Model-based Security Event Management

Julian Schütte[1], Roland Rieke[2], Timo Winkelvos[2]

[1] Fraunhofer Institution AISEC, Munich, Germany
[2] Fraunhofer Institute SIT, Darmstadt, Germany

Abstract. With the growing size and complexity of current ICT infrastructures, it becomes increasingly challenging to gain an overview of potential security breaches. Security Information and Event Management systems which aim at collecting, aggregating and processing security-relevant information are therefore on the rise. However, the event model of current systems mostly describes network events and their correlation, but is not linked to a comprehensive security model, including system state, security and compliance requirements, countermeasures, and affected assets. In this paper we introduce a comprehensive semantic model for security event management. Besides the description of security incidents, the model further allows to add conditions over the system state, define countermeasures, and link to external security models.

Keywords: security strategy meta model, security information and event management, complex event processing

1 Introduction

Today, more and more critical assets are managed by complex ICT infrastructures such as in SCADA systems or heterogeneous and large-scale company networks. Many of these systems are subject to attacks on a daily basis, ranging from mostly harmless drive-by attacks in the form of automated and unsighted scans to targeted insider attacks.

While traditional Security Information and Event Management (SIEM) solutions focus on the mere detection of incidents and usually work at a specific level of abstraction, support for multi-layer correlation and explanation of security implications is scarce. Thus, the relation of any results of these systems to certain security properties or requirements is uncertain. It is hardly possible to derive the consequences of detected incidents on a system scale or process level. Furthermore, it remains a challenge to include information from sensors that go beyond the traditional network and security scanners, especially if these sensors are specific to the domain the system.

Therefore, it is evident that organizations need to broaden their IT monitoring concepts, and incorporate technologies that are designed to look at the application layer and provide detection of application level attacks in near real time [1].

The aim of this work is to enable techniques for interrelating information of different levels of abstraction and of different domains in order to infer more

valuable statements about threats in a monitored system. We introduce a modeling approach that facilitates the definition of security probes on different levels of detail, allows to refer to security threats and requirements and enables to integrate a variety of information sources into a thorough information security monitoring.

Current SIEM engines, based on Complex Event Processing (CEP), suffer from mainly three weaknesses, which we try to address in this paper:

First, when incidents are described in a proprietary event processing language, without any semantics linked to the incident definition, it becomes hard for users to understand the actual implications of an incident. Incident definitions are thus more complex, error-prone and harder to maintain.
Second, as the definition of incidents does not follow a formal model, it is not possible to extend it by additional information, such as models of the possible security implications, affected assets, possible remedies, etc.
Third, without such a formal model, it is highly complex to include and correlate additional information from field sensors into the existing incident definitions.

Our approach is therefore to reduce the complexity of security probes by means of an *Security Strategy Meta Model* (SSMM), abstracting from the event processing language. The SSMM allows users to define security monitors at an abstract, less technical layer which is independent from the underlying CEP engine and can be linked to further information, describing possible countermeasures or violated security requirements.

This paper is organized as follows. In Section 2, we reference related work and point out the requirements for the SSMM. In Section 3, we introduce the details of the model and put it into relation to further existing models describing security and infrastructure-specific aspects. Section 4 demonstrates the approach by example of a misuse case and Section 5 concludes the paper and sketches future work.

2 Motivation and Related Work

In this section we first point out what the deficits of existing SIEM solutions are, and derive a list of requirements for an advanced model-driven security event management system.

2.1 State of the art

The most important types of current threats are identified in [1]; and advanced monitoring techniques such as file integrity monitoring, database activity monitoring, application monitoring, identity monitoring, and user activity monitoring are discussed. In [2], some challenges with respect to collecting and analyzing a multi-gigabit network stream are outlined.

The event and knowledge representations of traditional SIEM solutions show where current limitations of these systems are located, when it comes to a comprehensive analysis of security properties. The existing solutions are very specific and explicitly designed to solve a certain type of problem:

Akab [5] is a SIEM appliance which is mainly focused at monitoring network events. It is based on the proprietary *Akevent* format and stores collected events persistently in a database. Prelude [9] is an open source SIEM framework which relies on the open IDMEF [10] event format. Using the LUA language, developers can write their own correlation modules. The open source SIEM engine OSSIM [4] aims at detecting security events at network (i.e., IP) layer. Consequently, its event format contains attributes like *IP address*, *protocol*, *port number*, and *severity of an incident*. In [8] it is shown how OSSIM can be extended to allow for safety analysis by correlating information which is produced by the security devices adopted in a dam network scenario, with information produced by safety sensor devices. New OSSIM plugins had to be developed and new correlation rules are needed to implement this approach to combined security and safety analysis at runtime. All of these engines have in common that they rely on an event representation syntax, but do not foster a comprehensive event model which links aspects like detection, correlation, reaction, and impact explanation. Moreover, due to the lack of a clear semantics, such event representations cannot serve as a basis for thorough analysis of indicators, which is required to handle potentially huge indicator models. Zabbix, another open source solution focuses mainly at aggregating potentially security-relevant incidents in a common monitoring dashboard and allows users to define simple triggers, e.g. in order to set up notifications.

Among the most mature commercial products are ArcSight ESM and IBM Tivoli Security Information and Event Manager [7]. Their main strength is to relate incidents to compliance catalogs and corporate policies, but the observed events are also predefined by technical attributes such as *source* and *destination host*, *severity*, *user account*, and others [16]. The RSA Archer Threat Monitor maintains a catalog of assets and links it to security-relevant information such as known vulnerabilities, patch levels, etc. Archer itself does however neither collect nor aggregate this information. A common format is the Common Event Format (CEF) [6], also used by ArcSight, for example, but it specifies only the syntax for event representation but does not provide any semantics.

The *Engineering Knowledge Base* (EKB) [17] is an ontology relating sensor values and combining runtime with development time models to analyze industrial automation systems and is used to define SPARQL or SWRL queries over sensor definitions. As we have a similar goal of finding inconsistencies, we believe that an approach like the EKB could help defining which inconsistencies to look for in event streams, and thus, which measurement points might indicate violations of the security requirements. Other approaches of interest to this end are the modeling concepts in [14], where business, application, physical, and technical information is merged and related, as well as concepts to use event-triggered rules for sensing and responding to business situations in [18].

2.2 Requirements

Our aim is to overcome the contextual restrictions of existing solutions with their predefined and closed models and rather provide an extensible model that

comprises all parts of the security monitoring and decision support process: (i) detecting threatening events; (ii) putting them in context of the current system state; (iii) explaining their potential impact with respect to some security- or compliance model; and (iv) taking appropriate actions. Thus, we establish the rationale for the SSMM through a list of requirements that state a set of required properties of the meta-model. In Section 3, we will define the language and processes from which to form a model, which satisfies these requirements.

Requirement 1 (Abstract from event processing languages) *As system operators are not necessarily experts in security monitoring, or SIEM solutions, the model should abstract from the specific event processing languages and vendor specific incident definitions.*

Requirement 2 (Correlation across layers and incidents) *The SIEM engine must be backed by a comprehensive model which allows to correlate incoming events "vertically" and "horizontally".*

Requirement 3 (Inclusion of context information) *Although most current SIEM solutions lack the possibility of correlating alarms with additional context events, it might be necessary in many cases to take additional context information into account.*

Requirement 4 (Model reactions to incidents) *While most SIEM only focus at reporting security incidents, the SSMM should include different ways to handle the incident.*

Requirement 5 (Retro-traceability of security requirements) *It must be possible to automatically link security incident events to a security model that provides additional information about the actual impact of an incident, such as the violated security requirements, concerned assets, and possible countermeasures.*

3 The Security Strategy Meta Model

In this section, we introduce the actual *Security Strategy Meta Model* (SSMM) whose purpose is to describe in a simple and semantically concise way how security incidents should be detected and handled. In the following, we use the term Security Strategy Model for a concrete instance of the SSMM.

An *Security Directive* is the root concept of the SSMM and combines the semantically modeled concepts, which are translated into specific queries and processed by the SIEM engine at runtime. Thus, addressing Requirement 1, a *Security Directive* provides an abstraction from specific event processing languages.

There are two main ways to specify an Security Directive: one is to solely use the structure of the SSMM, but directly formulate queries for specific CEP engines and databases. The other is to describe the Security Directive exclusively using the meta-model. Each Security Directive is structured as follows:

- `on` (`EventStreamProperty`)
- `if` (`Condition`)
- `do` (`Action`)
- `why` (`SecurityPertinence`)

Requirement 2 is addressed by the `on` part, which models security-relevant event patterns by means of an `EventStreamProperty` concept.

Whenever an event pattern is detected, the condition denoted by the `if` part is checked and if it evaluates to true, a security indicident has been detected and requires for a reaction. This allows for inclusion of context and state information, and thereby addresses Requirement 3.

Reactions are modeled by the `do` part (addressing Requirement 4) and refer to an executable `Action`, whereas the model distinguishes between *internal* and *external* actions. Internal actions update the knowledge base, i.e., they can add facts to the domain model or meta data model, as well as they can be used to set system state parameters which can be used in subsequent condition evaluations. External actions, in contrast, refer to loadable plugins which can be used to notify users, write to a database, or take counteractive measures by reconfiguring a firewall, for example.

The `why` part addresses Requirement 5 and refers to an explanation of the incident and may help users to estimate the potential impact and the indicent's relation to the security model. It will be reasonable to link the `why` property either to some compliance catalog (e.g., ISO27004 [13]), or to a formal security model (e.g., the Security Modelling Framework presented in [12]), so as to allow for a quantification of security.

In this paper, we focus mainly on the recognition of security incidents, i.e. the `on` and the `if` property of the Security Directive.

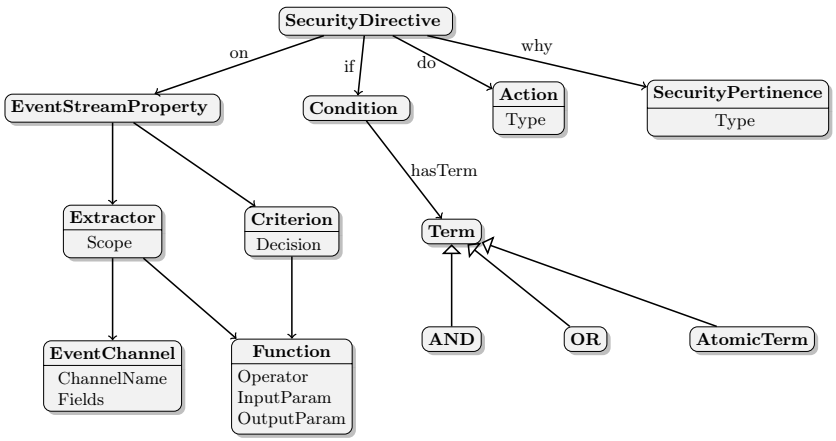

Fig. 1. Structure of a Security Directive

3.1 `EventStreamProperty`

An `EventStreamProperty` models patterns of events indicating a security incident. *anomalies*, meaning a property of the behave as expected, or attack *signatures* which are precise event patterns indicating a security incident. An EventStreamProperty comprises the following elements:

- `Extractor` [1 .. n] Extractors extract attributes from event channels and provide it for further processing.
- `Criterion` [1] There has to be exactly one `Criterion`, which triggers the Security Directive. The `Criterion` specifies how the attributes *extracted* from the event stream should be evaluated.

`Extractor` Points to those attributes of an event channel that should be selected from the stream.

- `EventChannel` [1] denotes the channel from which information is extracted
- `Function` [0 .. 1] denotes an optional function to be applied to the parameters

`Criterion`

- `Function` [0 .. n] can be applied to to `Parameters` provided by one or more `Extractors` (and with that, event channels)
- Makes use of provided values of `Parameters`
- `Decision` [1] The boolean parameter which indicates if the Criterion has been positively evaluated. Must be provided by one of the functions above.

`EventChannel` Identifies the event channel from which attributes should be extracted. An `EventChannel` is defined by the following properties:

- `ChannelName` [1] is the identifier of the channel
- `Fields` [1 .. n] determine which attributes will be *extracted* from the event channel and provided for further processing
- `SchemaName` [0 .. 1] identifies the schema of data of the event channel, if any.
- `ChannelSensors` [0 .. n] describe the sources of an event channel. This additional information might be helpful when porting or reusing the Security Directive.

`Function` A function takes the extracted attributes from an event channel as input, applies an operation to them and returns a value. This concept is where the actual processing of event attributes is declared and is thus an essential part of the model. At the moment, `Function` is predefined by a set of operators which we expect to be used frequently when setting up Security Directives.

- `InputParam` [0 .. n] specifies input parameters. These can either be static values (e.g., when using threshold functions) or refer to parameters provided by the `Extractor`.

- `Operator` defines n-ary operators. By logical concatenation, complex operators can be defined and re-used for other Security Directives. At the moment, we support the following operators:
  - Arithmetics: $+, -, \cdot, /, \%\,(mod), ln, log, exp$
  - Comparison: $>, <, >=, <=, ==, \neq$
  - Logical Concatenation: $\wedge, \vee, \neg$
  - Special Functions:
    - $corr(a, b, cor)$: Correlation (with specific *cor* or just + or -, the latter meaning that values of a, b positively or negatively correlate).
    - $avg(parameter, scope)$: Average of `Parameter` over the time frame defined by `Scope`.
    - $max(values)$, $max(field, scope)$, $min(values)$, $min(field, scope)$
    - $anomaly(norm, deviation)$ Defines an anomaly, meaning a deviation from the normal behavior.
- `OutputParam` defines the output. In case of the `Criterion`'s `Function`, this is always a boolean.

`Scope` A `Scope` defines the window over which aggregate operators like sum, avg, max, or min are applied to event channels. It has the following properties:

- `Type` refers to either number of events, or time in seconds
- `Value` denotes the actual number of events or second.

3.2 `Condition`

A `Condition` allows to match Security Directives only in certain system states and is evaluated whenever a `Criterion` is positively asserted. It refers to a boolean expression over `Terms` and when evaluated, returns a modal decision (`true`/`false`), along with a set of `Parameters`, representing the results of specific `Queries`.

- `hasTerm Term` [1] (`AtomicTerm` | `AND` | `OR`)
  `AND` and `OR` represent the boolean operators and refer to two `Terms` again.

`AtomicTerm` An AtomicTerm consists of a `Query` and a `Criterion`. The SSMM includes different query types, each comprising the actual query string, the set of parameters which the query provides to the SIEM engine, and a set of query-specific meta information, such as a database name for an SQLQuery, for example. Currently, the SSMM includes the following query types:

- `SQLQuery` Queries an SQL database. Besides the query string, database name, account, and URL have to be provided.
- `AMSECQuery` Queries the Attack Modelling and Security Evaluation Component [15]
- `PSAQuery` Queries the Predictive Security Analyser [19,11]

4 Modeling a Misuse Case

For illustrating the application of the model in a misuse case, we consider a hypothetical but realistic attack in a SCADA system for monitoring a dam infrastructure, based upon the security requirements that were derived in the project MASSIF [3] from the Terni hydroelectric complex, located about 150 Km in the north of Rome: the attacker, one of the workers at the dam system, steals the administrator password for the dam control station. He then uses his own legitimate RFID badge to enter the dam control station room and logs into the control system using the stolen administrator password. With the administration console under his control, he installs a software that intercepts and drops all control messages from the power plant. In the following, he sabotages the discharge sensor so that it would not indicate an increased discharge through the penstocks. Finally, he opens the discharge gates so water flows uncontrolled through the penstocks and the turbine starts to produce energy.

Due to the compromise of the water flow sensors, the dam control station does not indicate the increased flow through the penstocks. Furthermore, as requests from the power plant have been blocked by the malicious software, requests from the power plant to stop the discharge are ignored and the turbine continues to produce power in an uncontrolled way. This can lead to a situation called *islanding*, in which a part of the electric grid is separated from the rest of the grid, resulting in severe damage of the turbine and the grid infrastructure.

The following listing shows how this attack could be detected by correlating measurements across different layers. The security directive correlates sensor values from the water flow through the penstocks with the produced current at the turbine. As in normal operation, the values should always be positively correlated, the security directive detects the attack described above whenever the correlation falls below a threshold of 0.3.

Listing 1.1. Security Directive to monitor the correlation of current and throughput.

```
Security Directive Name=ManipulationSensorsSabotage {
:on [ :dischargeCurrentCorrelation
      :hasExtractor [
            :hasEventChannel [ rdf:type :DischargeLevel;
                  :hasFields "throughput";
                  :hasName "Discharge Level in Penstock"
                  :hasChannelSensors :dischargeSens ]
        :hasExtractor [
            :hasEventChannel [ rdf:type :PowerInductionLevel;
                  :hasFields "current";
                  :hasName "Power Induction Level"
                  :hasChannelSensors :currentSens ]
        :hasCriterion [ :hasFunction [:hasOperator :correlate
                                      :hasParam1 "throughput";
                                      :hasParam2 "current";
                                      :outputParam "correlation" ];
                                :hasFunction [:hasOperator :lt
                                      :hasParam1 "correlation";
                                      :hasParam2 0.3;
                                      :hasOuputParam "threshold" ];
                        :hasDecision "threshold" ] ]

:if [ rdf:type :SQLCondition;
      :hasQuery ="SELECT Role, Employer FROM PhysicalPresenceTable" ;
```

```
      :hasCriterion [ :hasFunction [ :hasOperator "==";
                                     :hasParam1 "Role";
                                     :hasParam2 "\"Admin\"" ]];
        :dbInfo [ :ipAdress "192.168.178.54"; :portNumber=31337]  ]

:do  [... left out for the sake of brevity ...]

:why [... left out for the sake of brevity ...]
}
```

Listing 1.2. Generated EPL Query

```
SELECT   sourceIP?,
         avg(cast(traffic?,float)) AS avgTraffic
FROM     SyslogChannel.win:time(30 sec)
HAVING   cast(avgTraffic?,float)>42
```

This scenario mainly served as a guideline for our prototype implementation. We wrote the model as an OWL2 ontology, which is parsed by our prototype engine and compiled into queries for the Esper CEP engine (c.f. Listing 1.2). Whenever an incident is detected, the SIEM prototype links back the detected event pattern to the incident model, so users receive a semantic description of the incident, which they can use as a starting point for a more detailled inspection.

5 Conclusion

In this paper we introduced a model-based approach to the definition of event driven security incident detection and handling. The model supports security monitoring by correlating events from different layers. Detected complex events can be matched against the current system state, where we intend to support different components representing the network infrastructure, as well as components providing attack and vulnerability information, and a predictive security analyser. Some of these components are currently developed within the MASSIF project. Furthermore, the model includes references to actions to be taken when an incident has been detected. In order to put incidents into the context of high-level security requirements, e.g. from compliance catalogs, the model comprises a SecurityPertinence link. We deem the strength of this model-based approach as threefold:

First, the model is comprehensive and brings together all parts of security monitoring which are to date covered by different systems: detection (IDS), reporting (SIEM), handling (like an IRS), and explaining (GRC) of security incidents. So, the model will support an integration of these existing systems into one coherent monitoring solution.

Second, the model abstracts from specific event formats, sensors, or query languages and thereby allows a mapping to different underlying event engines.

Third, representing Security Directives in a semantic model supports a separation of concerns, where a system administrator might provide details on the network infrastructure, the compliance department might provide a list of high-level requirements, and a security officer will combine them in a Security Directive definition, for example.

Our future work aims at extending the existing prototype, integrate it with an IF-MAP server and linking it to specific compliance catalogs, in order to test its practical usefulnes.

Acknowledgments. This work has been developed in the context of the project MASSIF (ID 257475), co-funded by the European Commission within the Seventh Framework Programme.

References

1. Monitoring up the Stack: Adding Value to SIEM. White paper, Securosis L.L.C., Phoenix, AZ (2010)
2. Applied Network Security Analysis: Moving from Data to Information. White paper, Securosis L.L.C., Phoenix, AZ (2011)
3. Project MASSIF website (2012), http://www.massif-project.eu/
4. AlienValult: AlienVault Unified SIEM. http://www.alienvault.com/ (2010)
5. Araknos: Akab2 (Jul 2012), http://www.araknos.it/en/prodotti/akab2.html
6. ArcSight Inc.: Common event format: Event interoperability standard. http://www.arcsight.com/collateral/CEFstandards.pdf (Aug 2006)
7. Buecker, A., Amado, J., Druker, D., Lorenz, C., Muehlenbrock, F., Tan, R.: IT Security Compliance Management Design Guide with IBM Tivoli Security Information and Event Manager. IBM Redbooks (Jul 2010), ISBN 0-7384-3446-9
8. Coppolino, L., D'Antonio, S., Formicola, V., Romano, L.: Integration of a System for Critical Infrastructure Protection with the OSSIM SIEM Platform: A dam case study. In: Flammini, F., Bologna, S., Vittorini, V. (eds.) SAFECOMP. Lecture Notes in Computer Science, vol. 6894, pp. 199–212. Springer (2011)
9. CS: Prelude SIEM (Jul 2012), http://www.prelude-technologies.com
10. Debar, H., Curry, D., Feinstein, B.: The Intrusion Detection Message Exchange Format (IDMEF). RFC 4765 (Experimental) (Mar 2007)
11. Eichler, J., Rieke, R.: Model-based Situational Security Analysis. In: Proc. of the 6th Int'l Workshop on Models@run.time at the 14th Int'l Conf. on Model Driven Engineering Languages and Systems (MODELS 2011), Wellington, New Zealand, CEUR Workshop Proceedings, vol. 794, pp. 25–36. IEEE Computer Society (2011)
12. Gürgens, S., Ochsenschläger, P., Rudolph, C.: On a formal framework for security properties. Computer Standards & Interfaces 27, 457–466 (2005)
13. Iec, I.: ISO/IEC 27004:2009 - Information technology - Security techniques - Information security management - Measurement. ISOIEC (2009)
14. Innerhofer-Oberperfler, F., Breu, R.: Using an enterprise architecture for it risk management. In: Proc. of the ISSA Conf. from Insight to Foresight (2006)
15. Kotenko, I., et al.: Analytical attack modeling. Tech. Rep. Deliverable D4.3.1, MASSIF Project (2011)
16. Lieberman Software: Common event format configuration guide (Jan 2010)
17. Melik-Merkumians, M., Moser, T., Schatten, A., Zoitl, A., Biffl, S.: Knowledge-based runtime failure detection for industrial automation systems. In: Workshop Models@run.time. pp. 108–119. CEUR (2010)
18. Schiefer, J., Rozsnyai, S., Rauscher, C., Saurer, G.: Event-driven rules for sensing and responding to business situations. In: Int'l Conf. on Distributed event-based systems (DEBS). pp. 198–205 (2007)
19. Verissimo, P., et al.: Massif architecture document. Tech. Rep. Deliverable D2.1.1, MASSIF Project (2011)