VEGAS: Visualizing, Exploring and Grouping AlertS

Damien Crémilleux
CentraleSupélec and INSA

Frédéric Majorczyk
DGA-MI and CentraleSupélec

Nicolas Prigent
CentraleSupélec

Abstract

IDS are well known to raise large quantities of alerts, many of them being false-positive. VEGAS is an intuitive visualization tool that allows to manage large quantities of alerts by grouping similar ones easily and dispatching these groups of alerts among security operators for further analysis. Once a group of alerts has been identified, any forthcoming alert that should belong to that group will be forwarded automatically to the operator in charge of the group for further analysis. Therefore, VEGAS reduces the amount of alerts that are received by the operator in charge of dispatching them and makes the flow of alerts more manageable.

Specifically, VEGAS proposes visual correlation of alerts based on principal component analysis (PCA), assisted creation of dispatching rules for alerts and automated dispatch of incoming alerts according to their features.

Keywords: Visualization, Intrusion Detection, CyberSecurity, PCA, Workflow, Teamwork, Collaboration

1 SNORT alerts as a data source

VEGAS uses SNORT (`https://www.snort.org/`) alerts as a data source. A SNORT alert is made of several fields that describe the packet that was identified as malicious: The *text description of the alert*, the *classification* that indicates the broad category the alert belongs to, the *priority* that describes the level of criticity of the alert, the *timestamp* that defines when the event occurred, the *source IP* and the *source port*, the *destination IP* and the *destination port*, and other arguably less important fields of the packets such as the *TCP sequence number*. These fields identified as less important are discarded in VEGAS. Even if they can be used during a forensic analysis, we believe that they are less useful for grouping and sorting alerts.

2 Actors and Workflow

VEGAS supposes two types of actors. First, the *front-line security operator* is in charge of receiving the raw alerts and quickly dispatching them using the interface that we describe in Section 3. *Security analysts* belong to the second type of actors. They analyze in details each group of alerts transmitted by the front-line security operator. In classical contexts, a front-line security operator generally dispatches alerts to a few security analysts.

Figure 1 summarizes the workflow in VEGAS. Alerts generated by the IDS are transmitted trough the network to a filter that dispatches them. Originally, the filter only has the *default* rule that sends all the alerts to the front-line security operator for display. When the front-line security operator identifies a new group of alerts, he or she performs a quick analysis of it, annotates it and adds a new rule to the filter so as to redirect these alerts to a new bucket to be analyzed in depth by a security analyst. From this moment, the alerts

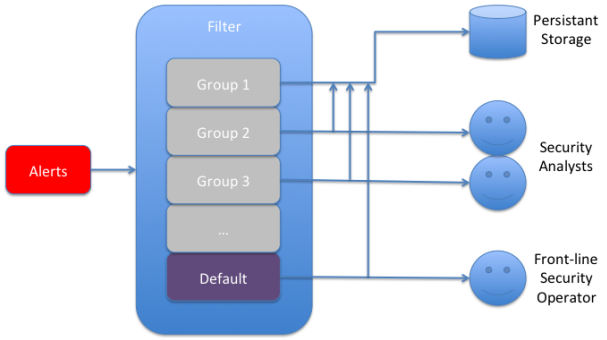

Figure 1: VEGAS workflow.

that have been identified and grouped by the front-line security operator disappear from his or her interface and are sent directly to the bucket. Forthcoming alerts matched by the rule are also sent directly to the bucket and won't be displayed on the interface of the front-line security operator anymore.

Each bucket is then assigned to a given security analyst for further investigation. The security analyst can then analyze the alerts that match the rule of the bucket, refine filtering rules if necessary and perform the relevant corrective actions. When this is done, alerts belonging to the bucket are no more symptomatic of a real threat, and the rule is modified to redirect already received and forthcoming alerts directly to a persistent storage in case they would be useful for forensics purpose.

3 Visual representation of alerts

As stated earlier, the main objective of VEGAS is to propose an efficient way to manage the flow of alerts. Representing the numerous fields/dimensions of many SNORT alerts on a single easy to understand representation would be very difficult, if not impossible. Therefore, VEGAS performs dimension reduction to offer the front-line security operator a simple enough representation to detect similar alerts. VEGAS uses Principal Component Analysis (PCA) [2] that has proven efficient for dimension reduction. To fit with PCA, categorical variables are transformed into numeric ones using the *dummy variable creation* technique: For each category, a new variable is created, and elements belonging to this category take the value "1" for the new dummy variable, else "0". Thanks to this technique, alerts are only made of numeric values and can be used as an input for the principal component analysis.

The outcome of the PCA is the projection of our dataset onto a smaller subspace made of two dimensions that is rep-

resentative of the original dataset. While these two new dimensions have no "real" semantic, they are the composition of the "most meaningful" dimensions in the original dataset. Therefore, in these two dimensions, alerts that are close in the original dimensions are still close in the newly computed ones while alerts that are distant in the original dimensions are distant in the newly computed ones.

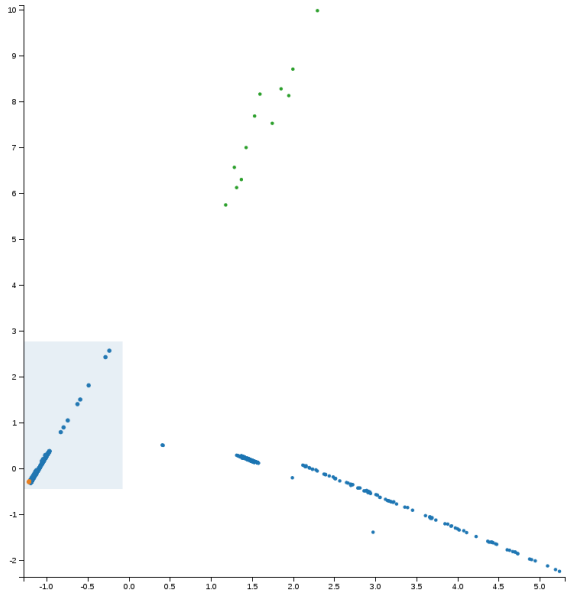

Figure 2: Alerts displayed as a scatterplot.

The two computed dimensions are then presented to the front-line security operator on a scatterplot. For instance, Fig. 2 shows the scatterplot generated based on 2000 alerts comming for the VAST Challenge 2012 dataset (`http://vacommunity.org/VAST+Challenge+2012`).

The front-line security operator can then use brushing to select alerts that visually appear to belong to the same group. Bar charts located underneath are then automatically updated to reflect the various values exhibited for the various fields (*Alert classification*, *Alerts by source IP*, *Alerts by destination IP*, *Alerts by source port* and *Alerts by destination port*) for the selected alerts (see. Fig 3). Bar charts have proven very efficient to represent categorical fields when that can take numerous differents values [1].

The front line security operator can then go through the bar charts representing the values exhibited by the alerts for each field to better understand the features shared by the selected alerts. Especially, he or she can verify that this group of alerts makes sense. Using his or her expertise and knowledge of the context, he or she can also inspect what fields are relevant in the group of alerts. For instance, on Fig. 3, the *Alerts by destination IP* and *Alerts by destination port* each exhibit only one value. In this case, this is due to the fact that the group of attacks are directed towards the DNS service (port 53) of a specific server (172.23.0.10). Therefore, in this group of attacks, these two fields are specifically important. Using VEGAS, the analyst therefore selects these two fields, adds a comment describing his or her understanding of the situation and clicks on a "Generate rule" link to generate a new rule to be inserted in the dispatching filter. All the selected values of the selected fields for the selected alerts are put in a dictionary and the rule is added to the filter. All the alerts matched by the filter are immediately sent to

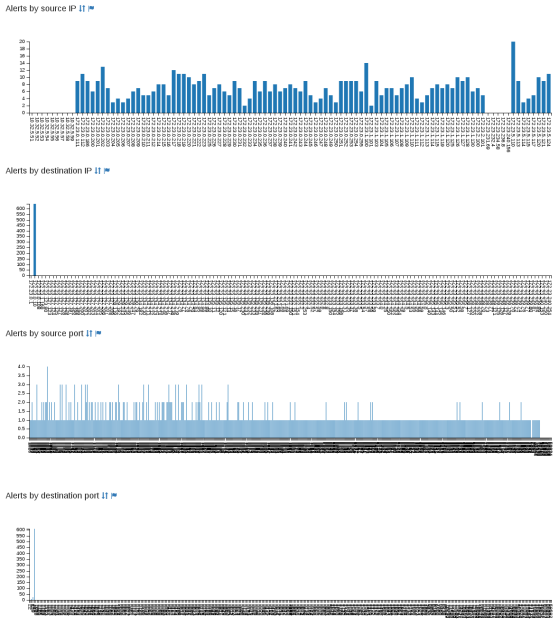

Figure 3: Values exhibited for the various fields by selected alerts.

the newly added bucket according to this new filtering rule, including both the alerts that were already displayed and the forthcoming alerts.

4 Implementation and future work

VEGAS is implemented as two distinct parts, a server and a client. On the server side, IDS alerts and rules are stored into Elasticsearch (`https://www.elastic.co/`). Logstash (`https://www.elastic.co/products/logstash`) is used to parse SNORT alerts. We developed a specific plugin to filter incoming alerts and tag them with the matching rules so as to dispatch alerts properly. PCA is computed with R (`https://www.r-project.org/`) using the FactoMineR package. The client side of VEGAS is implemented using web technologies: HTML5, Javascript, CSS and SVG. Charts and interactions are built using D3.js(`http://d3js.org/`) and Crossfilter.

Early experiments on the VAST Challenge 2012 dataset have produced encouraging results. We now need to perform supplementary tests on many different datasets to evaluate VEGAS versatility. We also need to evaluate the side effects of filters composition. Currently, dispatching rules that are too generic could cover less generic dispatching rules that should normally be evaluated later in the filter, preventing them to be be ever evaluated in practice. To solve this issue, we intend to find inspiration in the ways coverage is evaluated in firewall rules.

References

[1] Christopher Humphries, Nicolas Prigent, Christophe Bidan, and Frédéric Majorczyk. CORGI: Combination, organization and reconstruction through graphical interactions. In *Proceedings of the Eleventh Workshop on Visualization for Cyber Security*, VizSec '14, pages 57–64. ACM, 2014.

[2] I.T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2002.