On a Multi-criteria Clustering Approach for Attack Attribution

Olivier Thonnard
Royal Military Academy
Polytechnic Faculty
Brussels, Belgium
olivier.thonnard@rma.ac.be

Wim Mees
Royal Military Academy
Polytechnic Faculty
Brussels, Belgium
wim.mees@rma.ac.be

Marc Dacier
Symantec Research
Sophia Antipolis
France
marc_dacier@symantec.com

ABSTRACT

We present a multicriteria clustering approach that has been developed to address a problem known as *attack attribution* in the realm of investigative data mining. Our method can be applied to a broad range of security data sets in order to get a better understanding of the root causes of the underlying phenomena that may have produced the observed data. A key feature of this approach is the combination of cluster analysis with a component for multi-criteria decision analysis. As a result, multiple criteria of interest (or attack features) can be aggregated using different techniques, allowing one to unveil complex relationships resulting from phenomena with eventually dynamic behaviors. To illustrate the method, we provide some empirical results obtained from a data set made of attack traces collected in the Internet by a set of honeypots during two years. Thanks to the application of our attribution method, we are able to identify several large-scale phenomena composed of IP sources that are linked to the same root cause, which constitute a type of phenomenon that we have called *Misbehaving cloud* (MC). An in-depth analysis of two instances of such clouds demonstrates the utility and meaningfulness of the approach, as well as the kind of insights we can get into the behaviors of malicious sources involved in these clouds.

Keywords

Investigative data mining, attack attribution, threat analysis.

1. INTRODUCTION

There is no real consensus on the definition of “attack attribution” in the cyber domain. If one looks at a general definition of the term *attribution* in a dictionary, he will find something similar to: “to explain by indicating a cause” [Merriam-Webster]. However, we observe that most previous work related to that field tend to use the term “attribution” as a synonym for *traceback*, which consists in “determining the identity or location of an attacker or an attacker’s intermediary” [20]. In the context of a cyber-attack, the obtained identity can refer to a person’s name, an account, an alias, or similar information associated with a person or an organisation. The location may include physical (geographic) location, or any virtual address such as an IP address or Ethernet address. The rationale for developing such attribution techniques is mainly due to the untrusted nature of the IP protocol, in which the source IP address is not authenticated and can thus be easily falsified. An extensive survey of attack attribution techniques used in the context of IP traceback can be found in [20].

In this paper, we refer to “attack attribution” as something quite different from what is described here above, both in terms of techniques and objectives. Although tracing back to an ordinary, isolated hacker is an important issue, we are primarily concerned by larger scale attacks that could be mounted by criminal or underground organizations. For this purpose, we present an analytical method that can help security analysts in determining the plausible root causes of attack phenomena, and in deriving their *modus operandi*.

The method presented hereafter can be applied to a broad range of security data sets, or more generally, to many problems related to investigative data mining. This paper illustrates the application of this method through an empirical analysis of some attacks collected during two years by a set of low interaction honeypots deployed all over the world by the *Leurré.com project* [11]. Regarding this specific dataset, some typical phenomena that we want to identify vary from worm or malware families propagating through code injection attacks [10], to established botnets controlled by the same people and targeting machines in the IP space. As showed in the experimental results, all malicious sources involved in the same root phenomenon seem to form what we call a *Misbehaving Cloud* (MC).

The remainder of this paper is structured as follows: in Section 2, we formally describe the main components of our multi-criteria clustering method, i.e., a graph-based clustering process followed by a multi-criteria decision analysis. Then, Section 3 describes how we have applied this method to a specific dataset made of network attack traces. In Section 4, we detail some experimental results, and we provide a more in-depth study of two instances of misbehaving clouds. Section 5 concludes the paper.

2. MULTI-CRITERIA CLUSTERING

2.1 Introduction

In investigative data mining, an analyst must usually synthesize different pieces of evidence to eventually identify the potential root causes of attack phenomena. The goal is to determine how to “connect the dots”, i.e., how to discover important patterns and how to combine them meaningfully, so as to expose the “big picture” [19]. However, the amount of data of today’s systems used for gathering data far ex-

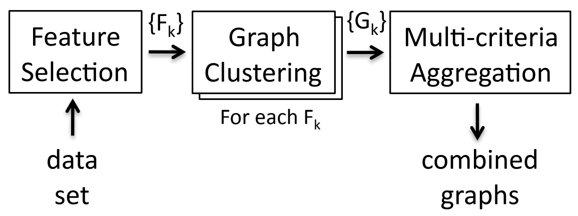

Figure 1: Overview of the multi-criteria clustering method. F_k refers to the set of features selected from a security dataset, and G_k refers to the set of corresponding weighted graphs of relationships resulting from the clustering component. The multi-criteria component combines all G_k so as to produce a combined link graph.

ceeds our capacity to analyze it manually. For this reason, we aim at developing a multi-criteria clustering approach that can systematically discover, extract and combine unknown patterns from a security dataset, according to a set of potentially useful features.

As illustrated in Fig. 1, our approach is based on three components:

1. **Feature selection**: we determine which *features* we want to include in the overall analysis, and we characterize each object of the dataset according to the set of extracted features F_k (e.g., by creating feature vectors for each object);

2. **Graph-based clustering**: an undirected edge-weighted graph is created regarding each feature F_k, based on an appropriate distance for measuring pairwise similarities. Strongly connected components can then easily be identified within each graph, so as to reveal relationships among objects sharing common patterns regarding some features;

3. **Multi-criteria aggregation**: the different graphs are then combined using an *agregation function* that somehow models the expected behavior of the phenomena under study.

The approach is mostly unsupervised, i.e., it does not rely on a preliminary training phase to classify objects to larger scale phenomena. Instead, we have only a data set of unlabeled observations, and we need to learn what patterns are present in the data in function of different characteristics that can hopefully bring some light on the underlying phenomenon.

2.2 Feature selection

In cluster analysis, one of the very first steps consists in selecting some key characteristics from the dataset, i.e., salient features that may reveal meaningful *patterns* [7]. The selection of these features may optionally be completed by a feature extraction process, i.e., one or more transformations of the input to produce features that are more suited to subsequent processing. Pattern representation refers to the number of categories, classes, or variables available for each feature to be used by the clustering algorithm.

More formally, we have thus a dataset $\mathcal{D}$ composed of N data objects. From $\mathcal{D}$, we can select and/or extract n different features so as to create our feature set $F = \{F_k\}, k = 1, \dots, n$. The purpose of this first component consists in creating one set of feature vectors X_k for each F_k, i.e.:

$$X_k = \begin{bmatrix} x_{11}^{(k)} & x_{12}^{(k)} & \cdots & x_{1p}^{(k)} \\ x_{21}^{(k)} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ x_{N1}^{(k)} & \cdots & \cdots & x_{Np}^{(k)} \end{bmatrix}$$

where a row $X_k(i, 1:p) = \mathbf{x}_i^{(k)}$ represents a feature vector for the i^{th} object of $\mathcal{D}$ obtained for the k^{th} feature F_k. The dimensionality of the feature vectors is p, the number of variables (or categories) that have been extracted or defined.

2.3 Graph-based clustering

We now turn to the description of the second component of our method, which implements a pairwise clustering. We formulate the problem of clustering objects from $\mathcal{D}$ using a graph-based approach. That is, for each feature F_k, we can construct a graph G_k in which the vertices (or nodes) are mapped to the feature vectors $\mathbf{x}_i^{(k)}$, and the edges (or links) express some degree of similarity between objects regarding the considered feature. As customary, we can represent the undirected edge-weighted graph (with no self-loops) obtained for a given feature F_k by $G_k = (V_k, E_k, \omega_k)$, where $V_k = \{\mathbf{x}_1^{(k)}, \mathbf{x}_2^{(k)}, \dots, \mathbf{x}_N^{(k)}\}$ is the vertex set, $E_k \subseteq V_k \times V_k$ is the edge set and represents the relationships between each pair of vertices, and $\omega_k : E_k \to \Re^+$ is a positive weight function associated with each edge of E_k.

In practice, we represent each graph G_k with its corresponding weighted *adjacency matrix* (or dissimilarity matrix), which is the $N \times N$ symmetric matrix $A_k(i,j)$ defined as:

$$A_k(i,j) = \begin{cases} \omega_k(i,j), & \forall (i,j) \in E_k \\ 0, & \text{otherwise.} \end{cases}$$

Note that the weight function $\omega_k(i,j)$ must be defined with a similarity metric that is appropriate to the nature of the feature vectors. In fact, there is a wide range of possibilities for chosing a distance, from rather simple ones (e.g., Euclidean, sample correlation, Mahalanobis, Minkowski) to more elaborate functions such as statistical distances (e.g., Kullback-Leibler, Bhattacharyya, etc). We further detail this aspect in the application of the method (Section 3).

Finally, the graph-clustering can be performed by extracting strongly connected components for each graph G_k. To do this, we use an algorithm based on *maximal cliques*, which are induced subgraphs in which the vertices are fully connected (i.e., complete subgraphs). In practice, this means we identify for each feature F_k which subgroups of objects are highly similar and thus share the same pattern regarding F_k. The Maximal Clique Problem (MCP) is a classical combinatorial problem that can be solved using different algorithms. In our method, we take advantage of the *dominant sets* approach of Pavan et al. [14], which generalizes the MCP to the edge-weighted case. Dominant sets (DM) are equivalent to maximum weighted cliques, hence to very coherent clusters. However, finding dominant sets is attractive

Symbol	Description
$\mathcal{D}$	raw dataset
N	Number of objects in $\mathcal{D}$
F	Feature set ($F = \{F_k\}, k = 1, \cdots, n$)
n	Number of features in F
X_k	Set of feature vectors contructed from F_k
$\mathbf{x}_i^{(k)}$	The feature vector of the i^{th} object obtained for F_k
$G_k(V_k, E_k, \omega_k)$	The undirected weighted graph built for F_k
A_k	The weighted adjacency matrix of G_k
$\mathcal{F}$	An aggregation function
$\mathbf{z}_{ij}$, or simply $\mathbf{z}$	A *criteria vector* built for a pair of data objects i, j

Table 1: Table of notations

from a computational viewpoint, since it can be done with a continuous optimization technique that applies replicator dynamics (from evolutionary game theory). As a result, we can find dominant sets by simply making a particular temporal expression converge. More precisely, consider the following dynamical system represented with its discrete time equation, and A_k being the adjacency matrix of G_k:

$$x_i(t+1) = x_i(t) \cdot \frac{(A_k\,\mathbf{x}(t))_i}{\mathbf{x}(t)^T A_k\,\mathbf{x}(t)}$$

with $i = 1, \ldots, N$. Starting from an arbitrary initial state, this dynamical system will eventually be attracted by the nearest asymptotically stable point. As it has been showed in [14], this corresponds to a dominant set, hence to a maximum weighted clique. Then, the DM algorithm will try to find a new dominant set with the remaining vertices of the graph until a stopping criterion is met (e.g., when the sum of the remaining edge weights is less than 0.05).

2.4 Multi-criteria aggregation

Aggregation functions are used in many prototypical situations where we have several criteria of concern, with respect to which we assess different options. The objective consists in calculating a combined score for each option, and this combined output forms then a basis from which decisions can be made. For example, aggregation functions are largely used in problems of *multi criteria decision analysis* (MCDA), in which an alternative has to be chosen based on several, sometimes conflicting criteria. Usually, the alternatives are evaluated from different attributes (or features) that are expressed with numerical values representing a degree of preference, or a degree of membership.

Definition (Aggregation function). An aggregation function is formally defined as a function of n arguments ($n > 1$) that maps the (n-dimensional) unit cube onto the unit interval: $\mathcal{F} : [0,1]^n \longrightarrow [0,1]$, with the following properties [1]:

(i) $\mathcal{F}(\underbrace{0, 0, \ldots, 0}_{n\text{-}times}) = 0 \quad \text{and} \quad \mathcal{F}(\underbrace{1, 1, \ldots, 1}_{n\text{-}times}) = 1$

(ii) $x_i \leq y_i$ for all $i \in \{1, \ldots, n\}$ implies $\mathcal{F}(x_1, \ldots, x_n) \leq \mathcal{F}(y_1, \ldots, y_n)$

In our multi criteria method, we have n different attack features, according to the F_k's, and thus a *vector of criteria* $\mathbf{z}_{ij} \in [0,1]^n$ can be constructed from the similarity weights, such that:

$$\mathbf{z}_{ij} = [A_1(i,j), A_2(i,j), \ldots, A_n(i,j)]$$

with A_k the weighted adjacency matrix of graph G_k corresponding to attack feature F_k, and (i, j) is a pair of objects from the data set $\mathcal{D}$. Our approach consists in combining the n values of each criteria vector $\mathbf{z}_{ij}$ (which reflects the set of all relationships between a pair of data objects), in order to build an aggregated graph $G^* = \sum G_k$.
A rather simplistic approach consists in combining the criteria using a simple arithmetic mean, eventually with different weights assigned to each criteria (i.e., a weighted mean). However, this does not allow us to model more complex relationships, such as "most of", or "at least two" criteria to be satisfied in the overall decision function, and this without having to know *which* set of criteria is more relevant for a given pair of objects. In other words, what we need is an aggregation function in which the combination of criteria of interest (and the associated weights) is not predetermined in a static way.

Ordered Weighted Averaging.
Yager has introduced in [21] a type of operator called *Ordered Weighted Averaging* (OWA), which allows to include certain relationships between criteria in the aggregation process. An OWA operator differs from a classical weighted means in that the weights are not associated with particular inputs, but rather with their *magnitude*. As a result, OWA can emphasize the largest, smallest or mid-range values. It has become very popular in the research community working on fuzzy sets.

Definition (OWA). For a given weighting vector $\mathbf{w}$, $w_i \geq 0$, $\sum w_i = 1$, the OWA aggregation function is defined by:

$$OWA_w(\mathbf{z}) = \sum_{i=1}^{n} w_i z_{(i)} =< \mathbf{w}, \mathbf{z}_{\searrow} >$$

where we use the notation $\mathbf{z}_{\searrow}$ to represent the vector obtained from $\mathbf{z}$ by arranging its components in decreasing order: $z_{(1)} \geq z_{(2)} \geq \ldots \geq z_{(n)}$.

It is easy to see that for any weighting vector w, the result of OWA lies between the classical **and** and **or** operators, which are in fact the two extreme cases when $\mathbf{w} = (0, 0, \ldots, 1)$ (then $OWA_w(\mathbf{z}) = min(\mathbf{z})$) or when $\mathbf{z} = (1, 0, \ldots, 0)$ (then $OWA_w(\mathbf{z}) = max(\mathbf{z})$). Another special case is when all weights $w_i = \frac{1}{n}$, which results in the classical arithmetic mean.

To define the weights w_i to be used in OWA, Yager suggests two possible approaches: either to use some learning mechanism with sample data and a regression model (i.e., fitting weights by using training data and minimizing the least-square residual error), or to give some semantics, or meaning to the w_i's by asking a decision-maker to provide directly those values, based on domain knowledge. In many attribution cases, we shall rely on the latter since the process is mostly unsupervised, i.e., we have no training samples for the phenomena we aim to identify.

Combined graph G^*.
The multicriteria aggregation leads finally to a combined graph G^*, represented by its adjacency matrix A^*, which can be obtained through following operation:

$$A^*(i,j) = OWA_w(\mathbf{z}_{ij}),\ \forall(i,j) \in \mathcal{D}$$

Finally, we can extract the *connected components* from G^* in order to identify all subgraphs in which any two vertices are connected to each other by a certain path:

$$\begin{aligned} \mathcal{P} &= components(A^*) \\ &= \{SG_1, SG_2, \ldots, SG_m\} \end{aligned}$$

which gives us our final set of subgraphs $\mathcal{P}$, where $SG_x \subseteq G^*$, and $\forall(i,j) \in SG_x : OWA_w(\mathbf{z}_{ij}) \geq t$, with $t \in]0,1]$. There exist several algorithms to extract connected components from a graph (depth-first search, breadth-first search, etc). We use here the Dulmage-Mendelsohn decomposition of A^*, which is a lightweight and efficient operation on matrices [6].

Each subgraph can now help the analyst to figure out which root phenomenon could have created the observations within SG_x. As we show with a practical application in Section 4, by analyzing and visualizing the resulting phenomena through the clustering results of their respective features, we can get a global picture of all important relationships among the observations of a same subgraph, and hence we get a better insight into the root cause and the behavior of the underlying phenomenon.

Note that we can optionally apply a thresholding function on A^* in order to eliminate combined edges that could result from an unfortunate linkage between two objects having some weak correlation for a number of features, which means they would otherwise end up in the same subgraph while not related to the same root phenomenon.

3. APPLICATION TO ATTACK TRACES

To illustrate our multi-criteria attribution method, we present a specific application of this method to a set of attack traces collected by honeypots in the Internet.

3.1 Experimental dataset

Our dataset is made of network attack traces collected from a distributed set of sensors (e.g., server honeypots), which are deployed in the context of the *Leurré.com Project* [9; 11]. Since honeypots are systems deployed for the sole purpose of being probed or compromised, any network connection that

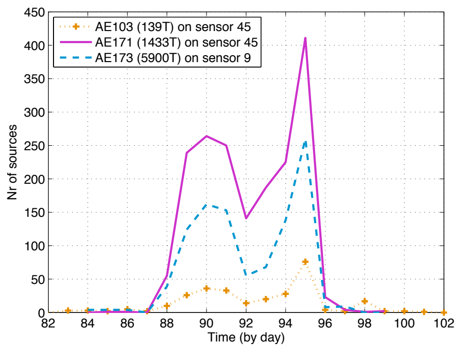

Figure 2: Illustration of a $\mathcal{M}$-event, composed of 3 μ-events that are correlated on 2 different sensors, and are targeting 3 different ports.

they establish with a remote IP can be considered as malicious, or at least suspicious. In Leurré.com, each IP source observed on a sensor is assigned to a so-called *attack cluster* [16] according to its network characteristics, such as the number of IP addresses targeted on the sensor, the number of packets and bytes sent to each honeypot, the attack duration, the average inter-arrival time between packets, the associated port sequence being probed, and the packet payload. Therefore, all IP sources belonging to a given attack cluster have left very similar network traces on a given sensor and consequently, they are considered as having the same *attack profile*. This leads us then to the concept of micro attack event:

Definition (μ-event). A *micro attack event* (or μ-event) refers to a set of IP sources having the same attack profile on a given sensor, and whose coordinated activity has been observed within a specific time window.

Fig. 2 illustrates this notion by representing the time series (i.e., the number of sources per day) of three coordinated μ-events observed on two different sensors in the same time interval, and targeting three different ports. By extension, a *macro event* (or $\mathcal{M}$-event) refers to the set of all μ-events observed over the same time period, and during which the time series are strongly correlated (e.g., the three μ-events in Fig. 2 belong to the same $\mathcal{M}$-event). How to identify such events from the spurious, nonproductive traffic collected by honeypots are issues that have been explained in [15]. For the purpose of this study, our dataset $\mathcal{D}$ comprises 2,454 μ-events that have been observed on 40 different platforms located in 22 different countries, on a period spanning from Sep 2006 until November 2008. Our set of μ-events accounts for a total of 2,538,922 malicious sources, which have been assigned to 320 distinct attack profiles.
In the rest of the paper, we show how we have applied our multi-criteria method to this set of attack events in order to establish connections between them. All μ-events that share enough features constitute a phenomenon that we call a *Misbehaving Cloud* (MC). We hypothesize that malicious sources involved in a MC have a common root cause. By identifying them and studying their global behavior, we hope to get a better insight into the modus operandi and

the strategies of those responsible for them.

3.2 Selection of attack features

The first key features used hereafter deal with the spatial distributions of malicious sources involved in μ-events, in terms of originating countries and IP blocks. Looking at these statistical characteristics may reveal attack activities having a specific distribution of originating countries or IP networks, which can help for instance to confirm the existence of "unclean networks" [3]. Concretely speaking, for each μ-event i, we create a feature vector $\mathbf{x}_i^{geo}$ representing the distribution of countries of origin (as a result of the IP to geolocation mapping), and a vector $\mathbf{x}_i^{sub}$ representing the distribution of IP addresses (grouped by their Class A-prefix, to limit the vector's size to 256 categories).

We have also selected an attack characteristic related to the *targeted platforms*. Looking at which specific platform has observed a μ-event is certainly a pertinent feature. Moreover, we combine this information with the $\mathcal{M}$-event identification, since $\mathcal{M}$-events are composed of μ-events that are strongly correlated in time (which indicates a certain degree of coordination among attackers). This leads to the creation of a feature vector $\mathbf{x}_i^{targ}$ for each μ-event.

Besides the origins and the targets, the *type of activity* performed by the attackers seems also relevant. In fact, malicious software (e.g., worm or bot) is often crafted with a certain number of available exploits targeting a given set of TCP or UDP ports. So, it makes sense to take advantage of similarities between the *sequences of ports* that have been probed or exploited by malicious sources. This gives us a feature vector $\mathbf{x}_i^{ps}$ made of several categories representing the targeted ports for each μ-event.

Finally, we have computed, for each pair of μ-events, the ratio of common IP addresses. We are aware of the fact that, as time passes, some machines of a given botnet (or misbehaving cloud) might be cured while others may get infected (and thus join the cloud). Additionally, due to the dynamic IP allocation of ISP's, certain infected machines can have different IP addresses when we observe them at different moments. Nevertheless, considering the huge size of the IP space, it is still reasonable to expect that two μ-events are probably related to the same root phenomenon when they have a high percentage of IP addresses in common.

To summarize, for each μ-event we define our set of features F as follows:

$$F = \{F_i\}\ ,\ i \in \{geo, sub, targ, ps, cip\}$$

where:

$$\begin{cases} geo & = & \text{geolocation of IP sources;} \\ sub & = & \text{distribution of sources IP addresses;} \\ targ & = & \text{targeted platforms} + \mathcal{M}\text{-event membership;} \\ ps & = & \text{targeted port sequences;} \\ cip & = & \text{ratio of common IP addresses.} \end{cases}$$

3.3 Cluster analysis

Recall that in the second step, we create an undirected weighted graph for each attack feature separately, on which we apply a graph-theroretical clustering algorithm in order to extract maximal cliques. To create the dissimilarity matrix A_k for each graph G_k, quite obviously, we need an appropriate distance function. When we have to deal with observations that are in the form of probability distributions (or frequencies), like in the case of features F_{geo} and F_{sub}, then statistical distances seem more appropriate. One such technique (which is commonly used in information theory) is the Kullback-Leibler divergence [8]. Let p_1 and p_2 be for instance two probability distributions over a discrete space X, then the K-L divergence of p_2 from p_1 is defined as:

$$D_{KL}(p_1||p_2) = \sum_x p_1(x) \log \frac{p_1(x)}{p_2(x)}$$

which is also called the information divergence (or *relative entropy*). Because D_{KL} is not considered as a true metric, it is usually better to use instead the Jensen-Shannon divergence (JSD) [12], defined as:

$$JS(p_1, p_2) = \frac{D_{KL}(p_1||\bar{p}) + D_{KL}(p_2||\bar{p})}{2}$$

where $\bar{p} = (p_1 + p_2)/2$. In other words, the Jensen-Shannon divergence is the *average* of the KL-divergences to the *average distribution.*

Now, to transform pairwise distances d_{ij} to similarity weights sim_{ij}, we still have to define a mapping function. Previous studies found that the similarity between stimuli decay exponentially with some power of the perceptual measure distance [17]. As customary, we can thus use the following functional form to do this transformation:

$$sim(i,j) = exp(\frac{-d_{ij}{}^2}{\sigma^2})$$

where σ is a positive real number that affects the decreasing rate of sim_{ij}.

Measuring pairwise similarities for the other considered features ($F_{targ}, F_{ps}, F_{cip}$) can be done using simpler distance functions, such as the *Jaccard similarity coefficient.* Let s_1 and s_2 be two sample sets (for instance with F_{ps}, s_1 and s_2 are sets of ports that have been probed by sources of two μ-events), then the Jaccard coefficent is defined as the size of the intersection divided by the size of the union of the sample sets, i.e.:

$$JC(i,j) = \frac{|s_1 \bigcap s_2|}{|s_1 \bigcup s_2|}$$

Extracting cliques of attackers.

We applied the unsupervised graph-theoretic clustering algorithm on each dissimilarity matrix A_k, as described in Section 2.3. The goal consists in discovering interesting patterns by extracting all groups of highly similar events. The obtained knowledge will be used to characterize the behavior of global phenomena identified by the multi-criteria component.

Globally, for each feature F_k, the clique algorithm could find on average about 85 clusters accounting for approximatively 70% of the data set, which seems to indicate that many strong relationships exist among μ-events. Note that the dominant set framework is quite attractive, since it does not require a number of clusters as input, and the algorithm will naturally extract the most significant groups in the first stages of the algorithm.

To illustrate this step, we have mapped on Fig. 3 some geographical clusters found for F_{geo}. We have used a dimensionality reduction technique called t-Distributed Stochastic Neighbor Embedding (t-SNE), which aims at converting a high-dimensional dataset into a low-dimensional representation that can be displayed, for example, in a scatter plot.

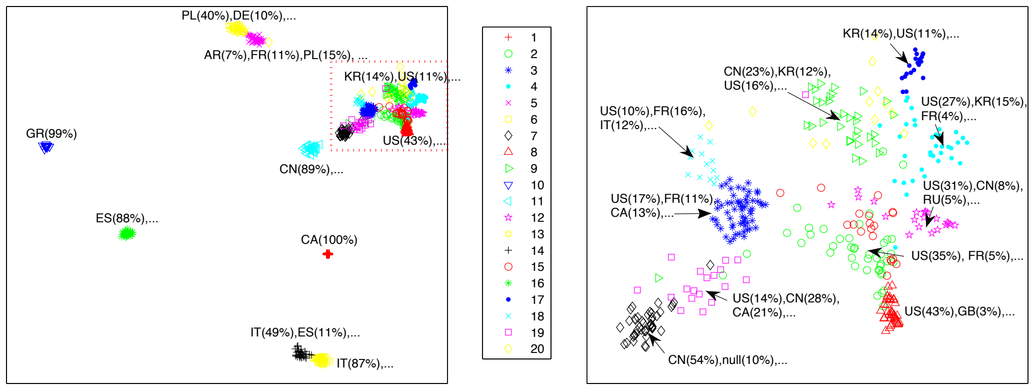

Figure 3: Left: 2D mapping of the 20 largest geographical clusters. Right: a zoom on the zone indicated by the dashed rectangle on the left.

The aim of dimensionality reduction is to preserve as much of the significant structure of the high-dimensional data as possible in the low-dimensional map. This can be helpful to visualize a certain dimension of the data set, but also to assess the consistency of the clustering results. t-SNE [18] is a variation of *Stochastic Neighbour Embedding*; it produces significantly better visualizations than other Multidimensional Scaling techniques (such as Sammon mapping, Isomaps or Laplacian Eigenmaps) by reducing the tendency to crowd points together in the centre of the map.

Figure 3 (Left) shows the resulting two-dimensional plot obtained by mapping the top 20 geographical clusters (encompassing a total of 720 μ-events) on a 2D map using t-SNE. Each datapoint on this map represents the geographical distribution of a μ-event. The coloring refers to the cluster membership of each event, as obtained by applying the clique algorithm, while the text labels indicate the cluster centroïd. We could easily verify that two adjacent events on the map have highly similar geographical distributions, while two distant events have clearly nothing in common in terms of originating countries.

However, as showed in Figure 3 (Right), we observe that several clusters of μ-events originate from some large or popular countries (e.g., US, China, Korea, France, Italy, etc). Although the clique algorithm did perform well, several clusters are overlapping with each other. Not surprisingly, the same kind of issue appeared for other dimensions as well (for instance, some well-known Windows ports are more heavily targeted than others). As a result, it leads to the natural intuition that multiple features need to be aggregated in a consistent manner so as to leverage the results of this knowledge discovery process.

3.4 Combining all features

This last step aims at combining all similarity values for each pair of μ-events (i, j), by applying the OWA_w operator to each criteria vector $\mathbf{z}_{ij}$ constructed from all graphs G_k, with $k \in \{geo, sub, targ, ps, cip\}$. However, we still need to define which values of the weighting vector $\mathbf{w}$ are the most appropriate to model the phenomena under scrutiny. In this case, we hypothesize that attack phenomena such as *misbehaving clouds* (e.g., worms, botnets) may perfectly evolve over time. That is, two consecutive μ-events of the same MC must not necessarily have all their attributes in common. For example, the composition of a botnet may evolve over time because of the cleaning of infected machines or the recruitment of new bots (which leads to a shift in the IP subnet distribution of subsequent events related to this botnet). Or, a botnet may be instructed to scan several consecutive IP subnets in a short interval of time, which leads to the observation of different μ-events having highly similar distributions for the origins, but those events target completely different sensors, and may eventually use different exploits (hence, targeting different ports).

Based on this domain knowledge, we have thus defined a weighting vector $\mathbf{w} = (0.1, 0.35, 0.35, 0.1, 0.1)$. It can be interpreted as: *at least three criteria must be satisfied, but the first criteria is of less importance* (because the first correlated feature between two μ-events might be due to chance only). These weights must be carefully chosen in order to avoid an unfortunate linkage between μ-events when, for example, two μ-events involve IP sources originating from popular countries (typ. US, China, Korea, Germany, etc), and are targeting common (Windows) ports in the same interval of time; but in reality, those μ-events are not necessarily linked to the same phenomenon. By considering different worst-case scenarios, we verified that this weighting vector $\mathbf{w}$ minimizes the final output value in such undesirable cases.

4. BEHAVIORAL ANALYSIS OF ATTACK PHENOMENA

4.1 Overview

We now turn to the description of the experimental results obtained from the application of the multi-criteria method on a 2-year dataset of honeypot traces, as described in previous Section. Starting from the 2,454 μ-events, the method has identified a total of 83 Misbehaving Clouds (MCs), which correspond to 1,607 μ-events, and 506,835 attacking sources. Singletons and MC's containing less than 3 μ-events have been discarded for further analysis. The phenomena involve almost all common services such as NetBios

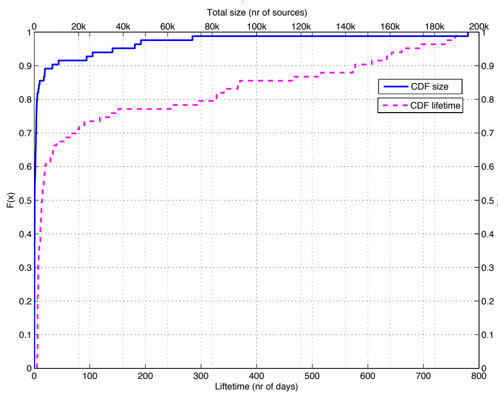

Figure 4: Empirical CDF of MC lifetime and size.

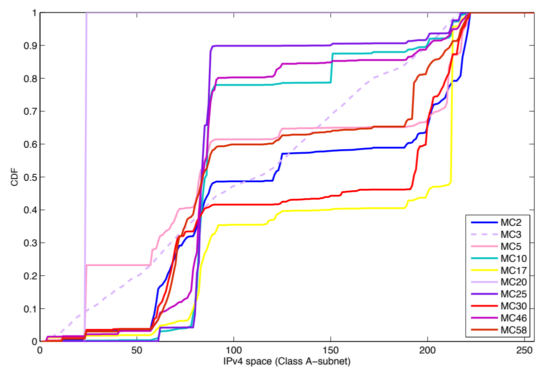

Figure 5: CDF of IP addresses involved in the ten largest MC's.

(ports 139/TCP, 445/TCP), Windows DCOM Service (port 135/TCP), Virtual Network Computing (port 5900/TCP), Microsoft SQL Server (port 1433/TCP), Windows Messenger Service (ports 1025-1028/UDP), Symantec Agent (port 2967/TCP), and some others. Figure 4 shows the cumulative distribution of μ-events per MC. As we can see, in most cases, the MCs contain rather few μ-events and sources. However, around 10% of MCs contain at least 20 thousand observable[1] sources, and some even contain up to 200 thousand sources (spread over 300 μ-events or more). Regarding the *lifetime* of these MC's (i.e., the time interval, in days, between the very first and the very last attack event), about 67% of MCs exist during less than 50 days but around 22% of them last for more than 200 days. Another global characteristic (not represented on the Fig.) is that, in 94% of the cases, the MCs are seen on less than 10 platforms.

These various characteristics suggest that the root causes behind the existence of these MCs are fairly stable, localised attack processes. In other words, different places of the world do observe different kind of attackers but their modus operandi remain stable over a long period of time. We are, apparently, not that good at stopping them from misbehaving.

Regarding the origins of MC's, we observe some very persistent groups of IP subnets and countries of origin. On Fig. 5, we have represented the CDF of the IP addresses involved in the ten largest MC's, where the x-axis represents the first byte of the IPv4 address space. Clearly, malicious sources involved in those phenomena are highly unevenly distributed, and form a relatively small number of tight clusters that are responsible for a large deal of the observed malicious activities. This is consistent with other prior work on monitoring global malicious activities, in particular with previous studies related to measurements of *Internet background radiation* [2; 13; 22]. However, we can show here that there are still some notable differences in the spatial distributions of those misbehaving clouds, even though there is a large overlap between "zombie-friendly" IP subnets. Moreover, because of the dynamics of this kind of phenomenon, we can even observe different spatial distributions within the *same cloud* at different moments of its lifetime. This is an advantage of our analysis method that is more precise and enables us to distinguish *individual* phenomena, instead of global trends, and even to observe their dynamic behavior.

Another interesting observation on Fig. 5 is the CDF of MC3 (uniformly distributed in the IPv4 space, which means randomly chosen source addresses) and MC20 (a constant distribution coming exclusively from the subnet 24.0.0.0/8). A likely explanation is that those MC's have used spoofed addresses to send UDP spam messages to the Windows Messenger service. So, this indicates that IP spoofing is still possible under the current state of filtering policies implemented by certain ISP's on the Internet. We refer the interested reader to [5] in which we provide a more in-depth analysis of this interesting UDP spam phenomenon.

4.2 Some detailed examples

Finally, we further detail two case studies from Table 2 to illustrate some typical behaviors we could observe among the misbehaving clouds identified so far, e.g.:

i) a move (or drift) in the origins of certain MC's (both geographical and IP blocks) during their lifetime;

ii) a scan sweep by the same cloud, targeting several consecutive class A-subnets;

iii) within the same cloud, multiple changes in the port sequences (i.e., exploits) used by machines to attack;

iv) a higher-level coordination among attackers of the same cloud.

Botnet Cloud.

An more in-depth analysis of MC28, in particular the shape of the time series of μ-events and the arrival rate of the sources, has led us to conjecture that MC28 is quite likely due to a botnet phenomenon. What is of interest here is that it has showed the behaviors *i)* and *iv)*. On Fig. 6, we can see this cloud had four main waves of activity during which it was randomly scanning 5 different subnets (note the perfect coordination among the time series). When inspecting the subnet distributions of the different attack waves, we could clearly observe a drift in the origins of those

[1]It is important to note that the sizes of the phenomena given here only reflect the number of sources we could *observe* on our sensors; the actual sizes of those armies are most probably much larger, even though some churn effects (DHCP, NAT) could also affect these numbers.

Table 2: High-level characteristics of two examples of *Misbehaving Clouds.* The colon *Root cause* refers to the presumed type of phenomenon, based on the results of the attack attribution method.

MC Id	Nr Events	Nr Sources	Duration	Root cause	Targeted ports
2	122	45,261	741	Worm-behaving cloud	1433T (MSSQL), 1025T (RPC), 139T (Netbios), 5900T (VNC), 2967T (Symantec)
10	202	71,157	577	Botnet cloud	ICMP, 445T (Microsoft-DS), 139T (Netbios), 80T (Web)

sources, probably as certain machines were infected by (resp. cleaned from) the bot software. Then, we found also that a smaller subset of μ-events in this MC were involving machines that directly attacked the Windows ports (without scanning them). However, this group of attacking zombies had quite different origins from those of the scanners, and seem to be ordered to attack only specific IP addresses on our sensors (i.e., the Windows honeypots). We conjecture that the attackers probably took advantage of the results given by the larger set of scanners. In Annex 1, we provide a graph visualization of this misbehaving cloud to further illustrate its behavior (i.e., the separation of duties between scanners and attackers, and the drift in the origins of the sources that leads to multiple geographical clusters).

Worm-behaving Cloud.
MC2 is an interesting case in which we can observe the behaviors *ii)* and *iii)*. It consists of 122 μ-attack events that have a shape which is fairly similar to the one left by a network worm: its trace exists for several days, it has a small amplitude at the beginning but grows quickly, exhibits important drops that can correspond to subnets being cured or blacklisted, and it eventually dies slowly [15].
The lifetime of this MC was fairly long (about 741 days!). It is composed of μ-events that have targeted a number of distinct services, including 1025T, 135T, 139T, 1433T, 2967T and 5900T. The results of the multi-criteria fusion algorithm indicate that those μ-events have been grouped together mainly because of the following three features: geographical location, targeted platform, and ports sequence. Moreover, a detailed analysis reveals that an important amount of IP addresses is shared by many μ-events composing this MC. A node-link graph is provided in Annex 1 to visualize the rather complex network structure formed by the μ-events of this worm-behaving cloud, in which we can observe its highly dynamic behavior. The cloud has been scanning (at least) four consecutive class A-subnets during its lifetime, while probing at the same time several ports on these subnetworks. It is not excluded that all these attacks could be due to two or three distinct worms (and thus, different groups of people). However, this result indicates that the same core piece of code has probably been reused, from a very similar starting point to launch a number of distinct attacks, which is an important piece of information for those who are in charge of identifying those misbehaving groups and their modus operandi.

5. CONCLUSIONS

We have presented a generic and systematic method to address the complex problem related to "attack attribution". Our approach relies on a novel combination of a graph-based clustering analysis and a multi-criteria aggregation process. We have applied our technique to 2 years of attack traces collected by 40 honeypots located all over the world, which

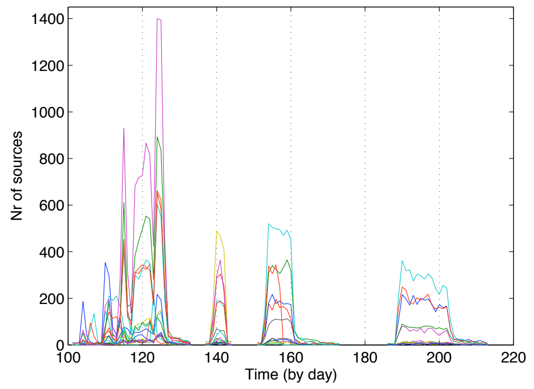

Figure 6: Time series of coordinated μ-events for MC28.

has delivered some interesting results showing the utility and the meaningfulness of this approach. It is worth noting that the method could as easily be applied on completely different threats-related events. In fact, the interim Symantec report published mid October 2009 on the analysis of Rogue Security Software [4] offers results of the application of this very same method to the problem of understanding the modus operandi of malicious users setting up Rogue AV campaigns.
It is our hope that people will be interested in trying to understand the rationales behind the *Misbehaving Clouds* we have identified. We are eager to share as much information as possible with such interested parties. Similarly, we are looking forward in having other opportunities to apply this method to other security datasets that future partners would be willing to share with us.

Acknowledgments

This work has been partially supported by the European Commission through project FP7-ICT-216026-WOMBAT funded by the 7th framework program. The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the European Commission.

6. REFERENCES

[1] G. Beliakov, A. Pradera, and T. Calvo. *Aggregation Functions: A Guide for Practitioners*. Springer, Berlin, New York, 2007.

[2] Zesheng Chen, Chuanyi Ji, and Paul Barford. Spatial-temporal characteristics of internet malicious sources. In *Proceedings of INFOCOM*, 2008.

[3] M. P. Collins, T. J. Shimeall, S. Faber, J. Janies, R. Weaver, M. De Shon, and J. Kadane. Using uncleanliness to predict future botnet addresses. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 93–104, New York, NY, USA, 2007. ACM.

[4] Symantec Corporation. Symantec Report on Rogue Security Software, http://www.symantec.com/business/theme.jsp ?themeid=threatreport [oct 2009].

[5] M. Dacier, V. Pham, and O. Thonnard. The WOMBAT Attack Attribution method: some results. In *5th International Conference on Information Systems Security (ICISS 2009), 14-18 December 2009, Kolkata, India*, Dec 2009.

[6] Timothy A. Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[7] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall advanced reference series, 1988.

[8] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics 22: 79-86.*, 1951.

[9] C. Leita, V.H. Pham, O. Thonnard, E. Ramirez-Silva, F. Pouget, E. Kirda, and Dacier M. The Leurre.com Project: Collecting Internet Threats Information Using a Worldwide Distributed Honeynet. In *Proceedings of the WOMBAT Workshop on Information Security Threats Data Collection and Sharing, WISTDCS 2008*. IEEE Computer Society press, April 2008.

[10] Corrado Leita and Marc Dacier. Sgnet: a worldwide deployable framework to support the analysis of malware threat models. In *Proceedings of the 7th European Dependable Computing Conference (EDCC 2008)*, May 2008.

[11] Leurre.com, Eurecom Honeypot Project. http://www.leurrecom.org/, [[s]ep 2009].

[12] J. Lin. Divergence measures based on the shannon entropy. *Information Theory, IEEE Transactions on*, 37(1):145–151, Jan 1991.

[13] Ruoming Pang, Vinod Yegneswaran, Paul Barford, Vern Paxson, and Larry Peterson. Characteristics of internet background radiation. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 27–40, New York, NY, USA, 2004. ACM.

[14] M. Pavan and M. Pelillo. A new graph-theoretic approach to clustering and segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[15] Van-Hau Pham. *Honeypot traces forensics by means of attack event identification*. PhD thesis, TELECOM ParisTech, 2009.

[16] F. Pouget and M. Dacier. Honeypot-based forensics. In *AusCERT2004, AusCERT Asia Pacific Information technology Security Conference 2004, 23rd - 27th May 2004, Brisbane, Australia*, 2004.

[17] Roger N. Shepard. Multidimensional scaling, tree fitting, and clustering. *Science*, 210:390–398, 1980.

[18] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.

[19] C. Westphal. *Data Mining for Intelligence, Fraud & Criminal Detection: Advanced Analytics & Information Sharing Technologies*. CRC Press, 1st edition (December 22, 2008), 2008.

[20] D. Wheeler and G. Larsen. Techniques for Cyber Attack Attribution. *Institute for Defense Analyses, Oct 2003*, 2008.

[21] Ronald R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Trans. Syst. Man Cybern.*, 18(1):183–190, 1988.

[22] Vinod Yegneswaran, Paul Barford, and Johannes Ullrich. Internet intrusions: global characteristics and prevalence. In *SIGMETRICS*, pages 138–147, 2003.

Appendix: Visualization of Misbehaving Clouds through Node-link graphs with multiple features.

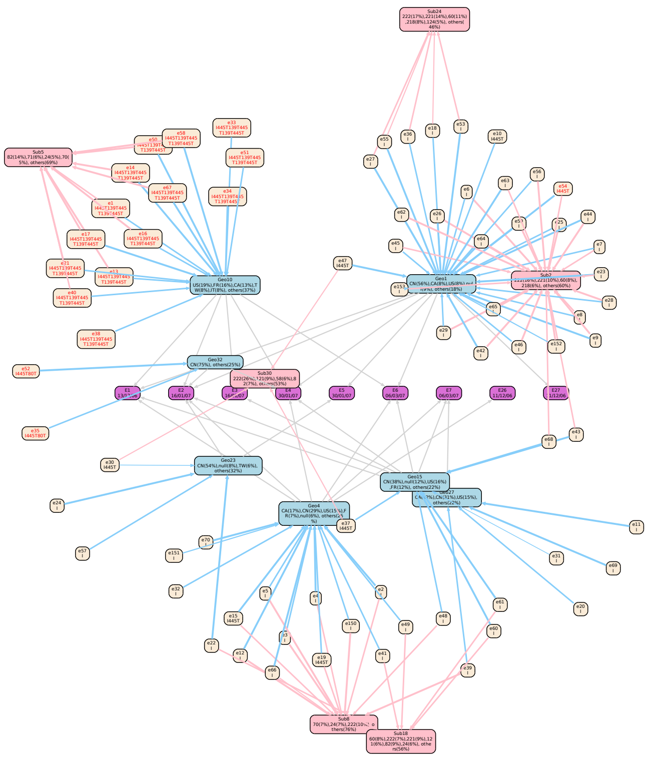

Figure 7: Node-link Graph of MC28 representing relationships between attack events (in beige). Following features are depicted: F_{geo} (in blue), F_{sub} (in pink), F_{ps} (in the labels of the events), and a time axis of events (in purple). In this botnet cloud, attack events involving machines attacking or exploiting directly the Windows ports are highlighted with red labels.

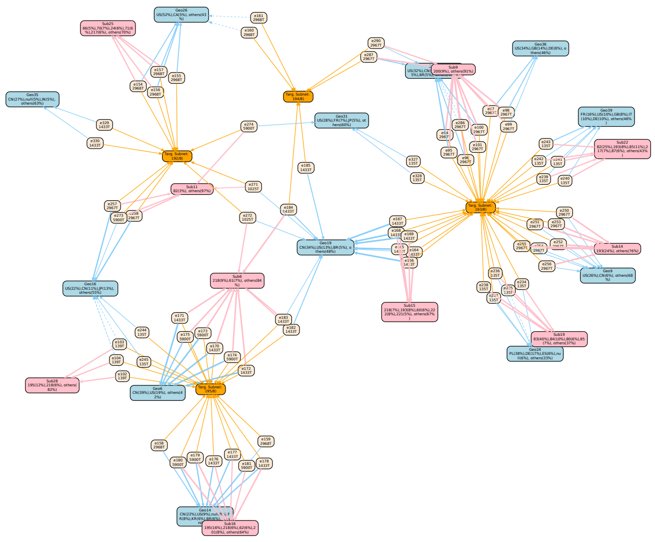

Figure 8: Node-link Graph of MC2 representing relationships for F_{geo}, F_{sub}, F_{targ} and F_{ps}.