STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LV**, Number 2, 2010

A NEW VIDEO SURVEILLANCE SYSTEM USING AUTOMATED SECURE SERVICE COMPOSITION

GENGE BÉLA AND HALLER PIROSKA

Abstract. We propose a new video surveillance system based on a new service-oriented middleware. The proposed system includes video capture services and saving services that are composed together in order to ensure that frames are saved for future replay. The main novelty of the proposed middleware and video surveillance system lies in the automated composition of security protocols implemented by system services and in the automated execution of security protocol specifications. This novelty allows us to implement a heterogenous system, where services implement heterogeneous security protocols, are capable of downloading new security protocol specifications and automatically execute them.

1. Introduction

Over the last decade, because of increasing security concerns, video surveillance systems have received significant attention from both the research and industry communities. The use of surveillance systems has become a necessity in airports, traffic, subways, stores and even homes. The requirements for developing such systems, formulated by Ostheimer et al [1], include affordable hardware requirements, low bandwidth consumption and access control procedures.

The use of the Internet to transfer video images between distributed nodes has made it possible to expand the applicability of these systems to remote surveillance [2], allowing the distribution of nodes across multiple countries and

Received by the editors: December 7, 2009.

2010 *Mathematics Subject Classification.* 68M14, 68Q55, 94A62.

1998 *CR Categories and Descriptors.* H.3.5 [**Information Systems**]: Information Storage and Retrieval – *Online Information Services* K.6.5 [**Computing Milieux**]: Management of Computing and Information Systems – *Security and Protection*;

Key words and phrases. Video surveillance, Web services, Security protocols.

This paper has been presented at the International Conference Interdisciplinarity in Engineering (INTER-ENG 2009), Târgu-Mureş, Romania, November 12–13, 2009.

continents. By using the Internet, surveillance systems also adopted modern security protocols such as TLS or VPN to secure data transfers.

However, using existing transport or network layer security protocols comes with a price. First, networks are usually closed environments protected by firewalls or NATs (i.e. Network Address Translation), where access is granted only to HTTP-based protocols. By implementing remote access, surveillance nodes can be placed behind multiple firewalls that can not be controlled by system developers. Second, the use of these protocols lacks the flexibility and extensibility required by applications running in heterogeneous environments. The heterogeneous property of video surveillance systems comes from the variety of protocols used by system nodes that must run for a long period of time and must face protocol changes without interrupting operation.

In this paper we propose a Web service-based approach for implementing video surveillance systems. Web services have been widely used to implement open systems running heterogenous protocols. In the proposed system, capture services (i.e. CAP-S) are used to capture frames and send them to client applications. Frames can be saved and replayed later if capture services transmit frames to saving services (i.e. SAVE-S). Replay is guaranteed by replay services (i.e. REPL-S). Connecting capture services to saving services is not a trivial task. Because each service can use different communication and different security protocols, interconnecting these services becomes a real challenge. Thus, coordinating and supervising this procedure becomes the task of the authorization and composition service (i.e. AUT-S). In order to automatically compose services, protocol specifications must be already available. These are stored and managed by the specification service (i.e. SPEC-S). Also, in order to automatically locate services, we use a name service (i.e. NAME-S).

Existing approaches [1, 2, 3, 4] assume a static interconnection of nodes, where nodes do not dynamically change the protocols or security protocols that are executed. The proposed system introduces many advantages over the mentioned existing systems. First of all, services are implemented as Web services that provide several advantages such as open and standardized protocols, technologies for service discovery and automated service composition. Second, the proposed system allows running different security protocols that provide secure mechanisms for accessing resources. Third, service capabilities are automatically composed together with security protocols, allowing automated service interconnection.

In order to enable the composition of Web services that implement heterogenous security protocols, first, we present a new middleware. The proposed middleware makes sure that both Web service capabilities and security protocols are composed. In our previous work we developed a new automated composition method of security protocols that has been implemented in the proposed middleware [6].

The main novelty of the proposed middleware and video surveillance system lies in the automated composition of security protocols implemented by system services and in the automated execution of security protocol specifications. This novelty allows us to implement a heterogenous system, where services implement heterogeneous security protocols, are capable of downloading new security protocol specifications and automatically execute them.

2. Security Protocol Composition and Specification

2.1. **Composition.** The role of the composition process is to create new, composed services, with an accumulated set of properties. These properties include service and security capabilities.

The composition of service capabilities ensures that the capabilities provided by services can be combined with the capabilities of other services. For each service, we define a set of preconditions and effects. Let ε be the set of preconditions of a service that denotes the type of data that can be received. Also, let ε' be the set of effects corresponding to a composed service, denoting the type of data generated by the service. Then, for a given service S_R, preconditions and effects are represented as $\varepsilon S_R \varepsilon'$. For N services, the composition of service capabilities reduces to finding the sequence for which $\varepsilon'_i = \varepsilon_{i+1}$, where $i = \overline{1, N-1}, N \geq 2$ and $\varepsilon_1 = \phi$. This means that the data corresponding to service preconditions must be available for each service and the set of preconditions corresponding to the first service must be empty, denoted by ϕ in the previous equations.

The composition of security protocols must ensure an accumulated set of security properties and must have a non-destructive effect on the security properties of the original protocols. This is not a trivial task, even when there is a human factor available. However, for the composition of security protocols in an on-line environment, the human factor can not be present, meaning that the entire composition process must be automatic. Such a composition method was developed in our previous work [5, 6], where we also proved that if security protocols are independent then these can be securely composed without losing their security properties. The key to eliminate the human factor is using an enriched protocol model, with the drawback that modifying security protocols in case of unsatisfied conditions is not possible.

2.2. **Specification.** Constructing a new specification is not a trivial task, if we just look at the information required to be included: preconditions, effects, types, message sequences, security properties, protocol roles, etc. This task becomes even more complex when we realize that by automatically composing security protocols we get new protocols that must be automatically implemented by services.

In our previous work [7], we have chosen WSDL-S [8] and OWL [9] to be the components from which specifications are constructed. Each protocol participant is described by a pair of WSDL-S and OWL files. The role of the WSDL-S component is to describe the message sequences and directions that must be executed by protocol participants. Each WSDL-S component contains the participant's role in executing the protocol (i.e. initiator, respondent or third-party), protocol preconditions, protocol effects and message sequences. The role of the OWL component is to provide semantic information such as the construction, processing and implementation of cryptographic operations (e.g. encryption algorithm, encryption mode, key). This is connected to the WSDL-S component via annotations.

3. Middleware architecture

3.1. **Service oriented architecture.** We define four types of services: name services, specification services, authorization and composition services and resource services. Name services (i.e. NAME-S) are implemented through UDDI [10] registries and are used to register, identify or locate existing services. Specifications, consisting of WSDL-S [8], OWL [9] and SAWSDL [12] files, are stored and managed by specification services (i.e. SPEC-S). Authorization and composition services (i.e. AUT-S) implement the verification mechanisms of client credentials and ensure the composition of service capabilities and security protocols. Finally, the resource services (i.e. RES-S) implement a set of capabilities provided for client applications.

The general middleware architecture is given in Figure 1. Each service contains several modules, a special attention was given to security aspects. Namely, each service contains an M-KEY module that handles cryptographic keys and certificates. These are used by security protocols in order to provide a secure communication between services. Another module that is present in all services except RES-S is M-SPEC, that implements all communication with the specification service. The RES-S receives the specifications from AUT-S when the composition process is executed.

In order to access NAME-S and SPEC-S without any previous connections with SPEC-S, the mentioned services provide the M-SPPR module for automatically downloading their own specifications. This module implements a standard HTTP/GET protocol for downloading own specifications.

The other modules are specific to each service. Thus, the M-DATA module implements communication with a database; M-CCAP, M-CPREF and M-CTERM are used in the composition of service capabilities and security protocols; M-CPERF is used to evaluate the performance of security protocols after the composition process is ended (based on this evaluation, the most performant protocol is chosen); M-NAME is used to implement the communication with the NAME-S; M-ROUT and M-RIN are used when resources are connected with each other in the composition process; M-CL is used to implement communication with client terminals; M-IOSPEC is used to store specifications; M-AUT-COMP implements composition-specific operations and M-RES implements resource-specific protocols (for AUT-S) and resource capabilities (for RES-S).

The theoretical aspects behind the implementation of these modules have been covered in our previous work [5, 6, 11]. The composition process [5, 6] ensures that the resulting protocols maintain their security properties. This has been implemented in three modules: M-CCAP, M-CPREF and M-CTERM. However, there can be multiple sub-protocols satisfying the same security requirements (i.e. mutual authentication, key exchange). In these cases we used a comparative performance evaluation process that we developed in our previous work [11], implemented in the M-CPERF module.

Resource access by client applications is done in several steps. In order to access a resource services, client applications must be authenticated, their rights must be verified and, if necessary, resources must be composed. This is done by accessing the AUT-S service, for which the specifications must be first downloaded. Clients request the location of AUT-S and SPEC-S from NAME-S, followed by the request of the specifications for AUT-S. The request for accessing RES-S is sent to AUT-S, containing the user credentials. These are verified and a security token is generated by AUT-S that is sent to RES-S. The client receives the generated token and sends it to RES-S, after which it is able to access the capabilities provided by the composed resource. All these steps are executed securely using an underlying security layer presented in the next section.

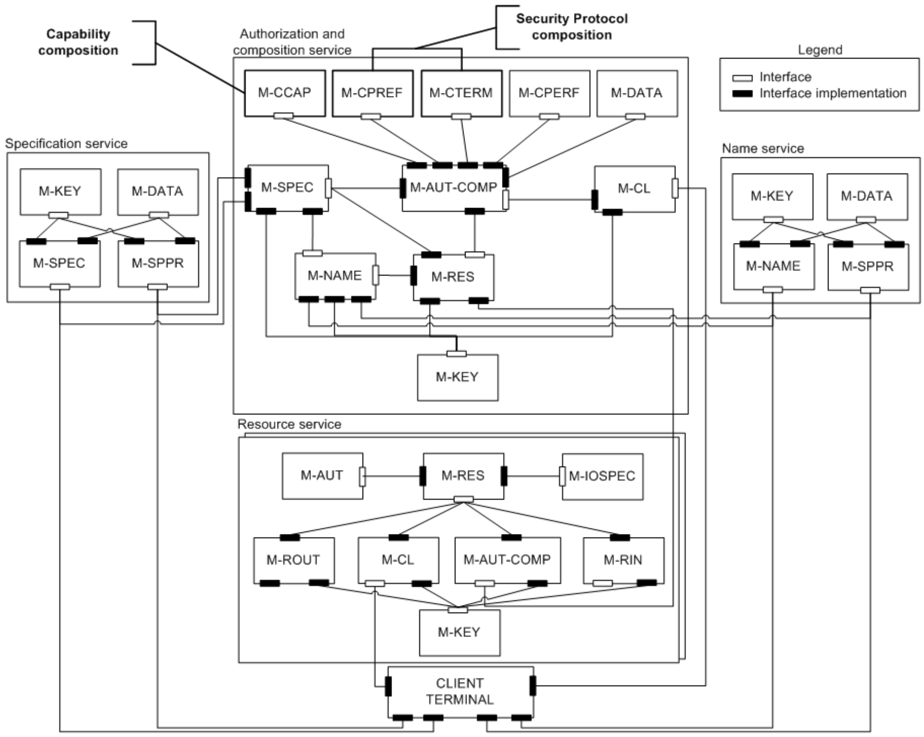


FIGURE 1. Middleware architecture

3.2. **Software architecture.** The architecture of the software stack is given in Figure 2. We identified two main layers: the communication layer and the service layer, given in Figure 2.

The *communication layer* implements service and security protocols. It is built on existing network and XML message-based protocols. Security protocols are implemented using extensions of the WS-Security standard, provided in our previous work [7]. The extensions consist of XML constructions for user names and binary keys required by key exchange and authentication protocols. The automated execution of security protocols is based on specifications developed using existing Web service technologies such as WSDL-S [8] and OWL [9]. Service protocols are described by SAWSDL [12] specifications and are specific to each service. Name services define a set of messages for interrogating service registry, while specification services define messages for downloading specifications. Authorization services define messages for requesting access to services and to send security tokens. Resource services provide two types of

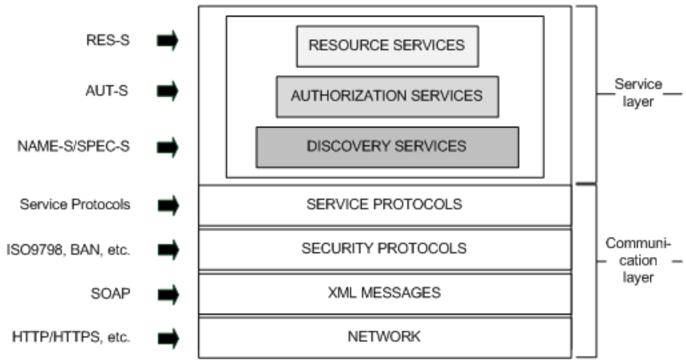


FIGURE 2. Software stack

specifications, for each external entity accessing them: AUT-S specifications provide messages for setting user security tokens, while client specifications provide client access messages.

The *service layer* provides the implementation of service capabilities. The capabilities of NAME-S, SPEC-S and AUT-S have already been discussed before. The capabilities of RES-S are specific to each implementation, ranging from video image capture capabilities to data storage capabilities. In order to implement new resource services, the only components that require change are the capabilities and the service protocol specifications. The security properties of communications with new resources are ensured by the underlying communication layer that remains unchanged.

4. Implementing a new video surveillance system

4.1. **System architecture.** In this section we describe our proposal for a new secure Web service-based video surveillance system. The proposed system is based on the platform presented in the previous section and comes with multiple advantages over existing video surveillance systems.

The proposed video surveillance system defines three types of resource services: capture services, saving services and replay services. Capture services (i.e. CAP-S) are used for capturing video images from physical devices. Frames can be sent directly to user applications or they can be saved for future replay. Frames are stored by saving services (i.e. SAVE-S) that also deal with the distribution of captured frames to storage hardware. Saved frames can be viewed by using replay services (i.e. REPL-S). The three service types are implemented as resource services using the proposed middleware.

To store the captured frames, we use the composition service provided by our middleware. Composition is used to compose SAVE-S with CAP-S services in order to save captured frames. Thus, frames can be replayed later in case a detailed analysis is required for a certain event from the past.

Clients can issue composition requests and the system automatically changes its connections, frames can be automatically redirected to new services. The main advantage of the proposed system is that composition is also possible with services implementing heterogeneous security protocols, because of the special security protocol composition and specification execution modules.

4.2. **Experimental results.** The experiments we conducted focussed on measuring the time required to access single and composed resources and on the time required to transfer data between services and client applications. For each resource type we used a different station with the same hardware configuration: Intel Dual Core CPU, 1.8GHz, 1GB of RAM, Windows XP OS. Our measurements were focused on capturing the processing performances of services, so stations were connected to a Local Area Network using several switches.

First, we measured the accessing time of single resources (i.e. that do not require composition). As shown in Figure 3, operations such as locating and downloading specifications clearly affect system performance. If specifications are not saved, accessing time of single resources is on average 580ms. If specifications are saved, the accessing time drops approx. 200ms. The peaks from Figure 3 denote the cases when the authorization service also needs to download specifications for the requested services.

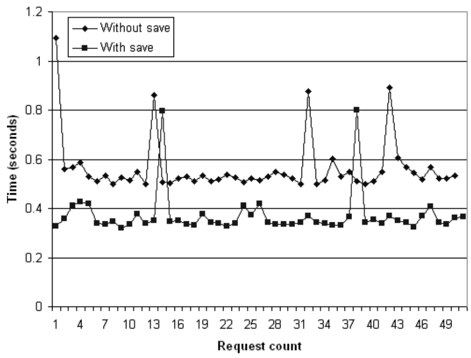
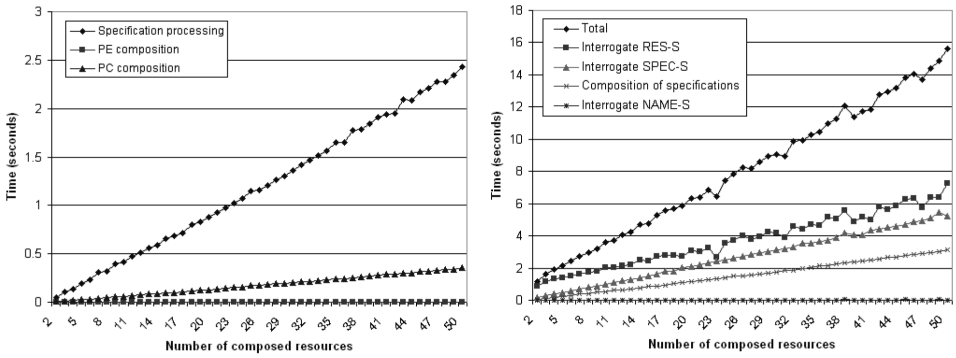


FIGURE 3. Accessing time for single resources



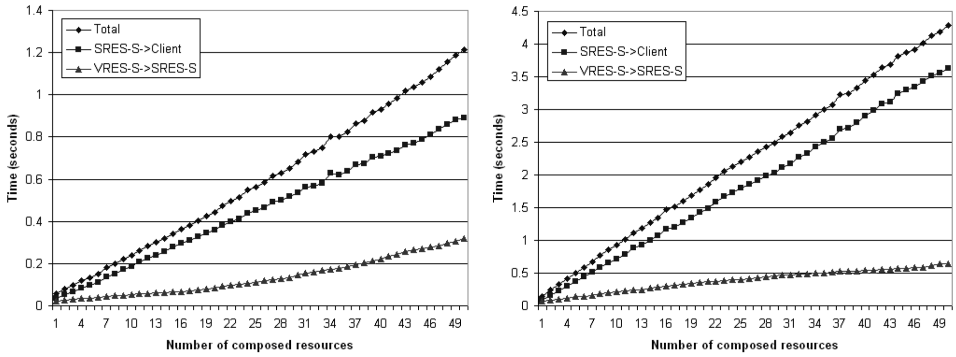
(A) (B)

Figure 4. Composition time: (A) Specification composition time (B) Accessing time of composed resources

Next, we measured the actual time required for the composition of specifications. Measurement results are shown in Figure 4 (A). The performance of the composition of preconditions and effects (i.e. PE-composition) and of the composition of protocol chains (i.e. PC-composition) is relatively good, considering that the composition takes around 450ms for 50 resources. However, the composition time is also influenced by the size and number of specifications, as shown in the same figure. Thus, the actual processing of specifications can reach almost 2.5 seconds for 50 resources.

While accessing composed resources, users do not only have to wait for the composition of specifications, but they also have to interrogate multiple services, such as the name service or specification service. Also, in the composition process the authorization service must connect to all resource services involved, must download their specifications and must compose them. These operations all contribute to the total accessing time for composed resources, as shown in Figure 4 (B).

Another aspect we measured was the time needed for frames to reach client applications in case of composed resources. We considered the two types of resources mentioned at the beginning of this section: SAVE (i.e. SRES-S) and VIDEO (i.e. VRES-S). We composed 2 to 50 VIDEO resources with a single SAVE resource and we measured the average time for frames to reach the SAVE service (i.e. from VRES-S to SRES-S) and the average time for frames to reach the client application from the SAVE resource (i.e. SAVE-Client).



(A) (B)

FIGURE 5. Transfer time: (A) 5fps, 10KB/frame (B) 10fps, 20KB/frame

The actual capturing of frames from hardware has only been simulated by the M-RES module using random bytes.

Because the transfer time is affected by the size of the frames and by the number of frames/second, we considered two cases. First, we considered a 5fps frame-rate, with the size of 10KB for each frame. For this case, results are shown in Figure 5 (A). For another case, we considered a frame-rate of 10fps with the size of 20KB for each frame. Results for this later case are shown in Figure 5 (B). By inspecting the measurement results from Figure 5 (A) and Figure 5 (B), we realize that the execution time increases almost 4 times in the later case. This is because we did not only increase the number of frames/second, but we also doubled the frame size, leading to an even more increased processing overhead consisting of cryptographic and XML processing operations.

5. CONCLUSIONS

We presented a new middleware for the automated composition of Web services that user heterogeneous security protocols. The approach used for the composition of security protocols was developed in our previous work and was implemented in the proposed middleware as a composition module.

The novelty introduced by the paper is the video surveillance system that can be used to create new, composed video resources. In this paper we composed video capture with video save resources in order to provide saving capabilities for frames that must be replayed in the future. The performances of the implemented system are highly dependent on the frame-rate and the size of the capture frames, which can be improved by using dedicated hardware for cryptographic operations or XML parsing.

References

[1] D. Ostheimer, S. Lemay, D. Mayisela, P. Dagba, M. Ghazal, A. Amer, *A modular distributed video surveillance system over IP*, in Proc. IEEE Canadian Conference on Electrical and Computer Engineering, Ottawa, Ontario, Canada, 2006, pp. 1001–1004.

[2] X. Yuan, Z. Sun, Y. Varol, G. Bebis, *A distributed visual surveillance system*, IEEE international conference on advanced video and signal based surveillance proceedings, 2003, pp. 199–204.

[3] J. Sun, S. A. Velastin, B. Lo, M.A. Vicentio-Silva, L. Khoudour, *A Distributed Surveillance System to Improve Personal Security in Public Transport*, Proceedings of the European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology (EWIMT-04), 2004, pp. 7–14.

[4] G. Gualdi, A. Prati, R. Cucchiara, E. Ardizzone, M. La Cascia, L. Lo Presti, M. Morana, *Enabling technologies on hybrid camera networks for behavioral analysis of unattended indoor environments and their surroundings*, Proceeding of the 1st ACM workshop on Vision networks for behavior analysis, 2008, pp. 101–108.

[5] B. Genge, I. Ignat, *Verifying the Independence of Security Protocols*, IEEE International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, 2007, pp. 155–163.

[6] B. Genge, I. Ignat, P. Haller, *Automated Composition of Security Protocols*, IEEE International Conference on Intelligent Communication and Processing, Cluj-Napoca, Romania, 2009, pp. 251–258.

[7] B. Genge, P. Haller, *Towards Automated Secure Web Service Execution*, Networking 2009, Aachen, Germany, May 11-15, Lecture Notes in Computer Science (LNCS 5550), L. Fratta et al. (Eds.), Springer-Verlag, 2009, pp. 943–954.

[8] World Wide Web Consortium, *Web Service Semantics - WSDL-S*, W3C Member Submission, 7 November, 2005.

[9] World Wide Web Consortium, *OWL Web Ontology Language Reference*, W3C Recommendation, 10 February, 2004.

[10] Organization for the Advancement of Structured Information Standards, *Universal Description Discovery and Integration*, available at http://www.uddi.org/pubs/uddi_v3.htm, 2004.

[11] B. Genge, P. Haller, I. Ignat, O. Ratoi, *Informal specification-based performance evaluation of security protocols*, IEEE International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, 2008, pp. 193–200.

[12] D. Martin, M. Paolucci, M. Wagner, *Toward Semantic Annotations of Web Services: OWL-S from the SAWSDL Perspective*, OWL-S: Experiences and Directions - workshop at 4th European Semantic Web Conf, 2007.

"Petru Maior" University of Târgu Mureş, Department of Electrical Engineering, Nicolaie Iorga St., No. 1, Târgu Mureş, 540088, Romania

E-mail address: {bgenge,phaller}@engineering.upm.ro