular username tried exists in this wordlist.

Although these enhancements could be successful for a short time period, the attackers can also adopt to their protection and the trials will not be statistically distinguishable from legitimate e-mails. Even if this protection would be successful to protect a single host, this approach will not help others to efficiently fight against DHA.

The filtering approach can be real-time or log-based. If the protection is based on log analysis and is not real-time, then then the attacker has large time windows to carry out an attack enabling him to test lots of addresses from a single attacking host.

The behavior of the server to the offending computers (computers on the list) can also be different: Some protection methods deny any IP traffic coming from those computers, while other servers only reject receiving e-mails from the attackers. It is also possible that the server only reports a temporary error (like in greylisting [8]) to the attacker thus the attacker won't know if he is filtered or the delivery should be retried later.

4.2 Proposed Network Based Protection Method

Our proposed solution is also based on filtering but with a centralized, network based approach. Parties of the protection are the following: the attacked server host and the centralized DHA RBL (Real-time blacklist) server. (check [9] for general information about anti-spam techniques like RBL)

If an attacker sends an email to an unknown address on the attacked server, the attacked server will send an error report to the central DHA RBL server. The error report contains the offending IP address, the recipient address tried, and the time of the attack. The centralized server collects the reports from the protected servers. If the number of trials (reports) exceeds a limit, the server inserts the address of the attacker into the blacklist. The server also stores the timestamp when the last e-mail was observed from the given attacker with an unknown recipient.

As usual, the RBL list can be queried by sending an address as a request to the server questioning if an address exists in the list. The server does not publish addresses on the list, instead it simply answers with yes or no.

4.3 Aging

For every blacklist-based method it is very important to decide how the addresses are cleared from the list. After removing the address from the blacklist the attacker can continue the attacks from that address, but if an address is remained too long on the list, then legitimate traffic might be blocked for a long time.

There are several methods for clearing the blacklist:

Administrator-driven aging: The administrator of the RBL server manually decides about the removal of the address. The owner of the attacking zombie computer can also ask the RBL administrator to remove the computer from the list. (e.g. the backdoor used to carry out the attack is removed from the computer)

Simple aging: After a given amount of time elapsed from the insertion of the address into the RBL the address is removed automatically. The problem with simple aging is that an attacker can easily estimate when the address is cleared from the RBL, therefore he can immediately restart attacking using that address.

Multi-phase aging: After the address of the attacker inserted into the list, we can expect that no more error report will arrive to the server corresponding to the given attacker. (Every server instantly filters out the traffic coming from the attacker) During multi-phase aging we define some protected hosts (automatically, manually, or randomly), which computers will not filter out traffic from the attacker, but report any attacking traffic. If the attacker renews or continues the attacks, the server may have fresh information about the attacker and so it can remain on the list for a longer time.

In our prototype implementation (described later) we use the combination of simple and administrator-driven aging methods.

4.4 Prototype Implementation

In order to test our proposed method and to resolve the issue of DHA attacks to our networks, we developed a prototype implementation. Our prototype application was planned as an extension of our previous network security efforts [2].

Our anti-DoS system presented in [2] contains a front-end component for SMTP servers to measure traffic levels and eliminate DoS attacks by statistical analysis. Our enhancements in this component enable us to use the proposed filter mechanism: The front-end can lookup the other party's IP address and check if it belongs to a possible attacker. If an attacker is found, the front-end drops the connection with an STMP (temporary) error. For collecting the data about the possible attackers, we developed new components. The role of these components is to analyze the SMTP traffic and report any possible attacker. This is done by real-time analysis of the log file generated by the SMTP server software (MTA). The structure of our prototype system is shown in Figure 3.

It was necessary to design the transport methods for our components. For administrative and informative tasks we simply built a dynamic web-page. For transferring information about attackers to third-parties (eg. ISP's) we selected XML format and the e-mail system as transport method. For transporting the information about possible attackers and queries about IP addresses we selected a scalable, fast, robust, and firewall-friendly transport method: DNS queries.

For filtering SMTP traffic, DNS queries are very popular: RBL servers get query information encoded into a DNS query and they return filtering information that looks like a regular DNS answer. (eg. the answer is encoded as an IP address. 127.0.0.1 means the host is not on the list, 127.0.0.2 means that the host mentioned in the query is an attacker. The actual code representation depends on the provider.) The DNS system supports clustering (i.e. multiple DNS servers can be created for

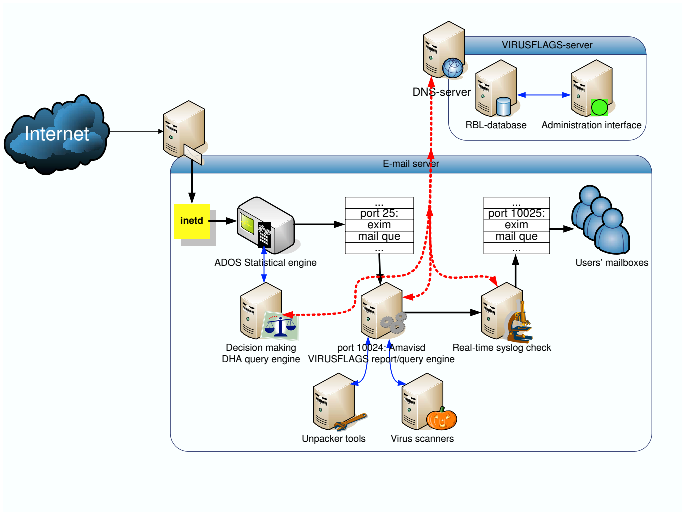

Figure 3: The structure of our prototype system

answering the queries). The DNS system is very transparent to the firewalls, because almost every firewall and filtering system lets the DNS traffic go through, and it can be easily limited by access rules. The DNS queries are very fast, as they mainly use simple UDP packets for transporting the information. The results of the query can be cached by a local DNS server: For a given amount of time, the re-check of an IP address can be faster and therefore the system can be more efficient during communication with sites with a high network traffic. The caching mechanism depends mainly on the settings of the authorative DNS server, i.e. the RBL server. If the RBL server tells the immediate DNS server that a given data is not cacheable (eg. a traffic report), then the local DNS server will repeat the query every time.

The procedure of incident reporting is presented in Figure 4.

Step 1. The attacker sends an e-mail to an internet mail server (MTA).

Step 2. The SMTP server answers with valid information: the user is unknown in the system.

Step 3. The SMTP server sends an incident report to the server. This is done by a DNS query with a special format. The queried DNS name contains the information about the offending host.

Step 3b. The DNS server of the MTA forwards the query to another DNS server or directly to the DHA RBL server. The RBL server decodes the query and processes it.

The filtering mechanism in our prototype system is very simple: After a low number of e-mails sent to unknown addresses (currently set to 10) we insert the offending address into the RBL list.

Figure 5 shows the procedure when a enlisted attacker tries to send a new mail to an arbitrary protected host.

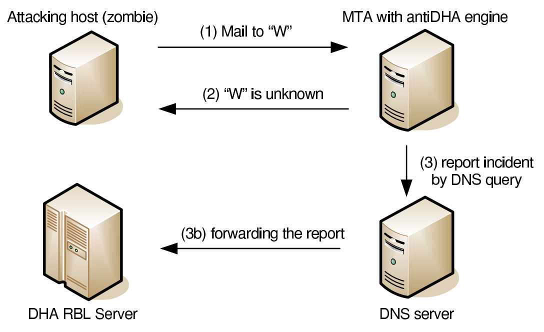

Figure 4: Reporting an incident to the antiDHA system

Step 1. The attacker tries to send an e-mail to an internet mail server (MTA).

Step 2. The SMTP server sends a DNS query with the address of the client (the attacker) embedded in the query.

Step 3. The DNS server of the MTA forwards the query to the DNA RBL server.

Step 4. The RBL server answers the query with a special form of IP address, meaning "Yes, the computer is in the RBL list". The DNS server can cache the answer for a given address, the caching time (TTL- time to live) can be controlled by the RBL server.

Step 5. The DNS server sends back the answer to the SMTP server (protected host).

Step 6. The SMTP server denies the connection with the attacker. This can be done at TCP level, or the attacker can be denied with a proper SMTP error code and explanation.

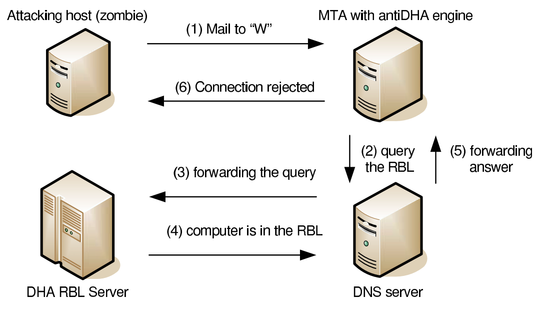

Figure 5: Filtering attacker using the data from the DHA RBL

The actual report from the SMTP server consists of the unix timestamp of the report, the reporter's user id, the suspicious ip address, and a keyed hash of the previously mentioned data elements whereas the key is a manually set constant key. The report format also can contain information (e.g. first characters) about the email address tried by the attacker.

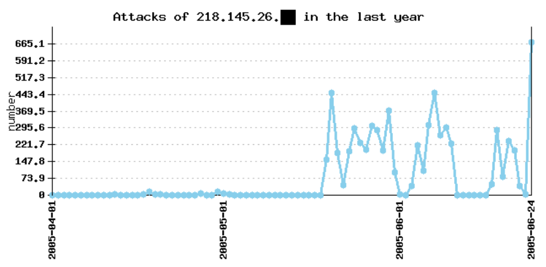

Figure 6: Example graphical statistics for attacking host

The data collected from the reporters is not only usable to setting up a filter list. The results are stored in a database and the behavior of attackers can be examined. Figure 6 shows the graphical representation of the attacks of a single host. We implemented numerous statistical elements into our management system to be able to drill down into statistical data and get actual information about attacks. Our system also makes it possible to send attack definition to the ISP of the attacker host based on standard format (idmef - intrusion detection message exchange format).

The results of centralized filtering is that the attacker can carry out only a very limited number trials against the protected domains. Of course, our method does not limit the attacks carried out against unprotected domains. The effect of the protection therefore modifies the expected income of the attacker. According to our experience, our model should be refined according to false positives: The system correctly detects spamming sources, but sometimes users do not want to filter out the relay server of big ISP's. To handle this situation we can use a whitelist or combine the filtering mechanism with other filtering methods.

4.5 Results of the Prototype Implementation

Using the prototype implementation we carried out some investigations in February, 2006 about the success of our proposed method. The short analysis of the log files on a typical day on two of our system is the following: On one of our systems (system A) the number of emails with unknown recipients were 4317 from 59 distinct hosts, for 5 days 27578 emails from 528 hosts. On system B the number of attacking emails on a typical day were 6421 from 2530 hosts, 47149 messages from 13820 distinct IPs in 5 days, the most attacking 5 hosts sent about 13000 messages in that five days. Only 6 hosts attacked both systems simultaneously if we only investigate the one-day data, 57 hosts are attacking both hosts in the 5 day interval. Interestingly, the attack on system A is almost disappearing during the night, while system B is under a constant attack.

The results show that on System A a host-based protection can be used to filter out the attackers (as number of attackers is very low, and the attacking emails coming from each host is high), but a network-based filtering is necessary to handle the attack of the thousands of hosts attacking System B.

Figure 7 shows the effect of the filtering in system A. When the filtering is turned on the number of incoming emails with unknown recipients sharply declined. The main reason of this good result is that the number of attackers was very limited.

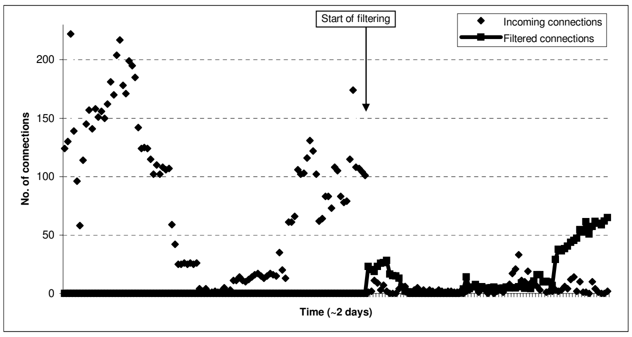

Figure 7: System A. Success of filtering

Figure 8 shows the typical variation of the emails on system B. After enabling the filtering mechanism, Figure 9 shows the change in the number of emails with unknown recipients and the number of filtered SMTP connections trials by our prototype implementation simultaneously. The number of filtered connections is very low, the reason behind this is that the attack was very distributed, a large number of hosts took place in the attack and most of them only sent a low number of emails. After some hours, the number of attacking emails lowered sharply. A possible interpretation of this effect is that the attacker measures the quality of the attack, and due to the filtration one of the attackers decided to stop the attack. Using higher number of hosts to collect data about attackers could enhance the protection.

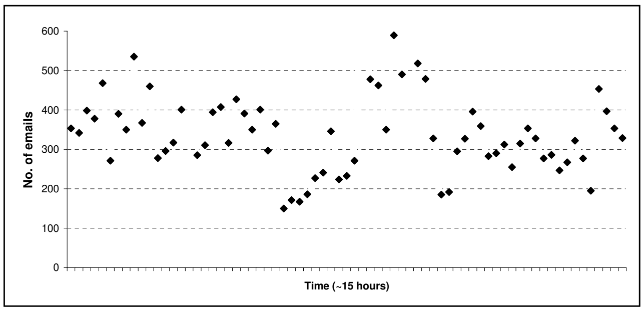

Figure 8: System B. Number of emails with unknown recipients without filtering

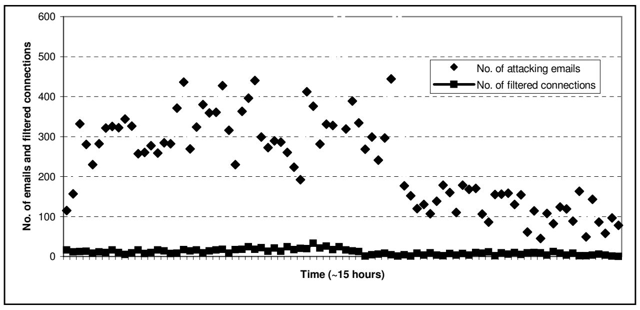

Figure 9: System B. Success of filtering

5 Conclusion

In this paper we analyzed the e-mail Directory Harvest Attack. We have shown that the generic DHA attack (found in-the-wild) can be optimized. An optimal attacking algorithm based on wordlist statistics has been presented. We derived formulae for the expected number of successfully attacked e-mail addresses for the presented algorithms. Our simulation results based on real data supported the feasibility and efficiency of our proposed algorithm. The description of the attack methods helped us to better understand DHA attacks and clearly see the limits of the attack and the protection against it.

We investigated the possible countermeasures in detail. We argued that to protect against a distributed DHA attack we should introduce a network-based protection. A possible solution should include methods and algorithms for the detection, filtering, list cleanup (aging), data transportation, etc. Our proposed centralized, blacklist based DHA protection system is implemented on the level of a prototype. The results collected from real-life data of the prototype system were also examined.

In the near future we plan to deploy our prototype system into real-life business environment. Meanwhile we try to expand the functionality of the individual system components as these components can be used to fight against a wide range of network attacks, e.g. DoS attacks and e-mail viruses and worms. Our system also helps us to collect statistical data about attackers and attack methods.

Acknowledgements

The work presented in this paper has been partially supported by the Hungarian Scientific Research Fund under project number T046664, furthermore the work is supported by DESEREC, a project of the Commission of the European Communities, contract number 026600. The authors are thankful to Levente Buttyán for his constructive comments on this work, and to Géza Szabó for his implementation efforts.

References

[1] L. V. Ahn, M. Blum, N. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Proceedings of Eurocrypt*, pp. 294-311, 2003.

[2] B. Bencsath, and I. Vajda, "Protection against DDoS attacks based on traffic level measurements," in *Proceedings of the 2004 International Symposium on Collaborative Technologies and Systems Simulation Series*, pp. 22-28, 2004.

[3] B. Bencsath, I. Vajda, "Efficient directory harvest attacks," in *Proceedings of the 2005 International Symposium on Collaborative Technologies and Systems*, pp. 62- 68, IEEE Computer Society, 2005.

[4] A. Costello, *Punycode: A Bootstring Encoding of Unicode for Internationalized Domain Names in Applications (IDNA)*, RFC 3492, Mar. 2003.

[5] eWEEK.com, *Special Report: Worm Attacks.* (http://www.eweek.com/category2/01874147646800.asp)

[6] GFI, *How to Protect your Network Against Trojans.* (http://www.gfi.com/whitepapers/network-protection-against-trojans.pdf)

[7] S. Gibson, *The Strange Tale of the Denial Service Attack Against grc.com.* (http://www.grc.com/files/grcdos.pdf)

[8] E. Harris, *The Next Step in the Spam Control War: Greylisting.* (http://greylisting.org/articles/whitepaper.shtml)

[9] S. Hird, "Technical Solutions for Controlling Spam," in *Proceedings of AUUG2002*, 2002.

[10] *Kerio MailServer - State-of-the-art Secure Email Server.* (http://www.kerio.com/kms_home.html)

[11] J. Klensin, *Simple Mail Transfer Protocol*, RFC 2821, Apr. 2001.

[12] C. Limited and S. d. Vries, *Corsaire White Paper: Application Denial of Service Attacks.* (http://www.corsaire.com/white-papers/040405-application-level-dos-attacks.pdf)

[13] *Project Honeypot - Distributed System for Identifying Spammers.* (http://www.projecthoneypot.org)

[14] *Postini Enterprise Spam Filtering, The Silent Killer: How Spammers are Stealing Your Email Directory.* (http://www.postini.com/whitepapers/)

István Vajda is a Professor at the Department of Telecommunications, Budapest University of Technology and Economics (BME). He is the Head of the Laboratory of Cryptography and Systems Security (CrySyS). His research interests are in Cryptography and Coding Theory. He has teaching experience in Algebraic Coding Theory, Cryptography, and Information Theory.

Boldizsár Bencsáth received the MSc degree in Computer Science at the Budapest University of Technology and Economics (BUTE). He also earned the MSc degree in economics at the Budapest University of Economics. From 1999 he is member of the Laboratory of Cryptography and Systems Security (CrySyS Lab). His work is focused on the technical aspects of internet security including DoS attacks, viruses, fine-grained access control, and secure smart card applications. Besides being the member of the CrySyS Lab, he also leads an SME that provides services in the field of network security, firewalls, security assessment.