How Taxonomy and Aggregation Help an Anomaly-based Correlation Engine

Alexie E. Ballon[1], Raven Hansel N. Ching[1], Joy Luville S. Mahinay[1],
Wyn Cuthbert C. Manlim[1], and Isaac Herculano S. Sabas[2]
[1] De La Salle University
[2] Pandora Security Labs
alexie.ballon@gmail.com; ravenching@gmail.com; luville_mahinay@yahoo.com;
wyn_manlim@yahoo.com; isabas@pandoralabs.net

Abstract: Intrusion Detection System (IDS), Antivirus, Antispyware and other security products are created to solve a specific type of problem and tend to be limited to the area of their focus. Security Information and Event Management (SIEM) is then created to increase threat detection by combining all the events generated by the security products and correlate the events to detect if an attack is occurring. The core of a SIEM is the Correlation Engine (CE) which is the one that processes events and tells the system if there is an attack occurring. Traditional SIEM utilizes rule-based correlation engine which suffer from limited scope of detection. This approach relies on the skills of the rule writer in order to effectively detect threats. Being rule bounded, this approach is not able to detect evolving threats outside the defined rules and therefore has a limited scope of detection. As a result, an anomaly-based system called "Hercules" is developed. There are vital modules in the anomaly-based correlation engine that solve the problem of having a limited scope of detection. For Hercules, these are the Taxonomy and Aggregation modules which handle the categorization and aggregation of grouped events. The Taxonomy Module produces a taxonomy name which contains the Common Weakness Enumeration (CWE) ID that is used by the Aggregation Module to group events together. This paper focuses on Snort IDS sensor only and analyzes whether or not CWE is feasible for Hercules. Results from the study show that there is a limited amount of CWE data associated with Snort and that using other tags within the Snort Community Rules would be a better way of grouping events. For the taxonomy and aggregation module, grouped events that were taxonomized utilizes the childOf tag, which is the parent CWE ID of the current CWE, to combine different events and helped the anomaly based event correlation engine to correlate events.

Key Words: Correlation Engine, Event Correlation, Anomaly, Common Vulnerabilities and Exposures, Taxonomy

1. INTRODUCTION

With the innovation of technology today, many threats are continuously emerging leading to the need to improve security. Part of the security process is monitoring where security devices or software are used to analyze and check the network for possible threats to prevent the network from being compromised.

Companies require security devices to defend themselves from threats and protect their assets. Some of the more popular security devices are the Intrusion Detection System (IDS) and Intrusion Prevention System (IPS). (Sourcefire, n.d.) However, having these devices is not enough to secure the network. The introduction of the Security Information and Event Management (SIEM) can make use of these devices to further improve the security of a network. With SIEM, false positives from IDS can be lessened by making sure that the other sensors from the network have also received a related alert like that produced by an IDS before concluding that there really is an alert. (Chapple, n.d.) Unlike IDS, SIEM monitors by analyzing logs from different sensors instead of just analyzing packets.

SIEM is a hybrid of Security Information Management (SIM) and Security Event Management SEM (Kibirkstis, 2009). SIM is mainly responsible for producing reports while SEM which contains the correlation engine is responsible for correlating data and detecting attacks. Like IDS, SIEM's correlation engine has two different detection types which are the rule-based and the anomaly-based detection. An example of a rule-based correlation engine is the Simple Event Correlation (SEC)

SEC is a rule-based correlation engine which makes use of rules or directives to correlate and detect an attack. (Vaarandi & Grimaila, 2012) However, the main disadvantage for rule-based correlation engines is its inability of detecting new or variation of common attacks.With the aforementioned problem, an anomaly-based correlation engine called "Hercules" is developed. Having an anomaly-based correlation engine allows detection of unknown attacks which therefore widens the scope of rule-based correlation engines in detecting attacks. "Hercules" solves this limitation; however, because of the users' and networks' uncertain behavior, this may lead to a great number of false positives (Du, 2006).

2. ANOMALY-BASED EVENT CORRELATION ENGINE

"Hercules" is an anomaly-based event correlation engine which detects attacks by applying an anomaly-based detection methodology. It is divided into 4 modules, namely Monitoring, Taxonomy, Aggregation and Risk Score Calculation Modules. The Monitoring Module keeps track of grouped standardized logs which are called monitored sessions. The Taxonomy Module is responsible for categorizing monitored sessions. Once the monitored sessions are categorized, they can now proceed to the Aggregation Module. The Aggregation Module is responsible for creating and keeping track of grouped or aggregated monitored sessions. Despite that, it is still possible for a monitored session not to be aggregated in this module if it did not find any correlation from any of the existing monitored sessions. After the Aggregation Module, the monitored session, whether aggregated or not, is passed to the Risk Score Calculation Module.

2.1 Taxonomy Module

The Taxonomy Module creates the taxonomy name for a certain monitored session. It makes use of a community vulnerability dictionary called Common Vulnerability Exposure (CVE) to acquire the necessary information to come up with a taxonomy name. The taxonomy name is used by the Aggregation and Risk Score Calculation Modules. The Aggregation Module to be able to relate and combine monitored sessions for better attack detection. The Security Analyst also utilizes the taxonomy name to give a better evaluation of what is happening in the system.

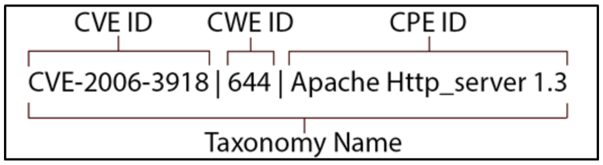


Figure 1: Taxonomy Format

Figure 1 shows the format of a taxonomy name with the pipeline (|) as delimiter for each standard and the tilde (~) as delimiter for the CPE name if more than one.

There are three (3) community standards that the taxonomy name contains:

Standard 1: Common Vulnerability Exposure (CVE)

As mentioned above, the CVE is used to acquire the other necessary information to build the taxonomy name. The CVE is composed of different information security vulnerability and exposures (MITRE Corporation, n.d.). It is used mainly by the Risk Score Calculation module for scoring the monitored sessions. Each CVE ID has its own equivalent CVE score or "threat level" which ranges from 0 – 10; with 10 being the highest. For every taxonomy name, there is only one CVE ID.

Standard 2: Common Weakness Enumeration (CWE)

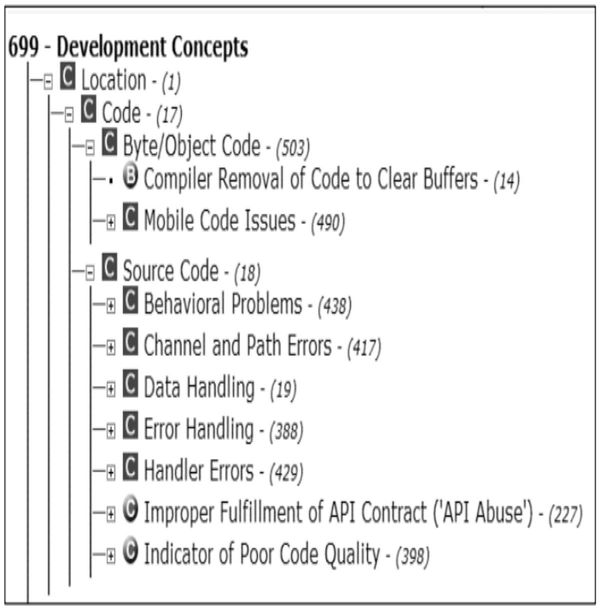

699 - Development Concepts
Location - (1)
Code - (17)
Byte/Object Code - (503)
Compiler Removal of Code to Clear Buffers - (14)
Mobile Code Issues - (490)
Source Code - (18)
Behavioral Problems - (438)
Channel and Path Errors - (417)
Data Handling - (19)
Error Handling - (388)
Handler Errors - (429)
Improper Fulfillment of API Contract ('API Abuse') - (227)
Indicator of Poor Code Quality - (398)

Figure 2: CWE Structure

The CWE is a set of software security weaknesses in a hierarchy (Refer to Figure 2) (MITRE Corporation, n.d.). Each CWE ID has different tags to relate it with other software security weaknesses. Some of the important tags it has are the childOf and parentOf tags which are the CWE IDs of the parent and children of a certain CWE ID. The Aggregation Module uses the CWE IDs for aggregating monitored sessions. The CWE is important for the implementation of the correlation engine because of three main reasons; first, it makes it easier for the analyst to evaluate events; second, it allows the analyst to see the bigger picture in a sense that multiple small events, when combined, creates a bigger threat and lastly, it enables more accurate scoring because of its CWE tag-matching. In the taxonomy name, there is only one CWE ID.

Standard 3: Common Platform Enumeration (CPE)

The CPE is a list of products, applications, hardware devices that follow a structured naming scheme (MITRE Corporation, n.d.). Like the CWE, the CPE is also acquired through the CVE ID taken from the sensor logs. Unlike CWE, each CVE entry can have more than one CPE names. The Risk Score Calculation Module uses the CPE names to score the monitored sessions. Since there can be more than one CPE name for every CVE ID, the taxonomy name segregates the CPE names by using the tilde (~).

2.2 Aggregation Module

The Aggregation Module is where the relation and combining of related monitored sessions happen. This module uses the taxonomy name to get the parent of the CWE ID. Based on the knowledge of the childOf tag of a CWE ID, this module looks for other monitored sessions with the same childOf tag as the given monitored session's CWE ID. If there is a match, the module then combines the two related monitored sessions. If not, the given monitored session will not be updated.

There are two conditions that this module takes into consideration before aggregating monitored sessions. In the first condition, the module must utilize the CWE tags available to relate the monitored sessions. In the second condition, the two candidate monitored sessions should both fit in the same time frame in order for them to be finally aggregated. The time frame for this condition is configured by the security analyst. The initial condition must be met before proceeding with the next condition.

There are two scenarios to consider with the first condition. The first scenario uses the childOf tag which is the parent node of a given CWE ID. The module checks if the new monitored session has the same parent with a previous monitored session. If there are monitored sessions with the same parent, they become candidates for aggregation which would then lead to the next condition.

The second scenario utilizes the parentOf tag which is the child node of a given CWE ID. The module checks if the new monitored session has any children or child nodes and finds a match with a previous monitored session. This is good because it can still be used to relate monitored sessions. Although, it is illogical to use because of its inefficiency in terms of obtaining the parentOf and childOf tags. When using this method, it applies a bubble sort algorithm which is not proposed in large collections because of its worst case performance.

The first one is the desirable scenario because it satisfies one of the goals of the anomaly-based correlation engine which is to detect the variation of common attacks or multi-stage attacks. This method is also more beneficial since it is less complicated and more efficient than the second scenario.

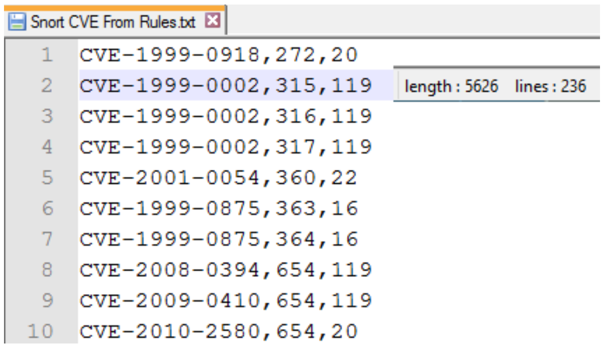
3. TESTING

The current implementation of rule-based correlation engines, allow detection of known attacks. With the evolution of attacks today, the current implementation will not be able to detect the variations of common attacks. To address this problem, the Hercules system develops an anomaly-based event correlation engine. The Taxonomy and Aggregation Modules allow the Hercules to detect multi-stage attacks or variation of common attacks. The focus of the experiment discussed in this section is on the usefulness of the taxonomy and aggregation modules in the anomaly-based correlation engine. Through the tests in this section, it can be verified if the anomaly-based correlation engine can successfully elevate alerts or scores.

This experiment was ran on a Windows environment where the command line was used to run the Perl scripts. XAMPP MySQL Server and MySQL workbench was used in conjunction to add or edit the database contents.

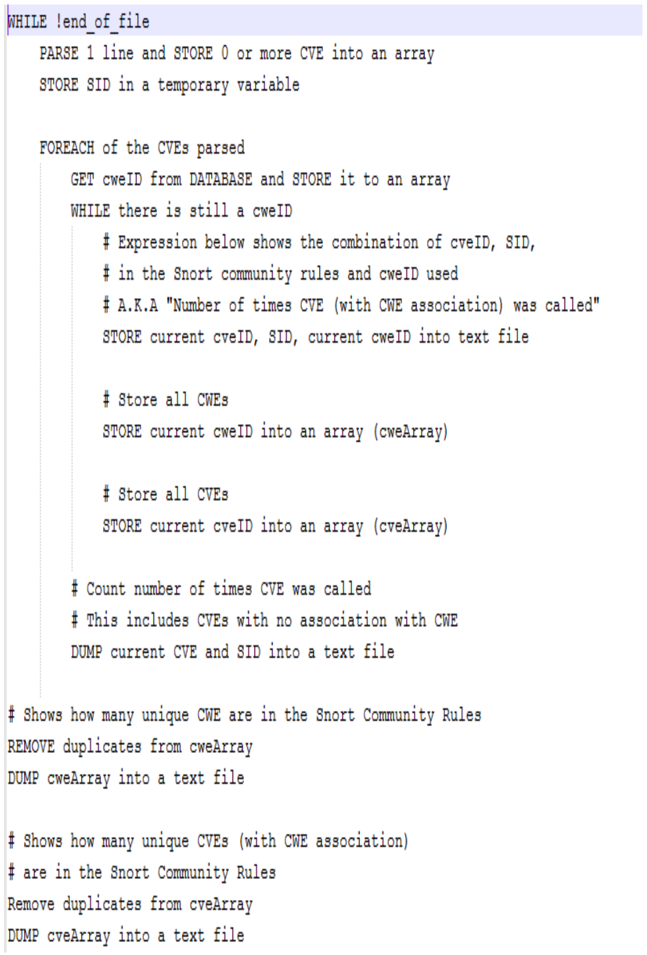
Since the experiment is focused on the Taxonomy and Aggregation Modules' capability, the standardized logs are placed manually to the database which was processed by the Monitoring Module instead of being simulated from the sensor logs to the parsers then the correlation engine itself. With the standardized logs the Monitoring Module groups together the logs and form a monitored session. The Monitoring Module groups based on three criteria; the event name, sensor name and timestamp. For this experiment, only Snort logs are used. These logs were generated based on Snort community rules. This experiment also assumes that all logs have complete information. Logs without port numbers or IP addresses are not processed.

The Snort logs are parsed with Regular Expressions in order to obtain the CVE ID and CWE ID associated with each SID. The Parser also checks for the number of times the CVE was called, the number of unique CVE (with CWE association) and unique CWEs in the Snort Community rules. Refer to Figure 4 below for the pseudo code



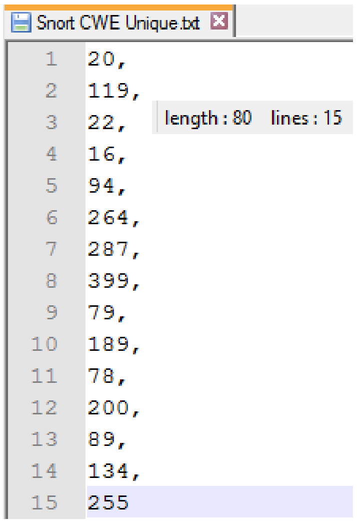
```
Snort CVE From Rules.txt
CVE-1999-0918,272,20
CVE-1999-0002,315,119        length : 5626   lines : 236
CVE-1999-0002,316,119
CVE-1999-0002,317,119
CVE-2001-0054,360,22
CVE-1999-0875,363,16
CVE-1999-0875,364,16
CVE-2008-0394,654,119
CVE-2009-0410,654,119
CVE-2010-2580,654,20
```

Figure 3: Number of CVE



```
WHILE !end_of_file
    PARSE 1 line and STORE 0 or more CVE into an array
    STORE SID in a temporary variable

    FOREACH of the CVEs parsed
        GET cweID from DATABASE and STORE it to an array
        WHILE there is still a cweID
            # Expression below shows the combination of cveID, SID,
            # in the Snort community rules and cweID used
            # A.K.A "Number of times CVE (with CWE association) was called"
            STORE current cveID, SID, current cweID into text file

            # Store all CWEs
            STORE current cweID into an array (cweArray)

            # Store all CVEs
            STORE current cveID into an array (cveArray)

        # Count number of times CVE was called
        # This includes CVEs with no association with CWE
        DUMP current CVE and SID into a text file

# Shows how many unique CWE are in the Snort Community Rules
REMOVE duplicates from cweArray
DUMP cweArray into a text file

# Shows how many unique CVEs (with CWE association)
# are in the Snort Community Rules
Remove duplicates from cveArray
DUMP cveArray into a text file
```

Figure 4: Snort Parser Pseudo code

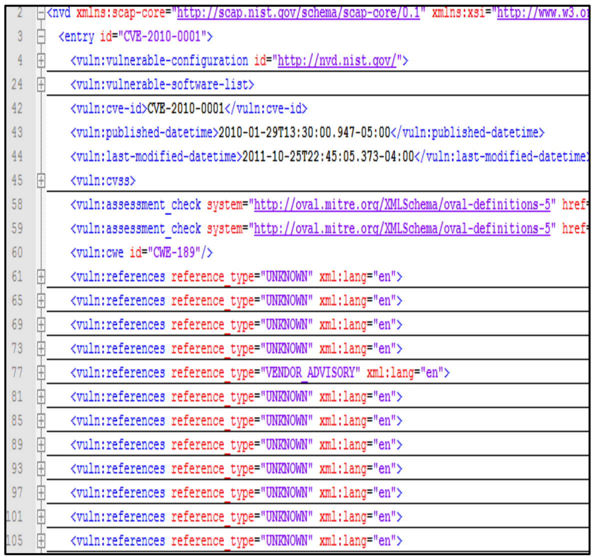


```
20,
119,
22,
16,
94,
264,
287,
399,
79,
189,
78,
200,
89,
134,
255
```

Figure 5: Number of Unique CWE

As shown above, Out of the 3034 Snort community rules, there are only 15 unique CWEs and 236 CVEs (duplicates included) with relation to CWE.

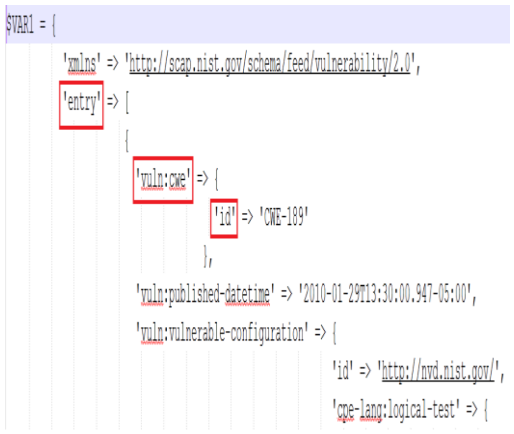
With the monitored session, the Taxonomy Module is able to create a taxonomy name from the CVE ID. This is used to get the other IDs; CWE ID and CPE ID. In the experiment, the CVE and CWE data are obtained by using the XML:Simple Perl package to parse the CVE and CWE definitions from National Vulnerability Database. The relationships between CVE and CWE, parents and children of CWE are also obtained by using the XML:Simple Perl package. Figure 6 below shows a portion of a CVE definition obtained from NVD.



```
<nvd xmlns:scap-core="http://scap.nist.gov/schema/scap-core/0.1" xmlns:xsi="http://www.w3.o
  <entry id="CVE-2010-0001">
    <vuln:vulnerable-configuration id="http://nvd.nist.gov/">
    <vuln:vulnerable-software-list>
    <vuln:cve-id>CVE-2010-0001</vuln:cve-id>
    <vuln:published-datetime>2010-01-29T13:30:00.947-05:00</vuln:published-datetime>
    <vuln:last-modified-datetime>2011-10-25T22:45:05.373-04:00</vuln:last-modified-datetime
    <vuln:cvss>
    <vuln:assessment_check system="http://oval.mitre.org/XMLSchema/oval-definitions-5" href
    <vuln:assessment_check system="http://oval.mitre.org/XMLSchema/oval-definitions-5" href
    <vuln:cwe id="CWE-189"/>
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="VENDOR_ADVISORY" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
    <vuln:references reference_type="UNKNOWN" xml:lang="en">
```

Figure 6: CVE XML File

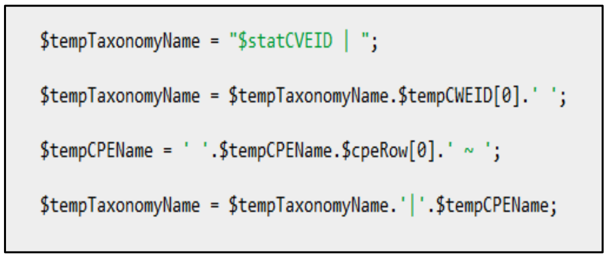
As shown in Figure 7, the XML:Simple produces easy to read xml structure that utilizes the tags to traverse into deeper nodes.



```
$VAR1 = {
          'xmlns' => 'http://scap.nist.gov/schema/feed/vulnerability/2.0',
          'entry' => [
                     {
                       'vuln:cwe' => {
                                       'id' => 'CWE-189'
                                     },
                       'vuln:published-datetime' => '2010-01-29T13:30:00.947-05:00',
                       'vuln:vulnerable-configuration' => {
                                                            'id' => 'http://nvd.nist.gov/',
                                                            'cpe-lang:logical-test' => {
```

Figure 7: XML:Simple Output

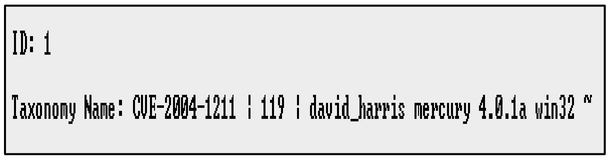
With the monitored session, the Taxonomy Module is able to create a taxonomy name from the CVE ID. This is used to get the other IDs; CWE ID and CPE ID. When these IDs are concatenated together, they produce a taxonomy name. This taxonomy name is used to retrieve the "childOf" tag using the CWE ID inside and is later on used by the Aggregation Module. The Monitoring Session Database is then updated with the "childOf" tag and the taxonomy name.



```
$tempTaxonomyName = "$statCVEID | ";

$tempTaxonomyName = $tempTaxonomyName.$tempCWEID[0].' ';

$tempCPEName = ' '.$tempCPEName.$cpeRow[0].' ~ ';

$tempTaxonomyName = $tempTaxonomyName.'|'.$tempCPEName;
```

Figure 8: Taxonomy Module Partial Code

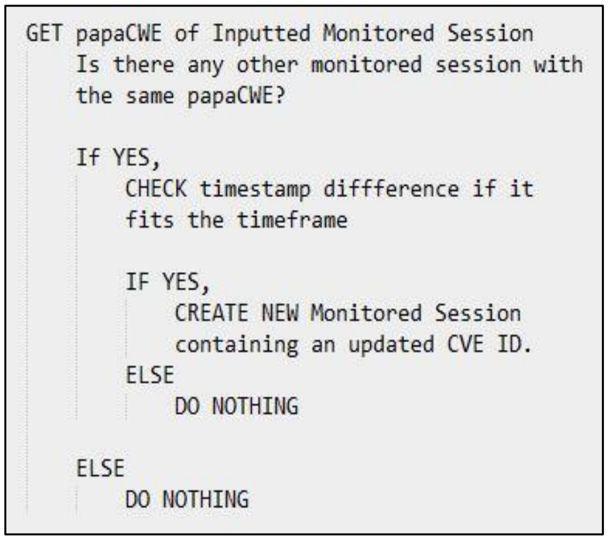
Figure 8 above shows a partial code of the script of how the taxonomy name is built. It shows the flow of building the taxonomy name. It first gives an initial value to the taxonomy name variable which is taken from the monitoring session database. Afterwards, the taxonomy name is then concatenated with the CWE ID and then CPE name/s. The test with the different standard names and IDs were able to give the expected output format which was shown in Figure 1. Figure 9 below shows the result to the taxonomy name test.



```
ID: 1

Taxonomy Name: CVE-2004-1211 ¦ 119 ¦ david_harris mercury 4.0.1a win32 ~
```

Figure 9: Taxonomy Module Result

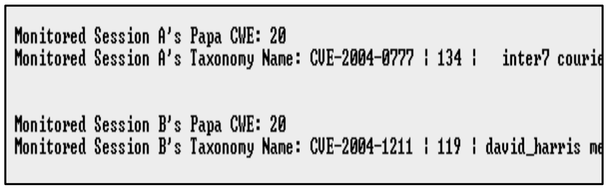
Once the Monitoring Session Database is updated, the Aggregation module makes use of the "childOf" tag and checks different groups based on their "childOf" tag to see if there are any groups that can be combined.



```
GET papaCWE of Inputted Monitored Session
    Is there any other monitored session with
    the same papaCWE?

    If YES,
        CHECK timestamp diffference if it
        fits the timeframe

        IF YES,
            CREATE NEW Monitored Session
            containing an updated CVE ID.
        ELSE
            DO NOTHING

    ELSE
        DO NOTHING
```

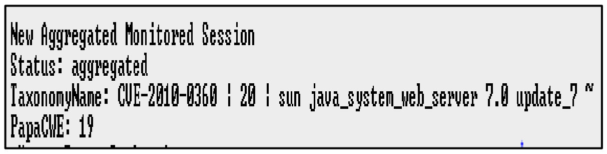
Figure 10: Pseudo code for Aggregation Module

Figure 10 above shows a pseudo code of how the Aggregation Module works. With the inputted Monitored Session which contains its own papaCWE (childOf tag), it checks if there are other monitored sessions that contain the same papaCWE. "papaCWE" contains the childOf ID that signifies that two different events are actually related. Given that there is a match, it checks the time difference between the two monitored sessions. For this experiment, the configured time frame is 7200 seconds. If the time difference is less than the 7200 seconds, it creates a new monitored session containing an updated CVE ID. This CVE ID will contain a score higher than the previous CVE IDs.



```
Monitored Session A's Papa CWE: 20
Monitored Session A's Taxonomy Name: CVE-2004-0777 ¦ 134 ¦   inter7 courie


Monitored Session B's Papa CWE: 20
Monitored Session B's Taxonomy Name: CVE-2004-1211 ¦ 119 ¦ david_harris me
```

Figure 11: Aggregation Module Input



```
New Aggregated Monitored Session
Status: aggregated
TaxonomyName: CVE-2010-0360 ¦ 20 ¦ sun java_system_web_server 7.0 update_7 ~
PapaCWE: 19
```

Figure 12: Aggregation Module Output

Figure 11 shows the inputs to the Aggregation Module. When the aggregation script ran, it was able to find two monitored sessions that can be combined (Monitored Session A and B). It then inserted the output found in Figure 12 into the Monitoring Session Database as a new monitored session. The CVE ID placed in that aggregated monitored session contains a score higher than the CVE ID of the two monitored sessions used to create it.

Table 1: Aggregated Monitored Session

Monitored Session	**childOf**	**Status**	**Threat Level**
A	20	Grouped	10
B	20	Grouped	7.5
C (A, B)	19	Aggregated	10

As seen in Table 1, there are three records. The first two are separate monitored sessions. After the second monitored session passes through the Aggregation Module, the module was able to find a common childOf tag between the first and second monitored sessions. This then produces the third monitored session “C”. With the combination of two monitored sessions, it now shows the score of the “childOf” tag of monitored sessions A and B.

4. CONCLUSIONS

Although signature-based correlation engines are widely used today, they are still limited to detecting known attacks. Because of the great threats continuously evolving, there is a need to develop another type of correlation engine which allows detection of variations of common attacks. The wider the scope of attack detection, the better is the security of a system. The development of the Taxonomy and Aggregation modules in the anomaly-based correlation engine satisfies the need for a better scope detection. These modules allows the correlation engine to elevate alerts that pose a bigger threat. Aside from that, the taxonomy name produced by the Taxonomy Module also allows the correlation engine to take note of the important assets and monitor for any anomalous behavior that might affect these assets. Other importance of the taxonomy name are the more efficient use of the CVE ID for scoring because of its availability in the taxonomy name, and the opportunity it gives to the security analyst to give a better security evaluation.

Although the implementation of CWE is possible, the amount of CWE data is not enough to construct a hierarchy of CVEs based from their CWE relationships (refer to Figure 3, Figure 5). One possible solution is to use tags from Snort community rules as a way of categorizing the events.

5. ACKNOWLEDGMENTS

We would like to thank Mr. Isaac Herculano Sabas for his guidance and support. We would also like to thank the members of Pandora Security Labs for their advices regarding our research.

6. REFERENCES

Chapple, M. (n.d.). *Should IDS and SIM/SEM/SIEM be used for network intrusion monitoring?* Retrieved November 12, 2014, from http://searchsecurity.techtarget.com/answer/Should-IDS-and-SIM-SEM-SIEM-be-used-for-network-intrusion-monitoring

Du, W. (2006, August 26). *Intrusion Detection System.* Retrieved October 18, 2014, from http://www.cis.syr.edu/~wedu/Teaching/cis758/LectureNotes/Intrusion_Detection.pdf

Kibirkstis, A. (2009, November). *What is the Role of a SIEM in Detecting Events of Interest?* Retrieved October 18, 2014, from http://www.sans.org/security-resources/idfaq/siem.php

MITRE Corporation. (n.d.). *Common Platform Enumeration*. Retrieved October 22, 2014, from https://cpe.mitre.org/

MITRE Corporation. (n.d.). *Common Vulnerabilities and Exposures*. Retrieved October 18, 2014, from http://cve.mitre.org/

MITRE Corporation. (n.d.). *Common Weakness Enumeration*. Retrieved October 18, 2014, from http://cwe.mitre.org/

Sourcefire. (n.d.). *The Case For The Next-Generation IPS.* Retrieved November 12, 2014, from http://de.security.westcon.com/documents/44750/Sourcefire%20Next-Generation%20IPS%20(NGIPS)%20White%20Paper.pdf

Vaarandi, R., & Grimaila, M. R. (2012). Security Event Processing with Simple Event Correlator. *ISSA Journal*, 30-37.