UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

RESILIENT EVENT COLLECTION IN SIEM SYSTEMS

Pedro da Silva Dias Rodrigues

DISSERTAÇÃO

MESTRADO EM SEGURANÇA INFORMÁTICA

2013

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

RESILIENT EVENT COLLECTION IN SIEM SYSTEMS

Pedro da Silva Dias Rodrigues

DISSERTAÇÃO

Trabalho orientado pelo Prof. Doutor Nuno Fuentecilla Maia Ferreira Neves

MESTRADO EM SEGURANÇA INFORMÁTICA

2013

Acknowledgments

First I would like to thank Prof. Nuno Neves for his guidance throughout this thesis. As an advisor his contributions were essential for me, not only during the research and planning phases but also with the valuable contributions to interpret the implementation results. His vast expertise in this area, as well as experience with international projects, boosted this work to a level that would be otherwise unattainable. I must also mention the professors I encountered during the two years at FCUL. Each one contributed to the success of my academic path, most noticeably Prof. Nuno Neves and Prof. Paulo Veríssimo in the fundamental security courses. I would also like to thank the support and opportunity to work within the European project FP7-257475 (MASSIF).

I express my gratitude to Eng. Paulo Moniz for his trust and for making the coexistence of my academic and professional life possible. He encouraged me to achieve additional goals in both fronts using his leadership skills and immense professional experience in the Information Security area. I also thank Eng. Vergílio Rocha, the person responsible for my enrolment in this Master's program. He is a true sponsor of my academic endeavors as well as professional responsibilities.

My colleagues, both at FCUL and EDP, were also immensely supportive throughout these years. I highlight Manuela Gaivéo and Ivo Rosa, members of the security team at EDP with whom I work closely everyday. I also thank Diana São Miguel for reviewing this dissertation and providing an outside opinion. Miguel Areias and Anderson Barretto were my companions through long studying weekends and nights at FCUL, providing not only support but also, more importantly, the unwinding moments that made me carry on.

For all the patience, friendship and love, I admire my fiancée, Cristiana Filipa Ferreira. You are the best!

Finally, I am grateful for my parents and their unconditional support throughout my life.

Pedro Rodrigues

Dedicated to Cris.

Resumo

A importância da Segurança da Informação tem crescido rapidamente nos últimos anos, com uma maior consciencialização da sociedade civil e das empresas para o problema. As notícias recorrentes de ataques direcionados e roubo de informação em larga escala que resultam em grandes prejuízos financeiros, por vezes tendo como consequência o encerramento das organizações envolvidas, justificam o investimento em mecanismos de proteção da informação.

No âmago da capacidade para monitorização da segurança em tempo-real está o Security Operations Center (SOC), o conjunto de pessoas, processos e sistemas onde se concentram as capacidades de análise e resposta a incidentes de Segurança da Informação. A base tecnológica do SOC é construída sobre o sistema de Gestão de Informação e Eventos de Segurança, vulgo SIEM. Este sistema permite recolher eventos de segurança de diversas fontes e encontrar padrões de ataque analisando relações entre eles. No entanto, tal como acontece com todos os sistemas informáticos, um atacante que tenha conhecimento da sua existência irá procurar ultrapassar as proteções implementadas, prevenindo que a equipa do SOC seja alertada para o ataque em curso.

A relevância dos sistemas SIEM tem vindo a aumentar no contexto da maior importância atribuída a questões de segurança da informação. Considerando um número cada vez mais elevado de eventos e as múltiplas origens onde estes são gerados, as equipas de monitorização estão cada vez mais dependentes de consolas únicas onde a informação é centralizada e processada. Como consequência existe também uma maior dependência dos sistemas centrais, tornando-os pontos únicos de falha.

Os sistemas SIEM são intrinsecamente complexos devido à necessidade de recolha de eventos de segurança a partir de fontes com tecnologias muito diversas, com localizações dispersas. O facto de desempenharem diversas funções aumenta esta complexidade, necessitando de módulos para recolha, consolidação, processamento e armazenamento de eventos. Para além destes módulos, que podem ou não traduzir-se em componentes fisicamente distintos, os sistemas SIEM estão fortemente dependentes dos sensores colocados junto às fontes de eventos, bem como da rede de comunicações que permite o envio desses eventos entre os diversos componentes, até à consola central.

A inexistência de investigação diretamente focada no aumento da resiliência dos sistemas SIEM resulta na implementação de soluções pouco adaptadas aos riscos e desafios associados a infraestruturas de segurança. Estando maioritariamente focada na proteção de segurança ao nível da rede, muitos dos desenvolvimentos recentes centram-se na capacidade de identificar padrões de tráfego maliciosos. Esta abordagem reflete-se em publicações direcionadas aos sistemas de detecção e prevenção de intrusões (IDS/IPS), com menos enfoque na implementação resiliente de sistemas SIEM. A nossa percepção, corroborada por uma pesquisa alargada de trabalhos desenvolvidos nesta área, aponta para um elevado número de implementações padrão, assumindo cenários teóricos e sem tomar em linha de conta o efeito de ataques contra o próprio sistema SIEM.

Neste trabalho começamos por efetuar uma análise às falhas de segurança que podem afectar o desempenho do processo de recolha de eventos de segurança, incluindo falhas acidentais mas também possíveis ataques deliberados ao sistema SIEM que possibilitem a uma entidade maliciosa ultrapassar os mecanismos de segurança implementados. Com base nessa análise endereçamos os problemas de fiabilidade que afetam qualquer sistema informático, apontando soluções que permitam lidar com falhas acidentais e, dessa forma, aumentar a disponibilidade do sistema. Ao reduzir a probabilidade de falhas que impeçam a recolha de eventos de segurança, estamos a contribuir diretamente para diminuir a janela de oportunidade disponível para que ataques à infraestrutura não sejam detectados. Focando o risco de falhas maliciosas, propomos soluções que impeçam os atacantes de explorar com sucesso vulnerabilidades no processo de recolha de eventos de segurança. Este processo envolve sistemas heterogéneos, desde a fonte dos eventos até à consola central, passando pela rede de comunicação responsável por interligar toda a infraestrutura. Consideramos fundamental atingir um nível de robustez elevado, mesmo na presença de infraestrutura parcialmente comprometida.

O principal objectivo deste trabalho passa por definir um método sistemático de recolha e correlação resiliente de eventos de segurança num sistema SIEM, mesmo na presença de componentes maliciosos sob controlo de atacantes. Para atingir este objectivo centramo-nos na robustez das regras de correlação, desde a sua concepção e desenho até à implementação final no sistema SIEM. Os sistemas SIEM contêm um conjunto alargado de regras padrão que, como demonstramos, partem de premissas demasiado optimistas relativamente ao processo de recolha de eventos. Descrevemos,

ao longo do trabalho, de que forma estas regras padrão podem ser melhoradas para lidar com as diversas possibilidades de falhas e ataques maliciosos, aumentando desta forma a resiliência total do sistema SIEM e o nível de confiança que a equipa do SOC pode depositar nesta ferramenta essencial. Utilizando casos de uso reais, demonstramos a metodologia proposta para aumentar a resiliência das regras de correlação. Tendo como ponto de partida uma regra base, aplicamos passo a passo a metodologia, detalhando e avaliando cada evolução da regra, até ser atingido um nível de robustez elevado.

Com o propósito de sistematizar a metodologia proposta para o aumento de qualidade das regras de correlação, desenvolvemos uma aplicação denominada AutoRule. Esta ferramenta recebe como entrada uma ou mais regras de correlação e efetua uma análise automática, detectando possíveis lacunas e sugerindo correções. Apesar de não suprir a necessidade de análise com base na experiência prática na definição de regras de correlação, a aplicação AutoRule permite à equipa de configuração do sistema SIEM atuar de forma precisa e direcionada, corrigindo as regras de correlação e, dessa forma, tornando-as mais resilientes.

Finalmente, para demonstrar e medir a eficácia da nossa proposta, foi posta em prática a metodologia através de uma implementação em cenário real, recorrendo ao sistema SIEM utilizado para monitorizar os eventos de segurança na rede corporativa da EDP – Energias de Portugal, S.A. Tratando-se de um grupo multinacional com mais de 12000 colaboradores ativos, a rede informática monitorizada por este sistema SIEM fornece a possibilidade de analisar em larga escala os efeitos das melhorias propostas.

A metodologia proposta para aumentar a resiliência das regras de correlação traduziu-se num acréscimo da eficácia das mesmas, resultando num sistema mais fiável. A consequência mais direta é uma melhoria operacional do SOC, que passa a dispor de informação mais precisa e mais adequada ao seu contexto de operação. Para além da proposta teórica, a implementação permitiu também validar a operação num cenário real da aplicação AutoRule, desenvolvida para automatizar a análise das regras de correlação. As melhorias introduzidas nas regras de correlação desenvolvidas no contexto da operação do SOC EDP, seguindo os passos da metodologia, foram sendo testadas com recurso à aplicação. Os resultados demonstram que a eficácia medida das regras correspondeu também a um melhor resultado obtido através da análise automática, existindo por isso motivos para confiar nesta análise. A aplicação AutoRule possibilitou ainda uma comparação entre as regras predefinidas, instaladas de forma

automática com a solução ArcSight, e as regras que seguiram o processo de melhoria preconizado pela metodologia proposta.

As avaliações finais que fazemos da implementação num cenário real são francamente positivas, ratificando a nossa proposta teórica e conferindo-lhe um elevado grau de confiança quanto à possibilidade de aplicação em larga escala, de forma independente da tecnologia de sistema SIEM escolhida.

Palavras-chave: SIEM Resiliente, Correlação de Eventos, Regras de Correlação, Falhas Acidentais, Ataques Maliciosos, Tolerância a Intrusões, Centro de Operações de Segurança

Abstract

Information Security has become a relevant subject in recent years, with greater awareness to the topic from major companies and general public. The frequent news regarding targeted attacks and large-scale information thefts resulting in major financial losses, sometimes even resulting in company bankruptcy, justify investments in protection mechanisms.

At the heart of real-time security monitoring is the Security Information and Event Management system, commonly known as SIEM. These systems allow for security event collection and pattern discovery, by analyzing relationships between those events in real-time. However, as with all computer systems, an attacker who is aware of its existence will seek to overcome the protection mechanisms in place, preventing the security experts from being alerted to the ongoing attacks.

We present an analysis of possible attacks to a SIEM system and seek solutions to prevent successful exploitation of those attacks, even if the attackers are able to take control over part of the infrastructure. Instead of suggesting massive changes throughout the multiple systems and network components, we propose an approach based on the capabilities of the SIEM system to collect and correlate security events from multiple sources. We advocate that it is possible to detect faults, malicious or accidental, though real time analysis of the collected events using carefully crafted and resilient correlation rules.

Our goal is to define a systematic method to resiliently collect and correlate security events in a SIEM system, despite the presence of components already under the control of attackers. The effectiveness of the proposed methodology is evaluated in a real production environment, simulating attacks and accidental failures and observing their effects in the capability of the SIEM system to identify abnormal behavior. We also develop and demonstrate an application capable of automatically analyzing correlation rules, identifying vulnerabilities and proposing improvements to increase their overall resilience.

Keywords: Resilient SIEM, Event Correlation, Correlation Rules, Accidental Failures, Malicious Attacks, Intrusion Tolerance, Security Operations Center

Table of Contents

List of Figures

List of Tables

List of Abbreviations

APT – Advanced Persistent Threat;

DMZ – Demilitarized Zone;

DoS – Denial of Service;

EDP – Energias de Portugal;

IdM – Identity Management System;

ICS – Industrial Control System;

IDS – Intrusion Detection System;

IPS – Intrusion Prevention System;

IPsec – Internet Protocol Security;

MAC – Message Authentication Code;

NTP – Network Time Protocol;

SIEM – Security Information and Event Management;

SOC – Security Operations Center;

SCADA – Supervisory Control And Data Acquisition;

SSL – Secure Socket Layer;

TCP – Transmission Control Protocol;

TLS – Transport Layer Security;

UDP – User Datagram Protocol;

WORM – Write Once Read Many.

Chapter 1

Introduction

A Security Information and Event Management (SIEM) [5] is a system that supports threat detection and security incident response through real-time collection and analysis of security events from a wide variety of event and contextual data sources [8].

Figure 1 represents a possible outline of the SIEM operational diagram depicting the various possible security event sources, for instance web servers or firewalls that generate operational logs. The events generated by the sensors in each asset are collected at the entry point of the SIEM, the event collector. Then, they are forwarded to the core of the system, the correlation engine, responsible for processing the information from the various sources and determining possible security issues, raising alarms in the console as needed.

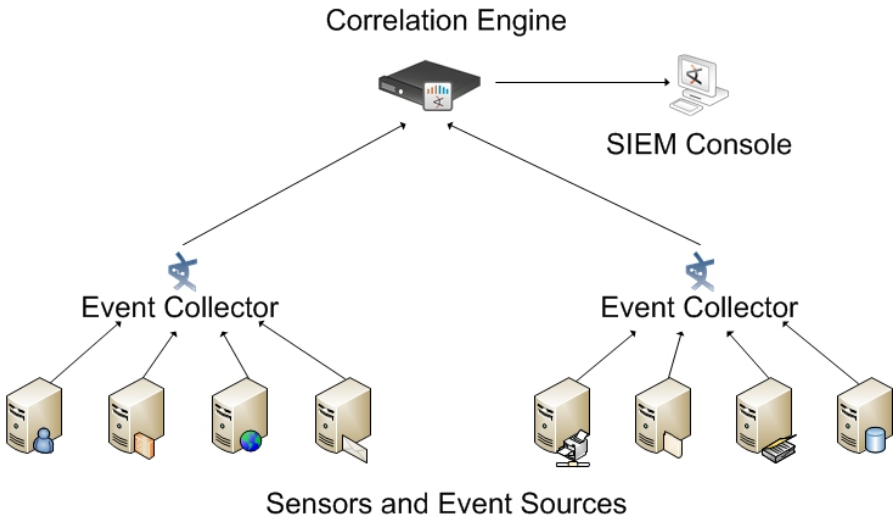

Figure 1 – SIEM Representation

System administrators commonly use SIEM systems to manage multiple security applications and to guarantee an automatic response to security incidents. The SIEM

provides not only real-time analysis and correlation of security events but also long-term storage, analysis and reporting of log data.

The main SIEM capabilities are:

- Data Aggregation: Events from many sources are aggregated by the SIEM, providing the ability to consolidate monitored data;
- Correlation: Data from the various sources can be correlated in order to determine meaningful events that would otherwise be considered independent and unrelated;
- Alerting: SIEM administrators can be immediately alerted based on the automated analysis of correlated events, therefore reducing the time interval between the start of an attack and the possibility of deploying countermeasures;
- Visualization: SIEM tools take event data and turn it into informational charts and tables, helping the analyst to identify activity that is falling outside the baseline pattern;
- Compliance: SIEM applications can be employed to automate the gathering of compliance data, producing reports that adapt to existing security, governance and auditing policies;
- Retention: The events collected by the SIEM can be saved in long-term storage to provide the necessary data retention for compliance requirements.

The correct operation of a SIEM system relies on the guarantee that events from the several sources arrive to the correlation engine correctly and inside an acceptable time frame. Therefore, it is of the utmost importance to ensure the resilience of the SIEM platform, composed of the sensors, collector agents and communication channels, in the case of failures in parts of the system.

Since the SIEM represents a central monitoring system on which security teams base their processes, it is a valuable target for attacks. A malicious entity that is aware of the existence of a SIEM in the infrastructure will aim at disrupting that platform before conducting the attack, thereby increasing the window of opportunity before that attack is detected. The importance of the overall security of the SIEM platform is proportional to the dependency the security team places in it.

To guarantee that all relevant information reaches its destination, one must account for accidental and malicious failures in the components of the SIEM platform and in the origin of the events itself, designing alternative ways to achieve the same level of available information in case of a partial failure in the infrastructure. Since the degree of interconnectivity between systems is increasing, we argue that it is possible to take advantage of those alternative communication channels to increase the reliability of the security monitoring processes and tools.

The MASSIF European Project [7], which provides the context for this thesis, aims to seamlessly integrate resilience into distributed SIEM systems by providing an enhanced framework architecture. The MASSIF SIEM system is modeled as a WAN-of-LANS, with several geographically separated facilities interconnected through a public network such as the Internet. There are core and edge components, with a Resilient Event Bus guaranteeing the communication between them. This thesis focuses on attacks against the edge components and LAN communication channels between those components.

1.1 Motivation

SIEM systems have become a fundamental security component of major IT infrastructures, public and private. The security drive behind investments in technology, something that is now of major concern to most Chief Information Officers (CIO), has put the SIEM at the center of security monitoring, making it an essential tool for security and incident response teams.

The problem with SIEM systems is that they can give operational teams a false sense of comfort, since security analysts are relying heavily on the accuracy of the SIEM to detect potential attacks, concentrating their monitoring efforts on a single platform. This represents a significant shift from traditional approaches based on multiple consoles, each dedicated to a specific component or technology, which enabled a distributed capability. Moreover, current SIEM implementations lack proper protection mechanisms capable of coping with advanced and focused attacks. By relying on the dependability of the sources and SIEM components, mostly without any redundancy, the detection of covert attempts to compromise the security properties of the systems is limited. These properties encompass the Confidentiality, Integrity and Availability triad (CIA). Targeted attacks, also known as Advanced Persistent Threats

(APT), in which the attacking party invests significant resources and time to achieve its goals, can detect the existence of a security monitoring platform and find ways to bypass it before launching a full-scale attack. As with any computer system, the SIEM also has frailties and design vulnerabilities that can be exploited. Even if those primary vulnerabilities are limited, the foundations of the SIEM and the simplicity of default approaches to event collection and correlation translate into easily discoverable attack vectors.

While some of the more common security measures, such as encrypted communications, may help to reduce the risk, they do not address the whole problem. To achieve that goal it is necessary to consider the fundamental processes to collect and correlate events, multiple fault scenarios and means to improve the way correlation rules and alarm triggers are designed. This thesis aims to define and implement techniques to improve the resilience of SIEM correlation rules by going further than traditional protection mechanisms, which have proven to be ineffective against sophisticated attacks. We intend to improve the effectiveness of alarm triggers in the presence of advanced attacks capable of compromising part of the SIEM components and/or part of the event sources. The idea is to increase and strengthen the possible manners in which attacks are detected by eliminating vulnerabilities and taking advantage of the multiple communication paths and connections between the components of the infrastructure. The major objective is to achieve a solution to improve the resilience of a SIEM system, specifically with regard to event collection, even in the presence of a partially compromised infrastructure.

This research intends to discuss possible attacks against event generation and delivery, as well as methods to circumvent the SIEM alert rules. We aim to provide a thorough and methodic solution to analyze and complement correlation rule sets, making them resilient to a limited set of compromised components in the SIEM system.

All the demonstrations and implementations were deployed and tested in a SIEM system connected to the corporate IT infrastructure of EDP, a Portuguese utility with responsibilities in the generation, distribution and commercialization of electric energy and natural gas. The implementation environment is not directly connected to Supervisory Control And Data Acquisition (SCADA) networks, instead monitoring the network where corporate systems reside, including Internet and e-mail gateways with public visibility.

1.2 Project Planning

The project described in this dissertation was developed over a period of twelve months, with sequential stages that built upon previous activities to achieve a coherent outcome. Our initial planning included four phases that consisted in: research and goal definition; studying and developing correlation rules; implementation; and, finally, the writing of this dissertation. The schedule aimed for a conclusion by the end of June 2013.

Throughout the project there were no major changes to these phases. The study of correlation rules consumed more time than we anticipated, resulting in a delayed start of the deployment phase. This factor contributed to the implementation being completed only in August 2013. Despite this schedule rearrangement, the writing of the dissertation started in June, as planned. The reporting of the implementation conclusions was concluded by the end of August, with just minor improvements and corrections being made in September 2013.

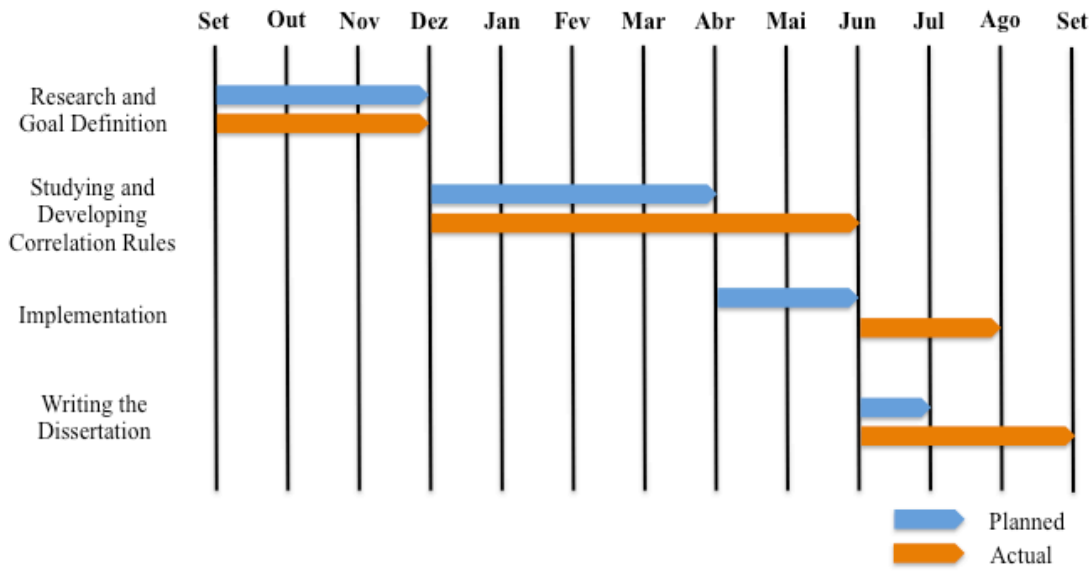

Figure 2 – Project Planning and Execution

1.3 Document Structure

The remainder of this document is structured as follows:

In Chapter 2 we provide an overview of a SIEM system, laying out a possible architecture and describing in some detail the components that are combined to achieve a security-monitoring platform. We also sum up relevant research in this field that can

be useful to contextualize the existing problems when there is a need to define, automatize and operate an information security team.

Next, in Chapter 3, we define possible attacks and faults that may impact a SIEM system, thoroughly emphasizing the different nature and consequently different approaches to deal with those threats.

Chapter 4 contains our proposal to improve the resilience and effectiveness of SIEM systems by improving the way correlation rules are defined and implemented. We provide strong reasoning to justify the fundamental changes in the way alarms are triggered to alert security teams of ongoing attacks. We also present AutoRule, an application to automatically process correlation rules, suggesting improvement possibilities and calculating a resilience score based on the strength of the defined conditions and the possible use of multiple event sources.

In Chapter 5 we demonstrate the improvements in correlation rules by presenting the results of applying our proposal to a real world application in a corporate environment. The ability to verify the effectiveness of the changes using actual security events from an operational infrastructure gives us additional comfort while also displaying opportunities for further developments.

Lastly, we conclude in Chapter 6, summarizing our proposals and outcomes of a sample implementation, while also pointing out further work to be done in this area.

Chapter 2

Context and Related Work

Although one of the key characteristics of a SIEM system is its flexibility, it is possible to define a reference architecture on which most SIEM implementations are based. The components may be rearranged according to specific limitations and objectives, but the information flows are maintained.

In this chapter we present the reference architecture for a SIEM system, detailing its main components. We also introduce the syntax used when defining correlation rules, allowing a better understanding of the possibilities and limitations of this tool. To contextualize our work we also refer to related work in attack detection, highlighting the useful contributions but also existing limitations in this area of research, especially when considering specific analysis of SIEM systems.

2.1 SIEM Architecture and Main Components

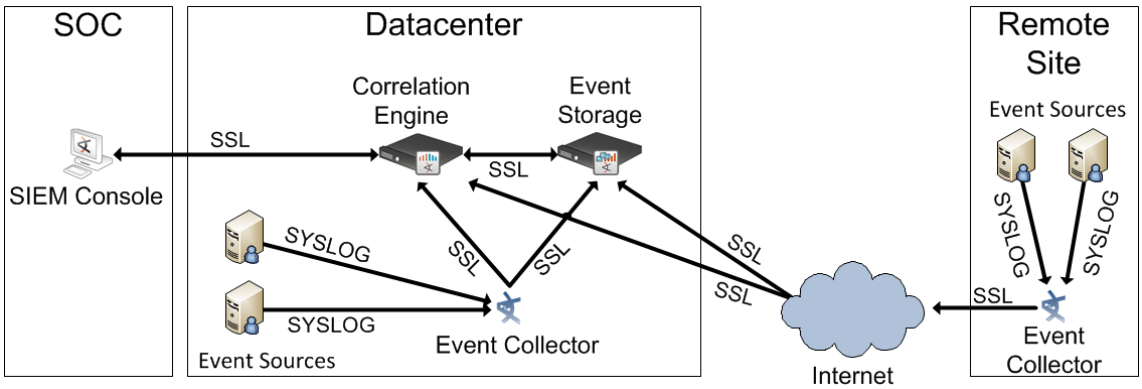

Figure 3 – SIEM Architecture

Figure 3 outlines a possible high-level architecture of a SIEM, encompassing both the central infrastructure as well as distributed network zones with relevant event sources. The most important connections between the SIEM components are

represented to indicate the flows of information, along with the most commonly used protocols.

Throughout the rest of this section we describe and analyze the components that are part of a SIEM system, making the continuous security monitoring possible, and the major reliability concerns with their operation.

2.1.1 Event Sources and Sensors

Although not a part of the SIEM system, the event sources are a fundamental part of the architecture as their capabilities, properties and location in the network are essential to the success of the monitoring effort. Event sources correspond to the existing infrastructure from which security events can be acquired. The complexity of implementing a computerized infrastructure is so great that, even within the same area of action, there are multiple differences in the way components are installed and configured.

The sources of events may range from the physical environment itself, for example the temperature being recorded by a sensor, to complex computer systems processing large quantities of data. What they have in common is that the teams operating them want to centralize the collection and processing of security events in a single platform, making it possible to efficiently manage the security operational context. This monitoring of security events is essential to detect anomalies in real time, triggering the necessary alerts for further investigation. Two possible examples of event sources are the authentication servers that generate an event for each authentication request; and the firewalls that log information from data flows including the source and destination addresses, network ports and the approval or denial of that network traffic.

There are typically multiple sensors spread throughout the monitored infrastructure, covering the various components. Those sensors are responsible for generating the security events, usually represented as blocks of text using a predetermined format, and sending them to the collectors. Hardware sensors are normally simpler in nature, usually measuring physical variables and outputting a single value that varies throughout the time. On the other hand, software sensors can be more complex, with the capacity to perform authentication procedures to access protected data.

Software sensors can be part of existing processes that generate security logs internally, as is normal in operating system processes, or dedicated components that act

as probes for the entire system and detect any changes to security properties. The security event log is maintained by the sensor and formatted so that the information can then be processed by the SIEM. Examples of software sensors are: the event logging process of the operating system, which records authentication procedures and any other actions performed by the users; and the processes responsible for verifying that certain conditions are maintained, for example that the antivirus agent is running, keeping a registry to enable the detection of changes in the security context.

In our work we have decided to consider that the sensors are not part of the SIEM platform but instead part of the system being monitored. The reasoning is that sensors are usually part of the event source processes and totally independent of the existence of the SIEM. The SIEM is responsible for collecting data generated by the sensors, either actively gathering that information or passively receiving it, depending on the type of event source and sensor.

2.1.2 Event Collectors

The event collector is the outmost component of a SIEM system. Directly connected to the event sensors, the collector receives all the raw data necessary to feed the correlation engine, performing a set of tasks that can range from simple event forwarding to aggregation. As we mentioned earlier, the way to collect security events from their sources can vary according to the specific manner in which the sensors are programmed and implemented. Some of the sensors periodically contact the event collectors and send the event data, while in other settings it is the responsibility of the collector to contact the sensors to retrieve the information. Either way, the collection process should be implemented in a secure fashion, ideally forcing both parties to mutually authenticate and establish a secure channel, especially if the security events contain confidential data that may reveal key information to possible attackers, disclosing existing vulnerabilities.

With respect to the event collector capabilities, the most important are filtering, aggregation and normalization. Since the sensors are not part of the SIEM platform, it may not be possible for the security team to adjust their parameters. Moreover, the sensor may be recording additional information besides security events, relevant for operational purposes but not from the security perspective. In these situations the event collector, usually placed closer to the sensors, can filter out unnecessary data to minimize the usage of communication channels towards the other SIEM components,

thereby reducing the load in the platform. With the same objective in mind, the collectors may also aggregate similar events, as long as it is possible to do so without losing relevant information. The collectors also have the responsibility of normalizing the events received from the sensors. The original events, also referred to as raw events, must be transformed to a format that is easier to process by the correlation engine. Metadata is also added to increase the information available for correlation rules. Parsing the raw events means organizing the text information into structured fields specified for that particular collector, depending on the type of events being processed. The collector is responsible for the correct forwarding of events to the other SIEM components.

2.1.3 Correlation Engine

The correlation engine is the brain of the SIEM system, capable of continuously analyzing thousands of events per second, matching them against a set of predefined rules. This component of the SIEM system is responsible for populating the user interface of the operators with the current security state of the infrastructure.

Since the data is stored in a local database and used for correlation purposes, it is important to define a reasonable duration for that storage, as a larger database translates into slower correlation and loading operations. Online events are stored in the correlation engine internal database and may be immediately checked against newly implemented rules. Events in the external databases are archived or offline, meaning that they are available for forensic analysis but the correlation engine cannot process them directly. The timeframe for each type of storage should be defined by policies, for example, stating that events should remain online for one month and then in the offline vault for an additional year. If a new correlation rule is created and the security team finds it necessary to test that rule against past events, it is possible to load events from the storage back into the correlation engine.

There is also the question of performance degradation if the number of events reaching the correlation engine increases past a predetermined threshold. The processing capabilities of the correlation engine are measured in events per second and the platform has to be designed with the size of the monitored infrastructure in mind. Nevertheless, SIEM platforms can scale either by increasing the computational resources of the machines or by adding more nodes. In the case of the correlation engine, if it is no longer possible to add computational resources, the solution may be to

have separate machines using the same event database, each one with a subset of the correlation rules, guaranteeing that the events are matched against all rules.

2.1.4 Event Storage

While the main reason for deploying a SIEM system is the capability of monitoring security events in real time, these systems also play a critical role guaranteeing the compliance requirements faced by major companies. Long-term storage of security events and logs not only allows a company to answer to legal requirements, but also enables the IT department to conduct forensic analysis after a security incident has been uncovered.

The logging detail depends on the objectives and legal constraints that the organization faces. Since unimportant events are filtered out be the collectors, the ones that reach the event storage should all be relevant and therefore maintained, but complex policies may be applied to optimize the storage process. For instance, the organization may choose to store events from specific sources for a longer period if there are particular guidelines to do so.

Since the event storage must in some cases abide to strict legislation, the events are also stored in their raw format, exactly as they were transmitted from the sensors. The processing done at the collector level could not only remove critical information but also have an impact on the integrity of the event, making it non admissible for legal procedures. To further guarantee this security property, the storage itself can possess integrity checks, for instance using cryptographic signatures, to validate that the data at rest has not been tampered. Storage solutions with information integrity guarantees are known as Write Once Read Many (WORM) and usually employ cryptographic checksums to detect changes in stored data.

2.1.5 SIEM Console

A team composed of security analysts and other technical experts must have access to the events and alarms to operate the SIEM system. A console provides that access, along with configuration options and correlation rules editors. The console is a software component installed in a laptop or desktop running a commercial operating system, without any particular requirements.

Using the console, the SIEM operating team can deploy, register and remove connectors, provided that the user has the necessary access privileges. Considering that

events are processed before being displayed in the console, the integrity of the information being presented on the screens is essential to the correct interpretation of those security events. Any errors may lead to either false positives or false negatives on the correlation engine, with potential catastrophic results to the overall security of the infrastructure.

2.1.6 Auxiliary Infrastructure

The SIEM system must rely on auxiliary systems for some fundamental functions that, in case of failure, can jeopardize its overall operation.

Time Sources – One of the most important supporting systems is the time source, which all components must contact to synchronize the clocks, usually using the Network Time Protocol (NTP). If the time source is attacked and starts to act arbitrarily, several correlations rules are jeopardized, since they rely on timestamp analysis. To guarantee the effectiveness of correlation rules, it is essential for the different components of the SIEM to be identically synchronized to a unique time source, with only a minimal drift between their internal clocks.

Authentication Services – The SIEM system collects events either by passively listening to data forwarded by the sensors, or by actively querying other systems for information. In the latter case, the SIEM system may need to provide valid credentials so that the sensors authenticate the request coming from the collectors. To make active event collection possible it is thus necessary to have authentication authorities to validate the provided credentials.

2.1.7 Network Channels

The entire flow of information has to be transmitted between the various modules of the SIEM architecture. The number and type of network segments varies according to the defined architecture, with some of the components placed together in the same machine, thereby eliminating the need for a network communication path. The type of channels and communication protocols has to be adapted to the specific technologies. It is common for events to be transmitted between the sensors and the collectors mostly using the syslog standard over the User Datagram Protocol (UDP), with all connections between SIEM components using Secure Socket Layer (SSL) for security reasons.

Syslog is a standard for data logging that labels messages to indicate the type of source and the severity of that message. By using this standard for most security event

sources it is possible to define uniform filtering and aggregation rules throughout the event collectors, taking advantage of the labels in each message. The syslog standard itself does not encompass security protections as data is passed in clear text throughout the network, without any confidentiality assurance. This is visible in Figure 4, a network capture of a transmission from a sensor to an event collector. Moreover, the employed UDP is a best effort protocol that does not guarantee the delivery of messages. Several efforts have been made to improve the resilience of syslog [9] but, since the standard has not been changed, most implementations rely on risk mitigation strategies such as establishing end-to-end secure channels.

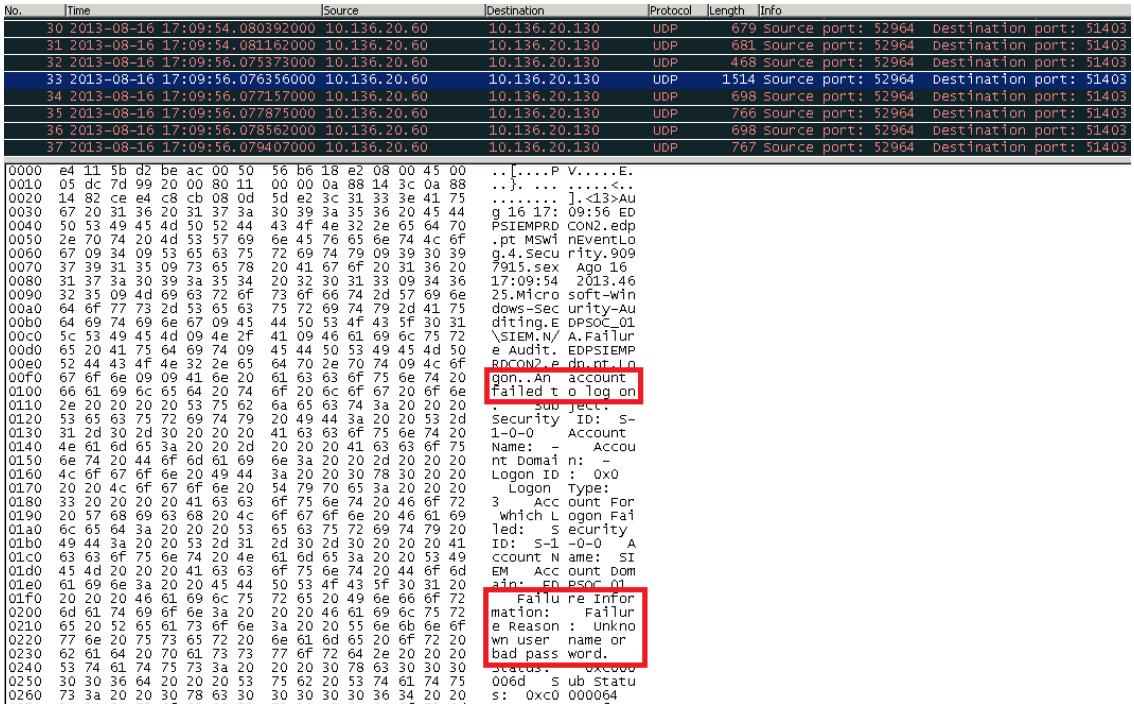

Figure 4 – Event Capture

2.2 Terminology for Correlation Rules

Before presenting and discussing our research, we briefly introduce the chosen terminology used to define the correlation rules. Throughout the document we will adopt the same terminology of the SIEM from ArcSight, as this was the product employed in the implementation section. We will describe the main concepts and syntax for specifying events and correlation rules.

Events – An event may be an action or measurement collected directly from a source system, in that case being equivalent to an entry in a log file.

Each event has a set of properties, translated into alphanumeric or Boolean fields. These properties encompass all the relevant information to determine the origin of that event, for instance the source and destination IP addresses, the type of event and the event outcome, which makes it possible to determine if the described action was successful or not. The severity of an event is determined using its properties and can vary depending on certain factors (for instance, the same type of event can be classified differently depending on its outcome and/or the network segment where it originated).

Figure 5 exemplifies a subset of the properties that constitute a security event. In this case the event was generated at a file server, thus the most relevant fields identify the type of event, source and destination of the request and some additional information regarding the authentication process. Due to privacy concerns, some of the fields in Figure 5 are redacted.

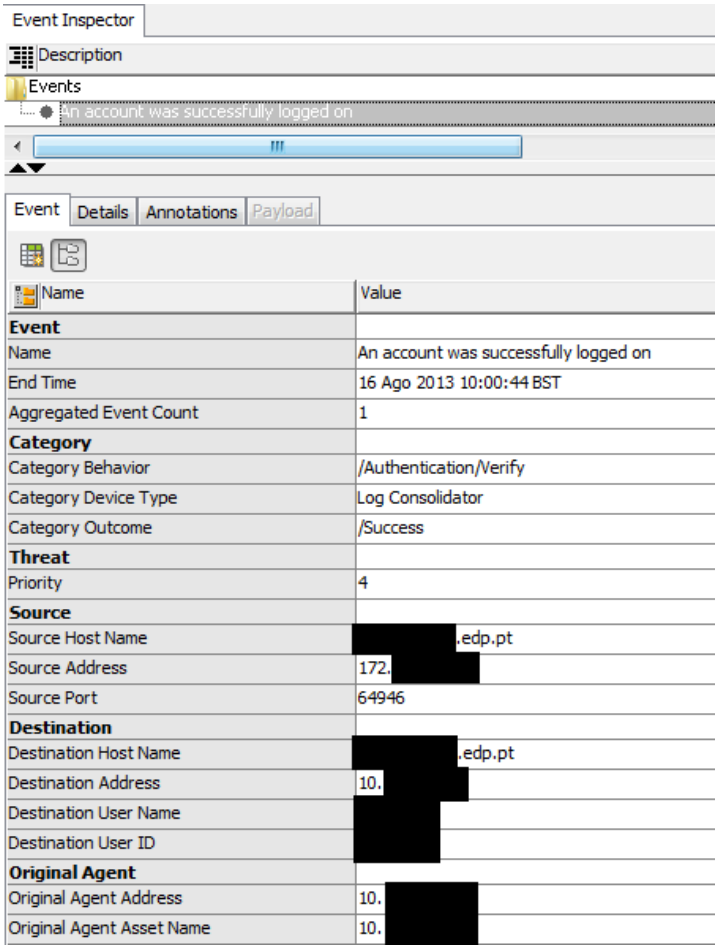

Figure 5 – Sample Event

Correlation Rules – One of the main purposes of the SIEM system is to make use of security events being collected from multiple sources, combining related information to trigger alarms. The correlation engine is responsible for that task and

must therefore be programmed to perform these actions, using correlation rules. A rule is based on a set of operators and expressions to process events, verifying conditions and, if necessary, triggering resulting actions.

Figure 6 shows an example of a correlation rule defined in ArcSight. The conditions shown here define the filtering parameters to trigger the rule, using information from various fields of two separate events.

```
Matching Event:
( Brute_Force.End Time <= Login_Success.End Time AND
Brute_Force.Destination Address = Login_Success.Destination Address AND
Brute_Force.Source Address = Login_Success.Source Address AND
Brute_Force.Source User Name = Login_Success.Source User Name )

Brute_Force :
( NotInActiveList("Trusted List") AND Category Behavior =
/Authentication/Verify AND Category Outcome = /Failure )

Login_Success :
( Category Behavior = /Authentication/Verify AND Category Outcome =
/Success )
```

Figure 6 – Correlation Rule Example

If the conditions are met, the correlation rule is able to perform automatic actions to immediately respond to possible attacks. Figure 7 shows an example of such actions, including setting information in the event fields and adding the suspected source to a list of monitored entities.

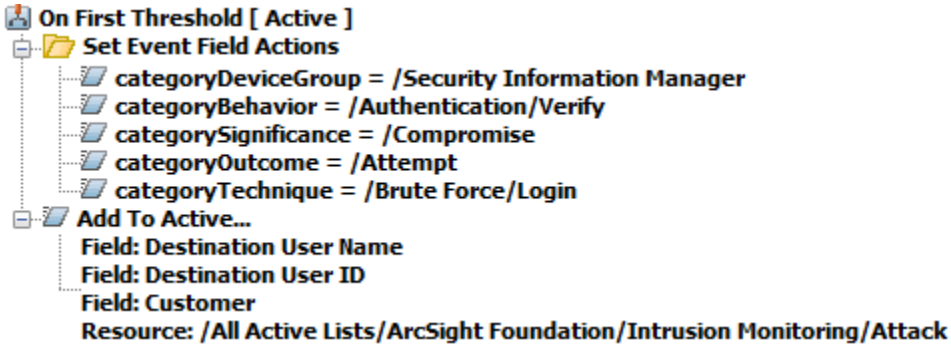

Figure 7 – Correlation Rule Actions

Correlated Events – An event can also be a change to a variable or condition calculated by the SIEM system, for instance a change in the event flow or the combination of events. Correlated events are always created as a result of a correlation rule, which means that they are the consequence of suspected patterns. Considering the

example in Figure 6, the SIEM would detect a successful authentication after an attempt to perform a Brute Force attack using the same credentials. The actions defined in Figure 7 would tag that correlated event, stating that it corresponds to an attack, while also adding the possibly compromised credentials to a monitoring active list.

Figure 8 shows an example of a correlated event, outlining the previous events that were identified and combined by the correlation rule.

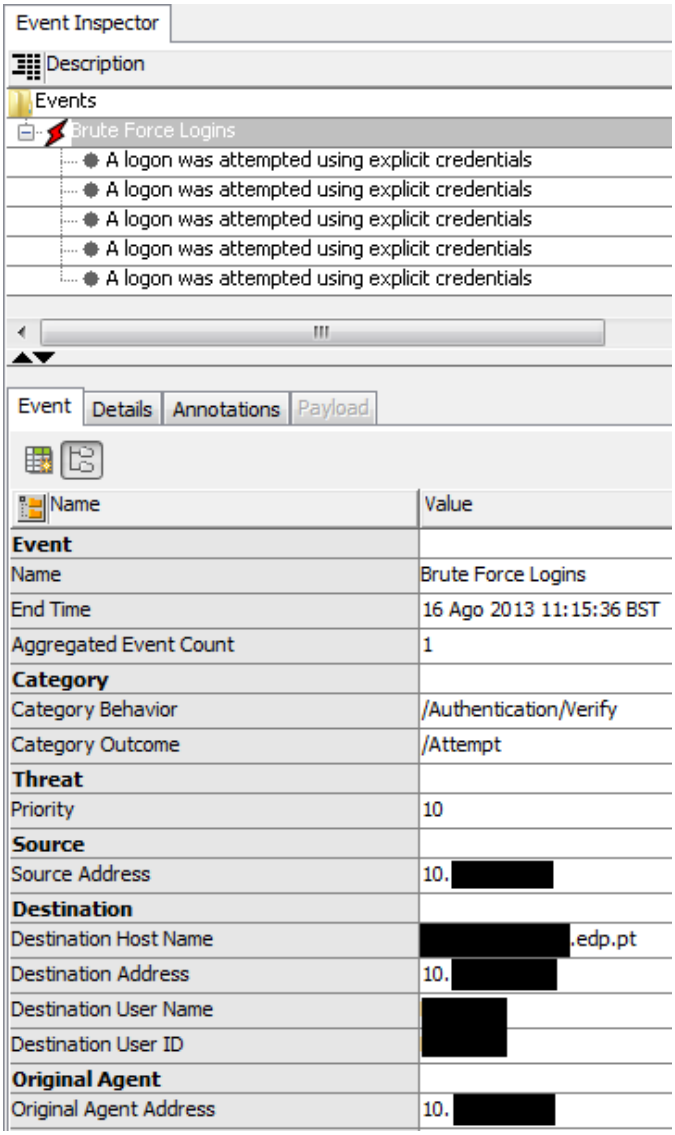

Figure 8 – Correlated Event

Actors – Events can have information from one to three actors, depending on the type of event. If it is an internal event generated inside the SIEM or collected directly from the source, such as a change in event flow, the only actor is the source of that event. On the other hand, if an event represents some kind of interaction between systems, there are two actors: the source and the destination. An example would be the

change of a certain configuration or a client-server transaction, where the event would show both where the request was made and also at the source of that instruction. Lastly, there are certain situations where one action has a source, a destination and also a third party. The most obvious example is an authentication process using a method based on a user repository. A system may need to validate the provided credentials using a third party, for instance a Domain Controller or RADIUS server. The resulting event would register the source, the destination and the repository that was consulted in the process. If the event is being collected from the authentication server, it will identify the source and destination as mentioned before, as well as its own address.

Operators – The operators are used when defining correlation rules. They are common logical prepositions, where the most frequent are: EQ (equal), NE (not equal), GE (greater or equal), LE (lesser or equal), GT (greater than), LT (lesser than), AND, OR, NOT, IN.

Active Lists – Another aspect to consider when defining correlation rules is the ability to use and update dynamic sets of information, known as Actives Lists. For instance it is possible to define a set of actors that share the same classification or characteristics. The lists can be employed to manage exceptions (e.g., whitelists of trusted systems) or limiting the scope of some rules to configured sources. The dynamic nature of lists makes them a powerful tool to enrich correlation rules. Active Lists can be fed by correlation rules that add information from collected events, such as IP addresses or hostnames. The information can be removed from an Active List either by an explicit action or using time-based triggers, for instance by defining a when a list entry expires. A simple example of the benefit of having Active Lists is to maintain a set of suspicious actors based on past events. Let us consider that a possible attack was detected; even in the case that the source of those events was not compromised and the suspicious actions originated in wrong system configurations, it could still be useful to add that system to the suspected sources list. Certain sophisticated threats use covert actions to disguise attacks, carefully probing the surrounding components before launching an attack. In that case, detecting that something is wrong is only possible by continuously monitoring that asset over a long period of time, triggering an alarm purely based on the recurrence of abnormal behavior.

Network and Asset Model – Building upon the concept of active lists, the SIEM allows the creation of a network and asset model of the monitored infrastructure. The Asset Model consists of information from the monitored infrastructure, ranging from

the operating system to the critically of that specific system or equipment. The Network Model includes not only the network addresses of the components, but also the architecture and the manner by which systems are connected. The information in these models can improve the efficiency of correlation rules by associating event sources and determining the relationship between them. For instance, one can consider the connections established between servers and network equipment to determine possible attack paths or to discover alternative communication channels to perform event collection in the presence of faults in the network. The asset model is also an important source of information, enabling the SIEM to determine if a certain asset is, for example, a web server, a router or a firewall. The type of attacks and accepted behavior can change considering the category of the asset, therefore empowering the definition of correlation rules. The severity of an event can also be adapted depending on the location of the target, increasing as the attacker is able to compromise components nearer the core of the infrastructure, since it is different to detect an abnormal behavior in an externally-faced server in a Demilitarized Zone (DMZ) or in an internal server behind the firewall. Both the network and asset models can be updated automatically using information from sources such as vulnerability scanners.

2.3 Related Research in Attack Detection

While SIEM systems have seen a fair amount of implementations in several industries, the major focus of academic publications is still being put on reliable pattern discovery at the level of Intrusion Detection Systems (IDS) or Intrusion Prevention Systems (IPS). Contrary to the SIEM system, these platforms are based purely on the analysis of data from a single asset, not on collecting and correlating events from various sources.

Todd et al. address the topic of accurate attack pattern discovery in [14], where alert verification evasion methods are presented. This paper demonstrates that it is possible to exploit the verification step of the intrusion detection process, modifying the behavior of a compromised server by crafting forged response messages to make a successful attack appear unsuccessful. The authors propose a set of methodologies that can improve the detection of attacks, even in the presence of forged communications, by analyzing the payload of each interaction with the server or by relying on mostly static vulnerability catalogs. The shortcomings of this approach are the implementation

complexity and the fact that it does not address attacks to the IDS itself. Moreover, even in a scenario where it is possible to implement the suggested improvements, encoding the payload could still thwart them and exploiting previously undiscovered vulnerabilities, also known as 0-day vulnerabilities, would still be a possibility.

The overwhelming number of events and alerts generated by the increasingly high number of implemented IDS/IPS has added to the necessity of strong correlation rules, capable of minimizing the number of false positives, thus increasing the efficiency of security analysts and network operators. In [16], Valdes and Skinner present a probabilistic approach to alert correlation, using a mathematical framework for correlating alerts. Starting from the premise that current IDS/IPS generate an overwhelming amount of event data from fundamentally different sources, the research intends to deepen the heuristic approaches to address this problem (which was previously presented by the same authors [15]). An alert fusion method is presented using Bayesian Inference to combine common features and similarity measures, creating threads for event aggregation. The most striking efforts were to prioritize errors originating from invalid requests, while downplaying errors caused by requests to already unavailable services, thereby reducing the number of false alarms when a component is known to have failed. By presenting an incident class similarity matrix, the authors are able to clearly define their view on how events can be aggregated, making it possible to uncover attack patterns based on normally consecutive actions by the attacker. Although results are encouraging, with a reduction in alert volume between 50% and 66%, the fundamental problem of guaranteeing the resilient delivery of events to the SIEM system still exists. The research is focused on diminishing the number of alerts while the SIEM is operating correctly, not addressing the resilience problem in the presence of accidental or malicious faults.

In fact, few authors have addressed the problem of getting events from multiple sensors to the collectors and then to a correlation engine in a resilient fashion. A starting point is related with detecting when components of the SIEM solution are under attack or have been compromised. Oliner, Kulkarni and Aiken propose an interesting solution based on detecting time-correlated anomalies in groups of identical assets, which are called communities [11]. By using information from an aggregated source, instead of individual components, the authors demonstrate that it is possible to reduce the proneness to false positives that is usually found in anomaly-based intrusion detection, while also reliably identifying when a subset of a community is having an abnormal and

potentially malicious behavior. A scoring approach is proposed to detect anomalous behavior by correlating events from multiple hosts, which can be used to establish voting mechanisms in order to eliminate false data from the SIEM event collection process. For instance, a client whose response times are unusually high, based on historical data and information collected from other members of the community, may indicate an anomalous score. The rational is that it is strange if only a subset of clients in the same environment start to behave differently, and this can be identified using aggregated data and time-correlated anomalies to determine if the change becomes more prevalent in that community. While useful to create alarms regarding possible security breaches, the paper does not address event delivery or redundancy considerations. The SIEM is able to generate more specific alarms considering a reliable and steady flow of events, but is none the wiser about an attack if those events are suppressed or modified at the sources.

Since this research is being conducted in collaboration with a company responsible for critical infrastructures, it is relevant to contextualize security monitoring in such environments. Specifically, it is important to understand the particulars of Industrial Control Systems (ICS), such as Supervisory Control and Data Acquisition (SCADA) systems. All entities must take their information system security seriously, but the impact of a successful attack is completely different in those critical infrastructures. Most organizations measure their business impact purely on financial losses, either direct or indirect, for instance through intellectual property theft or public image impact. On the other hand, in areas such as the military or utilities an attack may lead to loss of human lives, therefore raising the bar on the security alert accuracy requirements. However, even with those risks present, economical reasons have pushed those industries to a transition from closed, proprietary systems, protocols and networks to more open environments. That transition has exposed critical systems to cyber-attacks, connecting once isolated systems to public networks thus introducing the need for a novel approach in ICS security.

In [3] Briesemeister et al. focus on the regularity of traffic and the limited number of protocols present in ICS environments, showing a combination of signature based detection coupled with Bayesian methods and learning-based anomaly detection. The notion of network traversal attacks is also introduced, defining how an adversary may exploit trust relationships among hosts to attack high-valued targets that the attacker would otherwise not be able to access directly. These attacks are made possible not only

by the connection of critical infrastructures to public networks, but also because the data from those critical systems must be imported to corporate applications. The fact of the matter is that business priorities have driven corporate and industrial networks closer, creating a mesh of interconnections that is hard to control and even harder to protect against cyber-attacks. The authors lay out a conceptual architecture for connecting corporate and control networks using a dual firewalled demilitarized zone or DMZ where certain systems are accessible by both networks in a controlled fashion. The defined critically of the systems increases as an attack reaches deeper into the internal perimeter, creating alerts based on correlated events in different network segments, as if the SIEM is capable of tracking the movements of an attacker as the attack progresses. The results from this research point out not only the importance of correctly categorizing assets and defining correlation rules, but also the relevance of an advanced visualization tool to aid monitoring efforts. The authors give indications to define the architecture of a SIEM system in an ICS environment, but aim to enrich correlation rules and event information relying only on the same source, which does not address resilience issues.

Particularly interesting to the context of our work is the notion of collaborative SIEM operation between different domains, explained by Aguirre and Alonso in [1]. The fact that IT networks and domains must remain independently managed and supervised, albeit connected, raises questions on how to automate the sharing and correlation of relevant security events. Utilities companies have their ICS and corporate networks clearly segregated. Nonetheless, the communication channels between them, as well as common vulnerabilities, mean that sharing alarms among the SIEM systems would benefit both parties. Although this paper does not address the issue of reliable event collection, the notion of sharing information from multiple domains, correlating it in one SIEM system, is helpful. By analyzing the communication flows between separate domains we may find interesting alternative methods to collect information, since a simultaneous compromise of assets in more than one domain is less plausible.

Chapter 3

Faults in SIEM Systems

This chapter identifies possible faults that might occur on the various SIEM system components. The fault model is divided into two major classes:

- Accidental Faults: faults causing some or all of the components in the SIEM system to stop operating correctly. Faults occur without the direct involvement of an attacker;
- Malicious Attacks: faults introduced intentionally by malicious sources, with the goal of compromising the confidentiality, availability, integrity or timeliness of the SIEM system.

Throughout this chapter we describe and discuss the effects of faults in the components presented in Chapter 2, while also putting forward possible mitigation techniques. The idea is to present clear and systematic methods to deal with faults, enabling the timely discovery of those faults and, if possible, ensure the continuing operation of the systems. When considering malicious attacks, we turn to intrusion tolerance concepts [17]. Accepting the possibility of a successful attack against part of the infrastructure leads the system architect to devise solutions that will enable continual operation in those adverse situations.

3.1 Accidental Faults

This section explains how accidental faults can have a significant impact on the correct operation of a SIEM system. The emphasis is put on the most relevant type of accidental faults: crash faults that result in the halt of one or more SIEM components. Although not part of the SIEM system, crash faults on the event sources should also be addressed when designing correlation rules, as redundancy in event collection should be oblivious to the origin of event absence.

3.1.1 Event Sources

When analyzing a fault, it is essential to define its impact, determining if it resulted in a complete shutdown of the component or, on the other hand, if the failure only affected the event generation. For instance, if we consider a computer host – either server or a workstation – it is important to determine if the crash fault resulted in the total arrest of that host or if it was only the logging process of the operating system that failed. Since in some cases the sensor collects data from logs, if the logging process fails it might seem that the host is down when in fact it is still operating normally.

To correctly determine if the host failed it is thus necessary to consider the collection of events from that source in more than one manner, either by using two sensors operating differently or by using data obtained from other connected sources. Let us consider that a sensor attached to a host stops receiving events for a long period while still responding to heartbeat messages, indicating that the cause is not a sensor malfunction. The first step would be to have a configuration that indicates if it is acceptable or expected for that host to be silent. If that is not the case, then one must consider that the host failed. To corroborate or refute that assumption, there can be correlation rules to find out if any events related to that host are being collected in another source, for instance, a network component to which the host is connected. If the host is generating network activity or communicating with other hosts, it proves that it is still active and that the process generating the monitoring events or logs has either crashed or was maliciously shut down.

3.1.2 Sensors

Since most of the times there is only one sensor for each monitored host or network segment, the crash of a sensor may result in the loss of events generated by the monitored component. From the collector's point of view, it is as if that component no longer has any activity. To account for crash faults, the communication protocol between sensors and the collector should include periodic heartbeat messages, either proactively sent from the sensor or as a response to a request from the collector. With this mechanism it is possible for the collector to identify crashed sensors and generate an alert that will be received by the correlation engine.

One of the most often suggested approaches to increase the fault tolerance of a system is to make redundant services available by duplicating components. If we employ two identical and independent sensors, and account only for accidental faults,

the resulting reliability would be improved because it is probabilistically less likely that both sensors fail at the same time. There are two immediate consequences of this approach: one must build ways to deal with duplicate events; and the cost of the solution may rise abruptly, as sensors are one of the dominant components in terms of quantity.

If the sensor is hardware based, the duplication of sensors carries a steep cost when one considers large and complex systems. On the other hand, if the sensor is a software component there is an increased need for computational resources, and they may have common vulnerabilities, possibly making crash failures correlated. Moreover, having more sensors collecting data, either by polling or pushing, results in an increased system load at the host. In already burdened systems, adding this load could be unfeasible.

An alternative is the use of event correlation, an intelligent rule set to deduce events from a host when the sensor monitoring it crashes. Although it may be impossible to obtain the same amount of information with this approach, it mitigates the failure of a sensor, albeit with reduced data quality, with small extra costs or system load.

3.1.3 Event Collectors

Although in most cases it is possible for a sensor to forward events to multiple collectors, most SIEM products are unable to effectively cope with duplicate events. Therefore, the result of implementing more than one collector for the same event source would be in an unmanageable flow of data to the correlation engine, not to mention the performance and capacity issues. With this restriction in mind, we consider that the foremost solution to address the crash of a collector is to employ smarter correlation rules in order to gather information from a source by analyzing events from other adjacent sources.

For instance, one can consider a UNIX server and a Windows client workstation as the event sources, each with its specific sensor forwarding information to independent event collectors. Despite the operating systems being different, an interaction between them generates events at both ends. Therefore, the separate collectors for Windows and UNIX systems would be receiving similar events. In case the collector for the UNIX system has a fault resulting in a crash, it is still possible to determine part of the activity of the server by analyzing the events being forwarded by

the client. If we extrapolate this principle to all the components in the monitored infrastructure, it is possible to discover similar situations in which events from one source can be correlated and processed to deduce activity from other sources, in case of an accidental fault in one of the event collectors.

3.1.4 Correlation Engine

As the storage and archiving of security events is guaranteed by other SIEM components, the crash of the correlation engine should not result in event information being lost. Using the buffer capabilities of the collectors, the correlation engine would receive all events after resuming operation after a crash, processing them at that time. However, during the downtime of the correlation engine, the security operators would not receive alarms or any correlated events. The option available, besides the costly and complex replication of the core engine, is to directly access the event storage, analyzing the raw event logs. While this is not a perfect solution and drastically increases the difficulty of detecting malicious behavior, it is possible to improve the efficiency of monitoring operations by preemptively defining complex queries in the storage components, mimicking the correlation rules in the correlation engine. Although queries made over raw events are slower and less effective, it is an acceptable option for the situations where the correlation engine is unavailable for a short period of time as a result of a crash fault.

3.1.5 Event Storage

If storage components fail as a result of a crash fault, it is necessary to guarantee that no events are lost during the downtime. The straightforward solution of duplicating the storage databases is usually not acceptable due to the high cost of storing large quantities of data.

The common approach to deal with this risk is to enable data buffering in the event collectors, thereby ensuring that events are not lost due to momentary failures in the storage components. The amount of storage space available in the collectors, together with the rate of event generation, will determine the maximum admissible downtime without information loss.

3.1.6 SIEM Console

A crash on the SIEM console inhibits the operators from accessing relevant information and alarms. However, since the software can be easily installed in other stations and there are normally multiple consoles running at the same time, we do not anticipate the crash of a SIEM console to be a relevant risk.

3.1.7 Auxiliary Infrastructure

The SIEM depends on related systems for some of its functions, thereby needing to cope with crash failures of those systems.

Time Sources – Internal clock synchronization has to be performed regularly by all SIEM components to ensure that the clock drift does not become a problem. Having redundant time sources certainly decreases the risk but, since the synchronization requirements are not very strict, the difference between clocks is unlikely to pose a problem unless the time source remains unavailable for a long period.

By using the SIEM system itself to provide alarms in case of failure of the time source, the operators would have time to reestablish the service before it turns into a major incident. Nevertheless, basic integrity checks can be implemented, such as verifying the timestamp of an event against the system clock of the component receiving it, allowing an adequate tolerance for clock drift, but triggering an alarm if the difference in timestamps is too great.

Authentication Services – At least part of the event collection process is dependent on the correct operation of authentication services, usually LDAP or RADIUS. It is considered a best practice to have a central identity repository and authentication services, in order to guarantee the compliance with existing security policies regarding for instance user control and password complexity rules. However, it would be advisable to define a local user, preferably with a one-time password, to ensure that access to the SIEM system is always possible, even in the event of failure of the remote authentication services.

3.1.8 Network Channels

The failure of the communications network used by the sensors to collect events from the sources and to forward those events to the SIEM system is a major concern when developing resilient event collection architectures. While in case of sensor or host failure it is possible to infer information from related sources, the only option to prevent

failures in the network is to replicate the link or create a mesh network between the various components of the system, allowing multiple paths between any two nodes. Both solutions represent added cost and complexity, especially when considering that hardware sensors may be very simple components, making it extremely hard to configure alternative routing paths and having multiple network interfaces.

Naturally, the solutions presented above to deal with sensor and event sources crashes may also represent a way to overcome network failures. For instance, if we consider that there may be more than one sensor retrieving events from a single source, it is possible to ensure that those sensors are not sharing the same network medium, thus guaranteeing protection against limited failures in the network.

3.2 Malicious Attacks

Contrary to accidental faults that can be statistically predicted and addressed with normal fault tolerance principles, malicious attacks are deliberate attempts to compromise the security properties of a system. In the context of the SIEM system, the main motivation of an attacker will be to hide ongoing or future attacks. This objective may be attained either by making the platform unavailable or by compromising the collected information in a way that alarms are not triggered. Either method, if successful, results in the absence of alarms to the security team operating the SIEM, keeping them unaware of any anomalies.

Therefore, the most important security properties in this context are integrity and availability. This does not mean that an attacker could not profit from breaking the confidentiality of data flows, as they can include valuable information regarding the major components of the infrastructure and even the network topology. However, the confidentiality vector can be covered by standard approaches such as encrypted communication tunnels.

Event sources are not contemplated in this section focused on malicious attacks. The reasoning for this apparent inconsistency is that, contrary to the handling of accidental faults, dealing with malicious attacks involves changes in the architecture and/or source code. As the event sources are independent from the SIEM, and with most vendors not allowing changes in their products, the ability to take those security steps is very limited. On one hand, malicious attacks that result in the total arrest of the event source can be dealt with using the mechanisms proposed to cope with accidental faults,

meaning that the resilience is added resorting to neighbor systems and components. On the other hand, if the attackers target the integrity of events, and considering the mentioned limitations to changes in those components, one must rely on strong correlation rules to detect incoherencies and discard the events from that source.

3.2.1 Sensors

When an attacker has compromised a sensor, the generated events can no longer be trusted and should be discarded by the collector. The problem is that the collector may have no way of knowing that the sensor was corrupted and thus keeps forwarding those events to the correlation engine.

The attacker may employ distinct methods according to the intended outcome of the attack. An event duplication attack may target the availability of the system, requiring additional computational power to process a higher number of events. On the other hand, if an attacker intends to trigger false alarms to draw attention from the security analysts, he might try to manufacture events using information collected from the infrastructure. If the purpose were to hide ongoing attacks by suppressing alarms, then the attacker would have to elude the monitoring processes by making sure that the events do not reach the correlation engine.

Fundamental Sensor Security – Although sensors are simple components by definition, there might be different levels of sensor complexity, as we mentioned earlier. It is important to guarantee attestation, for instance, by recurring to a Trusted Platform Module (TPM) chip to verify the integrity of the software, with a signed version being kept on a secure ROM as a safeguard, making it possible to bootstrap a compromised sensor, returning it to a trusted state. Software based smart sensors should also run on top of trusted hardware with adequate protection and proper hardening processes at the different layers, specifically at the operating system level.

Information Integrity – An attacker may compromise a sensor in order to corrupt the information generated and sent to the collector. The attacker might alter the content of events, rendering them useless and impossible to process by the collector, or cleverly manipulate that data to either mask an ongoing attack or generate false positives that will flood the security analysts with alerts, making them unmanageable. Once again the solution for this problem can be based on having multiple sensors for the same host or network component, with the collector being responsible for managing a

voting mechanism capable of detecting outliers and discarding their information. An alarm can also be created when such outliers are observed.

Identity Spoofing – The events generated should carry within them a unique identifier of the sensor. If an attacker is capable of determining valid identifiers, then it is possible to impersonate other sensors thereby creating false positives or, more interestingly, making the collector assume that a sensor has been compromised. The effect would be that events from that sensor start being ignored and a recovery process initiated, if one is available. A possible solution is to employ cryptographic methods to generate that unique identifier or to sign the events.

Time Based Attack – After compromising a sensor, an attacker may be able to delay the transmission of events to the collector, making them temporally invalid and useless for the correlation engine. To perform this time based attack, it is not only necessary to be aware of the time frame inside which an event is still valid, but also to selectively delay some of the packets (the events), while immediately sending others (heartbeat messages).

Another option is to manipulate the timestamp of the event, achieving the same goal of rendering an event useless to the SIEM system. This method implies that the attacker knows the structure of the information and is capable of modifying part of the packet without compromising the integrity of the information. Once again, cryptographic methods could be used to prevent modification to the data, within the possibly limited processing capabilities of sensors.

3.2.2 Event Collectors

Although all the components of the SIEM system are susceptible to being attacked, some of its components are more exposed than others, like the event collector, as it is generally placed outside a safe perimeter. While the correlation engine and event storage are usually located in a datacenter or other protected network segment, the event collector is frequently installed at remote locations, closer to the event sources.

Due to their event processing and aggregating capabilities, the architectural decision of placing collectors closer to the sensors can result in significantly less network traffic. Moreover, since the collectors have increased computational capabilities when comparing to sensors, they can also implement mechanisms to secure the communication channel to the remaining SIEM components. On the other hand,

placing components of the SIEM outside the controlled and more reliable perimeter of the datacenter increases the risk of attacks.

As with any other software piece, vulnerabilities are bound to exist in the event collectors. Attackers may take advantage of these vulnerabilities to hamper the expected flow of events or the actions performed on those events by the collectors. It is also important to note that the attacks against sensors, described above, can also target event collectors, with the added risk of impacting events from multiple sources, as these components act as information aggregators.

3.2.3 Correlation Engine

It is indispensable for the correct operation of the correlation engine that the events arrive on time and their contents unaltered. All events receive two timestamps: one at the source and another when they reach the SIEM system; therefore it is possible to calculate average delays in event delivery as well as correlating events according to the source timestamp, reducing issues related to unreliable communication protocols.

If an attacker gains access to the correlation engine, it is possible to disable alarms, change correlation rules or even to alter the information presented to the operators, displaying past events as if they are recent.

An approach to minimize the risk of those attacks is to have an authenticated configuration, comprising the rule set and other relevant configurations, and periodically loading it from a secure location, for instance using on-chip cryptographic capabilities. The communication between the correlation engine and the SIEM console must also be secured to guarantee the authenticity and the integrity of the information.

3.2.4 Event Storage

The events arriving to the SIEM platform must be processed in order to detect ongoing attacks and to trigger alarms, but must also be stored in a secure vault for compliance reasons. Those repositories are essential for future investigations, forensic analysis and even to load events back to the correlation engine if new rules are defined and there is a need to verify those rules against historical data.

In heavily regulated sectors it is usual for authorities to demand data handover while performing investigations, either against a specific company or against individuals who could have accessed critical information through the computing systems of that company. In such moments it is of the utmost importance that the data

can be proved to the authentic, as well as readily available. Therefore, when considering secure event storage, one must focus mainly on the integrity and availability attributes of information. To achieve this goal, one can resort to available techniques like cryptographic signatures to detect data tampering and carefully chosen archival methods to reduce recovery times.

3.2.5 SIEM Console

The SIEM console itself poses security risks attributable to common software vulnerabilities. While the SIEM system core components are placed inside a secure perimeter, the console is commonly deployed in regular workstations, connected to the company's private network and the Internet. Specific firewall rules are then created between the office and the datacenter networks to guarantee the necessary accesses.

As with any other software component, the console has vulnerabilities, some of which are present at the middleware and operating system levels. For instance, the ArcSight console is based on JAVA, a technology with numerous security issues being uncovered frequently, some of them critical [12]. Therefore, if an attacker is able to access the workstation connected to the Internet and compromise the SIEM console, he can either completely disrupt the information being presented or, more interestingly, present false information to trick the operators. Even if the Internet access is cut off, a malware could still be introduced using something as simple as an infected pen drive. An entire scenario based on infecting a computer network with malware in order to display false information to system operators has already been put in place in the infamous Stuxnet attack [4].

More critically, as we mentioned, the workstation where the console is installed might present a path between the Internet and the datacenter network. If an attacker is able to circumvent existing protections, it may be possible to have direct access to the SIEM core components from the outside network.

3.2.6 Auxiliary Infrastructure

If an attacker is aware of the services on which the SIEM system depends to perform correctly, he may try to exploit known vulnerabilities on those services. Since the systems providing those services are usually shared infrastructures and, more importantly, rarely observe security policies as strong as those imposed on the SIEM, they are more exposed and therefore statistically more susceptible to being attacked.

Time Sources – By compromising the time source or the clock synchronization process, an attacker might be able to disguise attacks by making them seem as if successive events occurred a long time apart. To accomplish this, an attacker could, for example, use a compromised time source to synchronize two systems, feeding incorrect information to one or both, resulting in a significant difference in their internal clocks after the synchronization process ends. When the sensor uses that internal clock to timestamp the events, two simultaneous events would appear to have happened at a significantly different time, thereby bypassing time-based correlation rules.

One way to overcome this attempt to compromise data would be to apply algorithms to detect, at least, if the source timestamp is more recent than the SIEM timestamp and if the delay between time stamps is acceptable, since events are time stamped at the source and then again upon entering the SIEM system. Considering that it might not be possible to ensure clock synchronization of the source, one must rely on an analysis made by the SIEM, taking into consideration possible delays in the transmission.

Authentication Services – If an attacker is able to disrupt the authentication services, there may be relevant impact to event collection processes. While the more elementary collection methods are based on the sensors forwarding events to the collectors, there are more advanced and secure protocols that require authentication. In those cases, the sensors and collectors would be unable to mutually authenticate and the events would not be delivered.

An even more direct consequence of disabling authentication services is denying access to security operators, since they must also authenticate to access the SIEM console. The difference is that open sessions in the console will not be disturbed by an attack to the authentication services, which will only inhibit future authentication attempts, while the collection process will need to authenticate more frequently, on a per request or per session basis.

3.2.7 Network Channels

When access to the hosts, sensors or other SIEM components is not possible, or the cost of exploiting vulnerabilities in those components is too high, attackers may focus on the communication network, which is usually easier to get access to. The distributed nature of the SIEM and monitored systems translates to a disperse network, further adding to the complexity of securing those communication channels.

Furthermore, as we mentioned earlier, the edge components may not be able to cope neither with cryptographic mechanisms to secure communications nor with authentication protocols implemented to control accesses to the channel.

The level and complexity of security mechanisms can be increased if the communication between components of the SIEM system is made over public networks such as the Internet. One must consider alternative implementations of distributed SIEM systems, with the various components connected either using a private corporate network or public communication networks. Although assuming a communication channel to be secure is always dangerous, private networks allow more flexibility and avoid the necessity of having to deal with third parties such as service providers. By controlling the communication channels end-to-end, an organization is able to define security mechanisms to protect the traffic. On the other hand, if a public network is used, it is necessary to take into account possible restrictions imposed by third parties when defining the security enforcement methods, somewhat limiting the available options.

Considering the added complexity and risks, our analysis of possible attacks against the network is more detailed. We start by overviewing common vectors of attack to network channels, placing the problem into context, to then analyze possible solutions and protection mechanisms.

Network Attacks Overview – The major threats against communication networks [13] are eavesdropping, message modification and network flooding. By being able to access clear text network traffic, an attacker may disclose confidential information or gather data to perform future attacks. In the case an attacker is able to modify the content of a message without being detected, the recipient of that information may take actions based on incorrect facts, sometimes causing more harm to the system. Finally, a network flood can have significant impact on performance or even result in a communication breakdown.

Sniffing Attack – This attack enables a malicious entity to compromise the confidentiality of a communication channel by observing the packets passing through a network, making it possible to gather important information regarding the format and contents of the exchanged messages. To make sure that the information is transmitted in a secure fashion, the channel must be encrypted using reliable mechanisms, ensuring the confidentiality of messages. The issue with this approach is that it requires component

support for more secure protocols, which might be difficult to ensure when considering computationally constrained components.

Man-in-the-Middle Attack – While the sniffing the communication channel may enable an attacker to eavesdrop on the information being exchanged between the sensors and the collector, an attack based on modifying that information, thereby compromising its integrity, is much more powerful. The Man-in-the-Middle attack implies that the attacker is able to put himself between the sender and the receiving parties, intercepting the data and possibly altering its contents. By performing these actions, an attacker may be able to carefully craft bogus messages, disguising events that would lead to alerts as insignificant occurrences that will not be considered by the correlation engine. It is possible to prevent a Man-in-the-Middle attack by incorporating Message Authentication Codes (MAC) in the communications, enabling the collector to verify the authenticity of all the messages received. In conjunction with the cryptographic guarantees mentioned before, the communication channel can guarantee both the authenticity and confidentiality of the messages, making this attack unfeasible.

Denial-of-Service Attack – An attacker with access to the local communication channel might compromise its availability by flooding that channel with requests to the collector, thereby making it impossible for the collector to process all the information being received. The overflow of information may cause the collector to crash or, to avoid that, discard large quantities of packets, including relevant event information being sent by the sensors. While there are some satisfactory mechanisms to ensure communication confidentiality and integrity, there have been no conclusive achievements to effectively address the problem of a resourceful attacker compromising the availability of the network.

Protection Mechanisms – The most commonly used protocols to provide communication security over TCP are the Secure Sockets Layer (SSL) and Transport Layer Security (TLS). Both protocols are designed to provide security in the Application Layer of the Internet protocol suite, or Session Layer of the OSI model. The protocols provide confidentiality in a two-way communication through the use of symmetric encryption, after both parties negotiate a cryptographic key using the handshake protocol in which they mutually authenticate. They also provide message integrity by employing MAC.

SSL/TLS adoption can provide adequate security against attacks on the integrity and confidentiality of two-way communications over TCP, but they do not address the

availability property. Furthermore, these protocols can only run over reliable stream transport protocols, usually TCP, a restriction that limits their application in a complex environment where some of the components communicate using only UDP, or other unreliable protocols incapable of guaranteeing message delivery. Additionally, the cryptographic protocols require significant processing capabilities, something that not all components of the system are able to handle, especially the simplest ones like the sensors. Lastly, both SSL and TLS typically employ digital certificates signed by a trusted third party, creating the necessity for an additional actor to perform distribution and validation of those digital certificates. This approach is intended for strongly connected environments, being inadequate for a loosely coupled system such as a SIEM. The scope of application is therefore reduced to the SIEM core, the communication between collectors, correlation engine and event storage.

In contrast to SSL/TLS, IPSEC provides the same type of security in the Internet layer of the Internet protocol suite, the Network layer of the OSI model. By authenticating and encrypting each IP packet, the usage of IPSEC is transparent to the applications and may be used indifferently by upper layers protocols, such as TCP and UDP. A firewall or router can implement IPSEC, providing perimeter security and eliminating the overhead inside the private network. By operating at a lower layer, IPSEC can be more flexible and adapt better to the complex context of SIEM systems, as assuming that all components communicate using IP is less strict than requiring the communication to use TCP.

Additionally, since IPSEC can be implemented by the network active components, there is no need to consider performance impacts on the end nodes. In conclusion, implementing IPSEC is more adequate to the secure communications up to the network equipment connected to the sensors, leaving only the communication between the sensors and the event sources unprotected.

Chapter 4

Resilient Correlation Rules

In previous chapters we have outlined a possible SIEM architecture, described its components, discussed possible faults and presented ways to mitigate them. Those faults, either malicious or accidental in nature, result in events not reaching the SIEM correlation engine and, consequently, important alarms not being triggered. In this chapter we present techniques to improve the resilience of correlation rules, the heart of the SIEM, and a way to consider more thorough attack mitigation approaches by going further than just protecting the event collection process.

Improving the resilience of correlation rules is crucial to guarantee that all relevant information is collected and its integrity is maintained. It is also a stepping stone to the ultimate goal of acquiring a security monitoring capability that can guide the security team through the analysis of ongoing attacks, increasing their effectiveness by making information available and decreasing their response time by triggering relevant alarms as the events occur. To achieve this objective, it is vital to perform correlation using the various data included in the collected events, thus taking advantage of the inherent redundancy in the millions of events that are processed.

Correlation rules are at the core of the SIEM operation, which makes their definition an important part of the SIEM implementation, contributing to prevent attacks from circumventing the triggering of alarms as well as the possibility of an attack to go by unnoticed. Our goal is to improve current implementations of SIEM rules by expanding their resilience against attacks, even in the presence of compromised sensors, or other edge components, capable of interrupting, delaying or forging the event flow to the SIEM.

To make our approach more concrete, in the rest of the chapter we will examine some example correlation rules. These rules are built using the syntax of the ArcSight SIEM system.

4.1 Elementary Correlation Rules

The most straightforward purpose of collected events is to raise alarms based on their content, using events from each source to define specific trigger conditions. For instance, one could define types of events that should never be observed, since they are contrary to the defined security policy or, more commonly, trigger an alarm when an event is detected more than a predetermined number of times in quick succession. Throughout the section we give examples of these out-of-the-box rules that comprise what can be considered as the current status of SIEM correlation rules, while also pointing out their frailties and limitations.

4.1.1 Rules Using a Single Event Source

Each correlation rule starts by stating the frequency parameters that should trigger an alarm. In Rule 1 we show a policy violation that should trigger an alarm to the security team, even if it happens only once. The policy states that all changes to user accounts must be performed using the Identity Management system (IdM), which means that if there is any change not originating from that system an alarm should be triggered.

In line 1 a time constrain is defined to trigger the rule, a mandatory field, and state that it should be triggered by the first event meeting the criteria in the remaining lines. The criterion for triggering the rule is a conjunction of three conditions. Line 2 expresses that the attacker username is different than the account used by the IdM, line 3 matches the type of event to the known category of authentication and, lastly, line 4 indicates that the outcome of the event was successful. The entire rule can be read as such: match any successful events that resulted in changes to a user account and were not executed by the IdM account.

```
1   Matching 1 events in 1 Minute with conditions(
2   NE(event1.sourceUserName,IdMAccount);And;
3   EQ(event1.categoryBehavior,/Authentication/Add);And;
4   EQ(event1.categoryOutcome,/Success))
```

Rule 1 – User Changes outside IdM

This rule relies on events from a single source, the enterprise user directory, by scanning the logs to discover change commands of a specific type and then verifying its origin based on the username. The fact that the rule depends solely on the username to determine if the change is authorized means that spoofing that information may cause

attacks to go unnoticed, as long as the attacker knows the IdM username and is able to impersonate it.

The example in Rule 2 is a bit more complex, using auxiliary rules to label events, already identifying them as attacks or successful operations. The objective is to determine if a brute force attack was successful.

Once again line 1 indicates the time conditions, triggering the rule at each occurrence. Line 2 states that the successful login must have occurred at the same time or after the brute force attempts, with lines 3 to 5 verifying that the origin and destination of the events are the same. Line 6 excludes a subset of trusted actors, meaning that if the source of the event is in that list, the rule is not triggered. Lines 7 to 10 match the type of event and their outcomes, which should be failure for the Brute Force attempts and success for the completed authentication request. The resulting rule is: match any occurrences of brute force attacks being followed by a successful authentication from the same source, provided that source is not in the trusted actors list.

```
1   Matching 1 events in 1 Minutes with conditions(
2   LE(Brute_Force.endTime,Login_Success.endTime);And;
3   EQ(Brute_Force.sourceAddress,Login_Success.sourceAddress);And;
4   EQ(Brute_Force.destinationAddress,Login_Success.destinationAddress);And;
5   EQ(Brute_Force.sourceUserName,Login_Success.sourceUserName);And;
6   “Not" InActiveList(Brute_Force.sourceAddress, Trusted List);And;
7   EQ(Brute_Force.categoryBehavior,/Authentication/Verify);And;
8   EQ(Brute_Force.categoryOutcome,/Failure);And;
9   EQ(Login_Success.categoryBehavior,/Authentication/Verify);And;
10  EQ(Login_Success.categoryOutcome,/Success))
```

Rule 2 – Probable Successful Brute Force Attack

Like in the first example, this rule is based on an analysis of events from a single source, an authentication server. A set of events is previously analyzed and classified as a brute force attack using Rule 3. The rule then uses that information and relates it to successful authentication events to determine if the attacker achieved its goal.

```
1   Matching 5 events in 2 Minute with conditions(
2   "Not" InActiveList(Auth_Fail.sourceAddress, Trusted List);And;
3   EQ(Auth_Fail.categoryBehavior,/Authentication/Verify);And
4   EQ(Auth_Fail.categoryOutcome,/Failure))
```

Rule 3 – Brute Force Logins

In Rule 3, the time constraint in line 1 indicates that the rule is triggered only in the case five events meeting the criteria occur within two minutes. Line 2 exempts trusted actors from triggering the rule, allowing this type of behavior from

predetermined sources. Finally, lines 3 and 4 refer to the type of event and the unsuccessful outcome. Thus, the rule translates into: match five unsuccessful authentication attempts within two minutes, originating from a source that is not in the trusted list.

Once again there are clear limitations in this rule, for instance the fact that it is only triggered if the address of both the attacker and the target system are the same in the brute force attack and on the successful authentication. If an attacker is aware of this reasoning, he can use the several authentication servers normally present in a large enterprise to scatter the attack, keeping within the time limitation boundaries to avoid being detected.

To be more effective, the rule should consider the addresses of all authentication servers, although even then the attacker could still spoof its own address at each try to mask the true origin of the events. To cope with those more advanced attacks, more sophisticated rules are necessary, as we will demonstrate.

4.1.2 Rules Using Time Based Triggers

While Rule 3 took under consideration not only the attacker and target address but also the time interval between events, there are simpler rules that classify events as suspicious or even trigger alarms based solely on timing considerations.

```
1   Matching 1 events in 1 Hours with conditions(
2   EQ(event1.deviceEventClassId,Security:630);And;
3   InActiveList(event1.destinationUserName, CreatedAccountsActiveList))
```

Rule 4 – Windows Account Created and Deleted Within 1 Hour

Rule 4 uses event information from the user directory and relies on a related rule that adds newly created user accounts to the active list mentioned in line 3 (“CreatedAccountsActiveList”). The entries added to this list have a Time to Live (TTL) of one hour, after which they are automatically removed from the active list by the SIEM. If, during that hour, the account is deleted, identified in line 2 by the event code 630 in Windows-based domain controllers, this rule would be triggered and the action could be marked as suspicious or even display an alarm to the security team, who would then proceed to review the actions performed using that account.

By not relying on relating time constraints and the source or destination addresses, this rule can be somewhat sturdier than previous examples. Nevertheless, as in all time-based rules, if an attacker is aware of the restrictions imposed by such triggers, he can

easily bypass the rule and consequent alarms. In this specific case, creating the bogus account and waiting one hour before using it could successfully perform the attack.

4.1.3 Limitations of Basic Correlation Rules

The security team is highly exposed to possible faults by relying on only one event source and/or in time constraints to determine if a rule should be triggered. As soon as an attacker is aware of how the correlations rules are built, the loopholes become evident, thus making a targeted attack possible.

As we have shown before, basic rules are normally easy to bypass by spoofing part of the event details, such as the username or the IP address, something that can be done without much effort. Likewise, triggers based on the elapsed time between events can also be bypassed if the attacker is able either to change the pace of the attack, widen the scope of targets or simply delaying the sending of event information by the sensors.

Even if the attacker is unable to compromise the components of the SIEM system, he can circumvent basic rules just by compromising the sensor collecting events from the source under attack. The only option available to minimize the number of missed alarms, when one or more sensors are compromised, is to collect information from different sources, using the inherent relation between those sources as a way of enriching the correlation rules. Both the network and asset models can be very helpful when designing a robust set of correlation rules, since they contain precious information regarding the event sources, their inherent characteristics, location in the network and communication channels between them.

4.2 Improving Correlation Rules

As we have demonstrated in the previous sections, standard correlation rules can be ineffective against even moderately sophisticated attackers and are unable to cope with either accidental or malicious faults, such as compromised sensors. Our goal is to eliminate as many frailties in the correlations rules as possible, improving their resilience without adding any more complexity than strictly necessary.

Much like when improving the security in the configuration of a system, the first step should be to harden the correlation rule, considering non-straight-forward scenarios even when using a single event source. Our approach is to enrich the correlation rules using further information, also known as properties, included in the events, and take

advantage of SIEM resources such as the asset and network models. The default rule set takes the integrity of information for granted and focuses mostly on the best-case scenario, which results in the weaknesses mentioned earlier. Instead of considering only part of the information that constitutes an event, we take advantage of as much information as possible to detect malicious behaviors, even if the attacker is taking some precautions not to be noticed. Furthermore, by broadening the scope of properties considered when defining the rules we increase the difficulty of forging event information.

Understanding the properties of events and their idiosyncrasies is important when designing more resilient correlation rules. A subset of those properties is common throughout events from multiple sources, such as source and destination addresses, the event type or the outcome of the event. These fields can be used in any rule and constitute the basis from most correlation rules. However, there are many others that are exclusive to specific technologies, making them extremely pertinent when designing resilient correlation rules. By acknowledging the specificities of event properties it is possible to broaden fault detection capabilities and deepen the level of detail that will help to improve rules.

While hardening the correlation rules allows the SIEM to detect previously unobserved abnormal actions, basing the evaluation of events in a single source keeps the system vulnerable to the successful attacks on that source. This vulnerability results in situations where an attacker that is able to compromise a single component or system can completely control the events being generated in that source, thus thwarting correlation rules and allowing an attack to go by unnoticed. We demonstrate that it is possible to combine information from multiple sources in order to strengthen correlation rules, making them effective even in the presence of a partially compromised infrastructure.

The idea behind correlating events from multiple sources is that all systems are interconnected, and therefore, most actions result in associated events being generated at more than one source, thus creating some level of redundancy on the information that reaches the SIEM engine. Let us consider that an attacker is able to compromise a server without being detected and subsequently disables event collection from that source. If the attacker starts to use that compromised server to launch a new attack, each communication made with other servers will generate events on those destination servers, as well as in the network components that connect both assets. Therefore, even

in the presence of a compromised source, it is possible to collect events from other sources that convey the information needed to detect an attack. Taking advantage of these associated events in different sources, it is possible to design resilient correlation rules that not only increase the effectiveness of attack detection but also allow the security team to identify possibly compromised systems.

Even more interesting is to utilize events from multiple sources not only to detect but also to mask faults. As we exemplified earlier, some actions are expected to generate events both in the source and destination systems, therefore incoherencies between those sources are sufficient to raise an alarm, detecting a possible fault. However, in the case of actions that generate events in more than two sources it may be possible to go further, for instance employing a voting mechanism to determine which of the sources is reporting incoherent information. The remaining sources can then be used to discover the ongoing attack, so that the invalid source can be identified as reporting incoherent information.

4.2.1 A Method for Improving Correlation Rules

The systematic improvement of correlation rules can be performed accordingly to the methodology outlined in Figure 9. Depending on the type of rule and event sources, some of the steps of the methodology may not apply. There are exceptions to every rule, and in this case we opted for a generic approach that fits most cases, adapting it for specific situations when necessary.

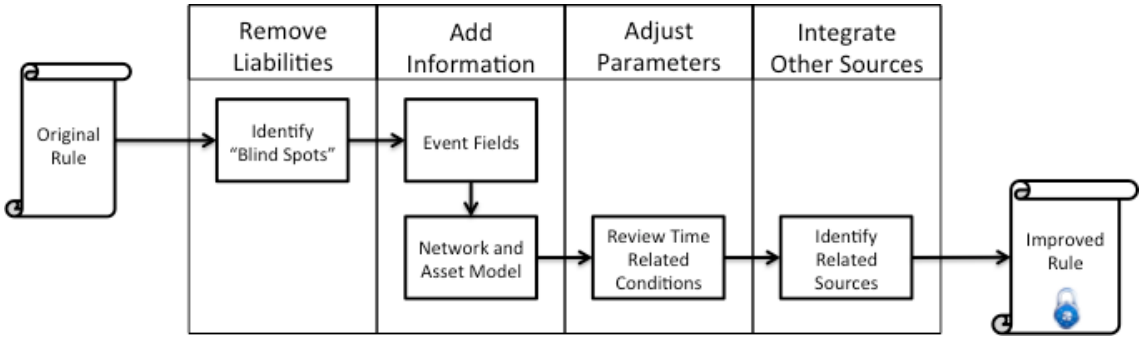

Figure 9 – Correlation Rule Improvement Procedure

To remove existing liabilities in the original, less resilient, rule, it is necessary to identify "blind spots" or possible vulnerabilities in the conditions of the rule. Having a whitelist or other exception mechanism is undesirable, unless strictly necessary, as it allows an attacker to circumvent detection by impersonating a trusted actor, sometimes

just by forging the source IP address. There are situations where the usage of whitelists is justifiable and necessary, for instance when considering specific systems that perform otherwise forbidden tasks like vulnerability scanners. However, in those situations, it is vital to carefully define the exceptions, providing comprehensive information not limited to IP addresses or hostnames.

The addition of relevant information is twofold, with the first and most simple approach being to take advantage of unutilized fields in the events. Information, such as the hostname, port or the agent that collected the event, can present valuable insight when analyzing events, increasing the necessary knowledge of the system that the attacker must possess to successfully manipulate the information entering the SIEM. The second step would be to incorporate information from the network and asset models, with the advantages we mentioned above. Since the integrity of the information in these models is verified and only a SIEM administrator can update their contents, the models can be used as baselines to compare against the event data.

The majority of correlation rules are partially based on the time lapse between events or the number of similar events within a defined time frame, and consequently these time constraints have to be carefully defined and reviewed. The time intervals defined to trigger alarms can be the difference between an attack being detected or not. On the other hand, it can also cause false positives or false negatives that decrease the confidence in the alert capabilities of the SIEM and flood the security teams with information. Both risks have to be taken into consideration when constructing the correlation rules, as they result in the loss of vital information, either because alarms are not triggered or due to an overflow of information that exceeds the processing capability. The solution must rely on a carefully designed learning process, tuning the parameters according to the normal operation of the infrastructure. Nevertheless, it is possible to define approximate default values for each scenario based on the experience of the security teams.

The final step when improving a rule would be to identify related sources that could contribute to verify the veracity of the information being processed by the SIEM, thereby making it resilient to a limited number of compromised sensors. The previous methods are effective against an attacker trying to inject bogus data in the network, and increase the overall robustness of the infrastructure, but they are not able to cope with compromised components where an attacker is able to penetrate the outer defense layer, consequently possessing all the necessary information to deceive the SIEM rules. The

possible solution is to collect information from multiple related sources, increasing the resilience of the system by considering that an attacker is only able to compromise part of the systems, at least initially.

Considering the fault model presented in Chapter 3, it is possible to state that all steps in the methodology contribute to lessen the risk presented by malicious attacks. The first two steps are more directed to personification attempts, making it harder for attacker actions to pass by unnoticed. To be successful, an attacker would have to possess thorough knowledge of the SIEM system and correlation rules, crafting highly detailed and coherent events that meet the criteria of the rules. The two final steps are aimed at detecting compromised sources or SIEM components. If an attacker is able to compromise part of the event creation or collection process, either changing the contents, delaying or deleting events. By enforcing time constraints and correlating events from different sources it is possible to improve the detection of manipulated information. The final step of the methodology is also paramount to cope with crash faults, adding source redundancy to ensure that as much information as possible still reaches the SIEM correlation engine in the event of faults that partially disable the infrastructure.

4.2.2 Correlation Rule Hardening

In this section we apply the proposed methodology to improve the resilience of the correlation rules presented in Section 4.1. We have established that Rule 1 is vulnerable to spoofing, as the knowledge of the IdM account name enables a bypass. This attack is possible because the rule verifies only the origin account, disregarding any additional information like the source system. The rule may thus be improved by adding information from other event fields. These properties can be used to include further details from that source system, forcing a possible attacker to have to spoof more information, thereby making the attack more complex.

```
1  Matching 1 events in 1 Minute with conditions(
2  (NE(event1.sourceUserName,IdMAccount);Or;
3  NE(event1.sourceAddress,IdMAddress);Or;
4  NE(event1.sourceOS,IdMOS));And;
5  EQ(event1.categoryBehavior,/Authentication/Add);And;
6  EQ(event1.categoryOutcome,/Success))
```

Rule 5 – User Changes outside IdM (improved)

Rule 5 is an improved version of Rule 1, where we added the underlined conditions in lines 3 and 4 to force additional checks. Using the attacker address and operating system signature it is possible to make the rule more robust, forcing a possible attacker to spoof not only the account name but also the address and OS fingerprint of the IdM system.

However, to make these conditions possible one would have to configure a large set of variables in the SIEM, consequently increasing the operational efforts and configuration complexity. Fortunately, the SIEM system includes the aforementioned network and asset models, which can be automatically updated with relevant information from the infrastructure including, but not limited to, the address and OS fingerprint of the servers. The methodology proposes the use of the asset and network models to ease the management of correlation rules, adding information maintained and updated by the SIEM to better identify source or destination systems.

The hardened rule is therefore not only more powerful but also easier to manage. Since the only source authorized to perform the action of adding a new user to the domain is the IdM, it is imperative for the source of such actions to be part of the asset model. The properties of the event source must be checked against the information present in both the asset and network models, something that can be easily enforced.

```
1   Matching 1 events in 1 Minute with conditions(
2   (NE(event1.sourceUserName,IdMAccount);Or;
3   NE(event1.asset, AssetModel.IdMAsset);Or;
4   NE(event1.sourceAddress, NetworkModel.IdMAddress));And;
5   EQ(event1.categoryBehavior,/Authentication/Add);And;
6   EQ(event1.categoryOutcome,/Success))
```

Rule 6 – User Changes outside IdM (hardened)

The condition in lines 3 verifies that the source system is part of the asset model and corresponds to the asset declared as the IdM, while line 4 focuses on the network information to establish the correspondence. The resulting Rule 6 would then be able to verify not only specific event attributes but consider the properties of two objects, matching them to encounter relevant discrepancies that indicate the attack source is not the same.

It is possible to use similar improvement techniques in Rule 2 and Rule 4. In the first case, the initial step would be to consider attacks coming from diverse sources and against distributed authentication servers. If an attacker is able to compromise several computers with the objective of performing a brute force attack against a privileged

account, for instance using a computer virus or worm, instructions could be sent to the infected machines to perform sweeps across the multiple authentication servers, thereby avoiding the time constraints on the rule.

```
1   Matching 1 events in 1 Minutes with conditions(
2   LE(Brute_Force.endTime,Login_Success.endTime);And;
3   EQ(Brute_Force.sourceAddress,Login_Success.sourceAddress);And;
4   EQ(Brute_Force.destinationAddress,Login_Success.destinationAddress));And;
5   EQ(Brute_Force.sourceUserName,Login_Success.sourceUserName);And;
6   ("Not" InActiveList(Brute_Force.sourceAddress, Trusted List));And;
7   EQ(Brute_Force.categoryOutcome,/Failure);And;
8   EQ(Login_Success.categoryBehavior,/Authentication/Verify);And;
9   EQ(Login_Success.categoryOutcome,/Success))
```

Rule 7 – Probable Successful Brute Force Attack (hardened)

The improved Rule 7 would consider the number failed authentication attempts by the same account, regardless of the origin address, followed by a successful authentication by that same account. The original rule also included a loophole by considering trusted sources, effectively ignoring events originating from systems with addresses on that list, which could be ranges of addresses inside a trusted network perimeter, therefore creating a blind spot if the attacker is able to breach that supposedly secure zone. The elimination of white lists that may introduce vulnerabilities is the first step of the proposed methodology to improve correlation rules. Since we are focusing in network information, the event fields used to construct and improve these rules are part of the set of properties that are common to events from all sources, without the necessity of resorting to specific properties from this event source.

Hardening Rule 4 requires additional efforts, since the simplicity of the objective would be undermined by a more complex construction, possibly increasing the number of false positives. Our only proposal, following the improvement methodology, is to widen the time window between the creation and deletion of an account, since it is not expected a user account to be active less than 48 hours when considering the normal life cycle of domain accounts. Although studies indicate that a security breach remains undetected on average for 416 days [6], the threshold of 48 hours seems appropriate to deal with the most eminent threats. A longer time interval or any further conditions would dramatically increase the number of false positives and the amount of information to be processed by the security team, in fact decreasing the probability of an attack being uncovered.

```
1   Matching 1 events in 48 Hours with conditions(
2   EQ(event1.deviceEventClassId,Security:630);And;
3   InActiveList(event1.destinationUserName, CreatedAccountsActiveList))
```

Rule 8 – Windows Account Created and Deleted Within 48 Hours

The resulting Rule 8 employs the deviceEventClassId property of the event to determine the originating action. This property is specific to events from Windows servers, more precisely domain controllers, therefore not part of the common set shared by all events.

4.2.3 Correlating Different Event Sources

Even after the process of hardening the basic rules, several limitations are still present. As we have mentioned above, relying on a single source of events to trigger alarms is ineffective when considering a fault model such as the one we presented in Chapter 3, where event generation might be affected. The final step of the methodology proposes the correlation of events from multiple sources, withdrawing data from separate systems or devices to increase the resilience of the process.

Validation Using Network Events – Computer networks are ubiquitous in any modern IT infrastructure, with each node being connected to one or more network components in order to communicate with applications, databases or other systems. This means that each request or command from a source system is bound to have passed by a number of network nodes before reaching its destination, enabling the correlation of events from those sources.

The first event sources to incorporate in a SIEM system are the network firewalls due to their extensive logging of established connections, detailing traffic classification, protocol information and used ports. Using this information, as well as the defined network model, it is possible to detect attempts to mask the real origin of the traffic by spoofing the source address.

We demonstrate this capability in Rule 9, based on the already modified Rule 6, to detect changes in user accounts not performed by the authorized IdM application. The first step would be to define a rule that processes firewall logs and identifies commands from the IdM application to the user directory server, adding those commands to an active list for one minute. The active list is checked in line 4, to guarantee that the action was based on a previous command from the IdM application. Using IPSec it is

possible to establish a cryptographic tunnel between the firewall and the SIEM, guaranteeing the origin and integrity of the events registered in the active list.

```
1   Matching 1 events in 1 Minute with conditions(
2   (NE(event1.sourceUserName,IdMAccount);Or;
3   NE(event1.asset, AssetModel.IdMAsset);Or;
4   NE(event1.sourceAddress, NetworkModel.IdMAddress)) ;Or;
5   (Not InActiveList(event1.command, IdMCommandsInLastMinute));And;
6   EQ(event1.categoryBehavior,/Authentication/Add);And;
7   EQ(event1.categoryOutcome,/Success))
```

Rule 9 – User Changes outside IdM (using firewall events)

By stating that if one of the conditions is not met an alarm is triggered, we are eliminating the possibility of an attacker using a compromised workstation somewhere in the corporate network to impersonate the IdM server and successfully create a user account. If the attacker tried to compromise the sensor collecting the firewall events, the change in the user directory would trigger the alarm, since by blocking the events from the firewall the attacker would also hamper the update of the active list, therefore triggering the alarm all the same. The last resort available to the attacker would be to stealthily compromise a machine in the same network zone as the IdM system, already a more secure perimeter, and only then spoof the origin of the command.

There is however an issue with Rule 9 that would render its application ineffective. As we mentioned earlier, event collection from the sources is sometimes performed using unreliable protocols, with no ordering or delivery guarantees. As this rule relies on the correct ordering of events, it efficiency is limited and might generate false positives.

A more reliable possibility is combining events from the Domain Controllers, where the action is performed, with events from the IdM database. When the IdM creates an account, an event is generated and stored in the internal database. By collecting those events to the SIEM it is possible to generate an event each time there are matching actions for the same destination account.

To combine events from multiple sources with common fields, Rule 10 uses event tags, defined in line 3 and line 8, respectively identifying the event from the Domain Controller as “Action” and the event from the IdM database as “DatabaseOperation”.

```
1   Matching 1 events in 1 Minute with conditions(
2   MatchingEvent(Action.destinationUserName, DatabaseOperation.destinationUserName);
3   Action {(EQ(Action.sourceUserName,IdMAccount));And;
4   EQ(Action.sourceAddress, NetworkModel.IdMAddress);And;
5   EQ(Action.assetID, AssetModel.IdMAsset);And;
6   EQ(Action.categoryBehavior,/Authentication/Add);And;
7   EQ(Action.categoryOutcome,/Success)}
8   DatabaseOperation{EQ(DatabaseOperation.assetID, AssetModel.IdMDBServer);And;
9   EQ(DatabaseOperation.categoryBehavior,/Authentication/Add);And;
10  EQ(DatabaseOperation.categoryOutcome,/Success)})
```

Rule 10 – User Changes outside IdM (using database events)

Line 2 expresses the condition to match fields from separate events, a method similar to a Join operation in SQL statements. Information from the Asset and Network models is used to guarantee the integrity of event data. Since this rule combines information from two events without using information from active lists, the order by which the events reach the SIEM is irrelevant. Contrary to Rule 9 that triggers an alarm when a condition is not met, this rule generates an event when both conditions are met. The resulting security analysis is that the creation of a domain account should translate into three related events – one from the Domain Controller, one from the IdM database and the event generated by Rule 10.

Considering the higher complexity of this rule, we present its graphical view in Figure 10, as it is shown in the SIEM console.

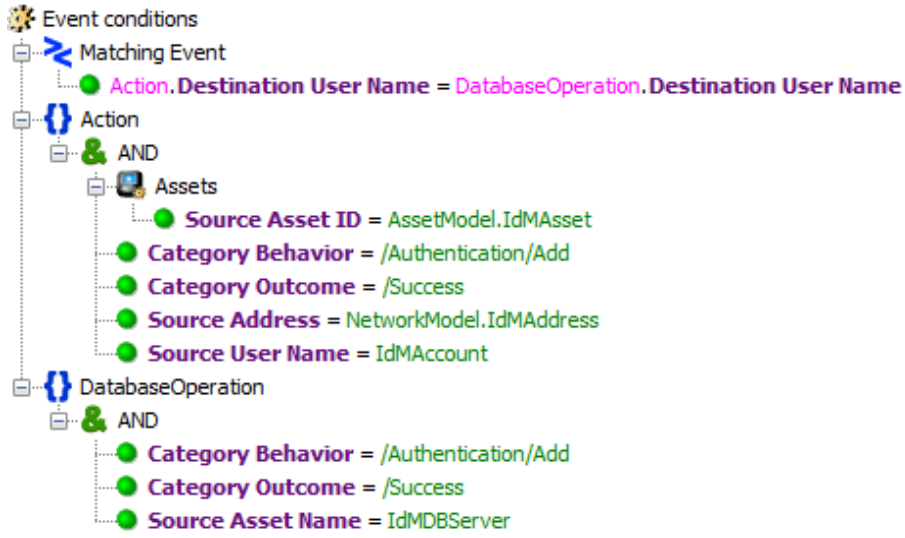

Figure 10 – Graphical View of Rule 10

Fault Detection Using Correlation – Event correlation can be used directly to detect incoherent information from multiple sources recording the same event, as we have seen in Rule 9. By taking advantage of the network and asset models, it is possible to define not only acceptable commands, but to verify how those commands align step-

by-step with defined workflows and procedures. The SIEM is able to interpret information such as the type of asset to detect abnormal behavior by analyzing the events coming from that asset.

For example, an institution might decide that some operations, like deploying firewall rules or software updates, can only be performed after working hours to avoid performance impacts. A simple rule can be employed to determine if certain types of events do not occur outside the allowed time window. However, this rule might only apply to production systems, while development or test environments have less strict policies. The asset model can be used to enrich the correlation rules with information pertaining to the infrastructure in which the systems are deployed, making it possible to accommodate these nuances.

Using the same principle the SIEM is also able to detect if specific changes to the configuration of the systems are being performed from the operations center or from the technicians personal laptops over a VPN connection. The company policy might dictate that critical operations can only be performed locally to ensure direct access to the systems in case a rollback is needed, therefore SIEM rules can be defined to detect such occurrences and trigger the necessary alarms.

Using the same approach it is also possible to detect faults in the infrastructure by spotting the absence of expected events. Suppose that an attacker decides to target a sensor attached to a web server with the objective of modifying its contents, which are in turn stored in a separate database. Unless the attacker is also able to successfully compromise the sensors in the database and in the firewall segregating the DMZ from the internal database servers, there would still be traces of the malicious actions. An alarm can be triggered upon the verification that events from the database and firewall indicate an action originating from the web server, while the associated event from that source is absent from the SIEM. The alarm would state that an expected event did not reach the SIEM, indicating a possible fault in that source or in the collection process.

With this scenario in mind, the goal would be to create pattern-based correlation rules that, once again using the network and asset model, are able to match related events therefore also detecting missing events that should have been received by the SIEM.

Fault Masking Using Correlation – More than just detecting faults, in specific situations correlation rules may go as far as permitting fault masking, which is to say that the SIEM system can reach the same conclusions and trigger alarms even in the

presence of compromised components. The principle of analyzing not only single events but also entire workflows, as we described above, enables the SIEM to process complex information by relating information from multiple sources.

We have shown how to use correlation to detect faults and trigger the correspondent alarms, but let us consider a situation where a command workflow is supposed to generate events in three different components. If, after correlating the information from all sources, it is discovered that one of the events is either missing or unaligned with the remaining, the SIEM could disregard the entire flow and merely trigger an alert to the security team. However, by employing a voting algorithm, it is also possible to assume that there was either an error in the outlier source, or that it has been compromised.

4.3 Limitations of Correlation Rules to Detect Attacks

There are inherent limitations to detecting attacks relying only on correlation rules, as information redundancy is not always present. Let us put forward a scenario where a software component installed on top of the operating system acts as a sensor for events in that source. If an attacker is able to introduce malware in that machine, for example using an infected USB drive, then the malware could immediately target the sensor, much like well-known malware that disables the anti-virus agent. Imagining that the system is the intended target for the attack, for instance to steal information stored in the hard drive, the attacker would have no need to use the network thus making it impossible for other sensors to detect the attack. The lack of information redundancy, i.e., information coming from only one source, means that those events will not reach the SIEM in case that source is compromised.

Countering these targeted attacks cannot be done using a SIEM system, with the answer residing in stricter security policies like disabling USB ports. However, since we are designing ways to improve SIEM resilience, not increasing its capabilities, we will focus on more common attacks that make use of the network to access remote systems and spread across the IT infrastructure.

Improving the resilience of correlation rules is also an exercise to increase attack and fault detection capabilities while ensuring that the rate of false positives is maintained or, preferably, improved. As correlation rules become more detailed, using specific information from the sources, and incorporate events from multiple sources, the

knowledge of the infrastructure must also be on par with those information requirements. The more specific the rule, the more susceptible it is to changes in the monitored systems, meaning that updates or changes in processes can result in the necessity of reviewing the correlation rules in order to avoid erroneous alarms. For instance, when information from the asset or network model is used, one must ensure that changes to the systems are readily updated in those models, one of the options being to populate the models using automated scanning tools.

4.4 AutoRule: Automatic Rule Analysis

Under certain conditions it may be possible to automate the analysis of correlation rules, helping to identify the need to review the rules before implementation. Taking into consideration the complexity of some correlation rules, the automatic process is expected to have limitations when compared to human reviews performed by security experts. Nonetheless, a systematic approach will enable the detection of the most common errors when constructing correlation rules, as well as pointing out improvement possibilities.

4.4.1 Designing AutoRule

The automatic analyzer could start by parsing the rules and identifying keywords. Heuristic analysis could then be applied to pinpoint possible frailties and suggest improvements. The proposed methodology should be followed step-by-step, firstly identifying the usage of white lists, then the lack of event information diversity, followed by an absence of references to the network and asset models in conjunction with other event properties.

Time related conditions could be compared to standard values based on the type of rule, however, as we mentioned earlier, there should be a learning process to adjust parameters accordingly to the specific characteristics of the infrastructure. Lastly, to identify possible related sources, the automatic process should have the ability to import data from the asset and network models, creating an internal knowledge base capable of adding relevant event information to the correlation rules. The tool should therefore enable the possibility of customization by the security team, adapting to the monitored systems.

4.4.2 Implementation Principles

The implementation of this proof of concept application was based on understanding the syntax of correlation rules, identifying the structure beneath their definition. It was necessary to establish a correspondence between the concepts in the proposed methodology and the specific manner in which they are put together in correlation rules.

AutoRule is based on keyword assessment, identifying major recurrences and the methods employed to process collected information. By recognizing elements that could present frailties or be explored by malicious entities it is possible to recommend improvements, as well as calculating a resilience score. The same principle is applied to point out important elements that are preconized by the methodology and absent from the correlation rules. The result is a static analyzer capable of evaluating the resilience of the rules by identifying the presence or absence of relevant components in their definition.

4.4.3 Deployment and Demonstration

To demonstrate an automatic systematic analysis of SIEM rules, following the methodology previously presented, we developed AutoRule (Automatic Rule Analysis), a proof-of-concept application in Java to parse correlation rules, suggest improvements and calculate the overall resilience score according to the verified level of redundancy. The score is estimated according to the identified shortcomings of the rule, with different weights being given to diverse occurrences, with a score closer to zero indicating a more resilient correlation rule.

The first step, as the methodology advocates, is to detect the presence of exceptions to the rule by verifying the employment of trusted lists. As we explained before, if an attacker is aware of that potential loophole it may be possible to forge data in order for the attack to pass unnoticed by the SIEM. Being a relevant source for attack misidentification, the presence of a list of trusted agents has a high impact in the overall score.

AutoRule also checks for network or account information used individually, therefore making the rules weaker. The combination of both conditions is recommended to perform account identification and network origin checks simultaneously. Additionally, resorting to the network and asset models instead of explicit and user-defined variables is also preferable. To exemplify the usage of AutoRule to validate

correlation rules, we resort to the evaluation of previous examples to detect account changes outside the IdM system.

```
Possible loophole in active list exceptions:
  Line 6: "Not" InActiveList(Brute_Force.sourceAddress, Trusted List);And;

Warning - Network conditions should rely on the network model
Warning - Account verifications should rely on the asset model
Warning - The rule does not use multiple event sources

Final Score: 12
```

Figure 11 – AutoRule Evaluation of Rule 2

Figure 11 represents the output of the automated verification process when applied to Rule 2. It immediately shows that this rule is not very resilient, considering that it includes a reference to a trusted list, identifies the originating agent solely based on the network address and makes no use of the network or asset models. Also relevant is the fact that the rule does not possess any source redundancy, relying only on events from a Domain Controller.

Applying the same validation methodology to Rule 7, an improved version of Rule 2, shows the differences in robustness and, consequently, in the attributed score as we can observe in Figure 12. By eliminating the possible loophole introduced by the exceptions in the Trusted List, while also not restricting the attacker identification to a single network origin, the overall score is highly improved.

Some of the warnings remain, as the rule still tries to identify the attacker without any verification of the asset model. By maintaining the use of only one event source, the resilience of the rule is still low and the triggering of alarms could be interrupted by accidental failures or successful attacks.

```
Warning - Account verifications should rely on the asset model
Warning - The rule does not use multiple event sources

Final Score: 6
```

Figure 12 – AutoRule Evaluation of Rule 7

Table 1 summarizes the outputs and score obtained by analyzing all the correlation rules presented throughout this chapter with AutoRule. We point out the improvements to demonstrate the gains acquired with the proposed methodology.

Rule	Improves	Output	Score
#1	**N/A**	Username reference should be complemented with network information: Line 2: NE(event1.sourceUserName,IdMAccount);And; Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**9**
#2	**N/A**	Possible loophole in active list exceptions: Line 6: "Not" InActiveList(Brute_Force.sourceAddress, Trusted List);And; Warning - Network conditions should rely on the network model Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**12**
#3	**N/A**	Possible loophole in active list exceptions: Line 2: "Not" InActiveList(Auth_Fail.sourceAddress, Trusted List);And; Warning - The rule does not use multiple event sources	**10**
#4	**N/A**	Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**6**
#5	**#1**	Warning - Network conditions should rely on the network model Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**7**
#6	**#5**	Warning - The rule does not use multiple event sources	**5**
#7	**#2**	Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**6**
#8	**#4**	Warning - Account verifications should rely on the asset model Warning - The rule does not use multiple event sources	**6**
#9	**#6**	Warning - The rule could be impacted by out-of-order events	**2**
#10	**#9**		**0**

Table 1 – AutoRule Analysis of Developed Correlation Rules

Chapter 5

Implementation and Experimental Evaluation

When considering the possibilities for improving correlation rules, we made an effort to conceptualize generic scenarios and discuss theoretical possibilities to account for multiple implementation contexts. However, more than proposing a framework for correlation rule improvement, we also aimed for an outcome that could translate into a practical application. The implementation in a real production environment poses additional challenges, considering the shortcomings of a complex infrastructure where functionality is the ultimate goal and security only a desired, sometimes neglected, property.

Since the goal of a SIEM system is to collect security events, the first difficulty is having access to that information. There are multiple reasons why the access to comprehensive and complete information may be limited, for example an inadequate level of logging due to performance issues or the fact that part of the infrastructure is managed by a third party. Additionally, the level of service externalization is increasing, either by outsourcing the management of the IT infrastructure or by resorting to cloud service providers, which represents a problem when security teams need to access security configurations or events.

Part of the functions of a Security Operations Center is to convey that message to the corporate management, contributing to the establishment of security policies that include specific requirements to be considered when committing to service contracts with third parties. Nevertheless, the framework we present encompasses multiple action points to improve correlation rules so that such technical or contractual limitations can be at least partially circumvented.

5.1 Experimental Environment

To evaluate the effectiveness of the proposed rule improvement methodology and obtain valuable information to increase the resilience of SIEM systems, we resorted to a SIEM system implementation in a multinational utility: EDP – Energias de Portugal, SA. While designing the architecture of the SIEM system, EDP took into consideration the reference architecture and best practices based on previous implementations in similar sized companies. The geographically distributed nature of the network and systems, and the necessary redundancy to guarantee the availability of the service in case of a single component failure, led EDP to implement two separate SIEM stacks, one in each datacenter in Portugal. After conducting a market research and analyzing proposals from several vendors, the chosen technology was ArcSight, consecutively considered to be best of breed by independent evaluations. On top of the technical capabilities, one of the deciding factors was the reference of successful implementations of the technology in multiple companies and the expertise demonstrated by the vendor.

The implemented solution is based on a set of three hardware appliances in each datacenter – event collector, event storage and correlation engine. The hardware specifications are detailed in Table 2. To deal with multiple and remote event sources, additional software-based event collectors were deployed both inside and outside of the datacenters. In addition, to comply with regulatory requirements, it was necessary to make additional storage space available, guaranteeing long-term data archival. Finally, the SIEM console is a software component that can be installed in a standard off-the-shelf computer running a Microsoft Windows OS.

	Event Collector	**Event Storage**	**Correlation Engine**
System OS	Red Hat Enterprise Linux 6.2 64-bit	Red Hat Enterprise Linux 6.2 64-bit	Red Hat Enterprise Linux 6.2 64-bit
CPU	1 x Intel Xeon 2620 6-Core 2.0 GHz	2 x Intel Xeon 2648L 8-Core 1.8 GHz	2 x Intel Xeon E5620 4-Core 2.4 GHz
RAM	32 GB	64 GB	36 GB
Storage	4 x 500 GB (RAID 5)	4 x 3 TB (RAID 5)	6 x 600GB (RAID 10)

Table 2 – SIEM Appliance Specifications

Figure 13 represents the architecture deployed in EDP, outlining some of the most relevant event sources and their location relative to the SIEM. This representation is

limited to datacenter equipment, with remote locations having dedicated event collectors that convey the information to the SIEM platform in the datacenter, in line with the reference architecture presented in Section 2.1.

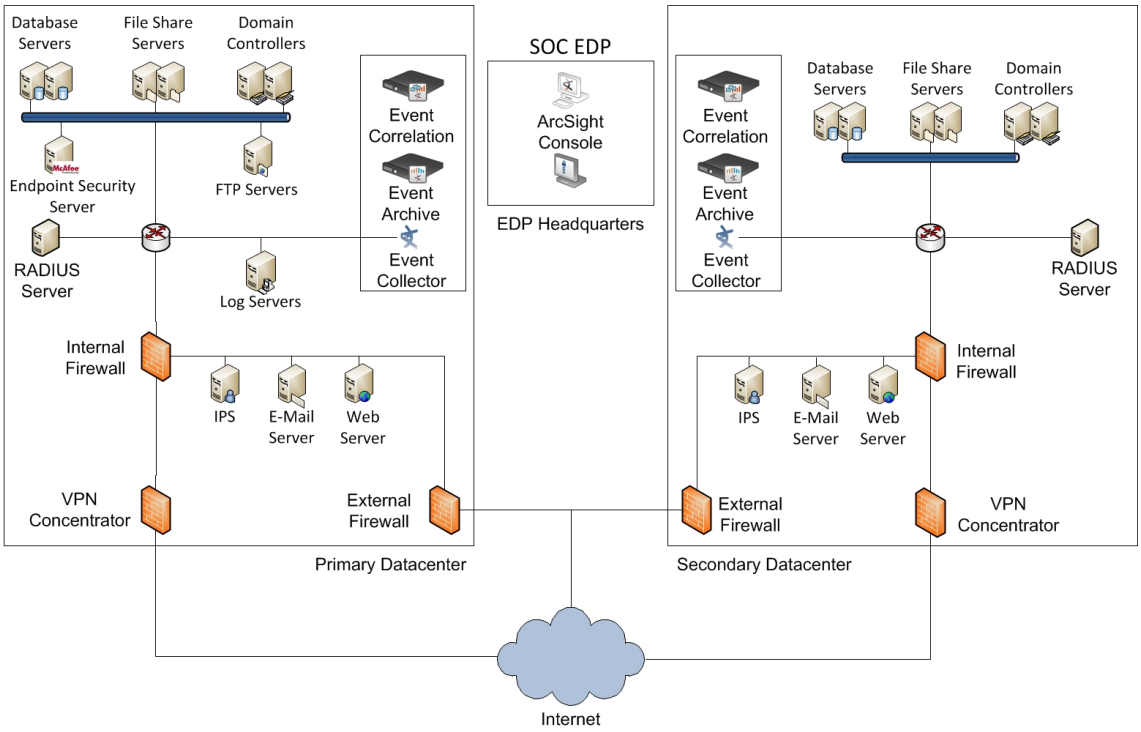

Figure 13 – EDP SIEM Architecture

The ArcSight SIEM platform includes a large set of correlation rules that are loaded out-of-the-box. The security team is expected to build upon this set of default rules, adapting the monitoring efforts to the specific characteristics of the infrastructure. We separate the default rule set from the internally developed correlation rules to allow the replication of our experiments. The default rule set includes 789 correlation rules, divided into the categories shown in Figure 14.

5.2 Analyzing Default Correlation Rules

To further demonstrate the applicability of AutoRule, we used it to perform an analysis of the default rule set that comes bundled with the ArcSight SIEM. It was considered that analyzing the 789 correlation rules would represent a significant effort, which would not warrant sufficient benefits to justify the lengthy process. Correlation rules share specific characteristics and approaches. Therefore, we believe that it is possible to select a representative sample, thereby reducing the number of rules to

study. Based on the previous analysis of correlation rules, we selected the most prominent characteristics to consider when classifying those rules. This classification is summarized in Table 3.

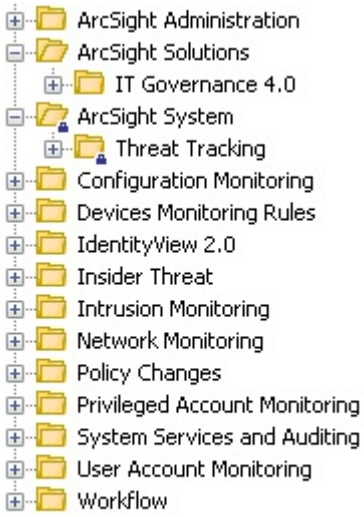

Figure 14 – Default Rule Set Tree

Considering the vectors established in Table 3, we sought the most frequently triggered rules in the EDP environment that could cover all possibilities. Table 4 shows the most relevant correlation rules, based on the number of detections in the experimental environment, indicating which classification vectors they cover. This subset of the default rules was used in the rest of the analysis, and their content is presented in Appendix A.

Correlation Rule Characteristics	Abbreviation
Based Primarily on Network Information	Net
Based Primarily on User Information	User
Oriented to Time Constraints	Time
Processes Events from Multiple Sources	Source
Detection of Repeated Attacks	Rep
Changes to System Setting	Sys

Table 3 – Significant Characteristics of Correlation Rules

It is possible to observe that the default rules rarely rely on multiple event sources, making them more vulnerable to attacks or accidental faults. Moreover, only one of the most frequently triggered rules contains both network and user related conditions.

Rule	Net	User	Time	Source	Rep	Sys
Activity from Badged Out Employee		X				
Failed Building Access	X					
Firewall - Application Protocol Scan	X		X		X	
Firewall - High Volume Accepts	X				X	
Firewall - Network Port Scan	X		X		X	
Multiple Failed Database Access Attempts		X	X		X	
Multiple Login Attempts to Locked Windows Account		X	X		X	
Multiple Windows Logins by Same User		X			X	
Pass After Repetitive Blocks	X		X		X	
Password Policy Changed						X
Physical Plus VPN Access	X	X		X		
Sabotage - Repetitive User Account Disabled		X			X	

Table 4 – Overview of the Default Rules Most Used in the EDP Environment

We used AutoRule to analyze the correlation rules and the results are shown in Table 5. The complete outputs of the analysis are presented in Appendix B.

Rule	Score
Activity from Badged Out Employee	8
Failed Building Access	5
Firewall - Application Protocol Scan	15
Firewall - High Volume Accepts	10
Firewall - Network Port Scan	15
Multiple Failed Database Access Attempts	5
Multiple Login Attempts to Locked Windows Account	5
Multiple Windows Logins by Same User	4
Pass After Repetitive Blocks	17
Password Policy Changed	5
Physical Plus VPN Access	8
Sabotage - Repetitive User Account Disabled	4

Table 5 – AutoRule Analysis of Built-in ArcSight Rules

The results show that there is significant room for improvements, especially in the correlation rules based on network conditions, which have the worst resilience scores. The reason is that these rules are focused only on information that can be easily forged, failing to corroborate that information with other variables. The most commonly found weaknesses are the absence of events from multiple sources, the isolated use of network or user information and the lack of conditions based on the network and asset models.

Figure 15 shows the score distribution, with the majority of correlation rules scoring between 5 and 9. We argue that it is possible to improve these scores using the proposed methodology, especially using source redundancy. The example given using Rule 10 shows an improvement from an initial score of 9, achieving a much more resilient correlation rule that scored 0 in the automatic analysis.

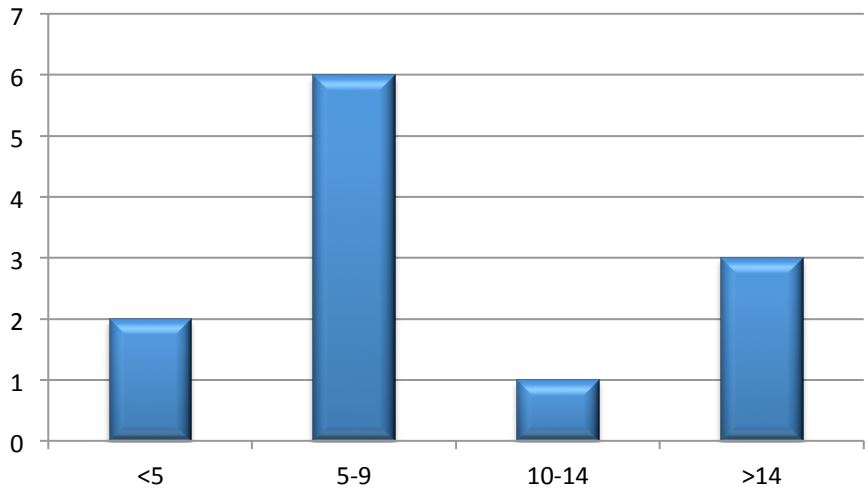

Figure 15 – AutoRule Score Distribution of Default Rules

5.3 Deploying Improved Correlation Rules

The validation of our proposal depended on the capability to improve existing correlation rules and demonstrate their added resilience in face of attacks. With that goal in mind, we adapted some of the existing rules in the SIEM, following the methodology introduced in Section 4.2.

In order to take full advantage of all our improvement proposals, we had to perform preparatory activities. These activities included careful identification of event sources, and the examination of event samples to determine relevant fields with meaningful information. We also loaded active lists with information regarding known actors and end-user equipment. Finally, we updated the network and asset models with

inventory information regarding application and database servers, including their identification, network addresses and criticality. As mentioned before, the level of infrastructure knowledge loaded into the SIEM is fundamental to increase the resilience and effectiveness of correlation rules.

Our approach was to test the correlation rules presented in Chapter 4 with information collected in the experimental environment. Information was recorded from the SIEM during two consecutive days of operation, with both the original and improved rules active in the correlation engine. Then, it was compared the number of times each rule was triggered, analyzing the quality of the alarms that were generated. Figure 16 shows the total number of events collected during the two days, nearly 171 million events, including the priority and overview of the event flow throughout the period.

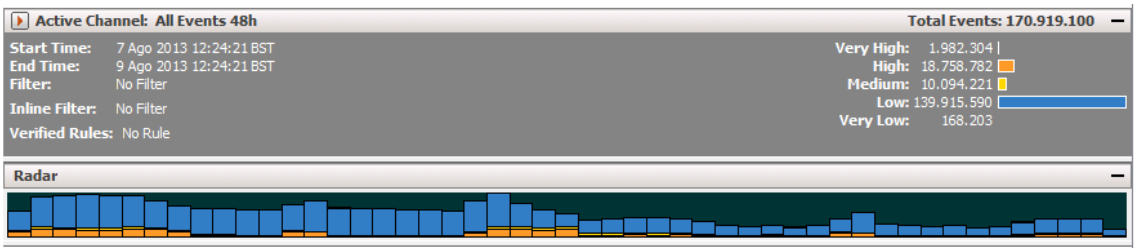

Figure 16 – Event Flow (48 Hours)

5.4 Results from the Improved Correlation Rules

Resorting to real events in a production environment has benefits and drawbacks. On one hand it is possible to observe how correlation rules can be applied in a real world scenario. But, on the other hand, the fact that this is a controlled environment limits the possibilities of observing ongoing attacks, and prohibits us from deliberately attacking the SIEM system.

Rule 1 was aimed at detecting changes to user accounts performed outside the IdM. However, the verification was performed using only the username of the entity performing the change. If an attacker is able to impersonate a valid source by using that username, changes can be performed unnoticed. Rule 5, Rule 6 and Rule 9 are improved versions of that correlation rule, using additional information. During the time period of our experiment, a single occurrence of this unauthorized action was recorded. Since there was no attempt to mask that action or to impersonate an authorized actor, both the original and improved versions of the correlation rule were able to detect that activity.

Rule 10 is also related to changes in user configuration through unauthorized channels, though it operates with an inverse logic. Instead of detecting unapproved changes, this correlation rule is triggered when sanctioned modifications are performed. During our experimental deployment, Rule 10 was triggered on 27 separate occasions, while there were 28 changes to user accounts recorded in the SIEM system. The analysis of these values also validates the data obtained from the previous correlation rules, resulting in a single alarm being raised due to unauthorized changes.

During the 48-hour period, Rule 3 was triggered 17259 times, signaling Brute Force Attempts. In other words, occurrences of five consecutive failed authentications from the same source within two minutes, using the same user name. The original Rule 2, a default rule from ArcSight, detected 1154 situations where the Brute Force Attempts were followed by a successful authentication from the same source. To demonstrate the effectiveness of individual steps of the methodology, we divide the results into two phases. The first step consisted of the elimination of the Trusted List from the rule and contributed to a 31% raise in the number of triggered alerts. Close analysis of the increased number of alarms showed that many systems have built-in processes and scripts that use stored credentials. These system accounts are loaded into trusted lists, since they are purportedly used only to integrate parts of the infrastructure and not for interactive logins. The result is that, by ignoring these failed attempts, Rule 2 is actually concealing a number of important events, especially considering that an attacker might be able to compromise the credentials of one of these system accounts.

The second phase was the completion of the methodology, resulting in the more robust Rule 7, which was triggered on 2318 occasions. While the original rule detected only successful authentications from the same source that generated the Brute Force Attempts, the improved version eliminates that restriction. The discrepancy shows that the credentials were being used in different systems or equipment at the same time. Upon further investigating a sample of the alerts generated by these rules, we concluded that there were no false positives, as all detections corresponded to effective authentication attempts. We also established that these are occurrences that result from the existence of scripts running with identical stored credentials in multiple systems, some of them not properly updated.

Table 6 summarizes the improvements measured as a percentage of the number of detections. As stated above, the increased information is relevant and can lead to infrastructure optimizations. After eliminating configuration problems in the monitored

systems, the number of raised alarms would diminish greatly, helping the efforts from the security team to detect actual malicious attacks.

	Rule 2	Rule 2 without Trusted List	Rule 7
Number of Alerts Triggered	1154	1507	2318
Improvement over baseline	N/A	31%	101%

Table 6 – Correlation Rule Improvements

Both Rule 4 and its altered version Rule 8 were not triggered during the two-day period when the experiences were conducted. As we mentioned in Chapter 4, this was the subtlest change in all the improvement examples, as the adjustment of time related conditions must be based on statistical data and experience. Considering the available information, we firmly trust that Rule 8 will enable the detection of additional occurrences. However, the inexistence of such attacks in the implementation environment limited our capability to demonstrate the added effectiveness.

5.5 Simulating Failures

As we mentioned earlier, the absence of malicious attacks in the experimental environment increases the difficulty to determine the real added value of improving correlation rules. This is especially true when we consider the addition of events from multiple sources, as performed in Rule 10.

The possibility of interfering with the normal operation of a production environment removes the option of performing attacks against the SIEM system. However, it is possible to simulate specific failures, such as lost messages. To accomplish that objective, it was necessary to apply a filter to the events collected during the 48-hour period of our experiment. We removed all the events originating from domain controllers and firewalls, which meant that changes to user accounts were only present in the IdM database events.

The results from this test are summarized in Table 7. For simplicity, the number of detections from Rule 10 is shown as the number of alarms raised due to the detection of incoherent events.

	Rule 1	**Rule 6**	**Rule 10**
AutoRule Score	9	5	0
Detections Using all Events	1	1	1
Detections After Event Filtering	0	0	28

Table 7 – Correlation Rule Resilience (Simulated Failures)

It is thus possible to observe that, in the presence of a failure in the event collection process from part of the sources, Rule 10 is still able to generate relevant alarms. Rule 10 considers information from the Domain Controller and the IdM database and expects to encounter corresponding events in both sources. When information coming from the sources is not coherent, an alarm is triggered. In this scenario, the absence of events from one of the sources results in alarms being triggered every time an event is collected and the correlation engine is unable to match it with the corresponding event from the other source. The outcome is that the security team is not only alerted to the unauthorized change to a user account, but also to the problem in the event collection process.

5.6 Result Analysis

By eliminating frailties in the correlation rules using the proposed methodology, it was possible to increase the number of behavior patterns detected, and alarms triggered, using the same sample of collected events, as shown with Rule 7. The usage of this improved version of the correlation rule means that security teams can benefit from additional information. Using events from multiple sources, as deployed in Rule 9 and Rule 10, can contribute to the detection of malicious actions, even in the presence of partial failures in the infrastructure.

The experimental environment, while possibly limiting the ability to demonstrate the effectiveness of all improvements to correlation rules in face of malicious attacks, provided a valuable real world scenario to deploy the proposed methodology.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This document presents a methodology for the design of resilient correlation rules and their implementation in a SIEM system. The resilience of correlation rules has great impact on the accuracy of these systems, especially in the face of malicious attacks that may compromise the security of the monitored infrastructure.

From the premise that improving correlation rules can contribute to improve the capabilities of SIEM systems, the proposed methodology focused on identifying and eliminating vulnerabilities in the rules. It is possible to do so by removing exceptions from the rules, increasing the information used by the correlation engine and combining data from multiple sources. The systematized approached made possible the development of AutoRule, an application capable of analyzing correlation rules, proposing improvements and calculating an overall resilience score. The possibility of performing guided improvements acting directly on the areas with most impact is extremely relevant, as the number of deployed rules in a SIEM correlation engine can easily reach the hundreds. Following the warnings and recommendations it is possible to implement correlation rules that achieve better results in detecting potentially malicious behavior.

Employing the methodology, and guided by the automatic analysis performed by AutoRule, we deployed improved correlation rules in a live scenario. Measurements show that the number of abnormal behavior detections increased following the removal of restrictive conditions and exceptions from the correlation rules.

Although the syntax to implement correlation rules may vary, the principles presented here are not specific to a vendor. The methodology presented is valid to any SIEM system with negligible adaptations.

6.2 Future Work

When considering correlation rules based on events from multiple sources, it may be possible to measure a level of confidence in alarms by using the correlated information, such as voting schemes. Information redundancy is therefore an important concept to consider when determining the capability to accurately trigger SIEM alarms. In this case, information redundancy translates into being able to collect related data from multiple sources, enabling the use of a voting scheme to detect outliers. The confidence in the outcome of the voting process can be measured by the ratio of outlying data against consistent reports.

Time redundancy consists in performing similar verifications of the same event flow at different instants in time against a common set of correlation rules, which, in the absence of failures, should produce equivalent outcomes [2]. Time redundancy can be employed to detect successful attacks against the SIEM system, namely the correlation engine component, with the cost of having more than one of those components operating simultaneously.

References

[1] Aguirre, I., Alonso, S., "Improving the Automation of Security Information Management: A Collaborative Approach", IEEE Security and Privacy Magazine, January / February 2012.

[2] Aidemark, J., Karlsson, J., "Using Massive Time Redundancy to Achieve Node-level Transient Fault Tolerance", ARTES Graduate Student Conference, 2000.

[3] Briesemeister, L., Cheung, S., Lindqvist, U., Valdes, A., "Detection, Correlation, and Visualization of Attacks Against Critical Infrastructure Systems", in *Proceedings of Eighth Annual Conference on Privacy, Security and Trust*, Ottawa, Canada, August 2010.

[4] Falliere, N., Murchu, L., Chien, E. "W32.Stuxnet Dossier", Symantec Tec Rep. February 2011.

[5] Gartner, http://www.gartner.com/it-glossary/security-information-and-event-management-siem/, accessed in July 2013.

[6] Haddix, J. et al, "HP 2012 Cyber Risk Report", March 2013.

[7] MASSIF FP7 Project, http://www.massif-project.eu/.

[8] Miller, D., Harris, S., Harper, A., Vandyke, S., Blask, C., "Security Information and Event Management (SIEM) Implementation", McGraw-Hill Osborne, 2010.

[9] Nawyn, K., "A Security Analysis of System Event Logging with Syslog", SANS Institute, 2003.

[10] Nicolett, M., Kavanagh, K., "Magic Quadrant for Security Information and Event Management", May 2012.

[11] Oliner, A., Kulkarni, A., Aiken, A., "Community Epidemic Detection using Time-Correlated Anomalies", in *Proceedings of the 13th International Conference on Recent Advances in Intrusion Detection*, Ottawa, Canada, 2010.

[12] Oracle Security Alert for CVE-2013-0422. http://www.oracle.com/technetwork/topics/security/alert-cve-2013-0422-1896849.html

[13] Stallings, W., “Cryptography and Network Security”, Fifth Edition, Pearson Education, 2011.

[14] Todd, A., Raines, R., Baldwin, R., Mullins, B., Rogers, S., “Alert Verification Evasion Through Server Response Forging”, in *Proceedings of the 10th International Conference on Recent Advances in Intrusion Detection*, Gold Coast, Australia, 2007.

[15] Valdes, A., “Blue Sensors, Sensor Correlation, and Alert Fusion”, in *Proceedings of the 3rd International Conference on Recent Advances in Intrusion Detection*, Toulouse, France, 2000.

[16] Valdes, A., Skinner, K., “Probabilistic Alert Correlation”, in *Proceedings of the 4th International Conference on Recent Advances in Intrusion Detection*, Davis, CA, USA, 2001.

[17] Veríssimo, P., Neves, N., Correia, M., “Intrusion Tolerant Architectures: Concepts and Design”, Technical Report DI/FCUL TR-03-5, Department of Computer Science, University of Lisbon, April 2003.

Appendix A – Default Correlation Rules

1 Matching 1 events in 5 Minute with conditions(
2 "Not" InActiveList(event1.sourceUserName,/All Active Lists/ArcSight Solutions/IdentityView/Book Keeping/Badged In Actors);And;
3 EQ(event1.name,Successful Building Access Event))

Activity from Badged Out Employee

1 Matching 1 events in 5 Minute with conditions(
2 EQ(event1.name,Failed Building Access Events))

Failed Building Access

1 Matching 3 events in 3 Minute with conditions(
2 (EQ(Protocol_Deny.categoryBehavior,/Access);Or;
3 EQ(Protocol_Deny.categoryBehavior,/Access/Start));And;
4 "Not" InActiveList(Protocol_Deny.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;
5 ("Not" InActiveList(Protocol_Deny.sourceAddress,/All Active Lists/ArcSight System/Threat Tracking/Reconnaissance List);Or;
6 "Not" InActiveList(Protocol_Deny.destinationAddress,/All Active Lists/ArcSight System/Targets/Scanned List));And;
7 EQ(Protocol_Deny.categoryDeviceGroup,/Firewall);And;
8 EQ(Protocol_Deny.categoryOutcome,/Failure))

Firewall - Application Protocol Scan

1 Matching 1 events in 5 Minute with conditions(
2 "Not" InActiveList(FirewallAcceptsMovingAverageEvent.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;
3 EQ(FirewallAcceptsMovingAverageEvent.name,Firewall Accepts);And;
4 EQ(FirewallAcceptsMovingAverageEvent.deviceEventCategory, rising))

Firewall - High Volume Accepts

```
1       Matching 5 events in 3 Minute with conditions(
2       (EQ(Deny_TCP_UDP.categoryBehavior,/Access);Or;
3       EQ(Deny_TCP_UDP.categoryBehavior,/Access/Start));And;
4       "Not" InActiveList(Deny_TCP_UDP.sourceAddress,/All Active Lists/ArcSight
System/Attackers/Trusted List);And;
5       ("Not" InActiveList(Deny_TCP_UDP.sourceAddress,/All Active Lists/ArcSight
System/Threat Tracking/Suspicious List);Or;
6       "Not" InActiveList(Deny_TCP_UDP.destinationAddress,/All Active Lists/ArcSight
System/Targets/Scanned List));And;
7       EQ(Deny_TCP_UDP.categoryDeviceGroup,/Firewall);And;
8       EQ(Deny_TCP_UDP.categoryOutcome,/Failure))
```

Firewall - Network Port Scan

```
1       Matching 3 events in 1 Minute with conditions(
2       NE(event1.type,Correlation);And;
3       EQ(event1.categoryBehavior,/Authentication/Verify);And;
4       EQ(event1.categoryObject,/Host/Application/Database);And;
5       EQ(event1.categoryOutcome,/Failure))
```

Multiple Failed Database Access Attempts

```
1       Matching 5 events in 2 Minute with conditions(
2       EQ(event1.type,Base);And;
3       EQ(event1.deviceProduct,Microsoft Windows);And;
4       EQ(event1.deviceVendor,Microsoft);And;
5       EQ(event1.deviceEventClassId,Security:531))
```

Multiple Login Attempts to Locked Windows Account

```
1       Matching 1 events in 1 Minute with conditions(
2       GE(event1.LoginCountActiveList,5);And;
3       EQ(event1.name,Successful Windows Login))
```

Multiple Windows Logins by Same User

```
1       Matching 1 events in 1 Minute with conditions(
2       (InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight
Foundation/Intrusion Monitoring/Attackers/Repetitive Firewall Block List);Or;
3       InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight
System/Attackers/Untrusted List));And;
4       (EQ(SuspiciousFirewallPass.categoryBehavior,/Access);Or;
5       EQ(SuspiciousFirewallPass.categoryBehavior,/Access/Start));And;
6       "Not" InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight
System/Attackers/Trusted List);And;
7       "Not" InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight
System/Threat Tracking/Suspicious List);And;
8       EQ(SuspiciousFirewallPass.categoryDeviceGroup,/Firewall);And;
9       EQ(SuspiciousFirewallPass.categoryOutcome,/Success))
```

Pass After Repetitive Blocks

```
1       Matching 1 events in 1 Second with conditions(
2       EQ(event1.name,Windows Event);And;
3       (EQ(event1.deviceEventClassId,Security:643);Or;
4       EQ(event1.deviceEventClassId,Microsoft-Windows-Security-Auditing:4739));And;
5       EQ(event1.message,Password Policy);And;
6       EQ(event1.type,Base);And;
7       EQ(event1.categoryOutcome,/Success))
```

Password Policy Changed

```
1       Matching 1 events in 2 Minute with conditions(
2       EQ(event1.name,Address or Username Present);And;
3       NE(event1.type,Correlation);And;
4       EQ(event1.categoryBehavior,/Authentication/Verify);And;
5       EQ(event1.categoryDeviceGroup,/VPN);And;
6       InActiveList(event1.ActorByAccountID,/All Active Lists/ArcSight
Solutions/IdentityView/Book Keeping/Badged In Actors))
```

Physical Plus VPN Access

```
1       Matching 1 events in 2 Minute with conditions(
2       EQ(event1.deviceProduct,ArcSight);And;
3       EQ(event1.deviceVendor,ArcSight);And;
4       EQ(event1.deviceCustomNumber1,3);And;
5       EQ(event1.filePath,Disabled Accounts);And;
6       EQ(event1.name,ActiveList entry updated))
```

Sabotage - Repetitive User Account Disabled

Appendix B – AutoRule Outputs

Username reference should be complemented with network information:
2 "Not" InActiveList(event1.sourceUserName,/All Active Lists/ArcSight Solutions/IdentityView/Book Keeping/Badged In Actors);And;

Warning - Account verifications should rely on the asset model
Warning - The rule does not use multiple event sources
Final score: 8

AutoRule Evaluation of Rule Activity from Badged Out Employee

Warning - The rule does not use multiple event sources
Final score: 5

Failed Building Access

Possible loophole in active list exceptions:
4 "Not" InActiveList(Protocol_Deny.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;
Address reference should be complemented with account information
5 ("Not" InActiveList(Protocol_Deny.sourceAddress,/All Active Lists/ArcSight System/Threat Tracking/Reconnaissance List);Or;
Address reference should be complemented with account information
6 "Not" InActiveList(Protocol_Deny.destinationAddress,/All Active Lists/ArcSight System/Targets/Scanned List));And;

Warning - Network conditions should rely on the network model
Warning - The rule does not use multiple event sources
Final score: 15

Firewall - Application Protocol Scan

Possible loophole in active list exceptions:
2 "Not" InActiveList(FirewallAcceptsMovingAverageEvent.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;

Warning - The rule does not use multiple event sources
Final score: 10

Firewall - High Volume Accepts

Possible loophole in active list exceptions:
4 "Not" InActiveList(Deny_TCP_UDP.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;
Address reference should be complemented with account information
5 ("Not" InActiveList(Deny_TCP_UDP.sourceAddress,/All Active Lists/ArcSight System/Threat Tracking/Suspicious List);Or;
Address reference should be complemented with account information
6 "Not" InActiveList(Deny_TCP_UDP.destinationAddress,/All Active Lists/ArcSight System/Targets/Scanned List));And;

Warning - Network conditions should rely on the network model
Warning - The rule does not use multiple event sources
Final score: 15

Firewall - Network Port Scan

Warning - The rule does not use multiple event sources
Final score: 5

Multiple Failed Database Access Attempts

Warning - The rule does not use multiple event sources
Final score: 5

Multiple Login Attempts to Locked Windows Account

Warning - The rule does not use multiple event sources
Final score: 4

Multiple Windows Logins by Same User

Possible loophole in active list exceptions:
6 "Not" InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight System/Attackers/Trusted List);And;
Address reference should be complemented with account information
2 (InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight Foundation/Intrusion Monitoring/Attackers/Repetitive Firewall Block List);Or;
Address reference should be complemented with account information
3 InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight System/Attackers/Untrusted List));And;
Address reference should be complemented with account information
7 "Not" InActiveList(SuspiciousFirewallPass.sourceAddress,/All Active Lists/ArcSight System/Threat Tracking/Suspicious List);And;

Warning - Network conditions should rely on the network model
Warning - The rule does not use multiple event sources
Final score: 17

Pass After Repetitive Blocks

Warning - The rule does not use multiple event sources
Final score: 5

Password Policy Changed

Address reference should be complemented with account information
2 EQ(event1.name,Address or Username Present);And;

Warning - Network conditions should rely on the network model
Warning - The rule does not use multiple event sources
Final score: 8

Physical Plus VPN Access

Warning - The rule does not use multiple event sources
Final score: 4

Sabotage - Repetitive User Account Disabled