2016 IEEE International Conference on Software Science, Technology and Engineering

Identifying Attack Propagation Patterns in Honeypots using Markov Chains Modeling and Complex Networks Analysis

Ariel Bar, Bracha Shapira, Lior Rokach and Moshe Unger
Department of Information Systems Engineering and
Telekom Innovation Laboratories
Ben-Gurion University of the Negev
Beer-Sheva, Israel
{arielba, bshapira, liorrk, mosheun}@bgu.ac.il

Abstract— **Honeypots are computer resources that are used to detect and deflect network attacks on a protected system. The data collected from honeypots can be utilized to better understand cyber-attacks and provide insights for improving security measures, such as intrusion detection systems. In recent years, attackers' sophistication has increased significantly, thus additional and more advanced analytical models are required. In this paper we suggest several unique methods for detecting attack propagation patterns using Markov Chains modeling and complex networks analysis. These methods can be applied on attack datasets collected from honeypots. The results of these models shed light on different attack profiles and interaction patterns between the deployed sensors in the honeypot system. We evaluate the suggested methods on a massive data set which includes over 167 million observed attacks on a globally distributed honeypot system. Analyzing the results reveals interesting patterns regarding attack correlations between the honeypots. We identify central honeypots which enable the propagation of attacks, and present how attack profiles may vary according to the attacking country. These patterns can be used to better understand existing or evolving attacks, and may aid security experts to better deploy honeypots in their system.**

Keywords - Cyber Security; Honeypots; Attack Propagation; Markov Chains; Complex Networks Analysis;

I. INTRODUCTION

During the last decade, we observe an extensive increase both in the magnitude and sophistication of cyber-attacks. Malicious activities on the internet such as botnets, worms, malware, viruses and denial of service attempts are required to be recorded, investigated and analyzed in order to cope with these threats. Thus, various approaches have been developed and deployed with intent to monitor and collect real world data about malicious activities, in order to identify new trends and patterns on the adversaries' strategies.

Honeypots are computer resources which are used to detect and deflect attacks on a protected system, as their value lies in being probed, attacked or compromised [1]. Honeypot frameworks (a set of deployed and monitored honeypots) are designed to attract attackers by presenting false or misleading information that an attacker will want to gain, attack or control. By definition, honeypots are supposed to have zero false alarms, since no legitimate or approved access to them is expected to be observed.

Analyzing the captured data from honeypots may shed light on new attack patterns or emerging threats, and aid security administrators or designers to support decision making and improve defense mechanisms. Propagation of attacks within the honeypot framework [2] is one of the possible analytical models that can be learned from the collected data.

Honeypot propagation models try to detect phenomena that reflect the propagation of attacks through different honeypots in the system. Examples of such situations might occur as a result of a scanning activity or from the propagation of worms within the honeypot framework. These models reveal information about attack trends which span over multiple attacked honeypots, rather than observing attacks on each honeypot at a time.

In this work we suggest a two-phased method for modeling honeypot propagation patterns within a honeypot framework. Our solution is based on a hybrid modeling approach which combines probabilistic Markov Chains with techniques originating from the area of complex networks analysis.

Our first contribution considers the questions that the generated model is capable of answering:

- What is the probability distribution of the next expected attacked honeypot in an attack session?
- Which honeypots are co-attacked together?
- Which attack propagation combinations are considered new and evolving?
- Which honeypots contribute the most to the attack propagation within the framework?
- Are there any contextual differences between attack propagations? (E.g. Do attack propagation patterns vary by the attacking country?).

Answering these questions can lead to the discovery of new attack trends, and aid security administrators at better deploying their honeypots. To the best of our knowledge this is the first modeling technique regarding attack propagations that is capable of answering all of these questions.

Our second contribution relies on the empirical assessment of our suggested solution. We evaluate our method on a massive dataset containing more than 167 million attacks on a honeypot framework that was deployed for more than a year. The monitored honeypot sensors are deployed in multiple locations around the globe and include several types of honeypot families which emulate a variety of network services and protocols.

The rest of this paper is organized as follows: in Section II we present related work considering analytical models for attack propagation; Section III includes a detailed review of the proposed method; the evaluation of the proposed solutions is described in Section IV; and finally, in Section V we conclude our work and outline several directions for further research.

II. RELATED WORK

According to a comprehensive survey regarding honeypot technology [3], one of the main research areas in this domain is the utilization of the output data that is collected by the honeypots. Indeed, in recent years much effort has been invested in designing new and advanced models to analyze captured data from honeypots. One of these models outlines the analysis of the propagation of attacks between the different honeypots in the monitored system.

The model suggested in [2] is designed to detect attack propagations which occur when the IP address of an attacking machine observed at a given honeypot, is also observed at another one. This scenario is most relevant in the context of scan activities or propagation of worms. Propagation is said to occur from honeypot P_i to honeypot P_j, when the same attacker attacks P_j immediately after P_i. The suggested model is a propagation graph, where each node represents a honeypot; a link between two nodes represents propagation between two honeypots; and each link is assigned with a probability. This propagation model serves as the basis for our suggested approach with two major contributions: first we give a formal representation to the propagation graph by considering it as the graph representation of a Markov Chains model [4]; and second, we extend it by adding a new layer of analysis methods that are adapted from the complex networks research area [5]. Using the suggested enhancements, we are able to identify more meaningful propagation patterns – e.g., Which honeypots are usually attacked together? Or, Which honeypots contribute the most to the attack propagation?

Previous works [6, 7] have tried to classify malicious web sessions into two families: "attacks" and "vulnerability scans". The web sessions were collected by honeypots, while the main modeling process included extracting various features from the web sessions, and training supervised machine learning models that distinguish between attacks and scans. We can consider these types of works parallel and complementary to ours, since scan activities are one of the examples of attack propagations.

On the same note, previous works that focused on the analysis of polymorphic worms can also be considered parallel and complementary to our suggested method. Two examples of such works can be found in [8, 9]. The first work identified signatures of polymorphic worms based on special tokens (substrings) which appear in the worms' payload data; while the latter identified worm signatures based on the worm's activity, e.g., examining sequences of operations in the SMB protocol.

Several studies that focused on proactive honeypots (honeypots which modify their configurations according to the attacks they observe) have used Markov Chains models or Hidden Markov Models [10] components in their architecture [11-13]. These Markov based components served as mechanisms which monitor the current state of the honeypot (e.g., "Normal", "Probed" or "Attacked"). In our proposed solution, we use Markov Chains in a different way, as we aim at modeling global propagation behaviors that we observe in the entire honeypot system.

III. METHOD

In this section we provide a detailed review of the methods for detecting propagation trends within a honeypot framework. The method can be applied on any dataset which contains basic information about the attacks. Moreover, the method is applicable both to low interaction honeypots, and high interaction honeypots alike.

A. Detecting Propoagation Trends

A propagation pattern among two honeypots is assumed to occur in situations where the same attacking machine attacks or scans for vulnerabilities several honeypots in a sequential order [2]. In other words, we aim to find sequential patterns which specify that the same attacker attacked Honeypot Y immediately after attacking Honeypot X.

The suggested method aims at finding these kinds of sequential patterns, and in addition tries to shed light on different interaction patterns between the different honeypots in the framework. For example, discovering which segments of honeypots are commonly attacked together.

In practice, we suggest a two-phased modeling process to achieve this goal: First, we train a Markov Chains [4] model in order to discover sequential attack patterns in the data. Second, we use the graph structure of the trained Markov Chains model and apply graph based algorithms in order to discover the interaction patterns between the honeypots.

The workflow of the suggested modeling process is presented in figure 1. The input for the process is a dataset of atomic attacks on the honeypot framework. Each attack record must consist of an attacker ID, the attacked honeypot ID, and the timestamp of the attack. The modeling process contains four steps:

1. "Create Attack Sessions": This process creates for each attacker, a list of his/her sequential attacks on the honeypot platform.
2. "Train Propagation Graph": This process is responsible for training a Markov Chains model based on the attack sessions from the previous step.
3. "Discover Honeypot Communities": Based on the trained Markov Chains graph representation, we identify segments of honeypots which are co-attacked together. This task is carried out by applying a community detection algorithm [14].

4. "Honeypot Ranking" – Each of the honeypots in the graph is ranked according to centrality based measures [15].

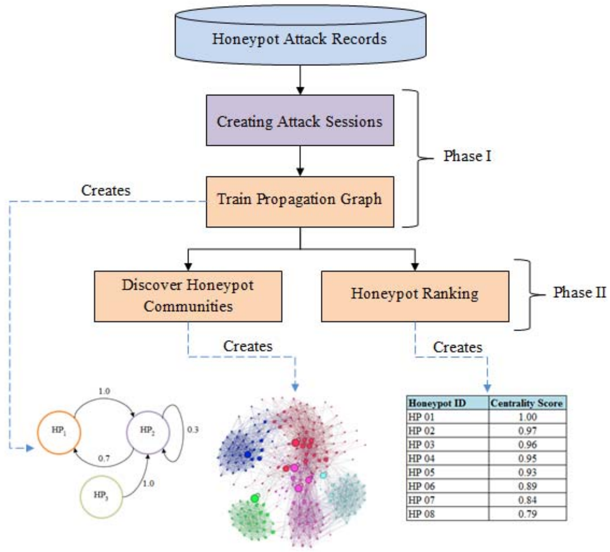

Figure 1: Modeling Workflow

After the compilation of these four steps, the resulting model includes three components: a) a Markov Chains model which holds the sequential attack propagations' probabilistic distributions, b) Segmentation of all honeypots into communities, and c) Ranking scores for each honeypot, which determine their centrality in terms of attack propagation. In the next subsections we provide additional details on these steps and their outputs.

B. Markov Chains Model for Attack Propagation

The first two steps in our modeling process (Phase I) involve training a probabilistic model which captures the dynamics of attack prorogation across the monitored honeypots in the system. In practice, we apply Markov Chains modeling for this task.

A Markov Chain [4] is a stochastic process which refers to a sequence of random variables in which the process changes over time. The assumption behind this is that given the present state of the variables, the future and past are independent. Markov Chains can thus be used to describe a system which follows a series of events, where what happens next depends only on the current state of the system. More formally, this property satisfies the probabilistic condition which appears in equation 1, where here x_i specifies the state of the system at time i.

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2 \dots, X_n = x_n) =$$
$$P(X_{n+1} = x | X_n = x_n) \text{ , when}$$
$$P(X_1 = x_1, X_2 = x_2 \dots, X_n = x_n) > 0 \quad (1)$$

Markov Chains are often modeled as a directed graph where the nodes represent the states and the edges are labeled with the probabilities of moving from one state at time n to another state at time $n+1$. A frequent assumption is that the Markov Chains are time-homogeneous, i.e., the probabilities on the graph are independent of n. Training a Markov Chains model requires estimating all transition probabilities from state i to state j.

In the context of analyzing attack propagation, the main idea is to model sequences of attacks on the honeypot system as a Markov process where an attacker attacks one honeypot after another. Given a dataset of atomic attack records on the honeypot framework, where each record contains the source IP of the attacker, a timestamp, and the attacked Honeypot identifier, we model these attacks according to the following approach:

- Creating Attack Sessions – We define an attack session as all atomic honeypot attack records from the same attacker in a certain time-interval (horizon window). In this analysis, the attacker is identified by the attacking source IP in a time-interval of one hour. Notice that relying solely on the attacker's source IP is misleading, since ISPs mostly allocate dynamic IPs that change regularly. We selected a relatively short time interval in order to maximize the possibility that the grouped attacks are under the same circumstances.
- Identifying Attack Sequences – We define an attack sequence as an ordered list of atomic attacks in the attack session. The attack list is ordered according to the timestamps of the attacks in chronological order.
- The honeypot IDs (denoted by HP_{id}) are considered as states in the Markov Chains model. Hence, we are interested in finding the probability P_{ij} that an attacker will attack HP_j immediately after attacking HP_i, in the context of the current attack sequence.
- Estimating P_{ij} is performed by calculating the ratio between the number of observed transition attacks from HP_i to HP_j, divided by the number of all observed translations from HP_i to any honeypot.

The resulting Markov Chains model reveals information regarding the propagation trends within the honeypot framework. It can be used to predict the next attacked honeypot in a sequence of attacks. Moreover, the model can be utilized to detect abnormal attack sequences (sequences which do not follow the learned transition probabilities in the model).

C. Complex Networks Analysis for Attack Propagation

The final two steps in our modeling process (Phase II) are based on several algorithms that originated from the Complex Networks Analysis [5] research area. We apply these algorithms on the graph representation of the Markov Chains model that was described in the previous section. In particular, we first apply a community detection algorithm in order to discover segments of honeypots which are attacked together. Second, we use centrality based metrics in order to rank the relative importance of each honeypot in the framework.

Formally, let the propagation graph $G = (V, E)$ be a weighted direct graph: where, V represents the set of all honeypots in the framework; and E represents a set of directed edges between two nodes (u, v), with $v, \in V$, iff the transition probability of the trained Markov Chains model (P_{uv}) is non-zero. In addition, all the edges of the graph are labeled with weights. The weight of edge (u, v) is the transition probability (P_{uv}) in the Markov Chains model.

In the above representation, the task of community detection aims to find a partition of the graph into segments or communities of dense nodes internally, while nodes from different segments are sparsely connected. One of the most popular algorithms for community detection is Louvain Modularity [16].

The modularity of a partition is a scalar value that measures the density of edges inside communities as compared to the edges between communities. The value of modularity measure ranges between -1 and 1, and can be extended to scenarios where the edges have weights as can be observed from equation 2

$$Q = \frac{1}{2}\sum_{i,j} A_{i,j} - \frac{k_i k_j}{2m}\delta(c_i, c_j) \qquad (2)$$

$A_{i,j}$ represents the weight of edge that connects node i to j. k_i is the sum of weights of all the edges that are connected to node i. c_i is the community to which node i is assigned. $\delta(c_i, c_j)$ is an indicator function which is mapped to 1 if $c_i = c_j$ and 0 otherwise. Finally m is the sum of all weights in the graph divided by 2. In order to maximize this value efficiently, the Louvain Modularity algorithm has two phases that are repeated iteratively:

- First, each node in the graph is assigned to its own community. Meaning that we start with |V| communities. Then, for each node i we consider how much gain (in terms of the modularity measure) we would achieve from removing it from its current community and merging it with an adjacent community. If the gain is positive, the algorithm selects the adjacent community that maximizes this value.
- The second phase of the algorithm groups all of the nodes in the same community, and builds a new graph. Nodes are the communities from the previous phase. The weights on the edges between the new nodes are given by the sum of the weights of the edges between the nodes in the corresponding two communities. The edges between the nodes of the same community are replaced with self-loops.

The algorithm repeats these two phases several times, until there is no modularity gain from changing the graph structure. The empirical evolution indicates that the number of required passes until convergence is relatively small (less than 5), resulting in short running times. An additional desired feature of this algorithm is that the number of expected communities is determined automatically, unlike some clustering algorithms such as k-means [17].

In the context of attack propagation analysis, the resulting partition of the propagation graph into several communities or segments reveals valuable information about the attack trends. This analysis highlights areas of the honeypot network that are co-attacked together on the same attack sequences.

In addition to identifying segments of highly correlated attacked honeypots, we would like to estimate the relative importance level of each honeypot for enabling attack propagation. Some honeypots may attract more varied attacks, while others may have a high spreading propagation factor. In addition, some honeypots may act as mediators and central bridges for the attack propagations.

Applying node centrality measures [15] on the propagation graph G may approximate the relative importance of each honeypot for attack propagation. We suggest applying the following three centrality measures for each node $v \in V$ to achieve this task

1. Out-Degree Centrality – measures the out-degree (in terms of adjacent nodes) of the honeypot on the prorogation graph G. High values on this metric suggest that attackers who attack this honeypot, tend to spread out, and continue to attack various other honeypots. Thus, we treat this centrality measure as the "Spreading Potential" of the honeypot for attack propagation. Similarly, a low out-degree indicates that attackers tend to finish or abort their attack sequences with this honeypot.
2. In-Degree Centrality – this centrality measure is symmetrical to the previous one. In this case we measure the in-degree of that honeypot on the prorogation graph G. This metric can be considered as the "Absorbing Potential" of the honeypot. High values on this metric suggest that the origin of the attacks on this honeypot (in an attack sequence) can result from multiple other honeypots.
3. Betweenness Centrality [15] – this measure quantifies the number of times a node acts as a bridge on the shortest path between two other nodes in the graph. More formally, calculating the Betweenness centrality for node v can be computed using equation 3, where σ_{st} denotes the number of shortest paths from node s to t, and $\sigma_{st}(v)$ denotes the number of shortest paths between s and t, that pass through node v.

$$C_b(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad (3)$$

Honeypots with a high degree of Betweenness centrality may act as mediators for attack propagation across the honeypot framework.

IV. EVALUATION

In this section we present the results from applying the suggested method on a massive data set of honeypot attacks. The analyzed dataset was collected via the T-Pot[1] Multi-Honeypot Platform. T-Pot is a globally deployed honeypot framework which contains multiple well-established honeypot technologies.

[1] http://dtag-dev-sec.github.io/

A. Data Analysis

The analyzed dataset contains meta-data about atomic attacks on the T-Pot honeypot framework. Overall, we analyzed 167,709,008 attacks on the framework between July 2014 and August 2015. The attacks targeted 267 distinct honeypots deployed around the world. The attacks originated from 901,549 distinct IP addresses that span over 14,108 different ISPs in 229 countries around the globe.

The analyzed dataset contains attacks on four families of honeypots:

- Glastopf[2] - honeypot which emulates vulnerabilities that are relevant to web applications.
- Kippo[3] - a medium interaction SSH honeypot.
- Honeytrap[4] - a low-interaction honeypot that aims at collecting malware in an automated way.
- Dinoaea[5]- malware capturing honeypot that emulates several well-known protocols such as SMB, HTTP, FTP, MSSQL and VOIP.

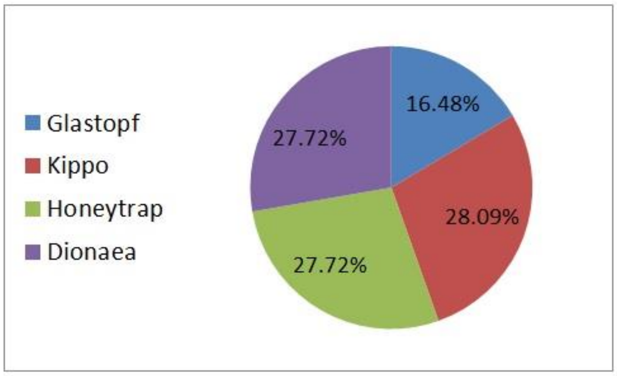

Figure 2: Honeypot Family Distribution

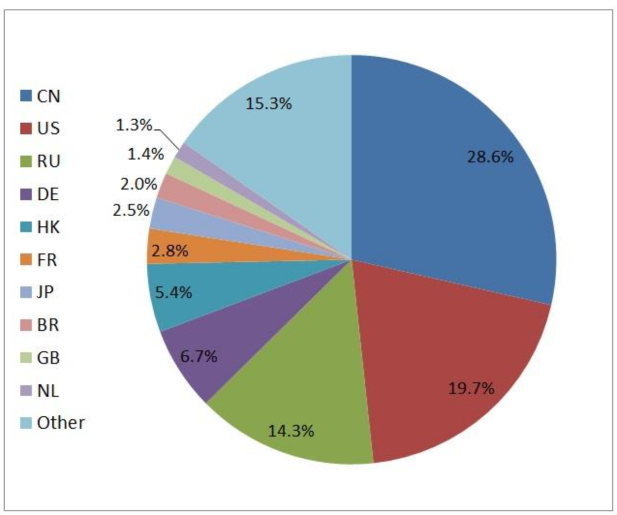

Figure 3: Attacking Countries Distribution

[2] http://glastopf.org/

[3] https://github.com/desaster/kippo

[4] https://sourceforge.net/projects/honeytrap/

[5] http://www.edgis-security.org/honeypot/dionaea/

The distribution of honeypot types within the framework is fairly uniform, except for the Glastopf family which has fewer instances, as is presented in figure 2. The top 10 countries which produced the most attacks on the honeypot framework are presented in figure 3. The distribution is long-tailed, as 219 of the remaining countries are responsible for 15.3% of the attacks. China, Russia and USA were recognized as the top attacking countries, responsible for a total of 62.6% of the attacks

In addition, we identified a phenomenon of "heavy-hitters" - specific IP addresses which produce an extensive amount of attacks. The top 10 attacking IP addresses were responsible for 24.6% of the attacks. Overall we identified 1,618 attacking IP addresses that produced more than 10,000 attacks on the T-Pot platform during the 13 months analysis period. We also observed that some honeypots were attacked more than others: the top 3 attacked honeypots absorbed 23.69%, 12.11% and 11.3% of the attacks. However 140 honeypots absorbed at least 10,000 attacks during that period.

B. Propagation Patterns Analyis

We applied the suggested two-phase method on the observed attacks. Based on the atomic attack records, we first created one hour-long attack sessions based on the attackers' IP addresses.

The transformation from atomic attack records into sessions produced 4,175,238 distinct attack sessions. However, when we further investigated the created sessions, we observed that the 92.7% of these where "singleton" sessions, i.e., sessions which contain only one attacked honeypot. Since "singleton" sessions do not have contribution leading to a better understanding of attack propagations, we filtered them out for the remainder of the analysis. Overall, we observed that 305,134 attack sessions contained a propagation pattern within the honeypot framework.

Next, we trained the Markov Chains model on the filtered sessions and produced a model with 231 states (honeypots) linked with 6,557 propagation edges between them. These numbers of nodes and edges suggest that the density of the graph is equal to 12.2%. Thus, the graph is neither extremely dense nor sparse. The medium density level is an indicator that we may discover well defined propagation patterns within the model. We also observed that although only 7.3% of the attack sessions had some propagation behavior, it covered 86.5% (231 from 267) of the honeypots in the framework.

To continue our analysis, we furthered examined the graph representation of the Markov Model: first, we applied the Louvain Modularity algorithm in order to detect honeypot communities; second, we calculated the three centrality measures for each of the honeypots.

The propagation graph of all 231 honeypots is presented in figure 4. Each node in the figure represents a honeypot, while the edges between the nodes represent a propagation pattern between the matching honeypots. Overall, we were able to detect six well defined honeypot communities. Each community was assigned with a matching label and color.

Finally, the size of each node is proportional to the "Betweenness Centrality" measure. Examining the propagation graph reveals some interesting findings:

- Most of the honeypot communities (5 out of 6) contain between 30 and 72 honeypot members. Only one community (labeled as "isolated") is extremely small and contains only two honeypots.
- The "Isolated" community acts as a separate connected component in the graph. There are no propagations to/from this community and the other five communities (henceforth, referred as the "Big 5").
- In most cases, the members of each of the "Big 5" communities belong to the same class of honeypots: the "cyan" community consists of 97% "Dionaea" honeypots; the "magenta" community consists of 75% "Honeytrap" honeypots; while the "green" and "blue" communities consist of 100% "Glastopf" and "Kippo" honeypots respectively. The only exception is the "red" community which includes all types of honeypots.
- The majority of attack propagations are within the communities themselves. We can observe this on the graph in the way that each community forms a dense "clique" area: the "Glastopf/Green" and the "Dionaea/Cyan" communities are almost complete cliques; while the other communities have some dense areas, along with more sparse ones. For example, the "Hybrid/Red" community has a dense area of honeypots in the center of the graph which almost forms a clique, while more isolated honeypots appear in the upper-right corner of the graph.
- We can observe that there are several attack propagation patterns between the "Big 5" communities: for example we can see that the attacks tend to propagate back and forth from the triangle of "Hybrid/Red", "Dionaea/Cyan" and "Honeytrap/Magenta" communities. However, propagations such as between the "Kippo/Blue" and the "Dionaea/Cyan" are less common.
- The "Glastopf/Green" community is more isolated than the rest of the "Big 5" communities. Hence, the web attacks are more self-contained
- In each of the "Big 5" communities, we can observe that a limited number of honeypots have a high "Betweenness Centrality" measure: The "Dionaea/ Cyan", "Kippo/Blue" and "Glastopf/Green" communities have a distinct, leading honeypot; the "Honeytrap/Magenta" has three; and the "Hybrid/Red" has one distinct leader, with additional minor leaders. When observing more closely, we can see that the reason for the high centrality score is that the attack propagations between communities tend to pass through these honeypots. Thus, these leading honeypots are very interesting as they act as a catalyst which enables attack propagations between different communities.

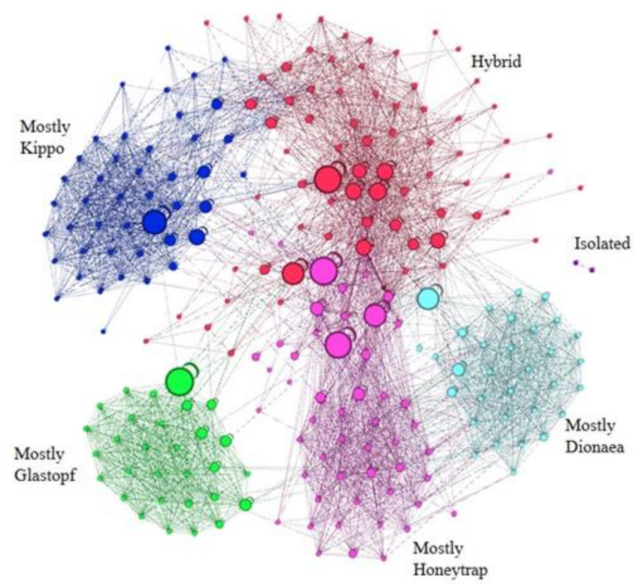

Figure 4: Propagation Graph - All Attacks

TABLE I. PROPAGATION TRENDS - BY IP ADDRESSES

From \ *To*	*Hybrid*	*Honeytarp*	*Dinoaea*	*Glastopf*	*Kippo*
Hybrid	35,605	3,728	197	183	444
Honeytrap	3,691	16,463	1,173	64	41
Dinoaea	192	1,261	3,515	63	59
Glastopf	193	63	72	635	1
Kippo	446	41	51	1	2,287
Hybrid	35,605	3,728	197	183	444

Based on the detected communities and propagation patterns we observed in the graph, we further examined the effect of each propagation trend between the "Big 5" communities. We measured the effect of each propagation trend by counting the number of attacking IP addresses that followed that trend. Table 1 summarizes each of the possible 25 combinations of propagation trends. For example, the number of distinct IP addresses which attacked a honeypot from the "Honeytrap/Magenta" community immediately after attacking a honeypot from the "Hybrid/Red" is 3,278.

As can be observed, the propagation trends presented in the table are consistent with the findings from the propagation graph: The noticeably higher numbers on the diagonal of the table indicate that propagations within the communities are very common. The relatively higher numbers in the Hybrid-Honeytrap-Dionaea area (the upper left region of the table) demonstrates the high interaction between these communities. The relatively lower numbers in the "Glastopf" row and column indicates that this community is more isolated and self-contained than the rest; for example, attack propagation from the "Glas-topf/Green" community, to the "Kippo/Blue" community was observed by one attacking IP address. In addition to these findings, we noticed that the table is almost symmetrical. This suggests that the attack propagation trends are also symmetrical, i.e., when we observe attack propagation from one community to another, there will most likely be a mirror image of that attack also.

We continued our analysis by comparing the centrality measures of the different honeypots. The distribution of a "Normalized Betweenness Centrality" score (the Betweenness Centrality of each honeypot divided by the maximum centrality in the graph) is presented is figure 5. As can be observed, the distribution is extremely skewed: almost 80% of the honeypots have a negligible betweenness centrality score, while only 7 honeypots have a normalized score that is above 0.75. The honeypots with the largest scores are the ones that were discussed earlier and identified as central enablers for attack propagation.

The Distribution of the "Out-Degree Centrality" score is presented is figure 6. In general, the out-degree ranged between 2 and 78. The distribution however is much less skewed compared to the "Normalized Betweenness Centrality", as the score of 61% of the honeypots ranged between 30 and 40. In addition, we observed that the distribution of "In-Degree Centrality" is very similar to "Out-Degree Centrality". This is another indicator that the propagation trends are symmetric in the analyzed data.

Our last analysis regarding centrality measures compared the correlation between the different centralities in order to examine whether they are consistent with each other. We observed that the "Out-Degree Centrality" and "In-Degree Centrality" were highly correlated (0.99 Pearson correlation). However, the correlation between the "Normalized Betweenness Centrality" and the "Out-Degree Centrality" and "In-Degree Centrality" was much lower (0.54 and 0.53 Pearson correlations, respectively). This can be demonstrated by examining the centrality measures of the top-seven leading honeypots according to the "Normalized Betweenness Centrality" score as presented in table 2. We can see that the In/Out centrality measures are highly varied, but consistent with each other. This indicates that several centrality measures are required to better understand propagation patterns.

TABLE II. TOP HONEYPOTS (ACCORDING TO NOMALIZED BETWEENESS CENTRALITY)

HP	Type [community]	Centrality Score		
		Normalized Betweenness	In-Degree	Out-Degree
1	Glastopf [Green]	1.0	35	35
2	Honeytrap [Red]	0.99	53	55
3	Dionaea [Cyan]	0.96	74	73
4	Honeytrap [Magenta]	0.93	44	46
5	Kippo [Blue]	0.83	39	40
6	Honeytrap [Magenta]	0.76	66	70
7	Honeytrap [Magenta]	0.75	27	27

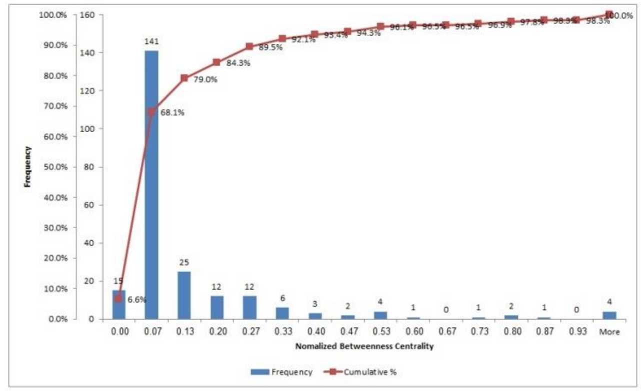

Figure 5: Normalized Betweenness Centrality Distribution

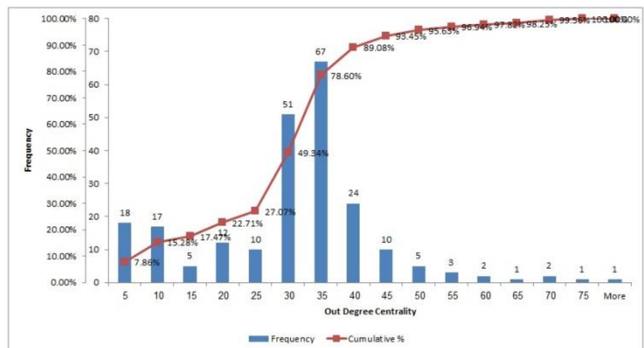

Figure 6: Out-Degree Centrality Distribution

Finally, we analyzed whether attackers from different countries share the same attack propagation patterns. We focused our analysis on China and Russia - the two countries that produced most of the attacks on the honeypot framework. Using the same analysis procedure, we produced propagation graphs which originated from China and Russia as presented in figures 7 and 8, respectively.

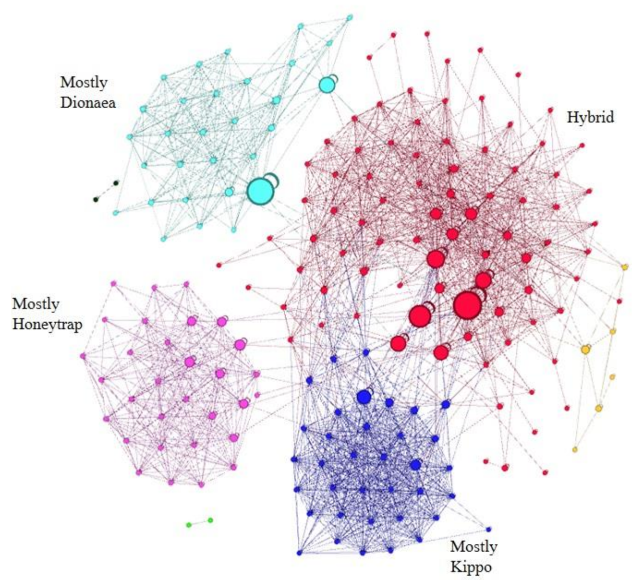

Figure 7: Propagation Graph – Attacks Originating from China

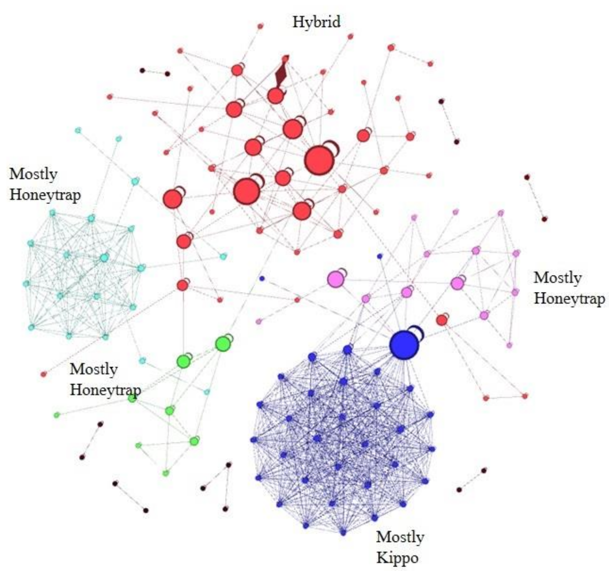

Figure 8: Propagation Graph – Attacks Originating from Russia

As can be observed, the two propagation graphs are different from one another. The propagation graph of an attack that originated from China has four well-defined communities. When we examined the communities more thoroughly, we observed that they overlap in four out of the five communities that were referred to as the "big 5" in the previous analysis. In practice, the only missing community was the Glastopf/Green. However, when we examined the propagation graph of the attacks that originated from Russia, the similarity was less obvious: the original "Honeytrap/Magenta" community is divided into three separate communities; there is no well-defined community of "Mostly Dionaea"; additionally, we observed more small and isolated communities. This analysis suggests that it is likely to expect different attack propagation patterns from different sources of attackers.

V. Conclusions And Future Research

In this work we presented a new method for analyzing attack propagation patterns based on data collected from honeypots. The method combines a probabilistic Markov Chains model with algorithms from the Complex Networks research area.

This is the first method for analyzing attack propagations that is capable of identifying highly co-attacked regions on the honeypot framework; highlighting common or evolving propagation trends; assigning a ranking score for all monitored honeypots according to their contribution to the attack propagation; and presenting whether contextual features have an effect on the attack propagations patterns.

We evaluated the proposed method on a massive dataset of attacks on a globally distributed honeypot. Analyzing more than 167 million attacks revealed that the honeypot system can be partitioned into several segments. Attack propagations within each segment are common, while propagations between different segments are rarer. Propagations between segments are more interesting since each segment empirically had its own characteristics. Hence, these kinds of attacks are expected to be more sophisticated. The model was able to identify which propagations between segments are more likely than the others. Thus, the suggested model is capable of identifying which propagation patterns are common, new and evolving.

Moreover, we were able to detect the top honeypots which contribute the most to attack propagation. Thus, security administrators can replicate the configuration of these honeypots in order to increase the magnitude of attack propagation in the system. Finally, our analysis demonstrates that attack propagation patterns may vary based on the origin of the attacking country.

Future research directions include the design of methods that are capable of handling scenarios where the identification of an attacker is not straightforward, e.g., attackers that are located behind a NAT, attackers that use an ammonization service, or attackers that control multiple IP addresses for their attacks. Moreover, additional methods for discovering of attack propagation can be designed and evaluated, e.g.: unsupervised machine learning techniques such as clustering or association rules. In addition, we would like to investigate how the discovered attack propagation patterns can be integrated into other honeypot's analytical models such as identifying bots, segmentation of attackers into botnets and predicting the expected number of attacks on the honeypot system.

References

[1] Spitzner, Lance. "Honeypots: Catching the insider threat." Computer Security Applications Conference, 2003. Proceedings. 19th Annual. IEEE, 2003.

[2] M. Kaaniche, Y. Deswarte, E. Alata, M. Dacier, and V. Nicomette, "Empirical analysis and statistical modeling of attack processes based on honeypots," in Workshop on Empirical Evaluation of Dependability and Security (WEEDS), Philadelphia, USA, June 2006, pp. 119–124.

[3] Bringer, Matthew L., Christopher A. Chelmecki, and Hiroshi Fujinoki. "A survey: Recent advances and future trends in honeypot research." International Journal of Computer Network and Information Security 4.10 (2012): 63.

[4] Grinstead, C. M., & Snell, J. L. (2012). Introduction to probability. American Mathematical Soc.

[5] Boccaletti, Stefano, et al. "Complex networks: Structure and dynamics." Physics reports 424.4 (2006): 175-308.

[6] Goseva-Popstojanova, Katerina, et al. "Characterization and classification of malicious Web traffic." Computers & Security 42 (2014): 92-115.

[7] Goseva-Popstojanova, Katerina, et al. "Quantification of attackers activities on servers running Web 2.0 applications." Network Computing and Applications (NCA), 2010 9th IEEE International Symposium on. IEEE, 2010.

[8] Newsome, James, Brad Karp, and Dawn Song. "Polygraph: Automatically generating signatures for polymorphic worms." Security and Privacy, 2005 IEEE Symposium on. IEEE, 2005.

[9] Su, Ming-Yang. "Internet worms identification through serial episodes mining." Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), 2010 International Conference on. IEEE, 2010.

[10] Rabiner, Lawrence R., and Biing-Hwang Juang. "An introduction to hidden Markov models." ASSP Magazine, IEEE 3.1 (1986): 4-16.

[11] Dantu, Ram, Joao W. Cangussu, and Sudeep Patwardhan. "Fast worm containment using feedback control." Dependable and Secure Computing, IEEE Transactions on 4.2 (2007): 119-136.

[12] Chen, Lin, et al. "Dynamic forensics based on intrusion tolerance." Parallel and Distributed Processing with Applications, 2009 IEEE International Symposium on. IEEE, 2009.

[13] Haslum, Kjetil, Marie EG Moe, and Svein J. Knapskog. "Real-time intrusion prevention and security analysis of networks using HMMs." Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on. IEEE, 2008.

[14] Lancichinetti, Andrea, and Santo Fortunato. "Community detection algorithms: a comparative analysis." Physical review E 80.5 (2009): 056117.

[15] Brandes, Ulrik. "A faster algorithm for betweenness centrality*." Journal of mathematical sociology 25.2 (2001): 163-177.

[16] Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." Journal of statistical mechanics: theory and experiment 2008.10 (2008):

[17] Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.