Towards malware inspired management frameworks

Jérôme François, Radu State, Olivier Festor

Towards malware inspired management frameworks

Jérôme François, Radu State and Olivier Festor
MADYNES - INRIA Nancy-Grand Est, CNRS, Nancy-Université
LORIA - Campus Scientifique - BP 239 - 54506 Vandoeuvre-Les-Nancy Cedex France
{jerome.francois,radu.state,olivier.festor}@loria.fr

***Abstract*—Scalability is a real challenge for network management due to the increase of the devices to be managed and their various locations. A potential solution is based on botnets basically used by attackers. To prove the efficiency of such a system, a model is needed. Since in a previous paper we propose an IRC botnet model, in this paper we introduce two kinds of P2P models, evaluate them and compare the three different models to determine a practicable solution for network management.**

I. Introduction

In the network management domain, most existing solutions are not well suited for today's Internet and its new services. Besides a huge number of various devices to be managed, the management domain has no clear boundaries and devices must be managed wherever they are by bypassing network equipments like firewalls or addresses translators. Furthermore, the management system shouln't be compromised for doing malicious malicious activities. Nowadays, a main Internet threat are the botnets which allow an attacker to control a huge number of computers. The supporting middleware communication used by botnets is highly adapted to deal with large scale networks and the firewalls protection. The scalability, the efficiency and the robustness of them is the reason why an evolution towards a malware inspired management could be efficient and scalable. We propose to use botnets for doing network management. To prove the scalability with thousands devices, a model is needed. Because there are several kinds of botnets, we will propose different models. Therefore we are able to evaluate each one and to compare them. Our objective is to describe model with realistic constraints and to simulate each one.

The paper is structured as follows: the main challenges and a case study are described in section 2. Moreover this section gives the different metrics to evaluate the models. The section 3 shows how a botnet can be used for network management. The fourth section details the two different possible P2P networks and the corresponding models. The section 4 evaluates them and compares the two P2P based solution and the IRC (Internet Relay Chat) [1] based solution described in our previous work [2]. The fifth section is about related work. Finally a conclusion and our future work are introduced.

II. Research challenges

Our work is motivated by a practical problem of managing a large scale distributed honeypot for detecting cyber-predators [3]. Specific keywords are announced by the honeypots on the file sharing network in order to attract the cyber criminals. The honeypots have to be configured and monitored in order to retrieve the potential cyber-predators list. In fact, our management plane aims to perform a mass configuration like in the Figure 1. Thus our simple scenario is composed of one management platform which sends a broadcast messages to all devices. In our vision, a large and voluntary users population installs the honeypot and therefore attackers are allowed to install it and to provide false results. The key features that we had to address are:

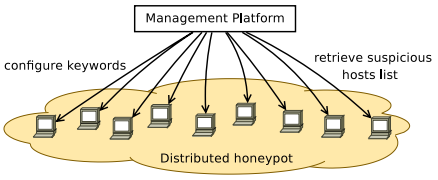


Fig. 1. The management plane

- scalability: in order to provide a good detection of cyber predators, many honeypots have to participate.
- efficiency: because the keywords are not content-linked, they can be changed quickly and thus the keywords list needs to be advertised quickly also.
- tracking prevention: an attacker should not identify other honeypots because he might disrupt them. Moreover, the manager who delivers the keywords list and retrieves the information of cyber-predators has to be hidden else it can be compromised to be used for criminal activities.
- security: an attacker should not be able to advertise another keywords list.
- reachability guarantees: an administrator needs to know the potential impact of sending a management request and above all the number of reached devices in order to plan complementary requests if necessary.

III. Malware based management approach

In our previous work [2], we showed that hackers dealt with the same challenges by using botnets. A botnet is a network of compromised computers and a controller (aka botmaster) sends requests to and gets replies from bots. A very huge botnet of 400 000 bots was already observed [4] and multiple defenses exist to prevent a botnet to be attacked.

Several communication channels can be used. In our previous work [2], an IRC [1] based model was introduced since

it is the most known botnet communication system. An IRC network is composed of few interconnected servers forming a spanning tree and each one forwards the requests to many connected clients. Many details about botnets are given in [5] and the second chapter introduces botnets based on other control mechanisms like P2P networks. Therefore, a malware based management plane can be inspired from P2P botnets. In fact the management request sent by the management platform is forwarded at each node which receives it. This architecture is totally decentralized, contrary to the IRC architecture which needs servers. However, there are several ways to organize a P2P network and to send a request to the maximum number of nodes *i.e.*, to send a broadcast message. Furthermore this kind of networks is able to easily preserve the anonymity of the botnet controller by forwarding the message through many nodes.

Two P2P networks are modeled and evaluated in this paper: Chord [6] and the Slapper communication mechanism [7]. Thus we don't focus on how the network is deployed. We evalute the networks once it is deployed. Basically it can be done in the same manner that agent are installed on different devices. Another possible way should be to propagate the software like a virus. However in this case, we need to use authentification to avoid to install attacker software. Thus a previous conifuguration is needed in all cases.

IV. Models

Some notations are shared by both models and the IRC model[2]. We denote by:

- m: the branching factor of a node is the maximum number of adjacent links,
- N: the total number of nodes in the network,
- β: the probability for a node to be compromised. The compromised node network communication can be monitored by the attacker. Therefore the other devices to be managed may be known and compromised too in the future. It is reasonable to consider that a compromised host cannot be used to inject commands (for probing the network) if encryption is used.
- $\alpha(m)$ is the availability factor. An available node is connected and is able to forward the request. Because maintaining a connexion needs ressources, more a node has to maintain connections, lower the availability is.

The models aim to determine the number of reachable devices regarding the previous mentionned parameters and the number of hops. An origin node sends a message and the reachability metric gives the average number of nodes at a given maximal distance *i.e.*, the number of hops (efficiency of a boradcast message).

A. Slapper model

Slapper [7] is a famous worm which appeared widely in 2002. The infected machines could be remotely controlled in order to perform an attack or to send Spam. In fact, the compromised computers did form a P2P network. Each peer knows all the other peers to build a full-meshed network. The attacker sends a message to a specific host with the maximal number of hops to reach the host. Thus the message is forwarded by random peers (chosen at each intermediary peer) to the destination. This mechanism prevents the tracking of the controller and a message might be forwarded by the same node more than one time. Each message has an identifier which is used to build a return path routing table.

Considering a mass configuration task, Slapper provides a useful broadcast technique named broadcast segmentation. For scalability and security reasons, the message is sent to two random peers in the network which will be in charge of choosing other two random peers and so on. One more time the manager cannot be tracked and a maximal number of hops needs to fixed. Therefore it is clear that there is no way to reach all nodes definitely. In the Figure 2, the nodes are organized in the same way that for the Chord model to show the difference easily. The branching factor is two and the origin node is the node 0. It sends the messages to two other nodes: 4 and 9. Each one of them will do the same thing independently. Moreover the non determinism property is illustrated by the figure showing another possibility for the second hop. Finally, we consider a general model where the number of chosen random peer is exactly fixed to m which can be different from two.

In a first step, considering t nodes which have already received the broadcast nodes among N. They have to forward the broadcast message and can send c duplicated messages. The probability $p(N,t,c,j)$ which is the probability to forward these messages to j nodes not yet contacted is:

$$p(N,t,c,j) = \frac{C_{N-t}^{j} \times \Gamma_{t+j}^{c-j}}{\Gamma_{N}^{c}} \quad (1)$$

where $\Gamma_n^k = C_{n+k-1}^k$ is the combination with repetitions of k items among n and C_n^k is the number of combination of k items among n without repetition.

In fact, the first term at the numerator is the number of possibilities to choose the j different nodes among the total number of N nodes except the t already contacted nodes. Considering that j duplicated messages are sent to these j nodes, the numerator is multiplied by the number of possibilities to choose the destination nodes of $c-j$ left messages among the total number of reached nodes *i.e.*, the previous contacted nodes and the j new contacted nodes. The result is then divided by the total number of possibilities to choose the destination nodes of the messages among all possible nodes in the network. This probability can be calculated only if $j \leq N - t$.

At the ith hop, the number of duplicated messages is m^i. However a node can be overloaded and so the number of new messages to be send is reduced to $msg = (m \times \alpha(m))^i$ because each node has the probability $\alpha(m)$ to be available. Although msg is also the maximum number of new reachable hosts, this one is limited by the number of already reached hosts at the previous hop. Thus we define:

$$max_{msg} = min(msg, N - reach(N, m, i-1))$$

The average number of new reached nodes is computed thanks to the formula (1) by considering all possible cases:

$$r(N,m,i) = \sum_{k=0}^{max_{msg}} p(N, reach(N,m,i-1), msg, k) \times k$$

$r(N,m,i)$ is the number of reached nodes at the exact distance i from the origin node and $reach(N,m,i)$ is the number of reached nodes at the maximal distance i. If $i = 0$, only the initial node is considered. Otherwise, the average number of reached devices is composed of the number of reached hosts at the previous hop and the new reached hosts. Moreover if one host is compromised, all the network is discovered. Thus the reachability is given by the following formula where $(1-\beta)^N$ is the probability for the network to not be discovered:

$$reach(N,m,i) = (1-\beta)^n \begin{cases} 1 & \text{if } i = 0 \\ reach(N,m,i-1) + r(N,m,i) & \text{else} \end{cases} \quad (2)$$

B. Chord Model

We expect to see an evolution in current malware towards a Chord based middleware since Chord has shown good performance in several studies [6]. Chord works as follows: an identifier is assigned to each participating node: $0 \leq id < N_{MAX}$ where $N_{MAX} = 2^k, k \in \mathbb{I}^{+*}$. N_{MAX} is the maximum number of participating nodes and the identifiers space size also. We consider a random and uniform distribution of nodes in the identifiers space by defining the average distance d between two consecutive node identifiers. It follows that $N_{MAX} = N \times d$. Contrary to a Slapper infected machine, a Chord node has a partial view of the network. It has a routing table with $log(N_{MAX})$ entries. Because the node distribution is uniform, only the first entries in the routing table could be the same. The first different entry i is different if $2^{(i-1)} > d$. Then, the number of different entries is:

$$log(N_{MAX}) - log(d) = log(\frac{N_{MAX}}{d}) = log(N) \quad (3)$$

This is the difference between the maximum number of entries and the first different entry plus the first entry of the routing table. Chord nodes are presented clockwise on a circle like in Figure 2. The i^{th} entry of the node p routing table is the first node having a distance to node p at least 2^{i-1}. [6] gives full details about the functioning of the chord lookup mechanism and how to maintain the routing tables. In our study we are interested to perform mass configuration over a Chord network. In [8], the authors introduce an efficient broadcast algorithm by avoiding that a node receives the message two times. Our approach is based on this algorithm. The initiator node sends the message to each node of the routing table with an exploration limit. This limit is the next different entry in the routing table because the corresponding

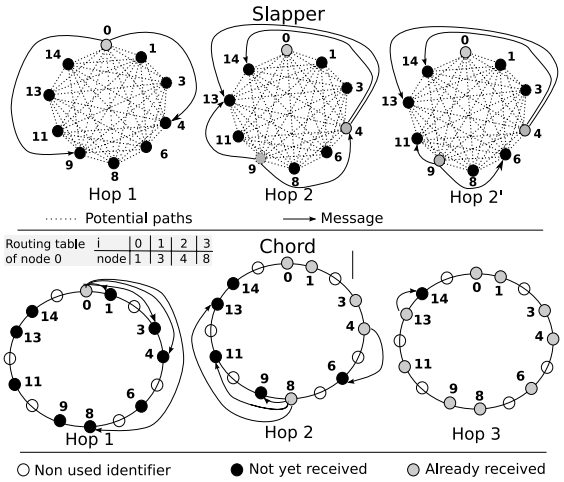


Fig. 2. The different broadcast algorithms

node will be in charge of broadcasting the message too. When a node receives a message, it is forwarded to each other peer of the routing table whose identifier is below the limit. Obviously a new limit is defined by the next node in the routing table or the current node itself. This mechanism is illustrated in Figure 2. The node 0 sends the message to all different entries of its routing table which are 1, 3, 4 and 8. For each of them, the node 0 gives a limit which is the next node *i.e.*, 3 for 1, 4 for 3, 8 for 4 and 0 for 8. This limit is an exclusive limit and that is why the node 4 forwards the message to the node 6 only since the limit is 8.

Each node has a limited routing table with $log(N_{MAX})$ entries. Thus when a node will forward the broadcast message, this value limits the number of new contacted nodes. So the branching factor m is fixed to $log(N_{MAX})$. Normally N_{MAX} is a power of two.

Because of the limited number of assumptions, the algorithm 1 allows to compute for each node the number of needed hops to reach it. Obviously, only participating nodes have to be considered i.e. the valid identifier are characterized by $\frac{id}{d}$ which has to be an integer. If the identifier of the node is a power of two then the node can be contacted directly in one hop. Else, if there is no node between the identifier and the previous power of two identifier, the node can be also contacted directly. Otherwise, there is at least one intermediary node. In this case the algorithm is reinitialized by setting the origin node to this intermediary node. We consider that the controller node is the node 0 but changing the controller node is like a renumbering due to the uniformity of the node distribution.

This algorithm is executed for each node. The reachability at the ith hop for a total number of nodes N is the total number of nodes having the distance from the origin node at i hops.

Finally, the results are affected by different kind of failures. In the worst case the first hop is affected by a failure and so another node with a lower identifier has to replace it. This one is able to contact the second original hop node because it is

Algorithm 1 Number of hops

$id = d \times k$ is the node identifier
$integer(x)$ return the integer part of x
$l = log(id)$
if $l \in Integer$ **then**
    return 1
else
    $total \leftarrow 0$
    $count \leftarrow 1$
    while $l \neq integer(l)$ **do**
        $total \leftarrow total + 2^{integer(l)}$
        if $id - total < d$ **then**
            return $count$
        else
            $l \leftarrow log(id - total)$
            $count \leftarrow count + 1$
        end if
    end while
    return $count$
end if

closer. By reiterating this process, for a node at a distance h, there are at most h failures. For each possible failure, an extra hop is necessary in the worst case. The probability $p(f \leq h)$ to have $f \leq h$ failures is defined by the binomial law. The corresponding probability is multiplied with the number of hops at the distance h and is added to the reachability value at the distance $h+f$. Thus algorithm 2 iterates over the different reachabilities from the lowest to the highest number of hops.

Algorithm 2 Node failure effect

h is the number of hops
n number of nodes at h hops
$total \leftarrow 0$
MAX_HOP is the maximum distance
$reach[x]$ is the average number of reached nodes at the distance x
for $i \leftarrow 1$ to MAX_HOP **do**
    for $i \leftarrow 1$ to h **do**
        $p \leftarrow n \times C_h^i \alpha(m)^{k-i}(1-\alpha(m))^i$
        $reach[h+i] \leftarrow reach[h+i] + p$
        $total = total + p$
    end for
end for
$reach[h] \leftarrow reach[h] - total$

Contrary to Slapper and the IRC model, protecting the network doesn't imply to preserve all nodes to be detected. In the worst case, an attacker can know all the network if he knows $n\_attacked = \dfrac{N}{log(N)}$ because each one knows $log(N)$ different nodes in average. Thus, discovering all the network is equivalent to take the control of at least $n\_attacked$ nodes. For each node, the probability to be compromised is β. Thus the probability to have the network entirely compromised can be calculated with a binomial law. It can be approximated by the Poisson law with $\lambda = N \times \beta$. Finally the network is detected entirely with the probability:

$$e^{-\lambda} \sum_{i=n\_attacked}^{N} \frac{\lambda^i}{i!}$$

This probability illustrates the robustness of our framework if an attacker try to misuse it for annihiling the distributed honeypot defense. Therefore the reachabilities computed by the algorithm 2 are multiplied by this limitative probability.

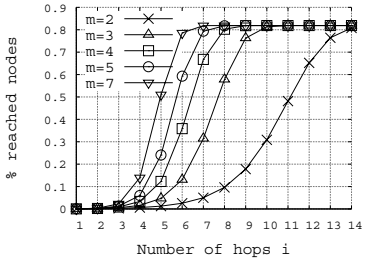


Fig. 3. Slapper model, N = 2000

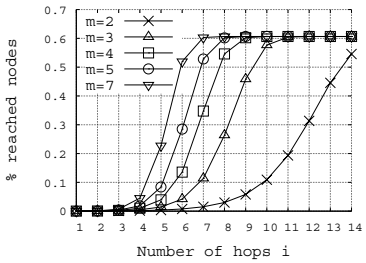


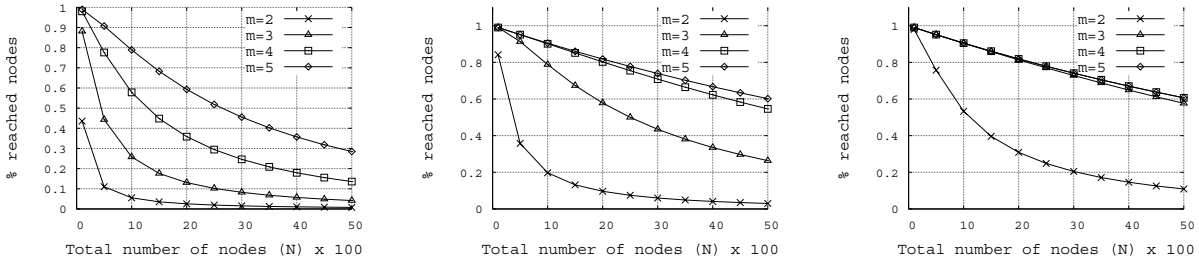
Fig. 4. Slapper model, N = 5000

V. Simulation results

The function and parameter $\alpha(m)$ and β have to be correctly fixed for a fair comparison with our IRC model in [2]. Contrary to P2P networks, this model is based on IRC servers and the results were generally interpreted by assuming 100 final hosts per server. Therefore, β_{IRC} in the IRC solution models the fact that 100 hosts can be detected. So, in P2P network models, β is fixed to $\dfrac{\beta_{IRC}}{100} = \dfrac{0.01}{100} = 0.0001$. In the same manner, $\alpha(m) = e^{\frac{2-log(N)}{100}}$ to perform comparisons between the different models.

A. Slapper model

1) Impact of the number of hops: The first experiments aim to show the reachability dependency on the number of hops and the branching factor. In figures 3 and 4, β limits the maximum reachability to $N \times (1-\beta)^N$. However $\alpha(m)$ which is more important for high branching factor has not a great impact on the results because in all cases, the higher the branching factor is, the better the reachability. Furthermore, whatever the number of nodes N is, a fixed number of hops allows to obtain the best reachability. For example, with $m = 5$, eight hops are needed to get the best result. Thanks to these figures, it is possible to determine a minimum number of hops to obtain attended performance. It is clear that a limit of four or five hops is totally useless. The total number of nodes N has an impact on the slope of the curves. When N increases, the reachability increases more at the beginning of the curve and so increases less before reaching the maximum value.



(a) i = 6 (b) i = 8 i = 10

Fig. 5. Slapper model, impact of the number of hops i

2) Impact of the number of nodes: Figure 5 shows the dependency between the reachability and the total number of nodes N for different distances from the origin node: $i \in \{6, 8, 10\}$. Obviously all the curves decrease because the number of hops is fixed and so the total number of reached nodes is limited by it. When the number of hops increases, the curves for the highest branching factor converge to the same values limited by β once again. This observation confirms that for a high fixed number of hops, obtaining the maximum possible reachability is possible whatever the total number of nodes N is. Else $\alpha(m)$ entails a difference between the curves. However for a considerable number of hops, which is higher or equal to 10, the branching factor has no impact on the reachability. Otherwise, the higher the branching factor is, the higher the reachability is. The branching factor $m = 2$ does not provide good results. Although it is the default value of the slapper worm, this case can be considered as impracticable for a management framework.

B. Chord model

1) Impact of the number of hops: The two Figures 6(a) and 6(b) display the reachability results for different numbers of nodes. In fact, $N_{MAX} = N \times d$ even if it is not a power of two. Therefore it is possible to test with different d and to compare the results with other models. In fact, it is like the chord ring was cut at the end which is managed by our algorithm. Obviously the number of routing table entries is the first integer lower or equal to $log(N_{MAX})$.

Firstly, all the curves are very close and the best curves are the curves with the smallest distance d. However, this conclusion is unfair because it depends on d being a power of two. The problem is that in the equation (3), the terms $log(N_{MAX})$ and $log(d)$ should be integers which is not the case and thus the real value is based on integer values which are the first integer lower or equal than $log(N_{MAX})$ and the first integer greater or equal than $log(d)$. This explains the differences for values which are not powers of two due to the truncations to integer, the number of different entries is reduced and so less nodes can be reached. By testing with other values of d from 1 to 16, two classes of d exist: (1)C_0: d is a power of two and (2) C_1: d is not a power of two. For an equivalent network size N, the first class is better and all the curves are similar. The second class has different curves but never better than for the first class. Moreover, a reachability close to one can be reached. In fact, β has not a great impact because an attacker needs to know at least $\frac{N}{log(N)}$ nodes to compromise all the network. Therefore, Chord is a solution to send the management request to all devices.

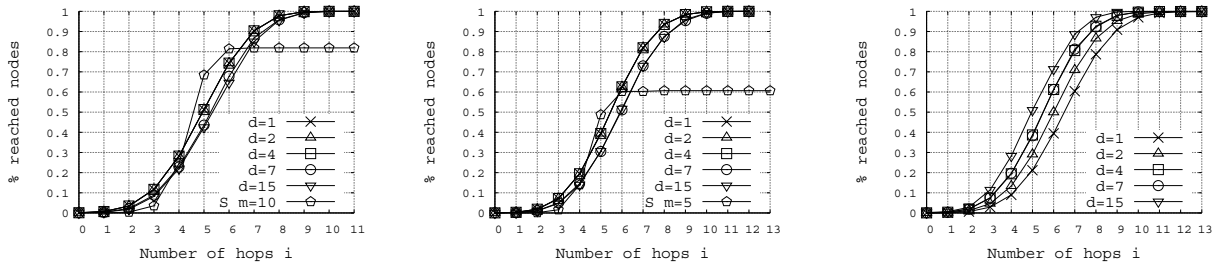
On the Figure 6(a) and 6(b), the corresponding values for Slapper are plotted by taking the suitable branching factor which is $log(N)$. The Slapper model has equivalent or better performance for a number of hops limited to 6. Therefore it seems that for a $i \leq 6$ hops, using Slapper architecture is sufficient. Indeed, in the previous section, it is shown that it is the minimum number of hops to attain the best reachability.

2) Impact of the nodes distribution: Because the branching factor m is fixed in the chord model, this experiment aims to show the impact of d, the average distance between two consecutive nodes identifiers. In the Figure 6(c), $N_{MAX} = 2^{14}$ and it is clearly highlighted that the higher the distance d is, the higher the reachability is. In fact, the performance are better because N decreases when d increases and each node has $log(N)$ entries.

Thus, when d is multiplied by a factor k, the number of different entries is decreased by $log(\frac{N_{MAX}}{d}) - log(\frac{N_{MAX}}{d \times k}) = log(k)$. In the same time, the number of nodes is divided by k and for high values of N it is clear that the number of nodes decreases faster than the number of different entries. That is why even if the number of different entries is reduced when d increases, the reachability increases due to the reduced population size.

3) Impact of the number of nodes: The curves plotted on Figure 7 illustrate the dependency between the reachability and the number of nodes. The curves have the same shape that for the Slapper model. However, because β has not a great impact, the maximal reachability is logically close to one. Once again the two classes can be distinguished. When simulating with other number of hops, the curves are similar but translated. Thus for a given reachability for N nodes, multiplying the number of hops by a quarter translates this value to $\frac{N}{4}$.

Because the real branching factor of the chord model (the number of different entries in the routing table) changes when the number of nodes changes (please see equation 3), on the Figure 7 the slapper curve is the maximal possible value *i.e.*, the limitation of β. For a huge network like more than 2^{12}



(a) Number of nodes $N = 2000$ (b) Number of nodes $N = 5000$ (c) Identifier space size $N_{MAX} = 2^{14}$

Fig. 6. The Chord model - Reachability (S stands for the slapper model)

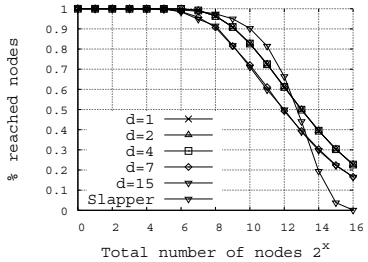


Fig. 7. Chord model - Rechability with i = 6 hops

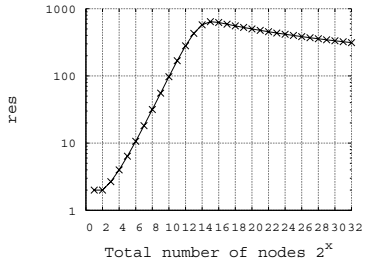


Fig. 8. Impact of attacks - $res(n)$

nodes, it is clear that the best performances are obtained by the Chord model. Else for a network size between 2^{10} and 2^{12} Slapper model can have better performances.

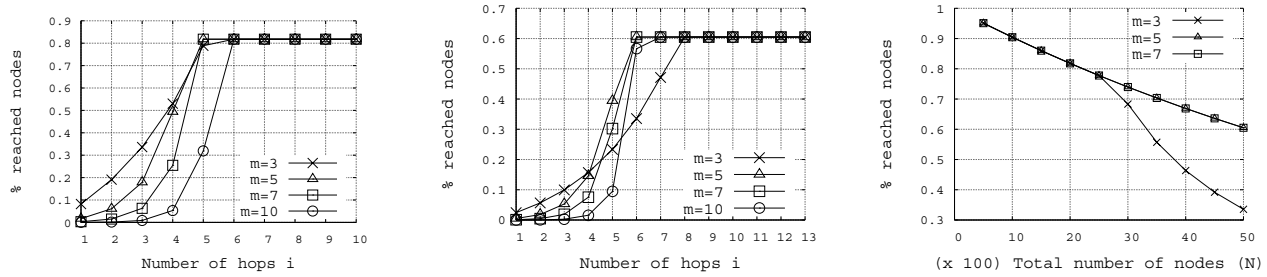
C. Impact of the attacks

This last experiment evaluates the resiliency of the different systems. $1-(1-\beta)^N$, is the probability to have the network discovered in the Slapper Model because each node knows all others. So the average number of detected nodes is $detected_{slapper} = N(1-(1-\beta)^N)$. In Chord, the average number of attacked nodes is βN. Each one has an average of $log(N)$ different entries in the routing table (see equation 3). So the average number of detected nodes is $detected_{chord} = \beta N \times log(N)$. We define a ratio: $res(N) = \dfrac{detected_{slapper}}{detected_{chord}} = \dfrac{1-(1-\beta)^N}{\beta \times log(N)}$. Firstly, this ratio does not depend on the distance d between two nodes in the identifiers space. Secondly, because a node knows every others in the slapper model, the impact of attacks is similar to the IRC model. The Figure 8 shows that the chord benefit is important because the curve increases quickly to reach a maximum for $N = 2^{15}$. $res(2^{15})$ is about 650 and means that a Chord based management plane divides the number of detected nodes by 650 in this case. Moreover the Chord model is always better than the slapper or the IRC model on this graph. For 2^{10} nodes in the networks, the model is 100 times more performant. However, after the maximum value for $N = 2^{15}$, the curve decreases slowly. However, for a huge network of 2^{512} nodes, the ratio is about 20 which means that it is reasonable to consider that the Chord model is more resistant to attacks due to the decentralization of the network knowledge.

D. Comparison with an IRC based botnet

As the IRC model considers only the servers, we assume that a server has 100 hosts connected and that the curves of [2] are modified to take in account this fact.

1) Impact of the branching factor: In the figures 9(a) and 9(b), the highest branching factors do not automatically imply the best performances contrary to P2P models. The network failures have an higher impact for an IRC architecture. Indeed, the more nodes there are, the more effective a high branching factor is. For example, a branching factor of 10 has poor performances with 2000 nodes. Moreover, the maximum reachability limited by β is the same for the Slapper and the IRC model. Furthermore, the curves increase quicker for the IRC model than other as showed by the Figures 9(a), 9(b), 3, 4, 6(a) and 6(b). Although for $m = 7$ in the Slapper model seven hops are always needed to obtain the maximum reachability, only five hops are necessary with the IRC model. The Figure 6(a) shows that seven hops are needed also whatever the value of d is. The IRC model has better performance because it is a centralized architecture with a limited number of servers. On the other side, as the number of hops is reduced, the anonymity is not as much protected as in the P2P networks. For the IRC model in[2], assuming a transmission time of t_{trans}, the total needed time to contact all reached hosts in five hops is $(5+100)*t_{trans}$ which is the time to contact the hosts of the last server (please see [2] for more details). For a P2P network only $(m+7-1) = 13*t_{trans}$ are necessary because the final hosts are directly considered and thus the time is limited by the delay to send the messages to all neighbors and the delay for the last message of them to



(a) Total number of nodes $N = 2000$ (b) Total number of nodes $N = 5000$ (c) Number of hops $i = 6$

Fig. 9. The IRC model

be transmitted to the last device ($7 - 1$ hops left). The time is divided by $\frac{105}{13} \simeq 8$. Furthermore in 6(a) and (b), it is shown that the slapper model is equivalent to the Chord model for a number of hops less or equal to six. Therefore, if there is a hard time constraint, using a P2P architecture is better. If reaching a reasonable proportion of hosts is sufficient, the limit can be fixed at six hops and a simple Slapper communication scheme could be used. Otherwise, the Chord model provides the best performances. Furthermore, only the Chord model guarantees a reachability very close to one.

2) Impact of the number of nodes: In the figure 9(c) the reachability of the IRC model is better than for Slapper model 5(a) for 6 hops only. For $N < 2^{12}$ the Chord model is also equivalent to the Slapper model. In fact, for the Slapper model, the highest curves did not converge to the same value and are not linear. It means that the maximum reachability is not reached which contrasts the IRC model where the high connectivity assured this. On the other side, if $N > 2^{12}$, the reachability is very limited by β and it is the same curve for the IRC model that for slapper on the Figure 7.

3) Comparison results: The goal of this section is to determine what kind of architecture is more suitable depending on the final management application. First of all, the reachability is better for the IRC model for a reduced number of hops (< 7) but the time delay performances are very poor in comparison with P2P models. If the application needs to configure the majority of the participating computers, the Chord model is more suitable than others due to low exposition to attacks. However, if the application doesn't need to configure many computers but only a part, the other models can be suitable. The Figure 7 shows that if only twenty percent of hosts need to be correctly configured, the other models are sufficiently performant if $N \leq 2^{14}$. Another attended property is the tracking prevention. In all P2P models, knowing the central authority is not possible as the messages are forwarded by several peers. The IRC architecture needs servers, the attackers knows where the management requests are issued from. Besides, the load of an IRC system entails to have a central authority to manage the IRC architecture. In [2], the load of an IRC server with 100 connected hosts is $load_{IRC} = (100 + m) * load_{host}$ if we consider that being connected with an host ($load_{host}$) and a server is the same for simplicity reason. In the case of a P2P network, the load is limited to $load_{P2P} = m * load_{host}$. The advantage of the IRC architecture is that only the IRC servers need to have open ports. In a P2P networks, each node needs to have open port to forward a request. The IRC architecture is the best solution ofr a pure management plane for security reasons. Moreover because of the professional usage of the computers and the firewall configurations, they should be less exposed to the attacks and the reachability could be improved. In our case of the detection of cyber predators, the participating computers are personal computers and so more exposed. Therefore a Chord approach is preferable. Moreover they run a P2P file sharing applications and thus in main case, they have already an open port. The Table I is a summary of the properties of each studied framework.

	IRC	Slapper	Chord
Efficiency	The lowest number of hops	The lowest delays	
Resiliency	very constrained by the probability to have a network failure or to be attacked	very constrained to be attacked, few connections to manage	high resiliency, few connections to manage, partial view of the network
Scalability	Network size $< 2^{12}$		Network size $\geq 2^{12}$
Security	The manager can be tracked	Tracking the manager is very difficult due to the intermediary nodes	
Interest	Large and closed networks with a main central authority (companies)	Large and open networks with checked partners (a research distributed honeypot)	Huge and public networks (honeypot where everyone can participate)

TABLE I
COMPARISON OF THE DIFFERENT FRAMEWORKS

VI. RELATED WORKS

Most of botnets are based on IRC [9]. In [10], a P2P botnet is described. With respect to these works we are the first to propose a model to evaluate the performances of P2P botnets. A malware based network management was already introduced in [11] where a worm patrols on different hosts but the experiments are very limited. The Code-Green worm [12] is a salutary worm which patches Code-Red infected computers. In [13], the authors propose to use a pastry overlay

network for distributing an alert about a malware propagation. It is complementary to our study as it is based on pastry and it is different because no model for helping an administrator is provided. The most common management solution is a direct communication between the management platform and all the devices. It is very close to the IRC based botnet solution with only one server. This kind of architecture is centralized and limits the scalability which other approaches aim to solve. The first one is the management by delegation [14]. An efficient way to improve the scalability is to use hierarchical cascaded managers [15]. In fact, each intermediary manager is in charge of creating the fitted queries to the hosts under its responsibility from the origin query. The replies of the hosts need to be transformed also before being transmitting to the main controller like in [16]. In fact, in these mentionned work the original request is only a basis for mid-level managers which need to adapt it. The introduced case in this paper is different because the goal of the framework is to do mass configuration. A single complete request is required and no mid-level manager is able to complete it which is better for security reasons. A P2P extension of the hierarchical management is proposed in [17]. The paper [18] aims to solve the distributed pattern matching to perform lookup with wildcards in P2P networks. The authors propose to use aggregation and replication. The authors in [19] show that the echo pattern, which is very close to a P2P network, has promising results in term of performances. Other solutions are based on active networks [20] where a light management program is forwarded to be executed at the network devices. However, dedicated network equipments are necessary which slows down the deployment of these solutions. In [21], a dedicated solution to ad-hoc networks creates groups and elects a manager inside it depending on the host connectivities. Besides, the network configuration to create the botnet is simple because it is possible to use the P2P file sharing system and dedicated ports. Furthermore, our approach does not need specific host except the management platform because every private user can participate because the needed resources are very limited contrary to other mentionned solutions. Finally the paper [22] focuses on the distributed hash tables (DHT) and shows that an efficient routing protocol should be associated with a good technique to compose and decompose the DHT due to the connections and the disconnections of the peers.

VII. Conclusion and future works

Botnets are an effective and scalable solution for management application. In our previous work [2], we studied the possibility to deploy an IRC-based botnet for network management. This paper proposes a P2P network as a communication channel for management request. Two networks are modeled: Chord and the Slapper communication mechanism by taking in account the possibility to have a network failure or to be attacked. A manager is able to know the efficiency of sending a request in term of reachability *i.e.*, the percentage of reached devices depending on the time delays or the number of hops. Using such an architecture for configuring honeypots does not need many requirements and provides good results. Considering the case of an open distributed honeypot where everybody can participate, the Chord approach has the best results due to a better robustness that Slapper communication. We plan to deploy the distributed honeypot on a real testbed.

Acknowledgment This paper was supported in part by the EC IST-EMANICS Network of Excellence (#26854).

References

[1] J. Oikarinen and D. Reed, "rfc 1459: Internet relay chat protocol," United States, 1993.

[2] J. Francois, R. State, and O. Festor, "Botnet based scalable network management," in *18th IFIP/IEEE Distributed Systems: Operations and Management (DSOM 2007)*, Silicon Valley, CA, USA, October 2007.

[3] R. Badonnel, R. State, I. Chrisment, and O. Festor, "A management platform for tracking cyber predators in peer-to-peer networks," *icimp*, vol. 0, p. 11, 2007.

[4] L. McLaughlin, "Bot software spreads, causes new worries," *IEEE Distributed Systems Online*, vol. 5, no. 6, 2004.

[5] C. Schiller and J. Binkley, *Botnets: The Killer Web Applications*. Syngress Publishing, 2007.

[6] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable Peer-To-Peer lookup service for internet applications," in *Proceedings of the 2001 ACM SIGCOMM Conference*, 2001, pp. 149–160. [Online]. Available: citeseer.ist.psu.edu/stoica01chord.html

[7] I. Arce and E. Levy, "An analysis of the slapper worm," *IEEE Security and Privacy*, vol. 1, no. 1, pp. 82–87, 2003.

[8] S. El-Ansary, L. O. Alima, P. Brand, and S. Haridi, "Efficient broadcast in structured p2p networks." in *IPTPS*, 2003, pp. 304–314.

[9] P. Barford and V. Yegneswaran, *An inside look at Botnets*. Springer, 2006, ch. 1.

[10] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," in *First workshop on Hot Topics in Understanding Botnets (HotBots 07)*, Cambridge, MA, April 2007.

[11] H. Ohno and A. Shimizu, "Improved network management using nmw (network management worm) system," in *Proceedings of INET'95: Honolulu, Hawai'i, June 27-30, 1995.*, 1995.

[12] P. Szor, *The Art of Computer Virus Research and Defense*. Addison-Wesley Professional, 2005.

[13] M. Costa, J. Crowcroft, M. Castro, A. Rowstron, L. Zhou, L. Zhang, and P. Barham, "Vigilante: end-to-end containment of internet worms," in *SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles*, vol. 39, no. 5. New York, NY, USA: ACM Press, December 2005, pp. 133–147.

[14] G. Goldszmidt and Y. Yemini, "Distributed management by delegation," in *15th International Conference on Distributed Computing Systems*. IEEE Computer Society, 1995.

[15] SNMP and Research, "The mid-level manager. http://www.snmp.com/products/mlm.html (accessed on 07/30/07)."

[16] K.-S. Lim and R. Stadler, "Real-time views of network traffic using decentralized management," in *Integrated Network Management, 2005. 9th IFIP/IEEE International Symposium on*, 2005.

[17] M. S. Artigas, P. G. López, and A. F. Gómez-Skarmeta, "Deca: A hierarchical framework for decentralized aggregation in dhts." in *DSOM*, 2006, pp. 246–257.

[18] R. Ahmed and R. Boutaba, "Distributed Pattern Matching: a Key to Flexible Efficient P2P Search," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 73–83, January 2007.

[19] K.-S. Lim and R. Stadler, "A navigation pattern for scalable internet management," in *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM'01), Seattle, USA, May 2001*, 2001.

[20] M. Brunner and R. Stadler, "Management in telecom environments that are based on active networks," *Journal of High Speed Networks*, 2001.

[21] R. Badonnel, R. State, and O. Festor, "Probabilistic management of ad-hoc networks," in *10th IEEE/IFIP Network Operations and Management Symposium - NOMS 2006*. IEEE Computer Society, 2006.

[22] L. Cheng, R. Ocampo, K. Jean, A. Galis, C. Simon, R. Szabó, P. Kersch, and R. Giaffreda, "Towards distributed hash tables (de)composition in ambient networks," in *DSOM*, 2006, pp. 258–268.