International Journal of Security and Its Applications
Vol.9, No.9 (2015), pp.77-94
http://dx.doi.org/10.14257/ijsia.2015.9.9.08

A Case Study on Converged Security with Event Correlation of Physical and Information Security

Koohong Kang[1] and Jinoh Kim[2]

[1]*Dept. of Information and Communications Eng. Seowon University, Cheongju 361-742, Republic of Korea*
[2]*Dept. of Computer Science, Texas A&M University-Commerce, Commerce, Texas 75428, USA*
khkang@seowon.ac.kr, Jinoh.Kim@tamuc.edu

Abstract

Today's security initiatives have encouraged incorporation of physical security and information security into converged security for greater effectiveness and capabilities. However, efforts for converging security have largely limited to the issues of the organizational structure with respect to streamlining processes and abstract frameworks for security management. To go beyond just a buzz word of converged security, it should be necessary to bring significant technical merits from this convergence. In this work, we consider "event correlations" that examine any associations between events coming from the above two distinctive worlds to provide greater capabilities for preventing unauthorized access to high-security computers, as a tangible step towards convergence of security. For this purpose, we introduce our approach using event categorization that maps physical events to a finite number of classes (five) instead of considering event types individually for feasibility, and also show how to define correlation rules with the categories. In addition, we present our prototype system that implements the incorporation of two typical physical security entities: a door/gate access control and a video surveillance system. Our exploration presented in this paper would be beneficial for guiding future development of a diverse range of converged security functions.

Keywords: *Converged security, event correlation, physical security, information security, rule matching*

1. Introduction

After 9/11, security technologies are changing so rapidly that the cycle of introducing new technologies has become smaller than ever less than six months, particularly in the fields of cyber-security, video analytics, the wireless network grid and imaging [1]. In particular, recent advances in IT technology has led to the evolution of physical security technology from simple closed-circuit television (CCTV) to the convergence of physical security, IP-based technology, and software technology, in order to effectively cope with malicious activities including terrorism and cyber-crimes. For example, recent physical access control systems, such as door locks and smart access terminals, provide sophisticated fingerprint recognitions with an internal camera that captures face image logs for extra level of security, in addition to the network-based communications for remote monitoring [2]. Due to these functional integrations of physical and IT technologies in a single device or a single infrastructure, physical and information security boundaries have become increasingly less distinctive.

Back to the early 2000s, many organizations such as the Alliance for Enterprise Security Risk Management (AESRM) [3] and Open Security Exchange (OSE) [4] offered some promising suggestions for integrating converged security programs. However, these early works discussed only conceptual converged programs such as how to integrate these

ISSN: 1738-9976 IJSIA

two departments in terms of the organizational structure, how to collaborate them in terms of the management platform, and how to reduce the management costs. Even the most recent work [5] still remains in the same phase without noticeable progress. This sluggish pace of converged security has made it less attractive, although we are forced to realize this initiative on real fields due to the current technical trends of social fundamental infrastructures such as the supervisory control and data acquisition (SCADA) system [6] and the electric power grid (smart grid) [7]; that is, such infrastructures use a tighter incorporation of the digital communications and computer infrastructure with the existing physical infrastructure, with all the inherent vulnerabilities. For a greater degree of security for the essential infrastructures as well as typical organizational computing environments, it is necessary to provide enhanced defense functions implementing converged security.

Since these information-centric architectures include many core computing servers for web, mail, databases, file, management, and monitoring, they have brought, with all its benefits, another kind of threat: that of people using illicit means to access, invade and attack these computers. To defend the organization against such threats, intrusion detection and/or prevention systems [8] have been widely deployed, and using such security functions have been proved to be effective. Moreover, high-security computers are placed in a safe, isolated district that has physical access controls such as door/gate locks and video surveillance systems, since if any malicious user gets access to any of the computers located in the secure room, the consequence should be devastated. What we consider in this work is to find and utilize possible correlations between physical security events and information security events, for more effective identification of malicious activities that could take place in such a secure computing room. Suppose an example of a root log-in failure. The event itself is not very serious and could be considered as a common mistake. However, if the event happened in the main computing room and the door to the room has been left opened for an unusually long time (due to door struck for example), it can be taken as a serious event because an unauthorized person could have infiltrated the server room physically while the door is left opened. In general, this is how the administrator reacts against the given event. However, human capabilities are often limited, they make mistakes, and they might not be consistent against the identical event, making it hard to consider a large set of possible combinations between events. Moreover, relying on human labors is relatively significantly expensive compared to automated, computerized techniques in general. In this work, we consider automated correlations between physical and information security events without human intervention, and develop a prototype system with two typical physical security entities, a door/gate access control system and a video surveillance system, in addition to the information security function.

A non-trivial challenge we observed along the prototyping is a significant number of possible combinations between events from physical devices and ones for information systems, making it impractical for correlating them. In particular, as physical security devices (PSDs) become intelligent, the devices generate more diverse types of events than ever. Thus, any naïve method considering the entire set of combinations should be neither scalable nor practical. In this paper, we introduce our approach using event categorization that maps physical security events (PSEs) to a finite number of classes (five) instead of considering event types individually for scalability. We also show how to define correlation rules with the categories by constructing a simple 2-dimensional table that specifies association rules between information security events (ISEs) and PSE categories. Additionally, we present automatic rule generation from the table to minimize human labors and potential errors.

The paper is organized as follows. We briefly discuss the early works of converged security and the well-known event correlation engines for information security in Section 2. Section 3 describes the security events and features discussed throughout this paper.

Then we introduce our rule-based deterministic event correlation engine that provides scalable correlation in Section 4. In this section, we also discuss automatic rule generation from the correlation table. Section 5 introduces our prototype system and preliminary evaluation results. Finally we conclude the paper in Section 6.

2. Related Works

2.1. Converged Security

Today, security can mean either physical security, as in physical access control to protect physical assets within the corporate, or information security (also known as logical security), as in virus detection or blocking of unauthorized access to protect information and intangible assets of the corporate computing servers. The departments that manage these two types of security are usually entirely separate (e.g., police department and IT department), and do not even collaborate. Now, the convergence of the IP network and the migration of legacy sensors and appliances to TCP/IP have helped drive to be integrated both physical and logical security technologies. As physical and information security boundaries have become increasingly blurry, many organizations are experiencing challenges with how to effectively and efficiently manage security within the corporate.

OSE developed a series of specifications for promoting interoperability in security management [4]. Specifically, Physical Security Bridge to IT Security (PHYSBITS) is a vendor-neutral approach for enabling collaboration between physical and IT security to support overall enterprise risk management needs. The PHYSBITS framework provides a modular approach to enable the integration of security management and presents a data model specification that describes a method to map physical security application data into an IT security framework. Mehdizadeh [9] addressed the benefits and value of the convergence of logical and physical security systems by using a common token such as a smart card. The author also presented some issues from architecture to pricing to user case scenarios. Other studies [3, 5] also discussed various frameworks in terms of "*conditions for convergence*", and noted that a business-focused framework will allow security elements to become part of the strategic landscape of the enterprise, thus legitimizing security as a critical element of the enterprise. Although these early works have put impassioned rhetorical expressions on converged security, we can hardly find any good solutions or products of converged security in market now. This is because most of the works have been trying to manage operational risk and to improve the overall risk profile; that is, all of these models focus on abstract security management platforms for convergence. Now, to achieve technical benefits from the convergence of security, there should be the following two considerations. Firstly, the events of physical security can be used to achieve the better defense of information security. Secondly, the converged security system should be scalable and implementable on real fields.

2.2. Event Correlation Engine for Information Security

Event correlation technologies have been applied to facilitate business event processing for Business Performance Management (BPM) [10] and to tackle a correspondingly greater effort for system managements of the increasing complexity of information technology (IT) systems [11]. Nowadays many researchers working in the field of Internet security consider event correlation as one of essential functions, given a large amount of data collected for network security analysis from dispersed networking resources [12, 13].

Jakobson and Weissman [14] create an alarm correlation model, which is a conceptual interpretation of multiple alarms such that new meanings are assigned to these alarms. The proposed alarm correlation based on the principles of model-based reasoning improves telecommunications network surveillance and fault management. Abad et al.

[15] show that correlating log information is useful for improving both misuse detection and anomaly detection. This is because an intrusion typically leaves multiple signs of its presence. The general idea in that work is to take advantage of attack traces by correlating information found in multiple heterogeneous logs, where they used data mining techniques to analyze and to filter out important data from different logs. Jiang et al. [12] also implemented a system that scans and correlates distributed events according to the signatures from users' high-level description of dynamic processes. While they mention it is important to integrate spatial and temporal event correlation together for intrusion detection, the proposed system only supports temporal-based event correlation.

Visualization also helps analyze event correlations by providing visual aids. Historically, visualization has been applied extensively to network monitoring and analysis, primarily for monitoring network health and performance. Initial visualization techniques for intrusion detection systems (IDSs) focused on simple scales and color representations to indicate state or level of threat. Recently, substantial efforts have been made to expand the visualization capabilities to provide the correlation relationships among security events over the last few years. Yin et al. [16] focus on visualization of network flows into and out of a network. The visualization of the data enhances a network administrator's ability to detect intrusions on the network by improving his or her situational awareness of current and recent network events. Livnat et al. [17] propose a visualization paradigm for the correlation of various network and host based alerts from disparate IDS logs. The paradigm is based on the observation that every network alert must possess three fundamental attributes i.e., the What, When, and Where, which in turn provide a consistent basis for correlation.

Because attacks against the information platform may involve multiple steps, evidence of attacks is typically distributed over the physical platform; that is, the event correlation between two different security realms must be very promising to enhance the security level of the converged security. However, there has been little discussion with respect to correlating events between the physical and information platforms, which is explored in this work.

3. Security Events

3.1. Physical Security Devices (PSDs)

The most popular devices for physical security are access controllers for entrance and the video cameras for surveillance. Recently, access controllers do not have space limitations by utilizing a convenient TCP/IP-based network [2, 18]. In other words, the entry and exit tracks and the status at the entry door can be easily monitored and tracked on a real-time basis with a computer in office, and a central security control center can manage the whole events collected from these devices [2]. In the meantime, computerized vision has been recognized as an attractive technology since watching and monitoring video streams by human beings should not be effective. The primary technology for video monitoring is with intelligent video systems (IVSs) which range from rudimentary motion detection to the state of the art intelligent video for multi-task complex scenarios detection [18].

These physical security devices (PSDs) generate *events* to indicate any unexpected incidents. As PSDs are getting more intelligent, one complication for dealing with such events is the fact that the types of events are also increasing significantly. It is also critical to event correlation that we are trying to address in this paper, as a space of possible combinations between physical security events and information security events should dramatically increase. For this reason, we consider categorizing events from PSDs. In the following section, we discuss event categorization more in detail.

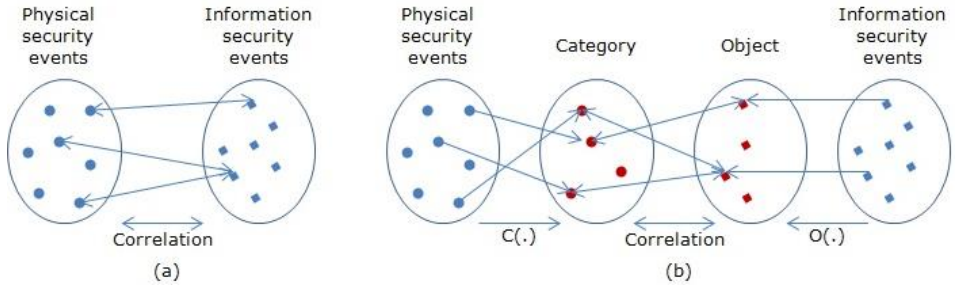

Figure 1. Event correlation between physical and information securities; (a) Direct mapping; (b) Indirect mapping using our Object&Category model (proposed approach)

3.2. Physical Security Event Features

In this paper, we consider event correlation between physical and information securities. We note that multiple PSEs can be possibly co-occurred when an ISE is occurred from the high-security computer. Suppose that we have N_p PSEs and N_i ISEs, the maximum number of possible combinations can be $N_i \times \left(2^{N_p} - 1\right)$ (see Figure 1 (a)). As getting more intelligent of PSDs as mentioned before, this number is not ease figure to propose a correlation engine scalable and implementable systematically. In order to reduce this figure, PSEs can be simply categorized according to their features (see Figure 1 (b)).

Table 1 illustrates how we manage PSEs for event correlation with a subset of events frequently observed in the above two PSDs (i.e., access control devices and intelligent video devices). Even if there could be many more events defined at the product manuals, we consider the events shown in the table for ease of exposition. However, as we will discuss later, our method presented in this paper is extensible with a concept of categorization, and any other event can be classified into the given set of categories in a straightforward manner.

Table 1. Physical security events for access control and intelligent video devices (Event Time Window (ETW), Category: Intrusion (I), Suspicion (S), System Overhaul (O), System Error (E), and Validation (V))

Device	Event	ETW	Cat.	Description
Access control system (ACS)	AC1	Semi-forced	V	Door is opened by a valid user
	AC2	Programmed	S	Authentication fails several times
	AC3	Forced	I	Door in opened by force
	AC4	Programmed	S	Door is held open too long
	AC5	Self	O	Alarm zone is disarmed
	AC6	Self	E	Network connection is closed
	AC7	Forced/self	S	Case is opened
	AC8	Self	O	System off
Intelligent video system (IVS)	IV1	Semi-forced	V	Valid object (or face or badge) for entrance
	IV2	Forced	I	Intrusion detection (perimeter monitoring)
	IV3	Forced	I	Detects disguised objects (bagged detection)
	IV4	Programmed	S	Person loitering in restricted zone
	IV5	Forced/self	E	Low signal quality
	IV6	Self	E	Network connection is closed
	IV7	Self	O	System off

As in Table 1, we classify PSEs into five categories based on event properties - Intrusion (I), Suspicion (S), Validation (V), System error (E), and System overhaul (O), the descriptions of which are as follows:

- Validation (V): Any event related to the valid access of ACS or the valid recognition of IVS is categorized as "V"
- Intrusion (I): The main purpose of PSDs is to detect unauthorized accesses, and any event related to this is categorized as "I".
- Suspicion (S): Any event occurred due to an illegal or misuse access of PSDs is classified into "S".
- System error (E): Any event telling that PSDs are working improperly or disconnected is categorized as "E".
- System overhaul (O): Any event in "O" takes place by administrator's purposes such as initialization or maintenance for security devices.

In detail, for any event in "E", it should be monitored carefully because someone maybe caused that problem on purpose and it is hard to expect the associated PSD functions correctly afterward. Unlike this, we assume that any event categorized as "O" is under control by the administrator because it was intended for maintenance. Thus it does not matter whether the associated PSD functions correctly or not in case of any event in "O". It would definitely be possible to consider a fine-grained categorization model with a greater number of categories, e.g., based on the degree of suspiciousness of events, but we use a model with five classes in this paper and show it is sufficient with less computational complexity than a fine-grained model.

Table 1 has also a column for *Event Time Window* (ETW). A security event is raised when a corresponding activity takes place. In general, an event first occurs, and then it is followed by another one when the security problem is cleared (i.e., up/down events). We refer to this time interval between "up" and "down" events as ETW of the corresponding event. We define four types of ETW as follows: (i) Self ETW, (ii) Forced ETW, (iii) Programmed ETW, and (iv) Semi-forced ETW.

- *Self* ETW: For Self ETW, an explicit "down" event is expected to be followed. For example, AC5, AC6, and AC8 in Table 1 are the typical instances of this class of ETW.
- *Forced* ETW: In contrast, an event can be effective until the security administrator takes some proper actions to clear the problem. We define it as forced ETW, and AC3 and IV3 belong to this class.
- *Programmed* ETW: In this type, we intentionally extend the ETW of some events in category "S" because we want to prolong their effect enough to count their latent period. For example, if we get an event that a person is loitering in the computing room (IV4 in Table 1), and later, someone fails several times to login a computer, there could be a high probability that the one who tried to login the main system is *that* person; that is, we need to inject notion of beware for some period after the event IV4 is happened. Hence the effect of event AC2, AC4, and IV4 in the table might last some pre-determined (programmed) time interval.
- *Semi-forced* ETW: Finally, AC1 or IV1 has no explicit "down" event corresponding to the "up" event for entrance by user's identifications with the smart card, fingerprint, or image recognitions. This is because access control systems often require the user to provide his or her access token only when entering the secure zone but not for his or her departure. Semi-forced EWT assumes that the user explicitly makes the corresponding "down" event when he or she leaves the place to clear the previous "up" event.

Note that AC7 and IV5 can be either self ETW or forced ETW, since the window can be closed either by a follow-up event or the system administrator.

3.3. Event Logs from High-security Computers

It is essential to track any unauthorized access to high-security computers, such as Internet application servers, since it is often one of the most important assets in an organization. There are several ways to collect information regarding intrusions, and we simply assume typical methods for collecting such information, e.g., by using SNMP or by regularly reading system logs. For example, Zeng and Wang [19] expanded MIB resources by defining MIB objects to monitor the resources of server. In addition, there is a lot of information placed in the event logs of servers (e.g., /var/log) that can be regularly imported to the correlation system. Abad et al. [15] show that correlating log information is useful for improving both misuse detection and anomaly detection. Table 2 shows a set of information security events considered in this paper. These events are selected from the event logs according to our objectives - preventing unauthorized access to the computer; this is, as shown in Figure 1 (b), the number of ISEs to be correlated with the PSEs must be reduced.

Table 2. Example of Information Security Events of High-security Computer

Event	Description	I/S	Regular expression
SV1	Local logon success	S	LOGON:LOCAL:SUCCESS
SV2	Local logon fail	I	LOGON:LOCAL:FAIL
SV3	Local logon fails several times	I	LOGON:LOCAL:FAIL:MUL
SV4	Remote logon success	S	LOGON:REMOT:SUCCESS
SV5	Remote logon fail	I	LOGON:REMOT:FAIL
SV6	Network traffic overload	I	OL:NET
SV7	CPU usage overload	I	OL:CPU
SV8	Disk usage overload	I	OL:DISK
SV9	Software configuration change	S	CC:SW
SV10	Hardware configuration change	S	CC:HW
SV11	Logout	S	LOGOUT

4. Two-tier Event Correlation Engine

Scalability is one of important challenges in designing an event correlation engine since the number of security events can be very huge. For example, a university campus may have tens of buildings and tens of thousands computers with a significant number of physical security devices and computing servers that should be well protected (e.g., web servers, file servers, grading servers, etc). In addition, events needed to be analyzed can be generated from dispersed sources. The smart grid and SCADA technology are also in this tough situation. For scalability and distribution of event sources, we consider a two-tier model that consists of the pre-processed step on the event observers of security devices and the rule-based deterministic event correlation on the second step.

Figure 2 shows an overview of the proposed rule-based deterministic event correlation system. In the figure, the PSDs on top and the high-security computers at bottom raise security events and send them to the converged security management (CSM) server in the middle. The CSM server determines event correlations by rule matching, and then generates alerts or warning messages when necessary.

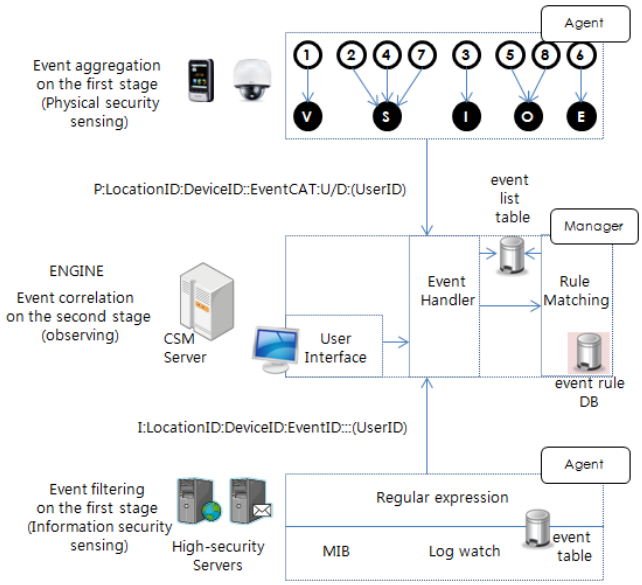

Figure 2. Two-tier Correlation Engine

4.1. Event Sensing on the First Stage

PSEs are sensed in the first stage of correlation, and then aggregated information is sent to the core engine (the top side in Figure 2). As mentioned, many PSDs today are networked with computers so that they can be managed remotely through some specialized applications (e.g., management or analyzer programs). These programs analyze received events from PSDs and respond by generating some proper messages such as alert or warning. Since security companies rely on proprietary event formats and transmission protocols for their PSDs, we assume an intermediate server (ACS&IVS server) that collects sensed events from PSDs and forwards them to the CSM server after converting to the standard representation. Figure 3 shows an interaction between PSDs and central servers for correlation analysis.

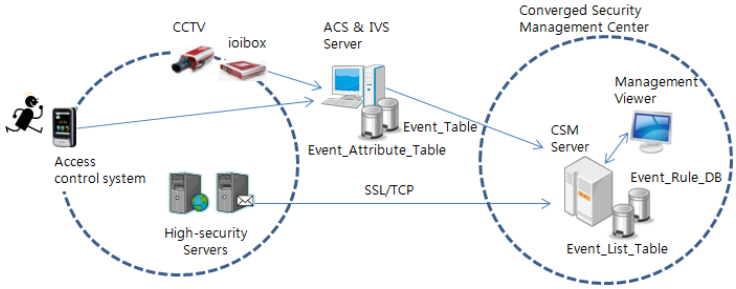

Figure 3. Experimental set-up for converged security management showing physical and information security devices (CSM: Converged Security Management, ACS: Access Control System, and IVS: Intelligent Video System)

An example of the standard format we use to transmit security events from the ACS&IVS server to the CMS server is as follows:

1: LocationID:DeviceID:EventCategory:U/D:(UserID)

Here, the first bit flag "1" indicates that this is a PSE (that is, "0" for an ISE), LocationID indicates the position of the place where the physical security device covers, and DeviceID is a unique ID of the device that raised the event. EventCategory is one of the

five security categories explained in Section 3, U/D (Up/Down) is a bit flag to determine ETW, and UserID is optional and provides the user information for that event. In order to translate the proprietary format of an event into the standard one, the ACS&IVS server keeps a table Event_Attribute_Table that contains every event feature shown in Table 1. Figure 4 gives an example of Event_Attribute_Table. The value of timer in Event_Attribute_Table is the pre-determined time duration of event whose ETW belongs to "Programmed ETW".

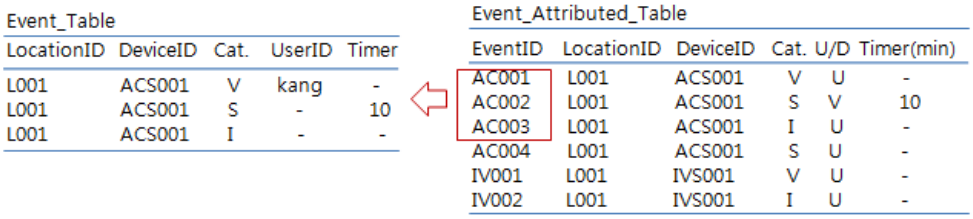

Figure 4. An Example of Usage Event_Attribute_Table and Event_Table

Aggregation is the operation of collecting (aggregating) multiple events in a relatively small number of events [13]. The ACS&IVS server performs aggregation to reduce the amount of events forwarded to the CSM server as well as the format translation. By doing so, it is possible to expect to save the CSM server resources, and as a result, it can be more scalable. For this operation, the ACS&IVS server should keep a table Event_Table which maintains PSEs currently active. Event_Table is simply an array of events, each of which contains: (i) LocationID, (ii) DeviceID, (iii) EventCategory, and optionally (iv) UserID and (v) Timer. Figure 4 shows a snapshot of Event_Attribute_Table and Event_Table, where three events (AC001, AC002, and AC003) are currently active.

Now, let us discuss how aggregation takes place in detail. Basically, the ACS&IVS server aggregates multiple events that have the identical EventCategory into a single event if their ETWs are overlapped. Further aggregation takes place through Event_Table, and there can be two options based on a field used for aggregation: (i) DeviceID for light-aggregation and (ii) LocationID for heavy-aggregation. Suppose a server that has been set to the heavy-aggregation option. If the ACS&IVS server receives an event IV002, then the server will not forward the event to the CSM server because one event with the same PSE location and category as the arriving event IV002 is still pending actively in Event_Table shown in Figure 4; that is, other PDSs located in the same zone (i.e., with the identical LocationID) do not have to issue redundant events. For a server with the light-aggregation option, on the other hand, the received event IV002 will be forwarded to the CSM server because the DeviceID of IV002 is different from the one of AC003. That is, in the case of the light-aggregation, we do not consider LocationID for aggregation. An entry in Event_Table is removed when the corresponding explicit or implicit "down" event occurs. The following pseudo code illustrates the aggregation operation when an "up" event takes place.

Program 1. Aggregation_UpEvent

```
while(true) {
    Sleep until an event is received from physical security device;
    if (light aggregation)
        Check Event_Table for an entry with the same DeviceID and
        the same EventCategory corresponding to the incoming event;
    else // means heavy aggregation
        Check Event_Table for an entry with the same LocationID and
        the same EventCategory corresponding to the incoming event;
    if (entry is not found) {
        if (the event is Programmed ETW)
            Set the Time-out value; // get the value from Event_Attribute_Table
        Create an entry with LocationID, DeviceID, EventCategory, and Time-out;
        Enqueue the event into Event_Table;
        Forward the event to CSM server;
    } // end if
    else // entry is found
        Do noting; // do aggregation operation
} // end while
```

Table 2 shows a set of information security events considered in this paper. Since there can be a variety of computing platforms, we establish a standard format as did for diverse PSDs.

0:LocationID:DeviceID:EventID:I/S:U/D:(UserID)

The second and third attributes in the above message (LocationID and DeviceID) have almost the same meaning as the PSE format except that DeviceID should point to an information security entity. I/S (Immediate/Store) is a bit flag that defines whether the event is stored to Event_List_Table in the second stage (will be discussed in detail in Section 4.3 and 4.4). We use a regular expression (e.g., LOGON:LOCAL) for EventID for rule matching, as will be further discussed in Section 4.3. Again, U/D (Up/Down) is a bit flag to determine ETW, and UserID is optional and provides the user information for the event.

4.2. Event Correlation

Let us define *inclusive* for any two events that should come together, and define *exclusive* for any two events that must not occur side by side. If any two events are related each other, then there exist the following four relationships between the events:

(i) time-ordered inclusive relationship, e.g., an ISE must be followed by a PSE or vice versa.
(ii) time-ordered exclusive relationship, e.g., an ISE must not be followed by a PSE or vise versa.
(iii) inclusive relationship regardless of time order, e.g., an ISE and a PSE must be observed together but the time order does not matter.
(iv) exclusive relationship regardless of time order, e.g., an ISE and a PSE must not be accompanied regardless of time order.

Based on this, we can define relationships for two related events. Table 3 shows an example of event correlation table to represent relationships between the access control system (for physical security) and the high-security computer (for information security). In the table, ◎ stands for inclusive, whereas ⊙ indicates exclusive. Thus, ◎ (P-I) is an inclusive relationship when an ISE follows PSE. In addition, ".a" and ".w" defines the message level: *.a: *alert* and *.w: *warning*, based on the associated severity. From the table, we can see that the number of combinations for correlating PSEs with ISEs is quite manageable by employing event categories.

Table 3. Example of an event correlation table for relationships between the access control system and the server computer (◎ (or ⊙)(P-I): Inclusive (or Exclusive) relationship when ISE follows PSE, ◎ (or ⊙)(I-P): Inclusive (or Exclusive) relationship when ISE is followed by PSE, and ◎ (or ⊙): Inclusive (or Exclusive) relationship, *.a: message level "alert", *.w: message level "warning")

PSE		ISE									
AC#	Cat.	SV1	SV2	SV3	SV4	SV5	SV6	SV7	SV8	SV9	SV10
1	V	◎(P-I).a	◎(P-I).a	◎(P-I).a	⊙.a	⊙.a	-	-	-	-	◎(P-I).a
2,4,7	S	-	-	⊙(P-I).w	-	-	-	-	-	⊙(P-I).w	⊙(P-I).w
3	I	-	⊙(P-I).a	⊙(P-I).a	-	-	⊙(P-I).a	⊙(P-I).a	⊙(P-I).a	⊙(P-I).a	⊙(P-I).a
5,8	O	-	-	⊙(I-P).w	-	-	-	-	-		⊙(I-P).w
6	E	-	-	⊙.w	-	-	-	-	-	⊙.w	⊙.w

In the table, the server events SV1-SV3 (local logon) or SV10 (hardware configuration change) must follow category "V" PSE (i.e., a valid user enters the area), and thus, it is an inclusive condition. In contrast, SV4 and SV5 (remote logon through a VPN connection) must not be co-existed with the corresponding user entrance of the local zone (i.e., mutual exclusive condition). Even if these inclusive and exclusive conditions are very direct event correlations, we can also consider the event correlation implicitly. For instance, as shown in the table, the server event SV3 (local logon fails several times) must not follow category "S" or "I", which means someone may intrude the area and try to login that system. The ISE SV3 also must not be followed by the PSE category "O" because the security administrator has to wait to complete some types of ISEs with the "down" events before he/she wants to shutdown (or stop temporally) the PSDs. Lastly, when the PSD experiences some faults (category "E") like network failures or low signal quality, we do not allow simultaneous occurrences of some ISEs since the corresponding ISEs are so important that they always actively covered by the PSDs. For example, we want to get a warning message if SV3, SV9 and SV10 are triggered because the PSDs suffer from some faults. Although we do not explain every column and row in the table, it would be straightforward for the security administrator to complete whole mapping in the table based on their own management policies.

4.3. Automatic Correlation Rule Generation

Jakobson and Weissman [14] present a correlation rule expression by extending the principles of model-based reasoning, originally used for the modeling of intelligent systems. We also extend the correlation rule expression obtained in [14] for our specific purposes. Correlation rules are used to recognize certain important activities from collected events. Each rule consists of two parts - conditional and action parts like IF-

THEN clause in C language. The conditional part is a Boolean pattern built upon primary and secondary terms. The first entry (PSE Cat="V" and SV1) in Table 3 can be presented by the following rule:

IF (I:LOGON:LOCAL:SUCCESS) // primary term

IF !(a PSE with PSC= " V" , and the same UserID and LocationID) //2nd term

THEN " alert"

where the primary term is the current incoming security event from the first stages, and the secondary term is the security event previously arrived and not terminated. Hence, in order to correlate the events, we need to store the incoming events if they are candidates for the secondary terms. In case of PSEs, every "up" event is stored, and then removed when its corresponding "down" event is happened. In case of ISEs, we only store the events which can be the secondary term of the correlation rules; that is, the ISE has some relationships such as ⊙ (I-P), ◎ (I-P), ⊙, or ◎ with any PSE. The first stage at the high-security computers marks the I/S bit of each ISE to notify *candidate the secondary term*. As shown in Table 2, each ISE has its I/S bit information which can be determined from the event correlation table shown in Table 3.

Although our reduction technique using event categorization enables us to dramatically reduce the number of correlations in the system, writing rules manually based on a given format would be difficult and even erroneous. For this reason, we develop a pseudo code for automatic rule creation from a given event correlation table, as follows:

Program 2. Auto_Rule_Generation

```
/* Given an relationship R and a message level L between ISE e and Physical
security category c from the correlation table */
primary_term = e; secondary_term = c; condition = null;
if (R == ◎ (I-P) or R == ⊙ (I-P)) {
   primary_term = c; secondary_term = e;}
if (R == ◎ or R == ◎ (P-I) or R == ◎ (I-P))
   condition = NOT;
if (R == ◎ or R == ⊙) {
   print.out.rule (
      IF (secondary_term)
         IF (condition) (primary_term)
      THEN
         L;)
} // create two rules for each time-order independent relationship
print.out.rule (
   IF (primary_term)
      IF (condition) (secondary_term)
   THEN
      L;)
end.
```

We store the generated correlation rules into the event rule DB in the second stage shown in Figure 2.

In order to reduce the size of event rule DB and the rules inside the DB, we can also compact the generated rule sets by using the property of regular expression form of ISE. Figure 5 shows a rooted tree on a set of 8 nodes with root LOGON. As explained earlier, a rule expression LOGON:LOCAL matches up the first three ISEs (SV1-SV3) in Table 2 . That is, when all vertices within the sub-tree rooted at LOCAL in Figure 5 are registered in the rule set, then they are reduced into a single rule.

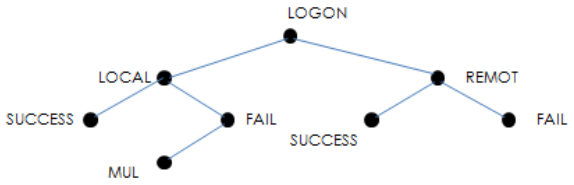

Figure 5. A rooted tree. The vertices of tree are mapped to the fields of regular expressions SV1 - SV5 in Table 2

4.4. Event Correlation Operation on the Second Stage

We so far discussed how to create an event correlation table to combine PSEs and ISEs and how to obtain corresponding rules from the correlation table automatically. We finally discuss what happens in the CSM server upon the reception of any event. As shown in Figure 2, there are several components including Event_Handler and Rule_Matching.

On arriving a PSE at the second stage (that is, in the CSM server) Event_Handler registers the event or deletes the corresponding event to/from the table Event_List_Table based on the U/D flag bit. Hence Event_List_Table always keeps up-to-date PSEs. When Event_Handler receives an ISE from a high-security computer, it performs exactly the same operation as receiving a PSE if the I/S bit flag is “S” (i.e., “Store”). However, if the I/S bit flag is “I” (i.e., “Immediate”), the event will not be stored in Event_List_Table because it never be the secondary term of the conditional part of correlation rules. Finally, as shown in Figure 2, Event_Handler forwards every received events to the Rule_Matching. As discussed in Section 4.3, there are two times matchings for the primary and secondary terms of the correlation rules. First, Rule_Matching selects the primary term from the event rule DB with the current event received. Second, if Rule_Matching finds any corresponding primary term, it retrieves the secondary term from the DB, and then search the event from the Event_List_Table. Finally, Rule_Matching generates the security message when it successfully matches two times.

5. Prototyping and Preliminary Evaluation

We developed a prototype system for converged security event correlation with a set of physical security devices and information security system. In addition, we reviewed a set of attack scenarios to evaluate how well our prototype system works.

As shown in Figure 6, we connected two Suprema XPass access controllers [2] (one for entrance and the other for exit) and an intelligent video analyzer ioImage

TRK100d [18] through fast Ethernet connections to our local network. We deployed these physical security devices to protect a computing room where a CentOS desktop computer is running. In order to ease manipulation of the doorlock according to the testing scenarios, we used a simple circuit for door sensor and relayed to emulate the doorlock that should be controlled by the XPass for entrance. We implemented three different agent programs for two different physical security devices for the first stage and one desktop computer of our experimental set-up, and installed the agent programs on the ACS&IVS server and the CentOS desktop computer respectively. We also implemented a manager program for the second stage and installed it on the CSM server. Finally we performed the predefined attack scenarios.

Figure 6. Experimental set-up for converged security management showing physical and information security devices (CSM: Converged Security Management, ACS: Access Control System, and IVS: Intelligent Video System)

Figure 7 illustrates the test scenarios we used that contain all types of physical security categories defined in this paper: categories "V" and "S" generated from XPass are tested in Figure 7 (a), and categories "I", "E" and "O" generated from ioImage are tested in Figure 7 (b). As shown in Figure 7, we also generated ISEs on the protected server such as "log on success or fail" by local and remote users and "hardware configuration change" by USB memory stick insertion. Figure 8 shows the event rule DB on the CSM server, which is generated based on the correlation table in Table 3. From the figure, we note that the highlighted entry LOGON:LOCAL shows an example of compacted rules discussed in Section 4.

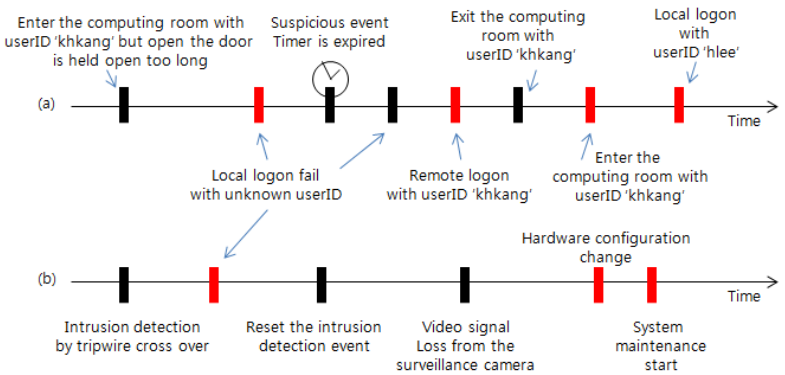

Figure 7. Test scenarios. (a) Access controllers (XPass) with a protected server; (b) Intelligent video analyzer (ioImage) with a protected server

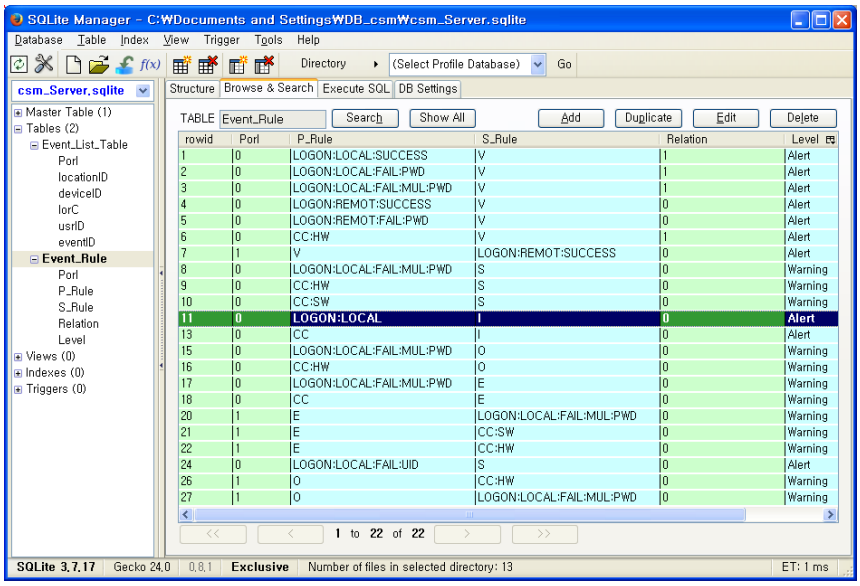

Figure 8. Event Rule Database Implemented with SQLite DBMS

Figure 9 shows the security events from agents during the testing, where each event is represented as the standard format discussed in Section 4. For example, the first action in Figure 7 (a) invokes two PSEs and sends them to the CSM server as follows,

Event Time : Sun Oct 13 15:12:05 2013

P=1 Location=0 Device=53788 Cat OR IS=V UD=U UserID=khkang INF=

Event Time : Sun Oct 13 15:12:31 2013

P=1 Location=0 Device=53788 Cat OR IS=S UD=U UserID=khkang INF=

Here, the second event follows the first one in a few seconds because of ignoring to close the door by a valid user.

```
CSM Server is running .... IP =[redacted]        port no =[redacted]
Event Time : Sun Oct 13 15:12:05 2013
P=1          Location=0     Device=53788    Cat OR IS=V   UD=U   UserID=khkang   INF=
Event Time : Sun Oct 13 15:12:31 2013
P=1          Location=0     Device=53788    Cat OR IS=S   UD=U   UserID=khkang   INF=
Event Time : Sun Oct 13 15:13:35 2013
P=0          Location=0     Device=200      Cat OR IS=I   UD=U   UserID=         INF=LOGON:LOCAL:FAIL:UID
Event Time : Sun Oct 13 15:14:34 2013
P=1          Location=0     Device=53788    Cat OR IS=S   UD=D   UserID=khkang   INF=
Event Time : Sun Oct 13 15:16:13 2013
P=0          Location=0     Device=200      Cat OR IS=I   UD=U   UserID=         INF=LOGON:LOCAL:FAIL:UID
Event Time : Sun Oct 13 15:17:18 2013
P=0          Location=0     Device=200      Cat OR IS=S   UD=U   UserID=khkang   INF=LOGON:REMOT:SUCCESS
Event Time : Sun Oct 13 15:18:13 2013
P=1          Location=0     Device=55711    Cat OR IS=V   UD=D   UserID=khkang   INF=
Event Time : Sun Oct 13 15:19:07 2013
P=1          Location=0     Device=53788    Cat OR IS=V   UD=U   UserID=khkang   INF=
Event Time : Sun Oct 13 15:20:21 2013
P=0          Location=0     Device=200      Cat OR IS=S   UD=U   UserID=hlee     INF=LOGON:LOCAL:SUCCESS
Event Time : Sun Oct 13 15:20:34 2013
P=0          Location=0     Device=200      Cat OR IS=S   UD=D   UserID=hlee     INF=LOGOUT:LOCAL
Event Time : Sun Oct 13 15:22:27 2013
P=1          Location=0     Device=500      Cat OR IS=I   UD=U   UserID=         INF=INT:TRIP:FENCE
Event Time : Sun Oct 13 15:23:32 2013
P=0          Location=0     Device=200      Cat OR IS=I   UD=U   UserID=         INF=LOGON:LOCAL:FAIL:UID
Event Time : Sun Oct 13 15:24:33 2013
P=1          Location=0     Device=500      Cat OR IS=I   UD=D   UserID=         INF=
Event Time : Sun Oct 13 15:25:47 2013
P=1          Location=0     Device=500      Cat OR IS=E   UD=U   UserID=         INF=LOSS:SIGNALNCE
Event Time : Sun Oct 13 15:26:59 2013
P=0          Location=0     Device=200      Cat OR IS=S   UD=U   UserID=         INF=CC:HW
Event Time : Sun Oct 13 15:27:37 2013
P=1          Location=0     Device=500      Cat OR IS=O   UD=U   UserID=         INF=
```

Figure 9. Security event sequences from agents during the testing . Device = 53788 (XPass for entrance), 55711 (XPass for exit), 200 (protected server), and 500 (ioImage)

Figure 10 shows a list of alarms generated throughout the testing. One example alarm among the list shows the following:

Event Time : Sun Oct 13 15:13:35 2013

[ALARM = Alert] [EXCLUSIVE CONDITION = LOGON:LOCAL:FAIL:UID] with

[EXISTING = S]

From this alarm, we can see that it consists of two parts: the first alert message with four fields including time, message level, correlation relationship and current event (primary rule pattern), and conditioned event (secondary rule pattern). The alarm indicates that the event "local logon fail with unknown user ID" detected by the protected server caused the CSM server to generate the alert message because the ISE cannot be coexisted with the "S" condition, and this is exactly what we expected from the scenario.

Although the testing was conducted in a controlled manner, the preliminary evaluation results we collected confirm the validity of our ideas. This suggests that our proposed system could provide a starting point to correlate physical security events with security event logs from high-security computers, and would also give an initial initiative for the converged security platforms.

Figure 10. Generated Alarm Events from the Testing

6. Conclusions

Despite the significant benefits from the convergence of physical security and information security, previous studies have been largely limited to integration of functions. In this work, we take a step to move forward the convergence by considering correlations between events coming from the above two distinctive realms, in order to recognize any important activity in a protected region. To this end, we designed a two-stage model that consists of event aggregation and filtering (first stage) and event correlation (second stage). By decoupling the two stages, the proposed model provides greater scalability which is one of the important requirements in the field. Another crucial challenge for scalability is a large number of possible combinations between physical and information security events. To deal with this problem, we applied a category-based approach when combining physical events with information events. We showed that the five categories presented in this paper would be sufficient to classify physical events, reducing the number of combinations dramatically helping scalability.

With the proposed techniques, we showed how we can construct a correlation table, and also presented how automatic rule creation would be achieved to minimize human labors and errors. We implemented core functions and showed the validity of our ideas by running them in a real prototype system. We believe that our exploration of the correlation engine for heterogeneous events would be beneficial for future development of

converged security systems. We plan to optimize our methods by considering a broad set of physical security devices.

Acknowledgments

ETRI Network System Security Team for free use of their Xpass and ioImage. We would also like to thank Jung Chan Nah and Dongho Kang with ETRI for the many valuable discussions regarding the experiments.

References

[1] D. Ritchey, "Proud To Be Security: How Roles Changed After 9/11", Security Magazine, Cover Story, **(2011)** September.
[2] Suprema-editor, "Introduction of Bio Star Lite", Technical Column, **(2011)** June.
[3] The Alliance for Enterprise Security Risk Management (Booz|Allen|Hamilton), "Convergence of Enterprise Security Organizations", **(2005)** November.
[4] Open Security Exchange, "Physical Security Bridge to IT Security PHYSBITS Framework and Data Model", **(2003)** April.
[5] S. M. Rahman and S. E. Donahue, "Convergence of Corporate and Information Security", International Journal of Computer Science and Information Security, vol. 7, no. 1, **(2010)** January, pp. 63-68.
[6] P. Tsang and S. W. Smith, "YASIR: A Low-Latency, High-Integrity Security Retrofit for Legacy SCADA Systems", In Proceedings of IFIP TC 11 23rd International Information Security Conference, Springer, **(2008)** September, pp. 445-459.
[7] H. Khurana, M. Hadley, N. Lu, and D. A. Frincke, "Smart-Grid Security Issues", IEEE Security & Privacy, vol. 8, no. 1, **(2010)** January/February, pp. 81-85.
[8] H. Debar, M. Sacier, and A. Wespi, "A revised taxonomy for intrusion detection systems", Annals of Telecommunications, vol. 55, nos. 7-8, **(2000)** July, pp. 361-378.
[9] Y. Mehdizadeh, "Convergence of Logical and Physical Security", SANS Institute InfoSec Reading Room, **(2003)** October.
[10] S. Chen, J. Jeng, and H. Chang, "Complex Event Processing using Simple Rule-based Event Correlation Engines for Business Performance Management", In Proceedings of the CEC/EEE'06, San Francisco, **(2006)** June, pp. 26-29.
[11] L. Stojanovic, J. Schneider, A. Maedche, S. Libischer, R. Studer, Th. Lumpp, A. Abecker, G. Breiter, and J. Dinger, "The role of ontologies in autonomic computing systems", IBM Systems Journal, vol. 43, no. 3, **(2004)**, pp. 598-616.
[12] G. Jiang and G. Cybenko, "Temporal and Spatial Distributed Event Correlation for Network Security", In Proceedings of the American Control Conference, **(2004)** June, pp. 996-1001.
[13] A. Muller, "Event Correlation Engine", Master's Thesis of Department of Information Technology and Electrical Engineering, Swiss Federal Institute of Technology Zurich, **(2009).**
[14] G. Jakobson and M. D. Weissman, M.D, "Alarm Correlation", IEEE Network, vol. 7, no. 6, **(1993)** November, pp. 52-59.
[15] C. Abad, J. Taylor, C. Dengul, W. Yurcik, Y. Zhou, and K. Rowe, "Log Correlation for Intrusion Detection: A Proof of Concept", In Proceedings of the 19th Annual Computer Security Applications Conference, **(2003)** December, pp. 255-264.
[16] X. Yin, W. Yurcik, M. Treaster, Y. Li, and K. Lakkaraju, "VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness", In Proceedings of the CCS Workshop Visualization and Data Mining for Computer Security, Washington DC, **(2004)** October. pp. 26-34.
[17] Y. Livnat, J. Agutter, S. Moon, R. F. Erbacher, and S. Foresti, "A Visualization Paradigm for Network Intrusion Detection", In Proceedings of the Workshop on Information Assurance and Security, New York, **(2005)** June, pp. 92-99.
[18] DVTEL Inc., "Intelligent Video System Technology", White Paper, **(2006)**
[19] W. Zeng, and Y. Wang, "Design and Implementation of Server Monitoring System Based on SNMP", In Proceedings of the International Joint Conference on Artificial Intelligence, China, **(2009)** April, pp. 680-682.

Authors

Koohong Kang, he received the BS and MS degrees in Electronics Engineering from Kyungpook National University and Chungnam National University, Korea, in 1985 and 1990, respectively, and then Ph.D. degree in Computer Science and Engineering from POSTECH, Korea, in 1998. He is currently a professor in the Department of Information and Communications Engineering at Seowon University. His research interests include computer networks and Internet security. From 1985 to 2000, he was a researcher and a senior researcher at ETRI participating in various research projects in TDX switching system, ATM networks, and network security.

Jinoh Kim, He received his Ph.D. degree in Computer Science from University of Minnesota, Twin Cities. He is currently an Assistant Professor in the Department of Computer Science at Texas A&M University-Commerce. The areas of research interests span from systems and networks, including distributed systems, big-data computing and analytics, and network security. Prior to that, he was a postdoctoral researcher at the Lawrence Berkeley National Laboratory for 2010-2011 and an Assistant Professor of Computer Science at Lock Haven University of Pennsylvania for 2011-2012. From 1991 to 2005, he was a researcher and a senior researcher at ETRI (a national lab in Korea) participating in various research projects in network security and ATM Networks.