Keywords: Anonymity, Log Analysis, Security Awareness, Threat Awareness, Threat Modelling.

Abstract: Modern information infrastructures and organizations increasingly face the problem of data breaches and cyber-attacks. A traditional method for dealing with this problem are classification zones, such as ‘top secret’, ‘confidential’, and ‘unclassified’, which regulate the access of persons, hardware, and software to data records. In this paper, we present an approach that finds classification zone violations through automated message flow analysis. Our approach considers the problem of anonymization for the source event logs, which makes the resulting data flow model sharable with experts and the public. We discuss practical implications from applying the approach to a large governmental organization data set and discuss how the anonymity of our concept can be formally validated.

1 INTRODUCTION

Information security is one of the crucial topics for modern industry and governmental organizations. Malicious attacks on industrial companies and governmental organizations can cause high annual damage. In 2017, the spending for IT security therefore increased dramatically, but at the same time, an all-high number of cyber-attacks and data breaches was reported (ENISA, 2018). Illegal knowledge transfer and economic sabotage are no longer a rare individual event, but a mass phenomenon. There are thousands of critical security incidents per year that produce an accumulated loss of billions of data records (RBS, 2018). Haystax Technology reported in 2017 that the largest threat for a company are managers with insider access rights, followed by consultants/suppliers and regular employees (Haystax Technology, 2017).

A common approach for dealing with sensitive data is the concept of information classification (European Commission, 2001). Different data items, such as paper documents and computer files, are classified according to their security sensitivity. The grouping takes place by the notion of classification zones, such as “TOP-SECRET”, “SECRET”, “RESTRICTED”, and “UNCLASSIFIED”. All data items in an organization, their storage facilities and the persons accessing them are categorized according to this scheme. The daily procedures and the IT infrastructure are now obligated to regulate the read and write access to data items in a way that their classification zone is taken into account. The necessary access control and monitoring can happen through software mechanisms, such as discretionary access control, electronic authentication procedures and automated security audits, or through more classical security means such as physical separation and entrance guards.

Given the fact that modern industries and institutions are forced to link their data silos and IT systems in complex networks, with some of them being constantly accessible from the Internet, it becomes increasingly harder to enforce classification zones and their protection. Several recent examples show that the detection and avoidance of classification zone violations become an increasingly relevant topic in the security community:

- In May 2015, the German federal parliament IT infrastructure was heavily attacked. Offices of parliament members were infiltrated and remotely searched for classified information (Zeit, 2015).
- A former external collaborator of the NSA was sued for a data theft of 50 Terabyte over a time period of 20 years, which is seen as the largest theft of classified governmental material of all times (WP, 2016).
In 2017, the credit office Equifax was attacked for stealing sensitive personal information of more than 147 million American people. The data breach leaked information such as name, driver license details and the social security number, which are important identity properties in the USA (Forbes, 2017).

### 1.1 Problem and Approach

Figure 1 shows the potential security threats for an infrastructure that relies on the idea of classification zones. Given the example of three protection categories (unclassified, restricted, secret), it can be seen that they include each other, e.g. every unclassified information can be treated the same way as a secret information, but not the other way around.

There are 4 possible information flows that can now take place:

- The first flow of information stores a lowly classified document on a lowly classified server, which is an allowed activity.
- The second possible flow of information stores a lowly classified document on a highly classified server. This flow of information is only allowed in one direction, an information backflow must be avoided. This is typically achieved with a data diode (Genua, 2016).
- The third flow of information attempts to transfer a highly classified document to a lowly classified server. This information flow is prohibited.
- The fourth flow of information stores a highly classified document on a highly classified server, which is again an allowed activity.

Given that general principle, the following security threats exist:

- Information flows from high to low zone
- Uncontrolled information flows from low to high zone (backflow problem - 2nd flow of information in figure 1)
- Information flows between uncategorized external stakeholders and internal servers

It should be noted that ‘unclassified’ is not equal to an unrestricted read or write access for external parties. The overall classification zone infrastructure is still treated as a closed ecosystem.

The problem scenario can be easily extended with human stakeholders, which also belong to a particular classification zone. They can be treated as a special kind of ‘server’ that gets read and write access to some classified document.

In this paper, we discuss our work on a practical approach for identifying classification zone violation as security threat. The investigation is performed based on protocols of past information flows in the organization, here given as simple textual log files from message exchange servers. The huge original raw data set is converted to an information flow model, which can be generated and investigated once, periodically, manually, or in an automated fashion.

The goal of this investigation is to find weak security spots and unknown attack vectors in the infrastructure.

A unique property of our work is the anonymization of the source data. This is a common expectation in organizations with mandatory classification zones, but is generally the opposite of open security research. The anonymization demand also normally prevents the collaboration with third-party consultants, or forces them to sign strong legal non-disclosure agreement (NDA) contracts. In our attempt, we try to tackle both issues at the same time.

### 2 RELATED WORK

The detection of classification zone violations, or suspicious activities in general, is a classical topic in security research and practice. One early example from the 80s is the Haystack system (Smaha, 1988), which was intended to detect intrusions in Air Force systems based on atypical system usage. The authors already considered the problem of an “event horizon”, meaning that the monitoring data needed to be reduced before further processing.

Modern computer systems constantly generate large amounts of event logs. These protocols are
intended for system administrators to understand the current system status and to pinpoint system downtime issues.

The usage of log files for finding dependability incidents is a classical topic in high-performance computing. In the past, there were attempts to use pattern discovery (Hellerstein et al., 2002), data mining (Stearley, 2004), (Vaarandi, 2004), (Yamanishi et al., 2005), clustering (Vaarandi, 2003), support vector machines (Bose et al, 2013), (Fronza, 2013), or decision tree learning (Reidemeister et al., 2013) for analyzing massive amounts of original log data. Oliner et al. (Oliner et al., 2008), for example, used the information entropy of message terms for finding failure-related events in HPC logs. Nearly all of these approaches focus on reliability events, such as hardware component failures, which are indicated through severity markers in the log files. This makes these approaches not directly applicable for our classification zone problem, but the available ideas for log information reduction can be directly applied (Cheng, 2015), (Azodi, 2014), (Saepigin, 2013), (Pantola, 2010). We also utilize security meta-data in the log analysis, something that is not commonly available in HPC-alike systems.

One part of our approach is the conversion of log data to data flow representations. The latter is a well-known tool in the security community. A classical approach for this is are data flow diagrams or similar techniques such as flowcharts (Gane, 1977), (DeMarco, 1978), (Youdon, 1989). Data flow diagrams are well known for their application within the STRIDE threat modeling methodology by Microsoft (Swiderski and Snyder, 2004). Another example is the work by Schmidt et al. who used data flow analysis for ranking security issues in automotive software architectures (Schmidt et al., 2014). Meinig et al. described the notion of extended classification zones (Meinig and Meinel, 2018).

The resulting data flow information can be investigated by sophisticated algorithms, so that security threats can be found automatically and at run-time. Data flow analysis in the security context is already applied for system calls made by applications (Li et al., 2008), network traffic (Su et al., 2018), (Tzur-David et al., 2009), (Hofstede, 2018), virtual runtime engines (Feng et al., 2010), program binaries (Chen et al., 2011), or program code (Baca, 2010). For this publication, we focus on the generation of data flow diagrams that can be inspected by human auditors from inside and outside the organization.

3 FROM EVENT LOGS TO DATA FLOW ANALYSIS

Given the demand for both anonymity and useful security analysis results, we created an approach that combines the following steps (see also Figure 3):

1. Event log sanitizing
2. Evaluation of message paths
3. Model Generation (Anonymization of source data, while keeping relevant security information and translation into a data flow model)

We applied the approach to the following data set from a large governmental organization:

- Unstructured raw logs about network communication for a set of servers
- Time range: 01/16 - 11/16
- One folder per server
- Multiple text files per folder, each covering a particular time span
- Log entries per file in text column format
- Per log entry: Timestamp, unique message identifier, send / receive operation, data size
- List of nodes and their security classification

Content and type of the exchanged data is a confidential information in itself and can therefore not be shared in the analysis result.

Folder Name: Server S
File Name: 2016-01-01XXX.log

01 00:02:12 Data received from Server S
YYY Bytes XXXMessage-ID-ABC

01 00:02:12 Forwarded to Server Sc
XXXMessage-ID-ABC

01 00:02:12 Data sent to Server Sd
YYY Bytes XXXMessage-ID-ABC

Figure 2: Raw log data.

3.1 Event Log Sanitizing

We developed a set of Python scripts that parse the original log files linearly and generate a database of node communication activities. This step removes invalid and incomplete information items, such as log entries with missing message classification information or invalid timestamps. Both problems were discussed with the administrators of the infra-
Figure 3: Steps to detection of classification zone violations.

structure and justified with implementation bugs, and not with a potential attempt for cleaning the logs from an earlier attack. The result of this normalization step is a set of records, as shown in Table 1, where each provides the following information:

- Column 1: Date
- Column 2: Time
- Column 3: Node generating the log entry
- Column 4: Data operation performed by the referenced node (send / relay / receive)
- Column 5: Data size of the message
- Column 6: Peer node
- Column 7: Unique message identifier

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Server Name</th>
<th>Data Direction</th>
<th>Byte Size</th>
<th>Server Name</th>
<th>Message ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016-01-01</td>
<td>00:02:12</td>
<td>S_a</td>
<td>Received from</td>
<td>YYY</td>
<td>S_a</td>
<td>ABC</td>
</tr>
<tr>
<td>2016-01-01</td>
<td>00:02:12</td>
<td>S_a</td>
<td>Relay to</td>
<td>S_c</td>
<td>S_c</td>
<td>ABC</td>
</tr>
<tr>
<td>2016-01-01</td>
<td>00:02:12</td>
<td>S_a</td>
<td>Send to</td>
<td>YYY</td>
<td>S_a</td>
<td>ABC</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

It must be noted here that message sending comes in two flavors: Sending triggered by the node itself, and sensing triggered due to relaying of a received message. This leads to the fact that a single receive activity may be reasoned by one of them, which is not directly visible from the entry. For that reason, we need to consider the message path as a whole.

### 3.2 Evaluation of Message Paths

Given the set of records about direct node-to-node interaction, we are now detecting message paths that may cross relay nodes. This can be easily done by clustering single transfer activities based on the unique message identifier. This is again implemented with a set of Python scripts, since the amount of log data makes such an approach still feasible. For larger data sets, according scalable analysis tools based on NoSQL databases can be applied.

The result is a set message transfers, where each transfer has a set of participating nodes. This can be the starting point for a multitude of analysis steps. In the approach here, we continue by only relying on the pure node-to-node message frequency statistics. Together with the given node security classification, this leads to a dataset that describes the interaction frequency of nodes in different, or the same, classification zone. Together with the overall amount of messages, it is now possible to generate the graph representation for further analysis.

Figure 4 shows an example for the resulting data file. It allows to determine the path of every message through the network.

### 3.3 Model Generation

The next step is to create a representation for the message flow data that can be shared with external parties.

The visualization of the identified information exchange flows is done with the *secure data flow diagram (DFDSec)* approach (Meinig and Meinel, 2018). It supports the representation of classification zones in an extended data flow diagram. The model consists of five elements: processes, external entities, data stores, data flows and security classification zones.

Processes are entities that perform activities by operating on incoming data and potentially generating an output. In our example data set, this relates to the different servers in their role as message sender, receiver, or relay.
External Entities are actors outside the modeled system which are the source or destination of information. They may be part of different security classification zones or a complete own one.

Data stores can contain information but perform no further processing that may lead to new or forwarded messages. Given the fact that the raw log data only describes active entities, and not passive storage facilities, we do not utilize the concept of data stores with the experiment.

Data flows are information flows among classification zones, processes, external entities and data stores. The most interesting ones cross classification zones and must therefore be investigated more specifically with respect to security threats.

Classification zones are the specific security level that includes processes, data flows, data stores, and external entities. They are drawn as dashed circles around other elements in the data flow diagram.

We generate the DFDSec diagram automatically from the message exchange data generated in the last step. This is done by producing a single GraphML file per data set. This file is modified by an auto-layout algorithm for determining the model element positions in the final picture.

One of the targets of our work is the anonymized and pseudonymized representation of sensitive information in a security model. We therefore take the following anonymization steps:

- The timestamp of the message exchange is not used in the model generation (column 1 & 2).
- Server names are translated in pseudonyms (column 3).
- Message size and message identifier are not used in the model generation (column 5 & 7).
- Data flows between processes are attributed with the fraction of message exchanges they contributed to the overall amount of data exchange.

In order to reduce model complexity in practice, we omit data exchanges between nodes in the same classification zone that contributes less than 1% to the overall traffic. We argue here that for a potential attacker, high-traffic nodes are more valuable than sparsely used servers. One obvious reason is that successful cryptographic attacks mostly rely on the availability of larger message flow probes.

Message transfers that cross classification zones are not capped, since each single message transfer may indicate a potential security breach.

Figure 5 shows a DFDSec diagram generated from the real-world data set in a governmental organization. It can be seen that there are two different classification zones in real use, RESTRICTED and SECRET. Each classification zone contains several operational nodes (sender, receiver or relay servers). Between these nodes there are information flows. Within the utilized data set, it turned out that message exchanges crossing a classification zone border did not take place. For the given governmental system, this is the expected and correct mode of operation.

When a DFDSec analysis shows a zone crossing, it depends on the kind of source and destination zone, as discussed in Section 1.1, that it is a problem. A typical approach would be the repeated investigation of the issue over time, for example by generating multiple DFDSec models periodically and compare them. This would clarify if the classification zone crossing is a regular effect at normal business hours with an expectable message frequency, or if this is an anomaly in the system operation. The analysis can also consider specific attributes of the participating nodes, such as their role in the organization.

For the given data set, it turned out that the most important operational node is S A. 26 percent of the traffic goes away from this node and 28 percent of the traffic goes in.

In the classification zone SECRET, the most important operational node is R A. 36 percent of the traffic is leaving this node. By contrast, this node has hardly any incoming connection.

From an attacker point of view, these two nodes seem to be the most interesting. Most of the information flows start or end here, so they have the highest potential for gathering classified information. From the defender’s point of view, IT security measures must be primarily taken on these two nodes. One idea would be to design these important nodes redundantly, so that, for example, in hierarchical organizations, they no longer act as single relay for messages. This would diffuse classified information over multiple paths in the organization, which makes it harder for an attacker to identify the relevant nodes in the system that are worth an attack.
Figure 5: DFDSec.
The most important information flow happens in the classification zone RESTRICTED. The connection SA - SC is responsible for 17 percent of the message flows, followed by the connection SA - SS at 10 percent. The most important information flow in the classification zone SECRET is the connection RA - R8 at 26 percent, followed by the connection R9 - R10 at 13 percent and the connection R1 - R11 at 11 percent.

From the point of view of an attacker, these connections are those that are eligible for a man-in-the-middle, e.g. by compromising switches and routers on the network path between the two nodes. From a defensive point of view, it would be therefore worth considering whether to redistribute traffic differently in order to avoid connections that are particularly busy, e.g. load sharing (Eager et al., 1986), or to harden the connection network accordingly.

4 ANONYMITY

Security analysis that relies on unfiltered classified data produces again a classified entity. It can rarely be made available directly to third parties. The challenge in the daily work is therefore to share such data without jeopardizing the disclosure of classified infrastructure details on the one hand, and not limiting the usefulness of the effort too much on the other hand.

While the DFDSec approach helps to make stripped results still useful, it remains an open question if the classified infrastructure knowledge is truly protected. It is not enough to reduce the published data set and hope for the best, instead, an objective and repeatable verification must be performed to validate this property.

One possibility for the evaluation of an anonymization approach was described by Samarati and Sweeney. Their model is called k-anonymity. A published data set is declared to have k-anonymity when identifying information for a single node, such as a server name, is indistinguishable from at least k-1 similar entities (Samarati, 1998), (Sweeney, 2002). A larger k means a higher degree of anonymization. The result is the confidence of 1 / k that a correct linking of correlating classified knowledge is not possible.

The model was later extended by the idea of l-diversity (Machanavajjhala, 2007) and t-closeness (Li, 2007). These extensions and the original approach prevent the possibility of de-anonymization, which could result from an unsorted matching attack, temporal attacks, or complementary release attacks (Ganta et al., 2008). These extensions fixed specific problems of k-anonymity such as the homogeneity attack or the background knowledge attack (Machanavajjhala, 2007).

For the approach described in this paper, we improve the k-anonymity by omitting 5 of the 7 original data columns. Only the sending server and receiving server are kept in their original meaning. k is determined here by the smallest number of server connections between two servers in the message flow data set. From the viewpoint of anonymization attacks, our approach covers the following security threats:

In unsorted matching attacks, the data columns are separated and disclosed in the same ordering. This allows the original data to be restored. This attack is prevented in our concept by only publishing two of the original columns. In addition, the sorting in the published graph representation is based on the node-to-node message frequency, which can vary. This automatically changes the order of the entries.

The temporal attack exploits the fact that data collections are dynamic. Adding, modifying, or removing entries can affect the k-anonymity of the analysis result. This attack is prevented by not changing the shared attributes (pseudonym server names) when publishing updated or extended analysis results. Subsequent versions of a DFDSec diagram therefore must re-use the same server pseudonyms as before.

In the case of the complementary attack, different releases of subsets of the base file may result in different anonymizations, each corresponding to k-anonymity. By combining the respective publications, the k-anonymity is canceled again. This attack is prevented by having all subsequent versions of an analysis result based on the original one. This is guaranteed by always using the same column attributes (inbound, outbound server) and column contents for unmodified data. The only changing variable for updated DFDSec should be the message frequency and the set of processes being modelled.

Homogeneity attacks rely on the idea that there are groups of data items, here model elements, that all have the same sensitive attribute. This attack is prevented in our concept by simply not using the sensitive attributes (e.g. message ID or server name) for the final published model. Additional diversity, as proposed in (Machanavajjhala, 2007), is therefore not needed.

The background knowledge attack exploits the fact that an attacker can unambiguously allocate data content despite k-anonymity through additional
knowledge. In our approach background knowledge is not made possible by omitting all sensitive attributes and by the physical separation of classified and anonymized public information. We publish only data necessary for external support, following the filter-in-principle described in (Pang and Paxson, 2003).

5 CONCLUSIONS AND FUTURE WORK

In this paper, we discussed an approach to identify classification zone violations by the means of automated message exchange analysis. The determination of critical information flows leads to the identification of sensitive operational nodes that are high-risk targets for security attacks. Since the raw data logs and the analysis results are classified information by themselves, we propose a straightforward anonymization approach for the original data. We discussed the potential for well-known anonymization attacks.

Our analysis of one year of real-world data showed that the system under investigation is in good health. Classification zone violations could not be identified. The visualization with reduced DFDSec diagrams turned out to be a feasible method for governmental organizations with high demands on data protection. The use of anonymized DFDSec models allows for sharing the analysis results with external security consultants and management personal that has a low security classification.

This paper is part of ongoing project work inside the governmental organization. Future work will focus on performing more advanced analysis, such as an anomaly detection that considers the time axis of the message exchange patterns or better consideration of message semantics. It is also planned to migrate these techniques from a pure offline analysis to online monitoring infrastructure.

REFERENCES


Bose, R. P. J. C., van der Aalst, W. M. P., 2013. Discovering signature patterns from event logs, IEEE Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, Singapore, Singapore, DOI: 10.1109/CIDM.2013.6597225.


