Federated claims based authentication and access control in the vehicular networks

Ramzi Debab, Yacine Challal

Federated claims based authentication and access control in the vehicular networks

Ramzi Debab (r_debab@esi.dz)[*]
Yacine Challal (ychallal@hdc.utc.fr)[**]

Abstract: The emergence of the vehicular networks has led automakers and researchers to develop connected embedded applications to exchange messages as part of road safety and entertainment. Secondly, the security has become a vital prerequisite for the deployment of such networks. In terms of controlling access to resources, more heterogeneous solutions are proposed based on X.509 certificates, Kerberos tickets, Password techniques, etc. This technologies heterogeneity pushes us to find a solution in order to federate the existing techniques and even future ones. WS-Federation is one of the keys aiming to solve this problem but only suitable for SOAs based on Web Services technology. Consequently, our approach was to integrate native distributed vehicular applications in a SOA and then apply the WS-* security-related Web services specifications while checking the federated access control to resources through WS-Federation and the notion of the claims. The result is a communicative architecture based on the ESB (Enterprise Service Bus) pattern that we baptized VSB (Vehicular Served Bus).
Keywords: Vehicular networks, Federation, claims, SOA, Web Services.

1 Introduction

1.1 Context

Thanks to the ITS (Intelligent Traffic Systems) technology, our cars are able to embed applications that aim to exchange so vital messages in coordination with road infrastructure units in order to ensure the road users safety. It will be possible, also, from his vehicle to consume services in the Internet or the Cloud.

Today, we know different schools (American, European and Japanese) in the field of ITS. Thus, many projects are emerging from research laboratories of several car manufacturers and universities. Nevertheless, the technological heterogeneity will be, undoubtedly, one of the thorny issues to resolve in order to build a network of distributed applications to communicate seamlessly [LSI08].

Furthermore, security is a crucial prerequisite [HUB07] for the deployment of vehicular networks and several solutions have been proposed. Access control to resources is a security aspect that has attracted the attention of researchers. Thus, varied and heterogeneous implementations have been developed based on X.509 certificates [LU08] [HUB07], Kerberos tickets, the Username / Password couples [GUO09], etc.

[*] Ecole nationale Supérieure d'Informatique, Algiers, Algeria.

[**] Université de Technologie de Compiègne Heudiasyc, UMR CNRS 6599 BP. 20529, Compiègne Cedex, France.

1.2 Problematic

Before presenting our problem, let us imagine the case of a vehicle in a country "A" that uses X.509 certificates to consume different services. This same vehicle travels to another country "B" and wants to consume the same services. Yet, in the country "B", Kerberos tickets are rather used. Consequently, our problem is to find a solution that unifies the various access control techniques in the vehicular networks. The solution must be open and agile to incorporate future technologies that do not exist yet.
Web services are the best technology to build SOA. This success is due to the richness of WS-* specifications that support different aspects: orchestration, transactions, security, etc.
WS-Federation is one of the WS-* security specifications that aims to federate heteregenous authentication and access control protocols. For this reason, we considered WS-Fedeartion as a good candidate to solve our problematic.
However, we have to think about how to integrate vehicular applications which are not necessarily Web Services into a SOA (based on Web Services) in such way that we can apply WS-Federation.

1.3 Our Approach

Initially, our objective is to develop a technique for integrating vehicular applications into a SOA using the bus pattern. Our inspiration is the Enterprise Service Bus (ESB) [ERL09]. We have adapted the ESB for the case of the vehicular networks and the new bus architecture is named VSB (Vehicular Service Bus). With VSB, it is possible to consider a vehicular network as a set of mobile and fixed nodes consuming and providing both services.

The second step is to apply the WS-Federation on our VSB architecture to solve the security heterogeneity problem.

1.4 Paper Structure

This paper presents the developed solution in three phases. The first one will be dedicated to the communicative architecture of the VSB. The second phase analyses the federated security module. Finally, we will close the paper by presenting some results of our work.

2 Communicative Architecture

In this section we present, first of all, the vehicular network as a SOA. After that, we describe our solution based on the VSB pattern. The VSB is designed using the WCF[1] technology [NAD10] allowing LOB (Line Of Business) applications integration into a SOA.

2.1 SOA Vision

We have studied a set of vehicular applications and the different communication paradigms (Push model like notifications from traffic lights system to vehicles, Pull model like a STOP command from a police car to an ordinary vehicle and other messaging models). Consequently, the SOA seems to be an excellent candidate to develop vehicular applications. According to our

[1] Windows Communication Foundation

vision, a vehicular network is a set of three domains (see Fig. 1):

- V2Internet: Vehicle to Internet where a vehicle is able to consume Internet or Cloud Services.
- V2V: Vehicle to Vehicle where a vehicle exchanges messages with other vehicles in the Adhoc domain.
- V2I: Vehicle to Infrastructure where messages are exchanged between the vehicles and the roadside infrastructure nodes.

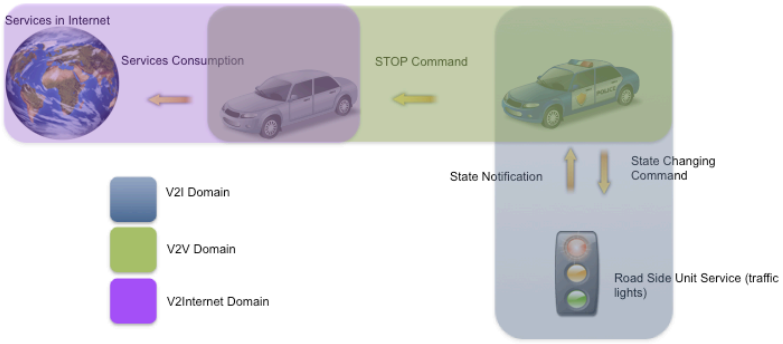

Fig 1. Vehicular ntwork domains.

2.2 VSB Structure

To integrate disparate applications and expose them as Web Services, we have designed a communication bus embedded into each vehicular network node. The VSB is based on the ESB pattern and is inspired from the Microsoft Biztalk[2] platform [DUN09].

The diagram (see Fig. 2) shows the different VSB components (and an example of an alert message lifecycle) by defining the following components :

- The VSB runtime: it is the execution environment containing a layer of engines exposed as WCF services (transformation of message formats [based on XSLT, for example], message routing, business rules, orchestration [based on workflows], Pub/Sub engine for notifications and events, resolving endpoints (like resolving a UDDI 3.0 URN into a URI). The VSB runtime is also equipped with a hosting environment for the various WCF services.
- Adapters: they are wrappers for the VSB applications and are exposed as input (input ports) and output (output ports) Web services. Their main role is to adapt the transport protocols, encodings and formats in line with the runtime input and vice versa at the output with the external system.

[2] http://www.microsoft.com/france/serveur/biztalk/default.aspx

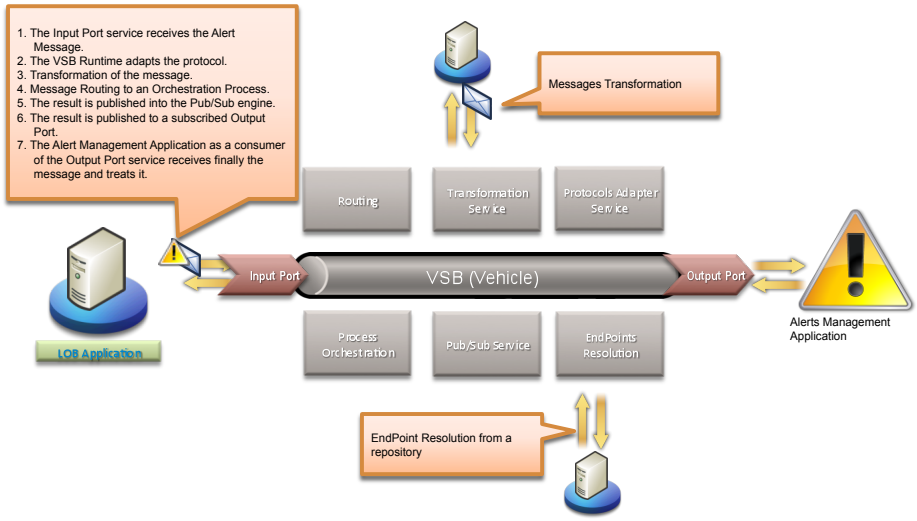

Fig 2. VSB components

2.3 Internal VSB Architecture

The figure (see Fig. 3) describes the VSB internal architecture that is similar to Biztalk architecture and contains the follwing components :

- Core web services layer: This layer contains the web services that implement the basic functionality of the VSB: the transformation of message formats, the resolution addresses of endpoints, and other customized features.
- Engines layer: this layer contains the different engines as the Pub/Sub engine that handles the registration messages as well as publications.
- VSB application layer: This layer contains a set of VSB applications as input and output Web Services. Each application has endpoints that implement different MEPs (Message Exchange Patterns) as Request / Reply, One Way, Pub / Sub, etc. Each application can be equipped with a discovery endpoint to be discoverable on the network. At each endpoint, there is an adapter, followed by a pipeline defined by an itinerary that is selected according to criteria defined by the rules engine.
- Tools layer: this layer is composed of the service and channel models as well as the WCF LOB SDK provided by Microsoft to create core services, VSB applications, adapters, etc.

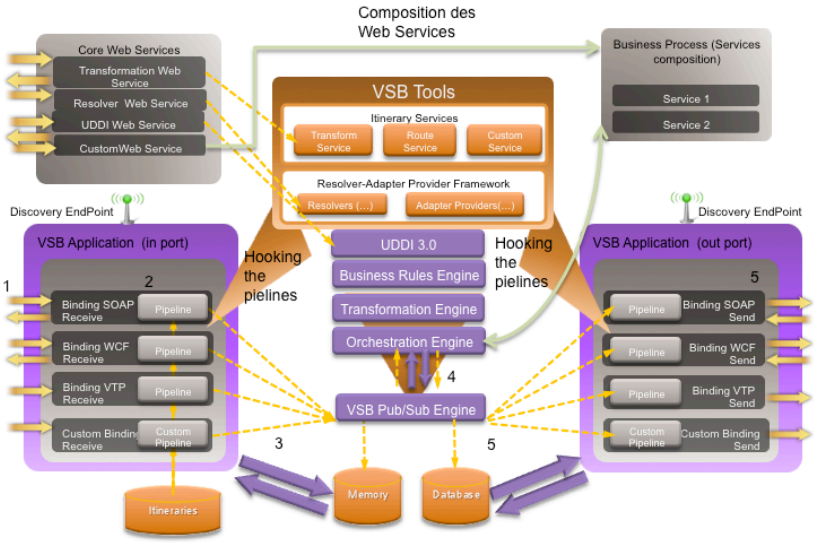

Fig 3. Internal VSB Architecture

2.4 Decentralised VSB Architecture

The figure (see Fig. 4) describes an example of a life cycle of an alert message corresponding to the slippery road alert generated by a LOB application of a vehicle (1). This alert message passed by the service road bus (2) is subsequently transmitted to all involved vehicles (3). The retransmission is based on the content of the message after a certain treatment on the service road bus. Finally, the message is transmitted through the bus of each vehicle for potential treatments to be managed through the appopriate application. The same message can be routed to a mobile device for the benefits of interactions with the driver.

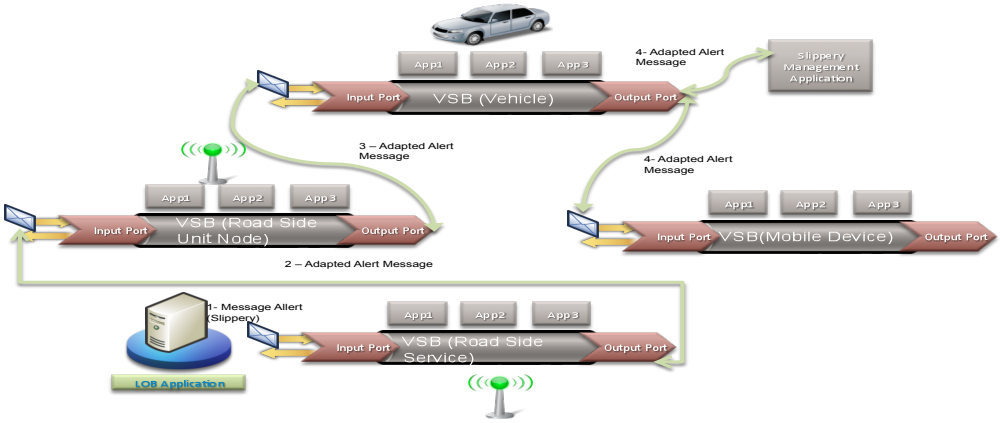

Fig 4. Decentralised VSB Architecture

3 Security Architecture

In this section we present the architecture of our security module comprising federated authentication and authorization based on the claims forming the security tokens.

3.1 Advantages of claims based access control

In this type of authorization, access control for the called service operations is done according to the content (the claims) of security tokens provided by the service consumers. Thus, access permissions are defined dynamically into security policies.

This model offers more flexibility than others (oriented roles, resources, etc.) for the following advantages [ERL09]:

- Decoupling authentication mechanisms from applications and services. Otherwise, an application consumer has to implement all supported authentication protocols. Nevertheless, the model is abstract and oriented tokens in such way it can support existing authentication mechanisms and even future ones.
- All credential types are supported. Just serializing them according to a given format is sufficient.
- Possibility of transforming the tokens from one format to another.
- This paradigm is based on defining security policies containing service requested claims. Thus, the service consumer has to deliver just tokens according to the policy and containing the requested claims.

3.2 Federated Access Control Architecture

In this section, we propose the architecture of the federated access control applied to our VSB platform. This architecture is based on the actif scenario of WS-Federation (WS-Federation Active Requestor Profile). The choice of this profile is supported by the flexibility of hosting services into different process types instead of just web servers. In addition, the active profile is applied to 'smart' applications, which correspond to the case of vehicular applications.
A federated architecture is generally composed of three entities :

- Client: in our case, it is a vehicular network node wishing to consume a service exposed in the same network, the Internet or on the Cloud.
- The service: it is a set of exposed functionalities to consume. Imagine the case of a control traffic light service.
- The Security Token Service (STS): it provides security tokens required by the service to consume via WS-Trust. A trust relationship relies the STS to the exposed service.

The following figure (see Fig. 5) shows the federated architecture of an example of two vehicles (ordinary car and police car) trying to update the status of a traffic light service :

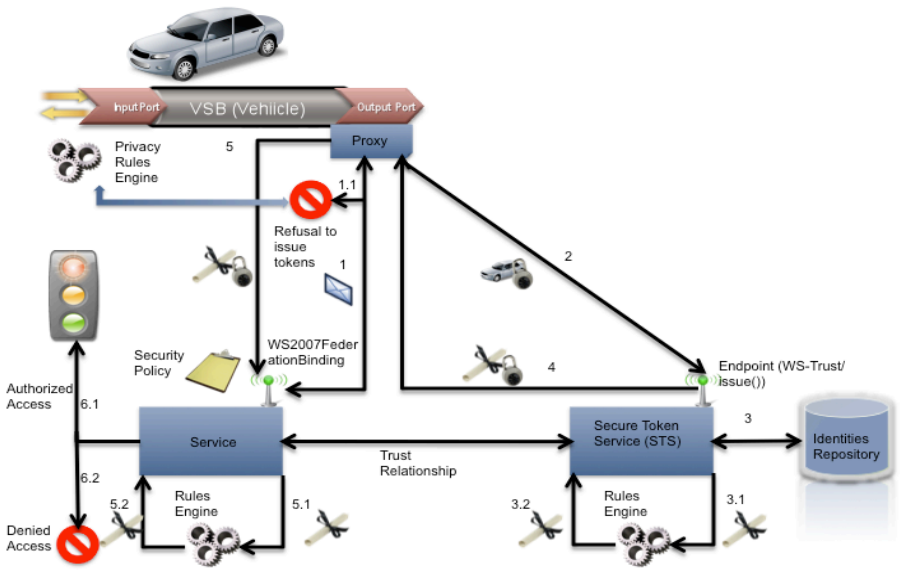

Fig 5. Federated Access Control Architecture

A vehicle which is equipped with a VSB, has an application in output (for example) and develops a proxy that is responsible for consuming the service. Thus, the federation active profile follows these steps: the proxy of the vehicle discovers the service through the WS-Discovery and retrieves the security policy set on a federated endpoint via WS-SecurityPolicy and WS-MetadataExchange. This policy expresses the claims to be included in the security token that must be issued by one ore more determinated STSs (1). Then, the native identity of the vehicle which does not correspond to the requested claims is sent to the STS within a security token request message (RST (Request Security Token)) (2). It should be noted, also, that the STS implements the WS-Trust that allows the tokens management. The STS then authenticates the identity of the vehicle and provides the required token (SAML[3] formatted for instance) (3) and generates, therefore, the response to the RST request as a data structure called RSTR (Request Security Token Response) (4) that contains the following :

- The serialized token, signed by the STS and containing the required claims and the proof key encrypted with the public key of the service to consume. The token is also encrypted with the public key of the service to consume.
- The proof key for the vehicle application.

The RSTR is protected by the policy expressed in the Binding of the STS. The vehicular application retrieves the proof key from the RSTR and sign the message to be sent to the service using the proof key (5). Thus, the service authenticates the token sent with the public key of the STS, retrieves the proof key from the token with its private key to authenticate the vehicle application as a real source of the request. Finally, the service makes a decision in terms of access permissions according to the generated claims (6).

[3] Security Assertion Markup Language.

3.3 Involved WS-* Specifications

The secured VSB architecture involves so many WS-* specifications in addition to WS-Fedreration.
The following figure (see Fig. 6) defines the involved specifications :

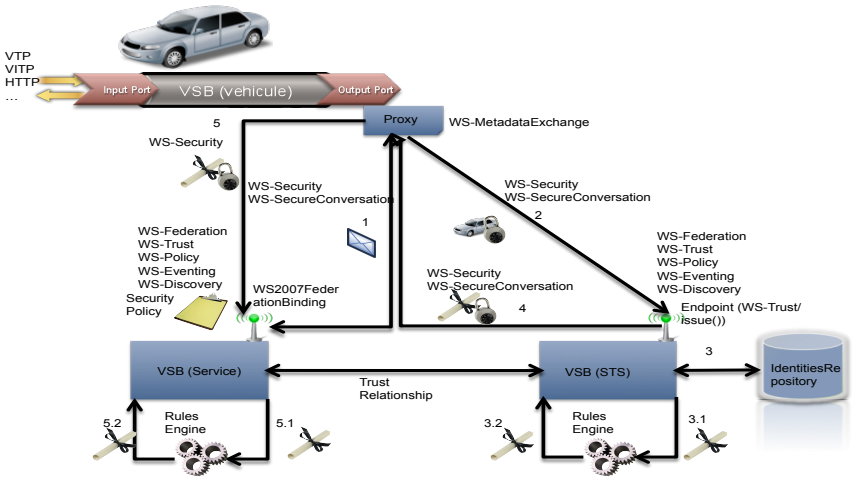

Fig 6. Security WS-* Specifications related to the VSB

The federated architecture defines foreach entity a set of specifications :

- Client: the proxy implements WS-MetadatExchange to discover services metatdata, WS-Security to protect the exchnaged messages and WS-SecureConversation to maintain a secure session.
- The service: its endpoints implement WS-Federation to support the federation principle, WS-Discovery to get the services discoverable, WS-Eventing to manage the events and the notifications, WS-Policy to define security policies, WS-Trust to manage the generated tokens, WS-Security and WS-SecureConversation.
- The Security Token Service (STS): it implements pratically the same specifications as the service to consume.

3.4 Towards a Vehicular Profile for Web Services

Thanks to the VSB architecture, it is possible de define a new profile adpated to the vehicular applications supporting all the necessary security specifications related to the Web Services.
The following figure (see Fig. 7) describes the profile layers which contains in addition to the DPWS (Device Profile for Web Services) components, different WS-* security specifications and supports also all transport protocols related to the vehicular networks as VTP (Vehicular Transport Protocol).

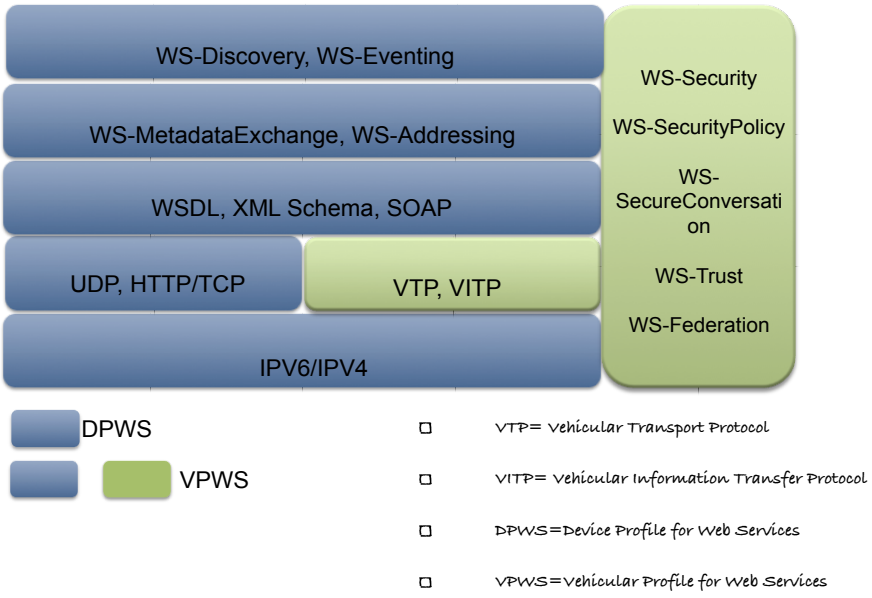

Fig 7. VPWS layers

4 Performance evaluation

To validate our work, we developed a POC (Proof Of Concept) application to solve the problem of the intersections management. In our POC, only the emergency vehicles are allowed to change the state of traffic light seamlessly. However, all vehicles are allowed to subscribe to the intersection management service in order to receive notifications about the road states.
The architecture of our application in terms of components is described by the following figure:

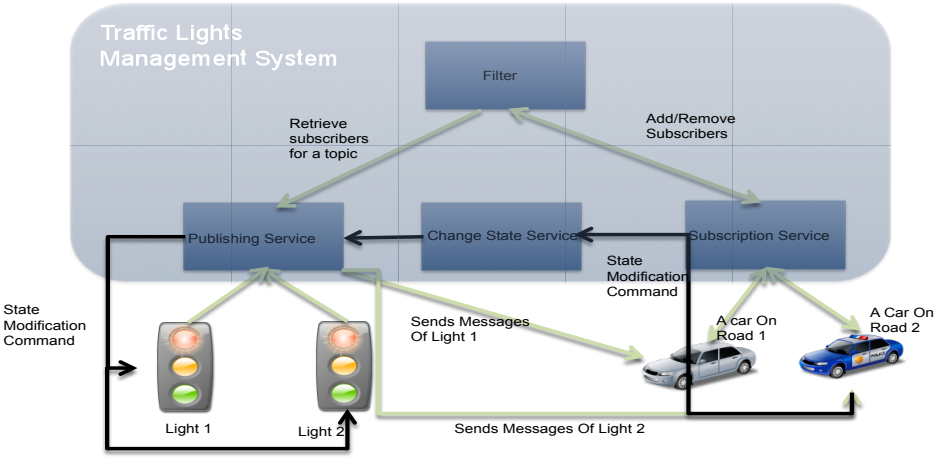

Fig 8. POC components

We have supposed that all vehicles implement X.509 certificates[4] as native identities. However, the traffic light service requires other types of tokens formatted into SMAL and defined in the following table :

Vehicle Type	Claims Types	Claims Values
Emergency Vehicle	http://claimsbased.VPWS.dz/samples/2009/06 /claims/role	/emergencyvehicle
	http://claimsbased.VPWS.dz/samples/2009/06 /claims/permission	/subscribe
	http://claimsbased.VPWS.dz/samples/2009/06 /claims/permission	/changestate
Ordinary Vehicle	http://claimsbased.VPWS.dz/samples/2009/06 /claims/role	/vehicle
	http://claimsbased.VPWS.dz/samples/2009/06 /claims/permission	/subscribe

Tab 1. Tokens types required by the Traffic light service.

It should be noted that we assumed that the architecture is already service oriented. Thus, our POC was designed to demonstrate the feasibility of SOA in the case of vehicular networks and test some important parameters such as time consumption during the message exchange.
We used the Visual Studio 2010 tool with C# language, the WCF 4 and WIF Frameworks to develop our POC.
The different application processes (vehicles processes, light services and intersection assistance system) were tested on a machine with the following characteristics :

- Model : MacBook of Apple.
- OS : Windows 7.
- RAM : 4Go DDR2 with 667 MHz.
- Processor : Intel Core 2 Duo T7250 with 2.16GHz.

We conducted performance tests on a vehicular application consuming the intersection service for two scenarios: direct and mutual authentication based on certificates. In each scenario, we modify the binding corresponding to the service and the application (transport protocol, security mode, cryptographic algorithms, etc.). After authentication, we record the average time between the date the message was sent and its interpretation at the application level during a cycle of light (green, orange, red).
Transport protocols that were subjects of our tests are TCP that has been used in [LSI08] and HTTP of which a similar protocol was defined in [NAD10].

[4] The choice of X.509 certificates is justified by the fact that it is the most common authentication mode in the literature of the vehicular networks security.

We have recorded the following times:

Binding	Time
HTTP without Security	6.73 ms.
HTTP with security Mode: Message. Algorithm: AES 256	10.73 ms.
HTTP with security Mode: Message. Algorithm: AES 256 SHA 256 RSA 155	10.85 ms.
TCP without security	1.84 ms.
TCP with security Mode: Transport.	3.59 ms.
TCP with security. Mode: Message. Algorithm: AES 256	5.59 ms.
TCP with security. Mode: Message. Algorithm: AES 256 SHA 256 RSA 15	6.24 ms.
TCP with security Mode: Message. Algorithm: 3 DES	7.65 ms.
TCP with security Mode: Message. Algorithm: 3DES SHA 256 RSA 15	8.07 ms.

Tab 2. Messages treatments times

Discussion. The real time is, perhaps, the most important criteria in vehicular communications. According to [CAV07], latency allowed for the warning signal in case of violation of the red light is 100 ms.
According to our tests, the worst time reached is just one-tenth of latency. That is very encouraging. Nevertheless, it is expected that the recorded times in real conditions are greater than those recorded in simulation.
Whereas concerning the durations of the federated authentication, the first time is estimated at almost 1.4 sec. However, thanks to WS-SecureConversation, a secure session is maintained between the communicating entities ; which avoids re authentication.

5 Conclusion

Our work is part of vehicular communications security and tried to answer a dual problem: how to integrate vehicular applications into a SOA? And how to federate different access control or authentication protocols?

[5] The RSA algorithm (asymmetric) is used to generate the symmetric keys used for encryption with AES, while the signature is based with SHA 256.

To answer the first problem, we developed an integration solution for vehicular applications, following the bus pattern. Our inspiration was the ESB pattern, which gave birth to a bus embedded in any vehicular network node and that we have named VSB (Vehicular Service Bus). To answer the second problem, we have implemented the WS-Federation and many other security-related specifications such as WS-Security. This solution is feasible thanks to the architecture of the VSB based on Web Services and all WS-* specifications.

References

[CAV07] Caveney D., Baliga G., Laberteaux K., et Kumar P. R. *Efficient Message Composition and Coding for Cooperative Vehicular Safety Applications,* IEEE Transactions On Vehicular 56 (2007): 3244-3255.

[DUN09] Dunphy George, Sergei Moukhnitski, Stephen Kaufman, Peter Kelcey, David Peterson, et Harold Campos. *Pro Biztalk 2009,* aPress, 2009.

[ERL09] Erl Thomas. *SOA design Patterns.* PRENTICE HALL, 2009.

[GUO09] Guo Huaqun, Lek Heng Ngoh, et Josef Chee Ming Teo. *An Anonymous DoS-Resistant Password-based Authentication, Key Exchange and Pseudonym Delivery Protocol for Vehicular Networks.* Singapore: International Conference on Advanced Information Networking and Applications, 2009.

[HUB07] Hubaux Jean Pierre, Maxim Raya, et Panos Papadimitratos. *Securing Vehicular Communications*, *Journal of Computer Security* 15, n° 1 (January 2007): 39-68.

[LSI08] L. Sichitiu Mihail, et Maria Kihl. *Inter-Vehicle Communication Systems: A Survey*, *IEEE Communications Surveys and Tutorials* (IEEE Communications) 10 (2008): 88-105.

[LU08] Lu Rongxing, Xiadong Lin, Chenxi Zhang, Haojin Zhu, Pin-Han Ho, et Xuemin Shen. *Security in Vehicular Ad Hoc Networks,* IEEE, 2008.

[NAD10] Nadeem Ahmed, et Muhammed Hamayun. *Performance Evaluation of Windows Communication Foundation's Interoperability.* Master Thesis, Ronnby, Sweden: Blekinge Institute of Technology, 2010.