International Journal of Advances in Computer Networks and Its Security
Volume 6 : Issue 3 *[ISSN 2250-3757]*
Publication Date : 30 December, 2016

An adaptive and smart security architecture for Web Service - SmartWSSec Architecture

El Houssain BEN MESSAOUD[1], Ouafaa DIOURI[2]
Mohammed V University in Rabat
Mohammedia School of Engineers
SIR laboratory
Avenue Ibn Sina B.P. 765 Agdal Rabat Maroc

Abstract – ***Vulnerabilities in Web based applications will always be present. Several measures were taken to extenuate the effects of this reality but with limited success. In fact, we are bombarded by new technologies to harden systems and monitor and respond to threats, like firewalls, IDS (intrusion detection system) and IPS (Intrusion Prevention System). However, the flow of attacks and threats is so important to the point that the configuration and reconfiguration of these tools becomes difficult to insure in time. In this paper we introduce "a framework for dynamic security Policy for Web services" called SmartWSSec. The main goal of our architecture is to guarantee better security for web services based on adaptive security models. It aims to identify the appropriate actions that must be taken when a zero day attack occurred resulting on a smart protection for web service in a self-adaptive manner. The proposed architecture uses a knowledge based mechanism to learn and adjust the system when new not-known before attack appears. The proposed architecture also includes an isolation faculty to protect the system when self-adaptation fails, which will notify and involve a system administrator. We have included on this paper also the design, the components, models and concepts of adaptive security architecture, and finally gives insights on a possible implementation by providing a POC applied to different use cases.***

Keywords*: Adaptive and dynamic security, Knowledge database, prevention, reaction, Security policy, Smart engine, Web service, WS-SecurityPolicy*

I. Introduction

Web services are distributed components that provide integration to applications over the network using open standards. They can therefore be used by applications written in different languages and executed in different platforms on different systems. To do so, they use a distributed architecture consisting of multiple computers and / or different systems that communicate over the network. A web service can typically offer business functions to be referred by consumers [1].

The security Web services technology has become an operational concern. In fact, so many concerns have transformed the information security to a critical issue, such as the growing need for the creation of new collaboration spaces between international partners, also secure manner for information exchange within the enterprise and beyond and finally the transition to a generalized digital economy.

In parallel, the dangers surrounding this transformation are growing. The security awareness brings organizations to describe the directions to follow in a document called security policy, specifying the objectives to guarantee or achieve. The implementation of security policy is a direct road to meet the requirements of integrity, availability, confidentiality and traceability, which forms, in addition to concepts of identity management and access control, the principal needs of protecting information systems of organizations.

Figure 1 presents data from the US database NVD (National Vulnerability Database). It shows that the number of vulnerabilities has increased the last three years. In view of the approximately 5200 entries registered till September 30, the total number of vulnerabilities for 2014 could exceed the record number reached in 2006 [3].

In this paper, we present a framework (called SmartWSSec) that outlining an intelligent mechanism for achieving end-to-end security for Web Services by providing regular review of their security policies which can enhance the global security level.

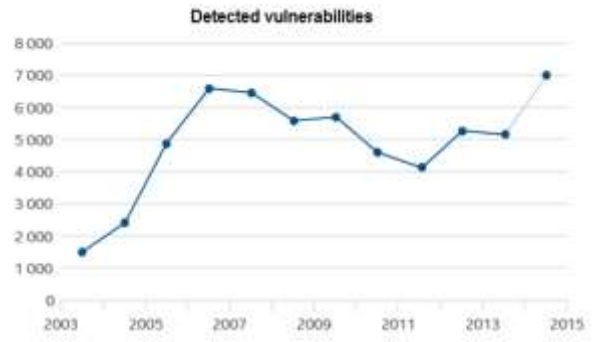


Figure 1 : Evolution of detected vulnerabilities (Source: National Institute and Standards and Technology – National Vulnerability Database)

Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

SmartWSSec enhances WS-Security which is the most proven standard for securing Web services. In fact WS Security stack is still limited to parametric and static protection and does not provide a model for enhancing security in a continuous and dynamic manner.

SmartWSSEC is a framework and a minimum viable architecture that can provide an adaptive and dynamic protection based on:

- Prevention mechanism: That uses proven security policy templates which contains best practices for protecting incoming and outgoing SOAP message and also a mechanism for regular review and update.
- Detection and reaction mechanism: through the integration of an intelligent engine that can take autonomous decision to protect the WS system against new -not known before- attacks.

It attempts to provide a framework for implementing self-protecting WS systems, by delegating a part of the responsibility of maintaining and updating the security configuration to the system himself and given it the capacity for analysis and learning, enabling it to anticipate a first level decision in case of attack or vulnerability detected.

The rest of this paper is organized as follows Section 2: Related work, Section: 3 Modeling of adaptability for software security, Section 4: presentation of our proposed intelligent and adaptable web services security architecture called SmartWSSec, Section: 5 Prototyping and Section 6. Conclusion.

II. Related works and motivation

For sure, the idea of self-adaptive system is not new.

Zhiwen Bai etl [18], proposed DTAD, a dynamic taint analysis detector aiming to protect malicious attacks and vulnerabilities comparing network data and log files to identify the attacks.

Fang Qi .etl [19], proposed Automatic Detecting Security Indicator (ADSI) for preventing Web spoofing on a confidential computer which is a harmless environment. It creates a random indicator to identify and detect bogus pages with URL screening data.

Much recent work has been reported in the literature about constraints imposed by the nature of tactical SOA implementations in which a security policy framework of suitable structural characteristics has been suggested, making use of description logic and ontological structures, that aims to identify suitable mechanisms for the on-line update of the security policy, aiming to maximize the node cooperation, while minimizing the cost of each policy decision in terms of resources [20]. Tarek Bouyahia etl [21], proposed also a context aware intrusion response based on argumentation logic.

Herve Debar elt [22], proposed an advanced security policy formalism, to define a contextual security policy that will be applied to the information system. It introduces a framework for verifying the security policy and for translating the security policy into custom configuration scripts that can be applied to policy enforcement points. The expression of the security policy allows the definition of simple responses to each Threat. The threat contexts vary according to alerts collected by various sensors. These alerts are mapped onto policy subjects, actions and objects and are used to activate specific contexts, which validate and transmit a new set of policy rules to the enforcement points.

Jose-Miguel Horcas [23], defined an approach called INTER-TRUST framework to maintain the correlation between the security policies that need to be enforced, the security aspects that are deployed in order to enforce those security policies approach has been integrated. This framework has been validated in pervasive systems field, however, it can also be applied to many other types of systems.

Tarek Bouyahia elt [24], confirm that having the ability for dynamic and intelligent enforcement of security policies becomes necessary to protect system from modern attacks. He suggest that the argumentative logic driven system are the most appropriate to achieve this objective.

All these rich works are more focused on detection/monitoring part in specific systems (pervasive/embedded) and do not take into account the specificities of web services. Also learning ability is not involved deeply.

Our proposed value added resides in combining self-adaptive models with web service security standards that are already proven and introducing artificial intelligence benefits to implement the learning faculty of the system regarding zero day Attacks.
In fact, SmartWSSec use the notion of adaptive security in its conception. And take action to enhance Secutiy policies written according to WS-Policy and WS-SecurityPolicy standards [2]

III. Modeling of adaptability for software security

The static security policies for web services are clearly no more appropriate to manage the risk associated with critical distributed applications in today's information systems. We must insure an accurate and regular customization to guaranty good protection against new

Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

vulnerabilities. Even that couldn't be enough, because we are protected only for a given period, and until a new change occurs within the system (new attacks, new vulnerabilities discovered). In addition, current web services security processes (aka WS Security Stack) produce a static security in time providing limited protection for known threats in advance. Also, there is a lack of decision-making ability (intelligence) intrinsic to the Web service, to confine a new attack or trigger an adapted reaction that needs a manual intervention to adjust the security policy and manage its redeployment.

We must therefore tune and optimize the security policy and its technical configuration continuously to maintain an acceptable level of safety. This is preferable to be in an automated manner, that what we can call an "adaptive security system" [8].

III.1. Definitions

Here are some definitions:

"Adaptability, is the property that defines the ability of a system, to take into account new events in its environment and to respond correctly. Indeed, adaptation is a mechanism that allows a system to provide its services under special or new conditions and allows him to make the necessary changes in a transparent manner. It means the faculty to adjust and respond to changing environmental constraints or business requirements [4].

Self-adaptive software, is software that modifies his behavior in response to changes in its operating environment [5].

In general, adaptability represents the modification of a system to respond to changes in its environment. Given the multitude of definitions related to this concept, we chose to withhold that is both generic and reduces ambiguities by its formal approach. According to Subramanian and Chung [6]:

Adapting a system S is caused by a ΔE change in an environment E which becomes a new environment E '. This adjustment results in a new system S' which ideally meets the needs of the new environment E'. Thus, the adaptation can be represented by a function [7]:

Adaptation: ***E×E'×S→S', where S' =f (E')***

We want to model the function f to be able de predict S'.

III.2. *Adaptive security model*

Moving from a static security model for web services to a dynamic and adaptive one is a highly needed wish. Otherwise, the web service will continue to suffer because the regular redeployment of a new security policy, following an intrusion for example, consumes time and effort that is proportional to the attack surface and what can open windows, give attackers to maximize damage to systems. Figure 2 shows the way to road to an adaptive security policy management that can react dynamically and redeploy autonomously:

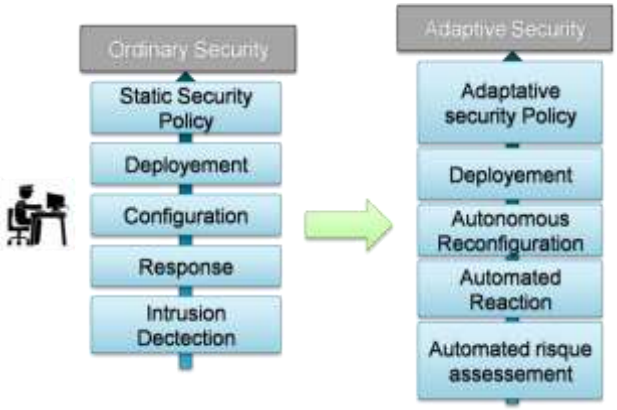


Figure 2: Towards an adaptive security model

III.3. *Protecting Vs monitoring*

The transition from a static model to a dynamic policy, shown in Figure 2, needs to be completed with a focused view on the link between the protection of a system and its supervision against vulnerabilities and attacks. Indeed, these two activities in a static model today are disconnected and linking between them, if any, is manually (using an administrator).

Example: In case of attack, intrusion detection tool will alert the system administrator (supervision) that will manually adjust the technical parameters of the security policy to counter this attack (act on the protection).

We are introducing the mechanism of a dynamic policy, that can be achieved by the introduction of a reasoning faculty to the system itself. It is a necessary feature to autonomous protection. In simpler terms, this means closing the protection/supervision loop, by enhancing the system capabilities to select/produce and execute itself some protective measures. This could be done using results analysis provided by the supervision and specifically the detection of intrusion.

Of course, we also close the loop by ensuring that the autonomous measures taken by the system have been effective and have the desired effect without or with minimal side effects. This proposed process is shown in Figure 3.

Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

Figure 3: Connected Protection and Supervision Model

IV. Design of Smart WS Security Architecture

IV.1. Overview

SmartWSSEC is a self-adaptive system for dynamically controlling the security of a Web services based systems. Its main features are:

- A dynamic and intelligent protection against new threats that can affect an information-based Web service system through an intelligent engine (E) to adapt and enhance the requirements of the security policy for a service provider (SP) and deploy it to clients / partners dynamically.
- Learning capability over the time to capitalize on the previous attacks to provide new answers to non-unknown threats.
- Isolation faculty that is activated in case of inability to find and derive an optimal solution. The isolation is fired after qualifying the severity of the attack and avoid false alert. This could reduce the scope of damages for an inevitable attack.

IV.2. Description

The SmartWSSEC Architecture introduces the concept of adaptive security providing continuous protection over the time. Indeed, conventional security mechanisms offer sometimes late threat detection and a relatively long delay to produce response. SmartWSSEC is based on the fact that instead of trying to prevent any attack, we propose instead to implement an adaptive model that can respond with new protective measures following confirmation of an attack on the Web service. We propose a mechanism with a dual goal: to quickly detect and categorize attacks, and respond automatically in a fast and efficient manner. This can be achieved using:

- An automated intrinsic response mechanism (E) facing the vulnerabilities of a Service Web application not necessarily known before.
- The learning ability that leverages the knowledge and experience of the system over time automatically

Indeed, the system autonomously strengthens requirements (without human intervention). The system also stores historical traces for all events on the architecture (K) to use it for learning and prediction of new solutions to new threats (E). The added value of this model is the ability to adapt the security policy of the web service to protect autonomously and dynamically, following the identification of a new unknown threat before. In others words, the proposed system is based on self-adaptive and intelligent engine, exceeding the static parametric protection through the possibility of autonomous decision-making, to enhance the level of Security policy requirements running on the web service. The Security policies are expressed (but not only) through the standard WS-Policy and WS-SecurityPolicy which express requirements on messages exchanged between services. It is used to specify which accreditations should accompany the message, and what data protection mechanism must be implemented in messages.

IV.3. Components

Our system is composed of three main modules. We distinguish a manager to interface with scanners and intrusion detectors called "sensor module". A "smart reaction module" that allows to interpret the alerts generated through the interfaces and to involve the corresponding treatments. A "Knowledge module" which includes knowledge database. The «smart reaction module" includes a learning unit (Ea) able to deduce and automatically inject new solutions, an update unit (Eu) which allows to implement the recommendations of the solution by updating the security policy dynamically and finally a qualification and reputation management unit (Q) responsible of avoiding blocking the Web Services for attacks that their severity is not justified :

- Proactive detection (D): The system compute the similarities by probabilistic analysis of detected threats vs attacks already categorized in the knowledge base.
- Knowledge Base (K): A core component of knowledge that concentrates the description of threats but also integrates solutions. This is a warehouse to store, process and update a database of vulnerabilities and possible solutions.

Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

- Reaction engine (E): Provides the ability to choose the best measure to face the detected threat. Based on its intrinsic rules engine and the available information on the knowledge base component (K). In the absence of corrective measures, the system must start a Contention process of the attack in order to limit the damage and after consulting the qualification unit (Q)

The figure 4 below shows the SmartWSSEC Architecture and its components:

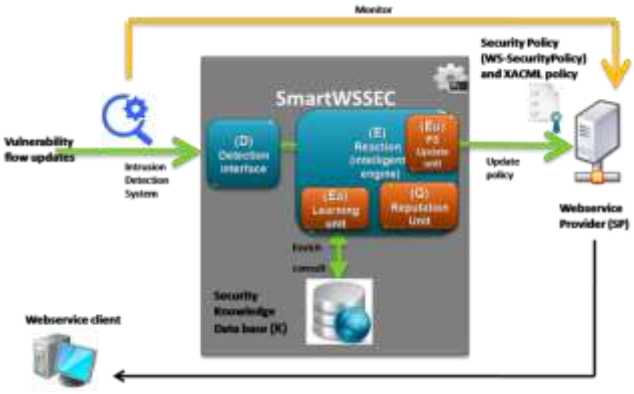


Figure 4 : Components of the adaptive control system dynamic application security Web services-based systems – SmartWSSEC

IV.4. Algorithm

The system initiates the detection module (D) which regularly monitors the system to protect and notifies the intelligent module (E) in case of suspect behavior. When a threat is detected, the intelligent module (E) consults the data base of knowledge (K) to check the existence or not of a possible protection solution. If it is the case, it will apply through the update unit (Eu) the new recommendation in the current security policy dynamically and launches then, the process of its redeployment. Otherwise, the intelligent engine (E) requires the learning unit and reasoning (Ea) to propose a feasible and optimal solution to the unknown threat by transmitting its description as well as the whole context and indicators that surround it. The (k) module will, based on the knowledge of the history and experience of previous attacks, try to categorize this threat and to derive an adequate solution. If the result of this unit is positive (New solution), it is inserted in the knowledge base for later use, and then transmitted to the unit (Eu) to update dynamically security policy accordingly. If the unit (Ea) fails to produce a solution to this new threat, the engine (E) will consult the qualification and reputation management unit (Q) to determine the degree of danger presented by this threat and if he deserves to isolate the system to limit any damage. This mechanism helps guard against the deterioration of the productivity of the system in case of false alerts. If the isolation process is fired by the engine (E), a notification will be also sent to a system administrator.

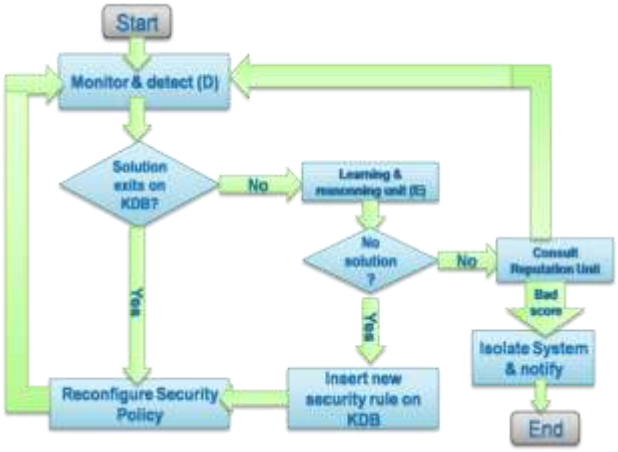


Figure 5: operation algorithm of adaptive control system dynamic application security Web services-based systems

SmartWSSEC works by classifying attacks (once detected) by categories in a tree in order to identify the shortest path to the field of possible solution, then this solution resulted (wherever possible) to integrate rules in the current security policy as an update.

IV.5. Learning faculty

The SmartWSSEC Architecture is adaptive and intelligent. This requires the ability to take into account the evolution of the information for which behaviors in response has been validated, the so-called learning. This allows improving the self-analysis and response feature. This of course is based on learning algorithms that can be categorized according to the learning style they implement. We have adopted the "supervised learning" family [10], because it suits the nature of our architecture and also because we can apply a training to the system through the already resolved attacks.
Indeed, in the supervised algorithm, classes are predetermined (family threats) and also known examples (previous vulnerabilities), the system learns to classify using a pattern classification corresponding to supervised learning (or discriminated analysis). An expert must first inject examples. The process happens in two phases. During the first phase (off-line, referred to as learning), it comes to determining a model of the tagged data. The second phase (online, called test) is to predict the label of a new data, knowing the previously learned model (solution to a new attack).

In our case, the detected attacks or vulnerabilities are categorized through the descriptions and rules already present in the knowledge base to assign a weight for belonging to a class of attacks in which we have an idea about the field of possible solution. In the case where the classification does not lead to a satisfactory result, the system chooses for safety and triggers the isolation

Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

mechanisms necessary to limit in a first response the damage of the attack / vulnerability identified (see Figure 4).

IV.6. Integration with the WS Security Stack

The SMARTWSSEC architecture acts, manages and maintains security policies, which are part of the Quality of Service of the web service. It is materialized by the WS-SecuriyPolicy [2] standard which itself is referenced by the WSDL (web services description Language). The "Reaction engine" component applies updates in the WSDL. And new requests will be based on the new WSDL and must comply with these new security requirements (ex: token type, encryption).

V. Prototyping and simulation

After introducing SmartWSSec, in order to validate our architecture and prove what claims. We had developed a prototype of our proposed security model and evaluated it on real uses cases of Web services and against two types of security attacks that were identified and mitigated. The prototype, which role is to prove the feasibility of the concept, is based on a lightweight implementation using some opensource tools that we combined and configured.

V.1. Uses case

The validation of the prototype is also based on the production of significant attacks, demonstrating the effectiveness of our solution. We also start from a minimum security policy that must be enforced as our Web service to protect is under attack. We chose to validate the architecture facing the two most dangerous attacks from OWASP top ten [29], namely "injection" and "Session Management":

- SQL injections: Is always one of the most common ways to attack Web services. The idea is to send SQL malicious code into a parameter field, hoping that the server will execute the code. It's an alteration attacks which can be faced using digital signature.
- Replay attack: This is to intercept and replay a valid message several times without authorization. One solution to this is the integration of timestamps (timestamp).

V.2. Prototype implementation

Our prototype is composed of a simple Web service that was developed using the apache CXF Framework [26]. The webservice is hosted on a tomcat application server behind an apache web server.

Bellow, the infrastructure we used to implement SmartWSSEC component. It contains the Web service server (Centos) tomcat + apache on a server, a server simulating the attacker (hacker), a server for detection (Mod Security ubuntu) and finally a server (centos) for SmartWSSecurity

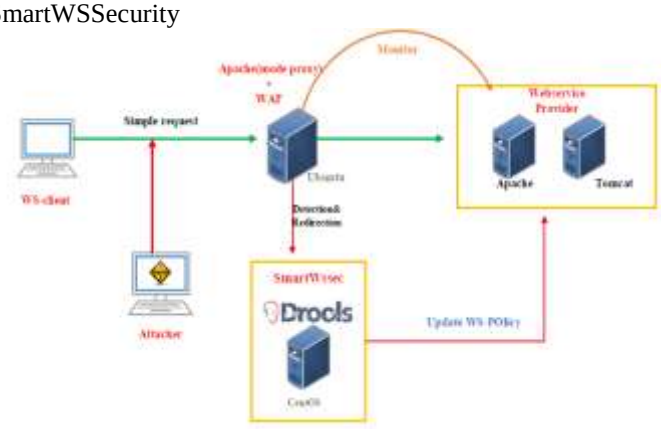
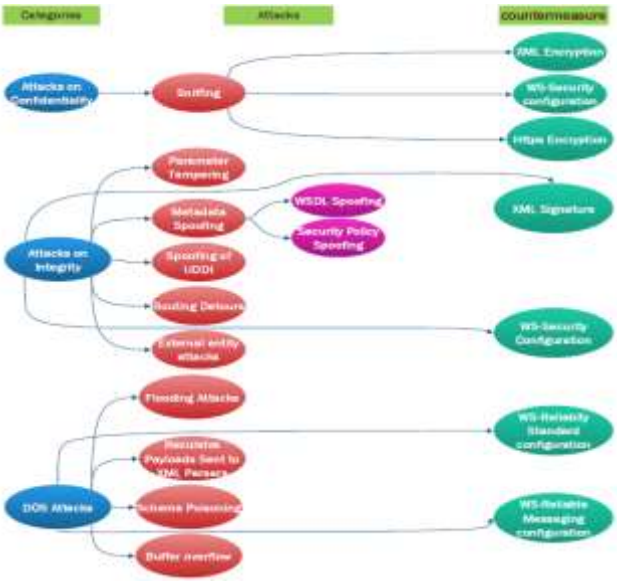


Figure 6: Infrastructure of the POC

V.2.1. Rule engine & Knowledge modeling

SmartWSSEC works by classifying attacks (once detected) regarding their closes families. This is done using a tree (figure 7) in order to identify the shortest path to the field of possible solution, then this solution (wherever possible) is translated in rules to be integrate into the current security policy as an update. As seen before, we are using supervised learning to predict new solution, below an example of our Decision trees that we injected on the rule engine. This hierarchical nature between the parent and children nodes will construct the rules that we need for our system:



Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

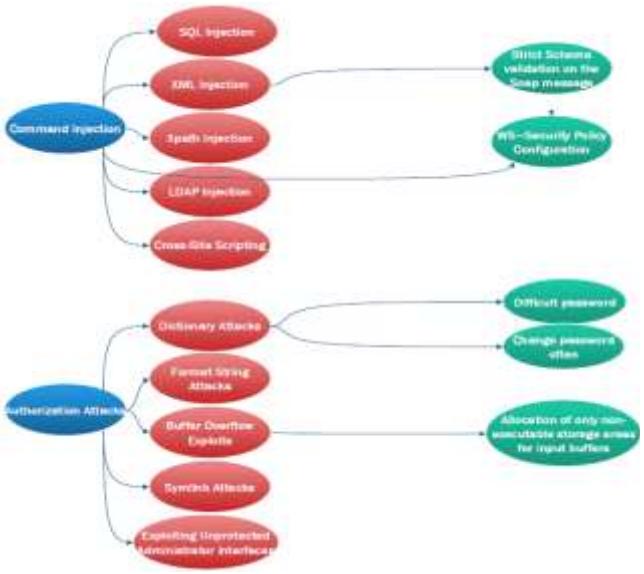


Figure 7: WS Threats and countermeasures knowledge modeling in SmartWSSEC

This part was implemented by JBoss Drools [28]. Drools is a business rule management system (BRMS). It is written in Java and extends and implements the model Rete matching algorithm. Bellow some examples of the rules we introduce in:

Rule 1: R1 if impersonation attempt then move to X509 certificate authentication
Fact1: F1 if 5 wrong attempts of password input in 30 seconds while identity theft attempt

```
User.java | UsurpationIdentité.java | AuthRule.drl
1  package authentification ;
2  dialect "mvel"
3  rule "usurpation identite"
4  when
5  User ( nbfailedAuth > 5 , authtime < 30)
6  then
7  System.out.println("WS-Policy Updated"); // consequer
8  end
9
10
```

Figure 8: Sample expression of a security rule (DRL language)

With Drools we have rules on one side and the working memory of the other side. The application code is responsible for loading the appropriate facts in working memory and launch appropriate rules.

V.3. *Proof of concept*

The initial security policy in place for our WS only requires a username and a valid password, which must be integrated on the header of each request:

```
<wsp:Policy
   xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/
policy"
   xmlns:wssp="http://www.bea.com/wls90/security/pol
icy"
   xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-
     wssecurity-utility-1.0.xsd"
   xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-
     wssecurity-secext-1.0.xsd"
   xmlns:m="http://example.org"
   wsu:Id="encrypt-custom-body-element-and-username-
token">

    <!-- Require messages to provide a user name and
password                                    token
        for          authentication          -->
    <wssp:Identity>
      <wssp:SupportedTokens>
        <wssp:SecurityToken   IncludeInMessage="true"
          TokenType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-
            wss-username-token-profile-
1.0#UsernameToken">
          <wssp:UsePassword
Type="http://docs.oasis-open.org/wss/2004/01/
            oasis-200401-wss-username-token-
profile-1.0#PasswordText"/>
        </wssp:SecurityToken>
      </wssp:SupportedTokens>
    </wssp:Identity>
```

We will see that such a policy does not provide a sufficient level of protection for the attacks as replays or injection and how SmartWSSEC can enforce the update of this policy with the appropriate rules.

V.3.1. *Attacks production*

To simulate the selected attacks, we will use the SoapUI tool which will create a query and add a token user name for the message. This test returned a result "failed" to hack this connection mechanism we will inject SQL code to bypass the authentication constraint present in the initial security policy:

Figure 9 : Attack production using SOAP UI

V.3.2. *Attacks detection*

We have chosen ModSecurity [27] which is one of the most popular and widely used firewall for web application security. ModSecurity is in the form of a module for the Apache Web server (httpd). The role of ModSecurity is to protect Web server application attacks by filtering upstream queries.

In our test, Modsecurity, have detected the malicious request but didn't block it because it is configured in log-only mode (transmission of the alert):

```
Message: Warning. Pattern match "(?i:\\bor\\b.?(?:\\d{1,10}|[\\'\"][^=]{1,10}[\
\'\"]).?[=><]+|(?i:'\\sx?or\\s+.{1,20}[+\\-!><=]|\\b(?i:x?or)\\b\\s+(\\d
{1,10}|'[^=][1,10}'|(\\b(?i:x?or)\\b\\s+(\\d{1,10}|'[^=][1,10}'|\\s*?[=><])"
at ARGS:username. [file "/usr/share/modsecurity-crs/activated_rules/
modsecurity_crs_41_sql_injection_attacks.conf"] [line "133"] [id "959071"]
[rev "2"] [msg "SQL Injection Attack"] [data "Matched Data: ' or true -- found
within ARGS:username: ' or true -- "] [severity "CRITICAL"] [ver
"OWASP_CRS/2.2.9"] [maturity "9"] [accuracy "8"] [tag "OWASP_CRS/WEB_ATTACK/
SQL_INJECTION"] [tag "WASCTC/WASC-19"] [tag "OWASP_TOP_10/A1"] [tag
"OWASP_AppSensor/CIE1"] [tag "PCI/6.5.2"]
```

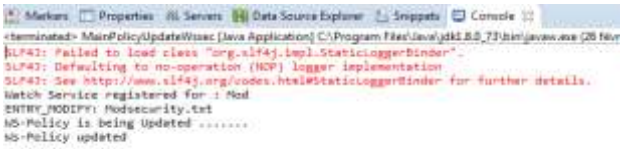
Proc. of The Fifth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2016

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

V.3.3. SmartWSSEC reaction

For this prototype, we used the WAF Mod Security to instantiate the detection part of our architecture. Mod Security uses a specific log file to trace all questionable behavior detected. So we implemented a listener / reader to monitor the log file and fire the alert informing us of an attack on our service.

This module is connected with the rule engine Drools by a rule which is triggered if we identify a keyword (like SQL injection alert) on the ModSecurity log file. This fact therefore triggers the rules to start the research process/generation of a consistent solution to the attack and identifying updates to apply on Web service security policy. SmartWSSEC fires the update task on WS-Policy of the Webservice, to add our new security requirements. The result of the execution of the facts (receiving an alert from detection module):



The WS-Policy file with our Security Policy was updated

V.3.4. Results

The attack on our web service can be countered by the digital signature requirement, to ensure the identity of the customer and protect the message from tampering during transportation. Also, it is necessary that the message can include a message timestamp determining the validity time to counter attack replays (Replay Attack). Following the launch of the attack, we can see its detection and response produced by our engine. The new security policy of our web service had been updated (in green):

```
<wsp:Policy
   xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/
policy"
   xmlns:wssp="http://www.bea.com/wls90/security/pol
icy"
   xmlns:m="http://example.org"
   wsu:Id="encrypt-custom-body-element-and-username-
token">
    <!-- Require messages to provide a user name and
password token
         for authentication -->
   <wssp:Identity>
      <wssp:SupportedTokens>
        <wssp:SecurityToken IncludeInMessage="true"
            TokenType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-
               wss-username-token-profile-
1.0#UsernameToken">
           <wssp:UsePassword
Type="http://docs.oasis-open.org/wss/2004/01/
              oasis-200401-wss-username-token-
profile-1.0#PasswordText"/>
        </wssp:SecurityToken>
      </wssp:SupportedTokens>
   </wssp:Identity>

 <wssp:Integrity>
     <wssp:SignatureAlgorithm
       URI="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"/>
     <wssp:CanonicalizationAlgorithm
       URI="http://www.w3.org/2001/10/xml-exc-
c14n#"/>

    <!-- Require the Timestamp header to be signed
-->
     <wssp:Target>
       <wssp:DigestAlgorithm
URI="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <wssp:MessageParts
          Dialect="http://www.bea.com/wls90/security
/policy/wsee#part">
          wls:SecurityHeader(wsu:Timestamp)
        </wssp:MessageParts>
     </wssp:Target>

    <!-- Require the message body to be signed -->
     <wssp:Target>
       <wssp:DigestAlgorithm
URI="http://www.w3.org/2000/09/xmldsig#sha1"/>
       <wssp:MessageParts
         Dialect="http://schemas.xmlsoap.org/2002/12
/wsse#part">
         wsp:Body()
       </wssp:MessageParts>
    </wssp:Target>
  </wssp:Integrity>
<wssp:MessageAge/>
<wssp:IncludeTimestamp />

</wsp:Policy>
```

Figure 10: Overview of the updated WS-Policy file

The updated Security policy generated by SmartSSEC, contains now the appropriate rules to face the detected attacks. Of course, it cannot be the case all the time, as the accuracy of the system increases with time and experience of the thwarted attacks and also regarding the richness of the security knowledge database we use. We will present in a future paper, the study we made to indentify the most suitable learning algorithm for our system.

VI. CONCLUSION AND FUTURE WORK

This paper constitutes our contribution in the long way of resolving problems of adaptability of systems from the security side. It introduces a framework for implementing smart and automatic protection systems for websevices-based applications. However, several improvements and optimizations remain possible to converge towards an industrialized product. The concept was patented on 2015 [25], further work must follow to provide solutions to current limitations of the prototype, such as handling of undesirable effects on disruptive reactions that can be the solution for a new attack but it may require a restart for example, which means low productivity, the creation of a dedicated and efficient learning algorithm that can improve performance and efficiency of the system, modeling internal attacks, because they can be more dangerous than external ones, integrating Offline learning feature, using logs and audit trail and finally improving the generic behavior of the framework for wide adoption for other technologies than Web services.

ISBN: 978-1-63248-104-7 doi: 10.15224/ 978-1-63248-104-7-18

References

[1] Mike P. Papazoglou, Web Service: Prinriples and Technology. Pearson — Hall, 2007.

[2] OASIS Web Services Secure Exchange TC. WS-SecurityPolicy 1.3. OASIS Standard, 2 February 2009. Available at http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/ws-securitypolicy.pdf.https://nvd.nist.gov/ The National Vulnerability Database (access date 2015)

[3] N. M. Belaramani, «A component-based software system with functionality adaptation for mobile computing » thesis, The University of Hong Kong, 2002

[4] Oreizy, P., Gorlick, M.M., Taylor, R.N., Heimbigner, D., Johnson, G., Medvidovic, N., Quilici, A., Rosenblum, D.S., et Wolf, A.L. « An Architecture-Based Approach to Self-Adaptive Software », IEEE Intelligent Systems, May/June 1999, pp. 54-62.

[5] N. Subramanian, and L. Chung, « Software Architecture Adaptability: An NFR Approach, » Dans IEEE Proceedings of International Workshop on Principles of Software Evolution, 2001.

[6] M. CHEAITO, «Un cadre de spécification et de déploiment de politiques d'autorisation ». Phd's thesis, Institut de Recherche en Informatique de Toulouse (IRIT), Mars 2012, pp. 57-58

[7] Frédéric Cuppens, Sécurité autonome et adaptative : quelles politiques ? 3ème colloque de l'Institut Mines-Télécom : Numérique : Grande échelle & Complexité, mars 2014

[8] Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. Proc. 22nd International Conference on Machine Learning (ICML'05).

[9] Arnaud CONTES, « Une architecture de sécurité hiérarchique, adaptable et dynamique pour la grille », UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS, Septembre 2005.

[10] FireEye, La ligne Maginot de la cybersécurité : une évaluation à l'épreuve des faits du modèle de défense en profondeur, mai 2014.

[11] R. Tiako, Student Member, IEEE, D. Jayaweera and S. Islam, Senior Members, IEEE;"A Case-Based Reasoning Approach for Dynamic Security Assessment of Power Systems with Large Penetration of Wind Power";Curtin University, Perth; 2013

[12] Gouenou Coatrieux Telecom Bretagne - LaTIM Inserm U650 Michel Embe Jiague LACL Stephane Morucci SWID;"Implementation of Web Services for Security Enforcement";AGENCE NATIONAL DE LA RECHERCHE;Decembre 2011

[13] Nils Gruschka, Ralph Herkenh¨oner and Norbert Luttenberger;"WS-SecurityPolicy Decision and Enforcement for Web Service Firewalls";Christian-Albrechts-University in Kiel, Germany, 2006

[14] Master of Science Thesis Stockholm, Sweden, 2007 ,"An approach for securing web service communication" Faisal Abdul Kadir

[15] "A Survey of Attacks on Web Services Classication and Countermeasures" ,Meiko Jensen Nils Gruschka Ralph Herkenhoner, 2009

[16] "Guide to Secure Web Services" Recommendations of the National Institute of Standards and Technology ;Anoop Singhal Theodore Winograd Karen Scarfone, 2007

[17] Zhiwen Bai,Liming Wang, Jinglin Chen,Jain Liu,Xiyang Liu on " DTAD A Dynamic Taint Analysis Detector for Information Security",IEEE, Web age Information system 2008, pp,591-597

[18] Fang Qi, Zhe Tang, Guojun Wang on" Attacks vs. Countermeasures of SSL Protected Trust Model", IEEE conference 2008, pp1886-1991

[19] Enabling Dynamic Security Policy Evaluation for Service-Oriented Architectures in Tactical Networks, Vasileios Gkioulos and Stephen D. Wolthusen, Norwegian Information Security Conference 2015 (NISK-2015)

[20] Context aware intrusion response based on argumentation logic, Tarek Bouyahia, Fabien Autrel, Nora Cuppens-Boulahia, Fr_ed_eric Cuppens, T_el_ecom-Bretagne, 35576 Cesson S_evign_e (France), 2015

[21] Herve Debar, Yohann Thomas, Frederic Cuppens, Nora Cuppens-Boulahia. Enabling automated threat response through the use of a dynamic security policy. Journal in Computer Virology (JCV), 2007, 3 (3), pp.195-210.

[22] Jose-Miguel Horcas1, Mónica Pinto1, Lidia Fuentes1, Wissam Mallouli2, and Edgardo Montes de Oca2, An Approach for Deploying and Monitoring Dynamic Security Policies, CAOSD Group, Universidad de Málaga, Andalucía Tech, Spain, 2015

[23] Tarek Bouyahia, Muhammad Sabir Idrees, Nora Cuppens-Boulahia, Fr_ed_eric Cuppens, Fabien Autrel. Metric for Security Activities assisted by Argumentative Logic. SETOP 2014: the 7th International Workshop on Autonomous and Spontaneous Security, Sep 2014, Wroclaw, Poland. 8872 - LNCS (Lecture Notes in Computer Science), pp.183 - 197, 2014,

[24] Patent N°38593 OMPIC

[25] https://cxf.apache.org/ (access date 2015)
https://www.modsecurity.org/ (access date 2015)

[26] http://www.drools.org/ (access date 2015)

[27] https://www.owasp.org/index.php/Top_10_2013-Top_10 (access date 2015)