Security of Composite Electronic Services

Jarosław Wilk
Military University of Technology / Comparex Poland Sp. z o.o.
Warsaw, Poland
jaroslaw.wilk@wat.edu.pl / jaroslaw.wilk@comparex.pl

ABSTRACT

This paper presents a new approach to the security of composite electronic services. They are defined as a part of integration platforms in the service oriented architecture. New security requirements are presented together with a short explanation why currently available models are not sufficient. Defined security model combines access and execution management with use of the lattice theory to allow automatic integration of electronic services originating from different information systems. Proposed solution is described using mathematical modeling with a short case study example. The direction of future work is indicated together with possible places of use for the author’s model.

KEYWORDS

information security, integration platforms, electronic services, composite services, service oriented architecture, SOA, lattice theory

1 INTRODUCTION

Rapid development of digital information systems in the last decades enforces the use of data from many different sources. It becomes necessary to integrate independent systems, which historically were created as separate elements. Integration platforms are considered to be a solution to integrate systems which provide electronic services. However, this new solution has brought new challenges. One of them is a difficulty in providing information security for complex electronic services consisting of many atomic services (originating from different systems).

Currently available information security management models for integration platforms “are still not fully satisfactory (despite many recommendations and best practice guides)” [1]. There is a need for a new model that will be suitable for the “service oriented architecture (SOA)” [2] and will allow dynamic management of security levels for integrated services.

The article discusses the author's information security model for service oriented systems. It addresses security management for systems like electronic services portals, cloud environments and integration systems based on web services.

The paper is organized as follows: first it defines the service oriented architecture, a composite electronic service and an integration platform (section II). Section III focuses on the importance of security in integration process, pointing differences in security management between SOA based systems and data direct access systems. Section IV presents a mathematical model of defined systems with a proposition of the new security model based on the lattice theory [3]. Section V presents an example case study indicating the usefulness of the proposed solution. In Section VI, the conclusions of the paper are presented.

2 INTEGRATION PLATFORMS

2.1 Service Oriented Architecture

Accurate definition of electronic services and the service oriented architecture is crucial in understanding role of integration platforms in modern information systems.

According to European Union Council Implementing Regulation – “electronically supplied services’ shall include services which are delivered over the Internet or an electronic network and the nature of which renders their supply essentially automated and involving minimal human
intervention, and impossible to ensure in the absence of information technology” [4].

Information systems which are designed these days are mostly focused on electronic services. This approach makes them user friendly, as typical end user is expecting specific services from IT systems. A service consumer is not interested in understanding what elements (servers, operating systems, databases, applications etc.) are involved in the whole process. He wants to achieve intended goal through the implementation of a service which can be supplied electronically.

The service oriented architecture is a reference model for IT systems based on e-services. “SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.” [2]

Summarizing Service Oriented Architecture enforces encapsulation and modularity by providing composite services to service consumers. Those composite services are created from atomic services originating from service components and operational systems.

### 2.2 Composite Electronic Services

Number of e-services is rapidly growing and they are covering different areas of the economic and public life. European Union Council Implementing Regulation [4] is pointing out some of them:

- e-government (public administration services using modern integrated digital techniques),
- e-banking services (access to bank accounts / trading systems using a computer or other electronic device via the Internet),
- e-insurance (services available on the Internet including: purchasing of insurance, contacting the agent, declaring a damage, interactive consulting, calculating the contribution),
- e-education (educational services available through electronic devices including:

  - language courses, vocational and professional courses for employees, studies, private lessons without a teacher),
  - e-culture (distribution, presentation, preservation and (re) use of cultural works using digital techniques).

Consumers are expecting composite services to perform their tasks with a minimum effort. That is why today’s e-services are very complex and originate from many different systems. For example, if a citizen wants to register a company he should use only one composite e-service. However, this electronic public service has to use many information systems from many public administration units like Ministry of Economy, Social Insurance Institution, National Court Register, Ministry of Finance and many more depending on the national law.

Those independent systems can communicate in two different models: bilateral and multilateral.

### 2.3 Bilateral and Multilateral Interoperability Models

A bilateral interoperability model is based on two components communicating directly with each other. It is not allowed to mediate the communication through the third element. This type of solution used for information systems is often called "point to point". In case of a larger number of elements in order to ensure full communication, it is necessary to create and maintain \( n \cdot (n-1) / 2 \) connections (where \( n \) is the number of elements). Each connection is independent and can only support communication between the two systems. In case of one connection line failure, affected elements will lose contact with each other even though they are still physically connected by another network element (another path). Figure 1 below is the full mesh network of connections for only 6 elements (systems).
A multilateral interoperability model is the answer to limitations of a bilateral model for complex systems rapidly growing with new elements. Managing a growing number of connections and security domains has become increasingly expensive and difficult. A multilateral model assumes the existence of a central element, which performs exchange of information between other systems. It reduces the number of connections to n-1 (where n is the number of elements) – adding at least one additional system when compared to the bilateral model.

Main advantages of this interoperability model are [1]:

- Fewer connections than in the bilateral model which ensures transparency and ease of adding new nodes (high adaptability and scalability).
- Thanks to the central point of integration it becomes easier to monitor and control performance of the whole environment.
- It forces data quality control and standardization (starting from the integration development processes).
- It ensures consistency in the data semantics.

Main disadvantages are:

- The difficulty of implementation in the already existing IT environment.
- Low level of reliability - a failure of the central system leads to a complete lack of communication in an integrated environment.
- It can lead to performance problems - the central system may become the bottleneck of the whole solution.

Figure 2 presents the multilateral interoperability model with full network of connections for 6 client systems (A - F) and one integrating system – hub.

In order to eliminate performance and reliability problems, systems are integrated using ‘message bus architecture’. A single integration system – a ‘hub’ is replaced by the ESB (Enterprise Service Bus) mechanism, which provides a high level of reliability and does not significantly affect the performance.

A detailed comparison of bilateral and multilateral interoperability models has been presented in other author’s publication [5].

2.4 Integration Platform – Definition

An integration platform is defined as a “set of interrelated elements, which aims to create a cooperation environment for systems in order to deliver functions requested by the users of these systems” [1].
With the aid of discussed elements of an integration platform, it can be defined more precisely as a set of information systems that cooperate with each other. Cooperation is based on multilateral or mixed (multilateral with some external systems in bilateral) interoperability models. It gives access to multiple (more than one) electronic services provided at the individual request of a customer through a unified point of access (integrating system). However, a hub is not the owner of these services - the originating unit is still responsible for the service and its implementation (a hub does not replace integrated systems). An integration platform generates composite electronic services which encapsulate operational systems below (Service Oriented Architecture).

3 NEW SECURITY REQUIREMENTS

3.1 Service Oriented Security

Switching from data direct access systems to Service Oriented Architecture entails the need to change the security model.

In popular security models used these days access to data is performed through a specific set of elementary operations like reading or writing. The service architecture model adds service between the user and the data. Service processes the data and then transmit it to the user. The data processing can be a simple mechanism just to forward it in a unified format using the specified interface or can be more complex for example by combining it with other data and performing a set of statistical operations.

This means that in the service architecture we also have to protect “a service” as an object additionally to “a data objects” and its “access operations”.

Another important and new aspect is the security of a composite service. According to Gartner report [6] secure data exchange and secure communication management are the basis for aggregate (composite) public services. This enforces key requirements for integration platforms security model. It has to implement two operations:

- assessment of atomic (basic) electronic services originating from different sources in order to determine if integration into a composite service is possible (in terms of security),
- integration of atomic electronic services to one composite service which includes calculation method of new security attributes (level, category, class).

3.2 Existing Solutions

Currently there are many security models available starting from military models like the Bell-LaPadula model [7] and ending on business commercial models like the Brewer-Nash model (also known as the “Chinese Wall” model) [8] but according to author they don’t meet integration platforms needs. Additionally information security requirements are well defined in the ISO 27001:2013 standard [9]. It describes steps which has to be taken like planning, operating, supporting and reviewing an information security management system but not indicating precisely which security model should be used in it. This is both advantage and disadvantage - giving more flexibility to security process stakeholders but making it more difficult to implement this standard. Usually a security model is already embedded in tools or systems available within organization. The problem begins when new architecture like SOA creates new objects and systems like integration platforms with their composite services.

In the previous author’s research [5] five assessment criteria were defined and 17 different security models were analyzed in terms of usefulness for composite electronic services security. Even if analyzed models had integration mechanisms like in the Smith-Winslett model [10] or the Jajodia-Sandhu model [11] they were strictly database models (almost impossible to implement in the service oriented architecture). On the other hand more general model like the Harrison-Ruzzo-Ullman (HRU) model [12] designed for protecting objects in operating systems can be easily converted to SOA environment but it doesn’t have any object security integration mechanisms.
3.3 Lattice Theory

According to the author’s research [5] security models that are based on the lattice theory for example – Denning [13] and Szafrański [8] models have features which make them suitable as a basis for a new integration platforms security model.

Lattices “are ordered sets, for which the condition is satisfied, that for every pair of elements of the set, there is an upper and lower bound.” [15] Lattice can be represented as:

\[(K, \leq, \oplus, \otimes)\] (1)

where:
- \(K\) is a partially ordered set,
- \(\leq\) is a partial order relation,
- \(\oplus\) is the operation of determining upper bound (supremum) of its arguments,
- \(\otimes\) is the operation of determining lower bound (infimum) of its arguments.

The lattice theory was used by Denning to create a secure information flow model [13] represented by:

\[MPD = (K, \leq, \oplus, \otimes, O, \rightarrow)\] (2)

where partially ordered objects ‘set O’ and operation ‘\(\rightarrow\)’ of allowed secure flow of information were added.

The lattice theory was also used by Szafrański [14] in his security model for data protection in distributed database systems. A database security is described with the data security lattice which is created from composition of flow and operation scope lattices. It can be used for distributed systems thanks to integration operation which transforms local security lattices (one for each database) to the ‘security super-lattice’ for the entire integrated environment.

This methodology allows to verify if integrated systems (in this case, distributed databases) can be integrated and allows to determine the security attributes for the entire integrated environment. A similar methodology can be used for integration platforms and composite electronic services.

Additionally to transformations presented by Szafrański [14] there are many new research [16, 17] on hierarchical lattices showing its usefulness not only in security but also in other fields.

A detailed analysis of lattice theory based security models has been presented in the other author’s publication [18].

4 PROPOSITION OF A NEW SECURITY MODEL

4.1 Mathematical Model of an Integration Platform

An integration platform is a set of integrated electronic service systems (meeting the additional requirements specified in Section II) so it is important to define basic characteristics of such a system.

An electronic service - \(e_l\) is the smallest executable resource that processes (using algorithms which are part of e-service) data from data units. An electronic service consists of algorithms which are permanently tied up with it and will not be considered separately. Different combinations of algorithms generate various electronic services. Set of electronic services:

\[E = \{e_1, e_2, ..., e_L\}\] (3)

An entity - \(p_r\) is the customer of electronic services. Set of entities:

\[P = \{p_1, p_2, ..., p_R\}\] (4)

An electronic service is run by an enforcement \(w_n\) triggered by the entity or other electronic service. Set of enforcements:

\[W = \{w_1, w_2, ..., w_N\}\] (5)

An enforcement launches electronic services - function \(g\).
Entities $P$ can launch enforcements within their permissions. Their permissions are defined by function $u$.

$$u : P \rightarrow 2^W$$  

(7)

Enforcements launched by entities form a set of initiating enforcements ($W_I$).

$$W_I \subset W$$  

(8)

Initial services ($E_I$) are registered on an integration platform and are the only services launched by entities (assumptions for the integrated environment).

$$E_I \subset E$$  

(9)

$$\wedge_{w_n \in W_I} g(w_n) \in E_I$$  

(10)

$$\wedge_{w_n \in W} (w_n \not\in W_I \Rightarrow g(w_n) \not\in E_I)$$  

(11)

$$\wedge_{e_l \in E_I} \vee_{w_n \in W_I} g(w_n) = e_l$$  

(12)

Electronic services which can launch another services form a set of developmental services ($E_w$). Each initial service is also a developmental service.

$$E_I \subset E_w \subset E$$  

(13)

Developmental services launch another services creating a services execution tree. Developmental services can’t launch initial services to avoid loops on the integration platform.

$$z : E_w \rightarrow 2^W$$  

(14)

$$\wedge_{e_l \in E_w} (w_n \in z(e_l)) \wedge (g(w_n) \in E_I) \Rightarrow w_n \in W_I$$  

(15)

Finally any electronic service can execute operations ($T$) on data units ($D$).

$$f : T \times D \rightarrow D$$  

(19)

A state of data unit $s_j$ is being changed as a result of the operation $t_m$ on the data unit $d_i$. A state change can be physical (for example: after "save" operation the data unit $d_i$ has a new value) and formal (for example: after "read" operation the data unit $d_i$ has still the same value, but it is in the new state "read" or even "compromised" if this operation was illegal). Set of all states:

$$S = \{s_1, s_2, ..., s_j, ..., s_J\}$$  

(20)

States generator:

$$gst : D \rightarrow 2^S$$  

(21)

$$gst(d_i) = S_i \text{ for } d_i \in D \text{ where } S_i \subset S; S_i \neq \emptyset$$  

(22)

The state of the whole system is any possible subset of all data unit states of the given system. The state of the whole system:

$$S_{system} = S_1 \times S_2 \times ... \times S_i \times ... \times S_I$$  

(23)
There are two possible ways to determine if specific system is still secure:

a) By controlling each system state after every operation. Impossible to implement in real systems because it requires a list of all states combinations with a description whether it is legal or illegal (from the security perspective).

b) By checking parameters of operations which lead to data unit state change. Implemented by functions (restrictions generators) and used in real systems.

The proposed security model is based on the second approach (b). It means that it has to consist of set of parameters and restriction functions. If result of a function is positive, it means that the system state after specific operation will be legal (from the security point of view). This allows to skip data unit states in further deliberations.

An integration platform consists of many electronic services which come from different systems and use different data sources. Figure 3 presents an example of a composite electronic service published in an integration platform environment. The entity (a service consumer) $p_1$ can invoke the composite e-service $e_1$ with the enforcement $w_1$. Selected service will trigger two data access operations $t_1$, $t_2$: saving some information to the data unit $d_1$ and reading from the data unit $d_2$. It will also invoke another e-service $e_2$ with the enforcement $w_2$. The electronic service $e_2$ will trigger one data access operation $t_3$: deleting the data unit $d_3$.

![Figure 3. Example of a composite electronic service.](image)

The composite electronic service is defined as:

$$ u = (E_Z, e_I, g, z) $$

(24)

where:

- $E_Z \subset E$ – electronic services (atomic services creating the composite service)
- $e_I \subset E_I$ with $e_I \subset E_Z$ – the initial service,
- $g : W \rightarrow E$ – the function defining services launched by enforcements,
- $z : E_W \rightarrow 2^W$ - the function defining enforcements launched by developmental services.

Additional assumptions for composite services:

$$ \land_{e \in E_Z} \lor (e^* e \in E_Z) \lor (e^* e \in E_W) \lor \exists_{w^* e \in (e^*)} g(w^*) = e_t $$

(25)

$$ \land_{(e_I \in E_Z)} \lor (e_I \in E_W) \land_{w \in e_I} g(w_{e_I}) \in E_Z $$

(26)

A set of composite services:

$$ U = \{u_1, u_2, ..., u_k, ..., u_K\} $$

(27)

An integration platform can be defined as a set of composite services, data units, operations, entities
and security management elements (a model with the security classification).

\[ \langle U, D, T, P, SM \rangle \] (28)

### 4.2 Security Management System

A integration platform security management system has to check entity (service consumer) parameters to allow or block execution of specified e-service. This means that all operations (data access and enforcements of next e-services) below have to be legal. There are two main problems that are solved by proposed model:

- **Composite e-services can be very complex** (a security model has to support process of triggering services by other services which is not present in currently available data access security models).
- **Elements of composite e-services can originate from different systems** (a security model has to support integration of security parameters from different systems).

The author’s information security management model for integration platforms consists of two elements:

- data access management model - AM,
- e-service execution management model - EM.

Data units \( D \) are described by data protection classes, which are members of set \( K \) for example: public, confidential, secret, top secret. Electronic services are described by categories of an authorized execution, which are members of set \( B \) - for example: universal, special, restricted.

Each entity \( p_r \) from \( P \) set has the following parameters determining its access level:

- a protection class from the \( K \) set - for example: confidential
- a range of an authorized operation from \( T \) set – for example: reading,
- a category of an authorized execution from \( B \) set – for example: restricted.

### 4.3 Access Management Model

AM model determines, which access operations \( (T) \) are allowed on data units \( (D) \) when triggered by specific entity \( (P) \). Controlling functions are based on the protection class and range of authorized operations. AM model:

\[ AM = \langle P, D, K, T, \rho, \tau, HF \rangle \] (29)

where:

- \( P \) – a set of entities,
- \( D \) – a set of data units,
- \( K \) – a set of protection classes,
- \( T \) – a set of operations,
- \( \rho \) – an access relation,
- \( \tau \) – a scope of operations relation,
- \( HF \) – a set of restrictions functions (restrictions generator for the access management model).

An access relation is built on pairs of protection classes: \( \rho \subset K \times K \) and determines the hierarchy of allowed accesses to the data – for example: top secret > secret > confidential > public. A scope of authorized operations relation is built on pairs of operations: \( \tau \subset T \times T \) and defines a hierarchy of operations – for example: update > delete > write > read > search. Both relations meet requirements of the lattice theory, which allows to build protection classes and operations lattices.

An entity is allowed to access a data unit using specific operations if its protection class is greater than or equal to protection class of a data unit and its scope of operations is greater than or equal to the operation being performed.

The restrictions generator for the access management model is defined as a set:

\[ HF = \{H_1, H_2, H_3, H_4, H_5, H_6, H_7\} \] (30)

\( H_1 \) - a function defining protection classes for entities \( (P) \) and data units \( (D) \):

\[ H_1: P \cup D \rightarrow K \] (31)
$H_2$ - a function defining if a data flow between $P$ and $D$ objects is acceptable ($1$ – yes, $0$ – no):

$$H_2: (P \cup D) \times (P \cup D) \rightarrow \{0,1\}$$

$$\land_{r_1, r_2 \in P \cup D} H_2(r_1, r_2) =$$

$$\begin{cases} 1 & \text{if } (H_1(r_1), H_1(r_2)) \in \rho \\ 0 & \text{if } (H_1(r_1), H_1(r_2)) \notin \rho \end{cases}$$

An access relation ($\rho$) is generating a hierarchical lattice from the protection classes set ($K \times K$). Fig. 4 presents an example of a protection classes lattice.

![Protection Classes Lattice](image)

**Figure 4.** An example of a protection classes lattice

Further properties of an access relation and operations allowing integration of different $K$ sets are not discussed in this paper.

$H_3$ - a function defining allowed operations which entity can perform on data units.

$$H_3: P \times D \rightarrow T$$

$H_4$ - a function defining an operation which is performed by an entity on a data unit object during an access request.

$$H_4: P \times D \rightarrow T$$

$H_5$ - a function defining if an operation execution is acceptable ($1$ – yes, $0$ – no):

$$H_5: (P \cup D) \times (P \cup D) \rightarrow \{0,1\}$$

$$\land_{r_1, r_2 \in P \cup D} H_5(r_1, r_2) =$$

$$\begin{cases} 1 & \text{if } (H_4(r_1, r_2), H_3(r_1, r_2)) \in \tau \\ 0 & \text{if } (H_4(r_1, r_2), H_3(r_1, r_2)) \notin \tau \end{cases}$$

A scope of operations relation ($\tau$) is generating a hierarchical lattice from the operations set ($T \times T$). Fig. 5 presents an example of a scope of operations lattice based on Linux permissions ($R$ – read, $W$ – write, $X$ – execute / access directory contents). In the proposed model execution is extracted from operations and applied only to services, so $X$ should be interpreted only as a ‘access directory contents’ operation.

![Scope of Operations Lattice](image)

**Figure 5.** An example of a scope of operations lattice.

Further properties of a scope of operations lattice and different $T$ sets integration mechanisms are not discussed in this paper.

$H_6$ – a function checking if an access request (a flow and an operation) is acceptable:

$$\land_{r_1, r_2 \in P \cup D} H_6(r_1, r_2) = H_2(r_1, r_2) \land H_5(r_1, r_2)$$

$H_7$ – a function checking if all access requests within a composite electronic service ($E$) are acceptable:

$$H_7(E) = \cap_{i \in I} H_6(\alpha_i, \beta_i) \text{ where } \alpha_i, \beta_i \in P \cup D$$

An entity is allowed (according to the AM model) to execute a composite electronic service only if all access requests within this composite electronic service ($E$) are acceptable:

$$H_7(E) = 1$$
4.4 Execution Management Model

EM model determines, which enforcements of electronic services (E) are allowed (which e-services can be executed by the entity (P) and other e-services (E)). EM model:

$$EM = \langle P, E, B, \delta, BF \rangle$$ (41)

where:

- $P$ – a set of entities,
- $E$ – a set of electronic services,
- $B$ – a set of categories of an authorized execution,
- $\delta$ – a relation of an authorized service execution,
- $BF$ – a set of restrictions functions (restrictions generator for the execution management model).

A relation of an authorized service execution is built on a pair of categories of an authorized execution:

$$\delta \subset B \times B$$

and defines execution permissions hierarchy – for example: restricted $>$ special $>$ universal. This relation meets requirements of the lattice theory, which allows to build authorized execution lattices.

An entity may execute an e-service if its ‘category of an authorized execution’ is greater than or equal to the ‘category of e-service’ (mandatory requirement). An e-service can invoke another e-service if its execution category is greater than or equal to the category of triggered e-service (strong optional requirement).

The restrictions generator for the execution management model is defined as a set:

$$BF = \{G_1, G_2, G_3\}$$ (42)

$G_1$ - a function defining categories of an authorized execution for entities (P) and electronic services (E):

$$G_1: P \cup E \rightarrow B$$ (43)

$G_2$ - a function defining if an entity $P$ is allowed to execute a electronic service $E$ (1 – yes, 0 – no):

$$G_2: (P \cup E) \times (P \cup E) \rightarrow \{0, 1\}$$ (44)

$$\bigwedge_{q_1, q_2 \in P \cup E} G_2(q_1, q_2) =$$

$$\begin{cases} 1 & \text{if } (G_1(q_1), G_1(q_2)) \in \delta \\ 0 & \text{if } (G_1(q_1), G_1(q_2)) \notin \delta \end{cases}$$ (45)

An authorized service execution relation ($\delta$) is generating a hierarchical lattice from the categories of an authorized execution set ($B \times B$). An example of an execution categories lattice is presented in Section V on Fig. 7 as a super-lattice.

Further properties of an authorized service execution relation and operations allowing integration of different $B$ sets are not discussed in this paper.

$G_3$ – a function checking if all service executions within a composite electronic service (E) are acceptable:

$$G_3(E) = \bigcap_{i\in I} G_2(\varphi_i, \omega_i) \text{ where } \varphi_i, \omega_i \in P \cup E$$ (46)

An entity is allowed (according to the EM model) to execute a composite electronic service only if all service executions within this composite electronic service (E) are acceptable:

$$G_3(E) = 1$$ (47)

In order to determine if a composite electronic service is allowed to be executed by an entity it has to be checked by the information security management system. It means it has to be checked in parallel or in series (order doesn’t matter in this case) using access and execution management models. Electronic services can originate from different local systems and their protection classes, operations and categories of an authorized execution will be checked and integrated.

The integration process of local lattices consists of several steps. In simplification four can be pointed out:

1. Set of local lattices is checked for consistency (a decision if they can be integrated).
2. Construction of a super-lattice.
3. Reduction of a supper-lattice.
4. Transformation of a supper-lattice.

5 CASE STUDY

5.1 Public e-services integration – the current model

In order to demonstrate usefulness of the proposed model it is necessary to compare it to the current solution. This case study presents execution of simple but integrated public e-service: registration of a newborn child.

Without systems integration citizen had to visit two different offices to complete this public service:
- Register Office to receive a birth certificate,
- Municipal Office (department of Population Registry and ID Cards) corresponding to the registered address to generate and receive personal identification number (PESEL) for a newborn child.

When those offices and their systems are integrated it is enough to visit only Register Office and an official can do both operations for a citizen as a one integrated service. However it is not delivered electronically to the citizen but is an e-service for an official (it can be treated as a public electronic service for this case study purpose).

Each office has different IT systems, different security user groups and policies. Electronic services in accordance with the applicable Polish law should be integrated using integration platforms – like ‘e-PUAP’ (Electronic Platform of Public Administration Services) or ‘Source’ (original name in Polish is ‘Źródło’ - a system that is integrating public registers).

Figure 6 presents simplified model of ‘registration of a newborn child’ e-service (blue services are Register Office services, yellow service is a Municipal Office service, red is a initiator service).

For this case study purpose we can assume that each office has similar protection classes and authorized operation (for the data):
- K: public < personal < confidential < secret
- T: read < write < update < delete

Each office has different user groups which result in different authorized execution classes.

For Register Office it is:
\[ B_{RO} : A1 \text{ (authenticated citizen) } < A2 \text{ (authenticated official) } < B1 \text{ (official with public documents authorization) } < C1 \text{ (head of Register Office department) } \]

For Municipal Office it is:
\[ B_{MO} : A1 \text{ (authenticated citizen) } < A2 \text{ (authenticated official) } < D1 \text{ (official with PESEL system authorization) } < E1 \text{ (head of Municipal Office) } \]

New security level of integrated public e-services is usually a decision of an expert. Different security groups are used in each office so according to the actual model there will be a new security group created – F1. Users in the new group will be allowed to run selected e-service (more precisely right to execute a initiator service). In most cases all users from B1 security group will be added to F1 group to allow all officials in Register Office to run both services as an integrated solution. For complicated integrated services data access rights are not checked for every component originating from other system.
so we can assume: ‘read and write – personal data / F1’ as the minimum security requirement for Register Office officials to run analysed integrated e-service (in the actual security model). In this case data access rights are checked locally and user has to be the member of security group created specifically for this new e-service.

This means that:
- there is no automation in e-service integration (need of expert every time new integrated e-service is created or changed),
- new security groups are often created specifically for new integrated e-services,
- security compliance is not checked for every component in every integrated system. In case study example proposed security level is too low as Municipal Office is processing confidential data in PESEL system so it should be ‘read and write – confidential data / F1’. This would result in the correct situation that not all officials from Register Office can execute integrated e-service, even if they are in the proper group (F1). They need additional data access rights permissions.

5.2 Public E-services Integration – the New Model

The same example is discussed using mandatory requirements from the new proposed security model. The first step is system integration which requires security attributes integration. All security attributes of analyzed systems meet lattice requirements - parameters K, T, B_{RO} and B_{MO} are partially ordered sets so they can be treated as local lattices. Local ‘protection classes – K’ and ‘authorized operations – T’ are the same in both systems, so integrated super-lattice will be identical to local. The super-lattice of authorized execution classes B_{L}, presented in Fig. 7, is calculated using lattice transformations.

Figure 7. Super-lattice of authorized execution classes.

After completion of described procedure, every electronic service consisting of any atomic e-services from analyzed integrated systems can be automatically checked for resulting permissions. Every user (entity) trying to execute these electronic services will be checked against access and execution security policies.

Register Office official from the example needs at least ‘read and write confidential data’ as all data access rights are checked in K and T super-lattices. He also needs at least B1 ‘execution authorisation’ as B1 is greater than or equal to D1 and A2 (from B_{L} super-lattice).

This means that:
- there is automation in e-service integration (possible need of expert only one time during systems integration, e-services can be created and changed with automatic security level recalculation),
- no need of new security groups (available security groups are integrated in super-lattices),
- security compliance can be checked for every component in every integrated system as they share common security schema defined with super-lattices.

Presented example is simple but integrated e-services can become very complex consisting of many electronic services from many different systems. Lack of well-defined security model for public integration platforms is a delaying factor in development of this area of public administration.
Long composite e-service creation time and need of experts makes integration projects very expensive.

Some complex calculations like lattices to super-lattice transformations have been deliberately omitted in this example, as purpose of this article is to present the idea of the new model, not its detailed mathematical proof. To achieve a complete secure solution in real environments, proposed security model has to be accompanied by additional functions like: authorization and authentication mechanisms, threat and antivirus protection, encryption and events logging.

6 CONCLUSIONS

This article presented only an introduction to the proper information security model described in the author’s doctoral dissertation (in preparation). New security model for composite electronic services published on integration platforms, which was introduced in this paper meets all the requirements placed on complex heterogeneous service oriented systems. Proposed AM and EM models (thanks to the lattice theory) can be used for integrated e-services originating from different systems.

Further, new model will be tested on polish public administration integration platforms. It is an answer to requirements defined by polish government to provide composite public e-services, which “will be produced in process of integration (interoperability) of different systems” [19]. Presented model can also be easily adopted to private sector integration platforms for example: banking and insurance integration systems. It can also be used for military integration mentioned in “Multilateral Interoperability Programme strategy to achieve interoperability of cooperating national Command and Control Information Systems at all levels of command, in support of multinational, combined and joint operations” [20].

Service oriented architecture and integration of separate systems is the main trend in 21st century IT. However, a new systems architecture requires a new approach to information security. Direct access security models are not suitable for e-service systems, like integration platforms and increasingly popular cloud environments.

7 REFERENCES

