A service oriented communication model for high speed intrusion detection systems

Mohsen Rouached*

College of Computers and Information Technology,
Taif University,
P.O. Box 888, 21974, Al-Hawiya-Taif, Saudi Arabia
E-mail: m.rouached@tu.edu.sa
*Corresponding author

Hassen Sallay

Information Security Department,
Al Imam Mohammad Ibn Saud Islamic University,
P.O. Box 5701, 11432, Riyadh, Saudi Arabia
E-mail: hmsallay@imamu.edu.sa

Abstract: The growing need for information sharing among different networks poses a great security challenge. One of the key aspects of this challenge is deploying intrusion detection systems (IDSs) that can operate in heterogeneous and large scale environments. This is particularly difficult because the majority of existing IDSs are not designed to work in a cooperative fashion. The integration becomes more difficult when we should reduce computing and memory costs incurred by the high speed IDSs communication. Service oriented architecture (SOA) is one of the key paradigms that enables the deployment of services at large-scale over the internet domain and its integration with IDSs may open new pathways for novel applications and research. Characteristics such as platform transparency and loose coupling make the web services technology a good choice for IDS integration. In this context, this paper presents a lightweight RESTful communication model for coordinating different entities of a high speed distributed IDS.

Keywords: high speed networks; HSN; intrusion; detection systems; service oriented architecture; SOA; web services; representational state transfer; REST.

Reference to this paper should be made as follows: Rouached, M. and Sallay, H. (2014) 'A service oriented communication model for high speed intrusion detection systems', *Int. J. Business Information Systems*, Vol. 17, No. 3, pp.323–339.

Biographical notes: Mohsen Rouached is currently acting as an Assistant Professor in the College of Computers and Information Technology at the Taif University. He received his MS and PhD in Computer Science from Nancy University in 2005 and 2008 respectively. His research interests span over several areas related to service oriented computing, business processes, security, privacy, and forensics management, services semantics, and wireless sensors networks. He has published over 40 research papers in these domains. He serves as program committee member and reviewer at many international journals and conferences and has been participating in several research projects.

Hassen Sallay received his PhD in Computer Science from Nancy University in 2004. He is an Assistant Professor in College of Computer Science and Information Systems, and a member of the Scientific and Technology Unit at Imam Mohammad Ibn Saud Islamic University. He is the leader of the Amansystem group focusing on excellence in building security technical intelligence to support academic institutions and professional bodies. His research interests are mainly in computer and network security management, digital forensics and privacy. He has conducted and participated in several research projects concerning these fields. He has published several research papers at international journals, conference proceedings as well as chapters of books.

1 Introduction

The internet and computer networks are exposed to an increasing number of security threats. With new types of attacks appearing continually, developing flexible and adaptive security oriented approaches is a severe challenge.

One of the primary approaches to the increasingly important problem of computer security is the intrusion detection system (IDS). IDSs are software or hardware systems that automate the process of monitoring and analysing the events that occur in a computer network, to detect malicious activity. Since the severity of attacks occurring in the network has increased drastically, IDSs have become a necessary addition to security infrastructure of most organisations. Today, intrusion detection is one of the high priority and challenging tasks for network administrators and security professionals.

Detecting intrusions in networks and applications has become one of the most critical tasks to prevent their misuse by attackers. Present networks provide critical services which are necessary for businesses to perform optimally and are, thus, a target of attacks which aim to bring down the services provided by the network. The cost involved in protecting these valuable resources is often negligible when compared with the actual cost of a successful intrusion, which strengthens the need to develop more powerful IDSs. Moreover, the complexity of the information, the computer systems, and the attacks are being more sophisticated, unpredictable, frequent and from a wider range of sources are exceeding current IDS ability. The problem becomes more serious with the emergence of high-speed networks like InfiniBand and Gigabit-Ethernet. One of the challenges is to keep up with the ever increasing internet usage and network link speeds, as more and more data has to be scanned for intrusions. Another challenge is that it is hardly feasible to adapt the scanning configuration to new threats manually in a timely fashion, because of the possible rapid spread of new threats.

Research on intrusion detection in distributed systems is currently focusing on two essential issues: scalability and heterogeneity. The IDSs in large distributed systems need to be scalable to accommodate the large amount of audit data in such systems. In addition, such IDSs must be able to deal with heterogeneous information from component systems of different types and that constitutes large distributed systems and can cooperate with other types of IDSs.

Service oriented architecture (SOA) is one of the key paradigms that enables the deployment of services at large-scale over the internet domain and its integration with IDSs may open new pathways for novel applications and research. The main idea of SOA

(Thomas, 2005) is to treat applications, systems and processes as encapsulated components, which are called services. These services are represented by input and output parameters and the semantic description of their functionalities. In fact, SOA is one of the core mechanisms for service deployment in the internet that has been adapted for making easier and more effective service deployment. It possesses an architectural style encompassing a set of services for building complex systems from existing components. As an architectural evolution and a paradigm shift in systems integration, SOA enables the discovery, access and sharing of the services, data, computational and communication resources in the network for multiple users. It also allows rapid and cost-effective composition of interoperable and scalable systems based on reusable services exposed by these systems. SOA inherently supports two major requirements: heterogeneous infrastructures and run-time adaptability, which are essential for large-scale and distributed IDSs in which multiple sensors run over diverse platforms and adopt different technologies.

When it comes to the implementation of web services, REST and Simple Object Access Protocol (SOAP) are the two major building blocks that define the usage and structure of different web services used today. SOAP consists of a set of functions that can be used to exchange information when implementing different web services. SOAP uses the XML as its main lingua franca. On the other hand, REST is a definition of an architecture or hierarchy for the design model of web services. In the recent years, REST has emerged as the pioneer in defining web services because of the simplicity in its design and ease of use. REST-based APIs offer innovation, ease and simplicity to programmers and users.

In this paper, we tackle integration of SOA with DIDs and propose a lightweight RESTful Communication model for coordinating different entities of a high speed distributed IDS.

The remainder of the paper is structured as follows. Section 2 exposes the background of the work to be presented. In Section 3, we present the global architecture and the different components of the developed system. Section 4 focuses on the details of the RESTful approach. Section 5 is dedicated to the performance studies. Finally, Section 6 concludes the paper and outlines some future directions.

2 Related work

Recently, new approaches and solutions addressing the huge amounts of transferred network data and increasing speeds of today networks were proposed (Yu et al., 2005; Foschini et al., 2008; Liu et al., 2010; Subbulakshmi et al., 2010; Su et al., 2009; Treinen, 2006; Wenbao and Shuang, 2006; Sekar et al., 1999). In this section, we focus on the use of SOA and web services for IDS integration. The literature on applying the service paradigm to intrusion detection is scarce and very few efforts were proposed. In what follows, we discuss the most important proposals.

The purpose of the paper (Bosin et al., 2008) is to describe an approach for overcoming limits of available solutions to intrusion detection problems. As first step, the authors propose to engineer ID processes as a set of services involving distributed tasks on networked resources. Specifically, they show how to model the ID processes as a set of plans that a security manager may go through on a network of cooperative nodes

interacting with one another in order to offer or to ask for services. Services correspond to specialised ID tasks and encapsulate problem solving and simulation capabilities. Complex ID activities are expressed by workflows. The work presented in Bosin et al. (2004) goes further in this direction by mapping ID tasks into reusable and customisable ID services which, in turn, can be composed to deploy a new generation of IDSs that can operate in heterogeneous and large scale environments, like grids. Some proposals can be found in Brandao et al. (2006b, 2006a), Rao et al. (2009), and Park et al. (2003), where the web service technology provides the basis for integrating existing intrusion detection elements and systems, without no general model for reengineering the ID processes according to a SOA approach. However, no infrastructure to implementing such a model was provided.

IDS composition using web services was introduced in Brandao et al. (2006a). For the paper (Brandao et al., 2006b) the infrastructure has been expanded, with a new format compatibilisation layer that enables the use of commercial off-the-shelf (COTS) IDS elements. Also, this paper introduces and discusses in detail several aspects of the use of service orchestration for building dynamic compositions, a key functionality that was missing from in Brandao et al. (2006a). Brandao et al. (2006b) presents a model for integrating IDSs in heterogeneous, large-scale environments. The main idea is to build compositions of IDSs that work as unified systems, using a SOA based on the web services technology. The necessary interoperability among the elements of the compositions is achieved through the use of standardised specifications, mainly those developed by IETF, W3C and OASIS. Dynamic compositions are supported through service orchestration. However, they do not perform any quantitative assessment of the proposed infrastructure, including its operational and computational costs, and the security service needs more refinement.

In Mauro et al. (2006), the authors present a model, an architecture and an implementation of a remote IDS using the technology of multi-agent systems, web services and model-driven architecture (MDA) was presented. This model adapts and extends the Network Intrusion Detection System (NIDIS based on intelligent agents) to provide a remote IDS on the internet. The purpose is that users that do not have a local IDS can use the services provided by the remote IDS. NIDIA is an IDS whose architecture consists of a set of cooperative agents. The IDS functionalities are provided as a set of accessible services on the internet through web services. The architecture uses MDA to support metadata management such as profiles of machines, profiles of users and profiles of services.

In Grzegorz and Agnieszka (2012), the authors propose an SOA-based framework for a distributed anomaly detection system and detail its functionality. According to the SOA paradigm, they treat a security as a service that is delivered on different levels of granulation. They introduced the procedure for behavioural pattern extraction. The patterns of users, services and system behaviour are then used by the agents to discover any anomalies in a network system. The functionalities related to the determination of the profiles of the system usage are delivered by the web services. However, the proposed framework was not tested and implemented and remains at the proposal level.

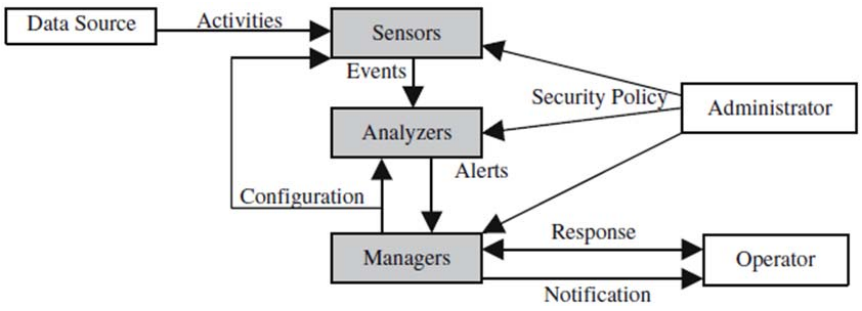
Web services-based IDS that uses the IDWG model is described in Park et al. (2003). Unfortunately, the model uses only one method of detection and it centralises the analysis. Although it uses the IDMEF, originally, it is not possible to integrate it with other IDSs. Our framework may be used to make this integration.

A common pattern of the above approaches is that they are using SOAP-based web services. However, with the goal of attracting a larger user community, more and more service providers are switching to REST in order to make it easy for clients to consume their web service APIs. This emerging technology advocates a return to the original design principles of the World Wide Web to provide for the necessary interoperability and enable integration between heterogeneous distributed systems. Thus, in this paper we apply the REST to define extensible and lightweight interfaces for controlling and monitoring the operations and messages exchanged between the different entities of our distributed IDS model.

3 Architecture overview

Recent standardisation efforts related to the exchange of security information are being developed mainly by the IETF through its IDWG and INCH working groups. IDWG is finishing up the intrusion detection message exchange format (IDMEF) (Debar et al., 2006) and Intrusion Detection Exchange Protocol (IDXP) (Feinstein et al., 2002) specifications. These efforts aim at the exchange of information among complete IDSs and IDS elements. The INCH group is working on the exchange of information and statistics about security incidents among incident response teams (CSIRTs). The requirements and the data model (IODEF – Incident Object Description Exchange Format) (Danyliw et al., 2006) for implementation are still in specification phase. All these specifications are based on XML.

Figure 1 Basic elements of the IETF intrusion detection model



The IDWG also defines a general model for intrusion detection (Wood and Erlinger, 2002), as illustrated in Figure 1. This taxonomy is adopted in our infrastructure. A sensor is an element that collects data from one or more data sources. The sensor is setup to forward events to the analyzer. An analyzer inspects data collected by a sensor looking for signs of unauthorised or undesired activity or for events that might be of interest to the security administrator. A sensor and an analyzer can be part of the same component. A manager manages the various elements of an IDS. Management functions typically include (but are not limited to) sensor configuration, analyzer configuration, event notification management, data consolidation, and reporting. IDS elements can be parts of a monolithic IDS or can be distributed on more than one location. In our work, we envision a new generation of IDSs defined by a set of services supporting security managers in improving the overall network security. Implemented algorithms and

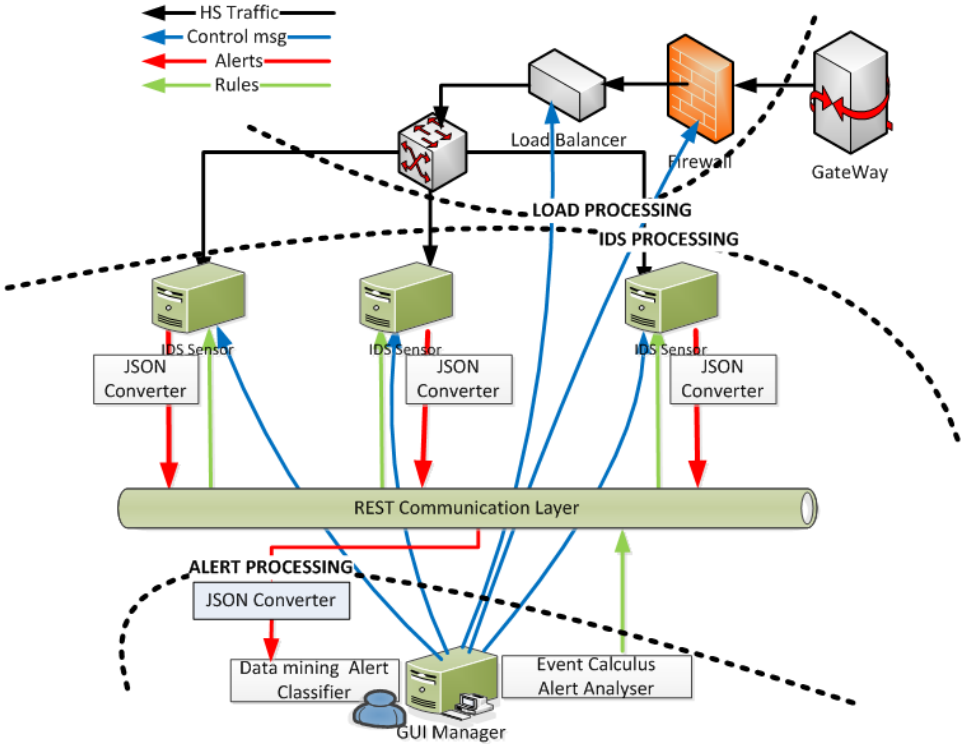
methods may be very easily exchanged according to the characteristics of the monitored network and requirements with respect to the level of security that must be maintained. These services can be realised in two ways:

1 SOAP-based services

2 RESTful services.

For the service providers, RESTful services can improve system flexibility, scalability, and performance as compared to the SOAP-based web services. It is equally attractive to end users as it is consume less resources (i.e., battery, processor speed, and memory). Additionally, REST-based services do not include complex standards and heterogeneous operations; and hence are easier to consume and compose as compared to SOAP-based web services. Thus, in our approach, communications between managers, sensors, and analyzers follow the REST architectural style and use JSON data format. Figure 2 depicts our global IDS architecture.

Figure 2 Global architecture (see online version for colours)



As shown in Figure 2, the major objective is to design and develop an efficient real time NIDS for high speed networks (HSN). We believe that this could be satisfied through the fulfilment of four main tasks:

1 Design and development of an underlying scalable and adaptive parallel and distributed IDS architecture for high speed network.

2 Design and development of algorithms and techniques improving the accuracy of NIDS alerts generation and correlation and real-time malicious attack detection by minimising the false alerts.

3 Design and development of an efficient and integrated management platform coupling these aforementioned algorithms and techniques to the underlying architecture.

4 Testing and performance study of the system and simulation for large scale scenarios. In this context, we developed an integrated IDS framework (Sallay, 2011).

The framework has four research themes pillars. The first one is modelling the brain theme which is targeting to improve the accuracy of detection and attack alert prioritisation (Ammar and Sallay, 2011). The second one is making the mind which aims to add some intelligence to the system by introducing a type of reasoning on the alert logs to discover new attack scenarios and therefore make the detection process more efficient The architectural theme targets to design distributed architectures performing adaptive traffic load balancing algorithms and splitting schemes to take over the HSN bottleneck caused by the IDS scanning tasks inside the network (Sallay et al., 2009). Finally, the management theme tends to manage efficiently the overall self-defence process (Sallay et al., 2011). The work to expose in this paper is situated in the architectural theme and focuses on providing a lightweight RESTful IDS communication model for coordinating high speed IDS entities.

4 Lightweight RESTful IDS communication model

4.1 RESTful communication model

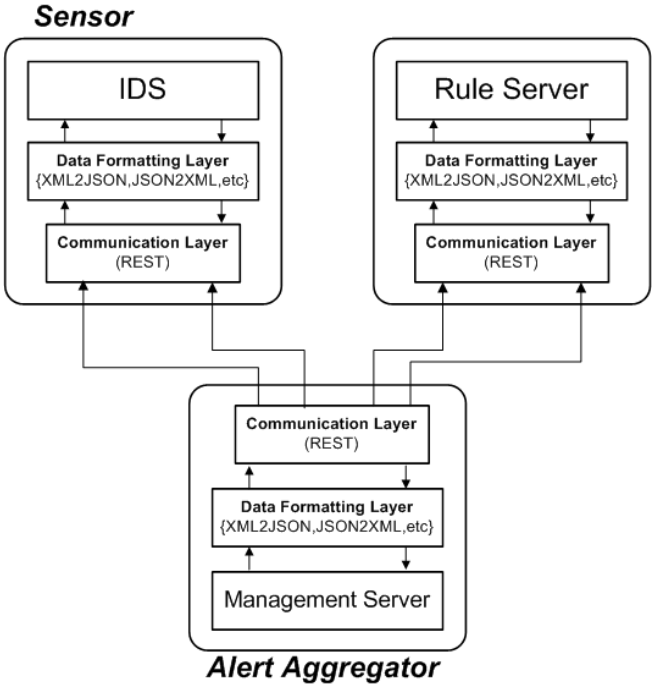
This section mainly focuses on the exchange protocols and communications between the sensors, the alert aggregator, and the rule manager. The description of the load balancing process was described in Sallay et al. (2009), where the authors presented an optimised scalable distributed architecture based on switch-based splitting approach that supports intrusion detection on high-speed links by balancing the traffic load among different sensors running Snort placed in each point of access to the internet. The event calculus analyser is also described in Rouached and Sallay (2012).

For clarity reasons, we will consider just one sensor throughout the paper to illustrate our ideas. The validation process is done by taking into account several sensors from different types. The communication model is depicted in Figure 3.

REST (Thomas, 2000) is an architectural model for how distributed applications are built. REST emphasises scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems. In a REST architecture a resource is an abstraction controlled by the server and identified by a Universal Resource Identifier (URI). The resources are decoupled by the services and therefore resources can be arbitrarily represented by means of various formats, such as XML or JSON. The resources are accessed and manipulated by an application protocol based on client/server request/responses. REST is not tied to a particular application protocol. REST uses standard HTTP methods; i.e., GET is used as a safe and idempotent

operation to access a resource, PUT is an idempotent operation that can be used to create or update a resource with a known URI, DELETE is idempotent as well and used to remove a resource and lastly POST is used for anything else. REST architectures allow IDSs and network applications to be developed on top of web services which can be shared and reused. The sensors become abstract resources identified by URIs, represented with arbitrary formats and manipulated with the same methods as HTTP. As a consequence, RESTful communications may drastically reduce the application development complexity. Indeed, REST interfaces are easy to design and implement: verbs, exception semantics, caching semantics, versioning semantics, and authentication and access control are already defined. It does not require a separate resource discovery mechanism, due to the use of hyperlinks in content.

Figure 3 Communications between sensors, alert aggregator, and rule server



As shown in Figure 3, we proposed to directly implement a RESTful application programming interface (API) on each entity (sensor, rule server, alert aggregator) in order to grant an easy communication between them. Concerning the data exchange format, we use a lightweight and language independent text format, which is the JavaScript Object Notation (JSON). JSON is more compact than XML as it provides an implicit data structure format. JSON does not require any XML parsing on the network nodes. Browsers can consume large amount of JSON much more efficiently than they can consume large amount of XML and the gap is widening because the latest versions of the browsers are now providing native, safe support for encoding and decoding JSON. JSON permits to reduce the messages size and the transmission time as it will be shown in the performance study section. We distinguish between communications from sensors and alerts aggregator, and alert aggregator and rules server.

4.1.1 From sensors to alerts aggregator

The idea is to propose a JSON-based format for security message exchange between sensors and the alert aggregator. The JSON objects are roughly inspired from the IDMEF format. Figure 4 shows an example of an IDMEF message.

Figure 4 IDMEF message

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE IDMEF-Message PUBLIC
"-IETF//DTD RFC XXXX IDMEFv1.0//
EN" "idmef-message.dtd">
<IDMEF-Message version="1.0">
<Alert ident="abc123456789">
<Analyzer analyzerid="hq-dmz-analyzer62">
<Node category="dns">
<location>Headquarters Web Server</location>
<name>analyzer62.example.com</name>
</Node>
</Analyzer>
<CreateTime ntpstamp="0xbc72b2b4.0x00000000">
2000-03-09T15:31:00-08:00
</CreateTime>
<Source ident="abc01"><Node ident="abc01-01">
<Address ident="abc01-02" category="ipv4-addr">
<address>192.0.2.200</address></Address>
</Node>
</Source>
<Target ident="def01"><Node ident="def01-01"
category="dns">
<name>www.example.com</name>
<Address ident="def01-02" category="ipv4-addr">
<address>192.0.2.50</address></Address>
</Node>
<Service ident="def01-03">
<portlist>5-25,37,42,43,53,69-119,123-514
</portlist>
</Service>
</Target>
<Classification origin="vendor-specific">
<name>portscan</name>
<url>http://www.vendor.com/portscan</url>
</Classification>
</Alert>
</IDMEF-Message>
```

This example illustrates a port scan attack alert. The alert was received from the hqdma-analyzer62 located at the headquarters web server. The attack was instituted by abc01 (192.0.2.200). The system being attacked was a DNS server named def01 (192.0.2.50). The list of ports that were scanned is 5–25, 37, 42, 43, 53, 69–119, 123–514. The analyzer has characterised the attack as a port scan attack. The alert is composed by about 1,070 characters. The JSON representation corresponding to this port scan attack is given in Figure 5.

How to automatically obtain this representation from the XML IDMEF source file will be detailed in the validation part of the paper. The gain in terms of data buffer size and transmission will also be measured and analysed.

Figure 5 JSON representation of the IDMEF message

```
{
"alert":{
"Ident":" abc123456789",
"Analyzer":{"id":"hq-dmz-analyzer62",
"Node":{ "category":"DNS","location":"
 Headquarters Web Server",
"name":" analyzer62.example.com")
},
"createTime":(["0xbc72b2b4.0x00000000",
"2000-03-09T15:31:00-08:00"]),
"Source":{
" ident":"abc01",
"Node":("ident"="abc01-01",
"Address":("ident":"abc01-02",
"category":"ipv4-addr","ip":"192.0.2.200"
}})
"Target":{" ident":" def01",
"Node":("ident"=" def01-01",
"category":"dns","Name:"www.example.com",
"Address":("ident":" def01-02",
"category":"ipv4-addr","ip":"192.0.2.50"
}},
"Service":{
" ident":"def01-03",
"portlist":["5-25","37","42","43","53","69-119",
"123-514"]
}},
"Classification":{
"origin":"vendor-specific","name":"portscan",
"url":"http://www.vendor.com/portscan"
}}
```

4.1.2 From alert aggregator to rule server

IDSs are controlled by decision rules (signatures), which contain technical description of intrusion accident and action that must be taken when this intrusion occur. Choosing the suitable IDS, configuration and management is a very hard and responsible task because each of well known system is using their own signatures standard. Because there is no common standard, administrators who know one system must learn other IDS from the beginning. Also there is a problem when independent institutions would like to do an audit of IDS. Because signatures are not standardised an audit becomes very hard or even it is impossible to realise. In this kind of audit it is necessary to analyse signatures, which is a hard task in the absence of a common format of signatures of different IDSs. The same difficulty is when a company will try to integrate different (may be inherited?) Intrusion Detections Systems in common security policy. To overcome these difficulties, the purpose of the Common Intrusion Detection Signatures Standard (CIDSS) is to define a common data format for storing signatures from different IDSs. CIDSS is XML-based, so it is ready to automatically parse, verify and extend signatures. Because we want to integrate various signature standards, CIDSS is now the most extensible language that can describe network intrusions. Supporting other IDS signatures make CIDSS to implement features such as stateful rules. In effect it is necessary to make CIDSS more flexible and extensible. However, its rely on the XML data format extremely limits its use in the context of real time systems and HSN for the reasons so far mentioned. In our work, we propose to adopt CIDSS by considering JSON data format instead of the XML.

A signature can be divided into two basic, logical parts. The first part, that includes the elements ‘Sources’, ‘Destinations’, ‘Protocols’ and ‘Patterns’, is used to define building blocks of a signature definition. The main XML element of the second part of a signature is the ‘Session’ element. A ‘Session’ element defines the main signature behaviour. In this second part of a signature, the information contained in the first part is combined using logical expressions.

Each rule resides in Signature mark. In every Signatures tag there could be one or more Signature tags. Enabled tag defines which of rules included in Signatures should be considered active by IDS (which signature is enabled). The JSON representation of a sample CIDSS rule creation process is as follows.

```
{    “Signatures”:
       {“noNamespaceSchemaLocation”:
      “common.xsd”,
 “Signature”:{“sid”: “RULE_NUMBER”,
     “Enabled”: “true”
                }
              }
}
```

In each rule we must define used protocol and characteristics of a protocol. Protocol tags depends on protocol type as described in internet Draft. CIDSS protocol information can be presented in JSON as follows.

```
{“Signatures”: {“noNamespaceSchemaLocation”:
                  “common.xsd”, “Signature”:
{“sid”: “RULE_NUMBER”, “Enabled”: “true”,
     “Protocols”: {“Protocol”:
{“Type”: “icmp”, “ICMP_Icmp_Id”: “123”,
  “ICMP_Icmp_Seq”: “0”,
“ICMP_Itype”: “0”}}}}}
```

As shown above, Protocols tag contains one Protocol tag, but if rule applies to more than one protocol then it can contain more Protocol tags – e.g., information for UDP or TCP protocol. The JSON representation of multiple protocol tags is:

```
Protocols”: {“Protocol”: [{“Type”: “tcp”,
                “TCP_Ack”: “0”, “TCP_Window”:
“34000”}, {“Type”: “udp”, “UDP_Dsize”: “40000”}]}}.
```

It is similar when we define source (Source) and destination (Destination) of potential attack – it is possible to describe more then on source and destination in each rule. The following depicts an example for source and destination description presented in JSON.

```
Protocols”: {“Protocol”: [{“Type”: “tcp”,
               “TCP_Ack”: “0”, “TCP_Window”:
“34000”}, {“Type”: “udp”, “UDP_Dsize”: “40000”}]}}
```

In this example, there could be any destination (any IP address – TFN client) and sources are hosts with prefix 83 or 84 or host with 194.154.2.142 address (TFN daemons). When there is more than one Source tag, we can use logical expression to define dependencies between sources. There can be used even complex logical expressions with one of the keywords: AND, OR, NOT and brackets.

Patterns tag is one of the most important tags in whole signature. It contains Pattern tags which describe packets using particular pattern. Some pattern types are: PCRE, Expression, string, decimal and hexadecimal value. In what follows, we illustrate a rule using Patterns in JSON.

```
{“Signatures”: {“noNamespaceSchemaLocation”:
                    “common.xsd”, “Signature”:
{“sid”: “NR_REGULY”, “Enabled”: “true”,
      “Protocols”: “...”, “Sources”:
“...”, “Destinations”: “...”, “Patterns”:
 {“id”: “1”, “Pattern”:
{“pat_id”: “1”, “Pattern_Type”: “string”,
“Pattern_Content”:
{“CaseSensitive”: “true”, “#text”:
  “shell bound to \\\nport”},
“Pattern_Offset”: “36”}}}}}
```

Finally, we can define an action of rule and describe the whole signature by:

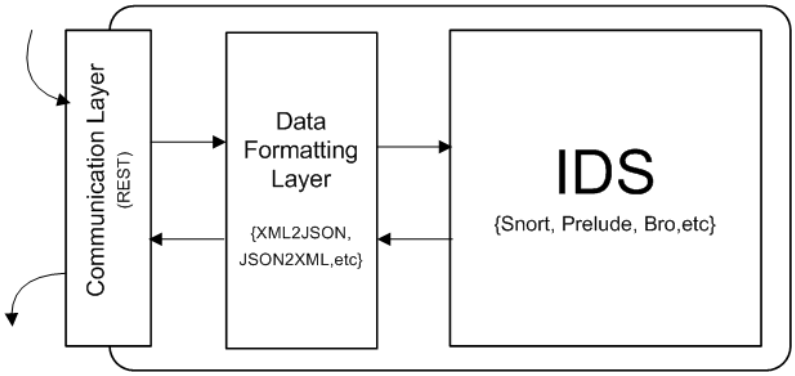
```
{“Signatures”: {“noNamespaceSchemaLocation”:
            “common.xsd”, “Signature”:
{“sid”: “NR_REGULY”, “Enabled”: “true”,
“Sig_source”: “Snort”, “Description”: “DDOS TFN
 server response”, “Action”: “alert”, “Protocols”
: “...”, “Sources”: “...”, “Destinations”: “...”,
“Patterns”: {“id”: “1”, “#text”: “...”}, “Message”:
  “DDOS TFN server response”, “Comment”:
[“reference:arachnids,182”, “classtype:attempted-dos”,
“rev:6”]}}}.
```

Sig_Source contains information about source of translation for the particular signature.

4.1.3 Data formatting layer

In order to automatically ensure the mapping between XML format and JSON of the patterns explained in the previous sections, we have developed a XML2JSON convertor. This convertor will be used in a high speed network sensor as depicted in Figure 6. The interaction between the data formatting layer can be implemented as a GLUE-layer linked with each modified IDS or as a separated process with inter-process-communication (IPC) capabilities.

Figure 6 XML2JSON module integration in an ids sensor

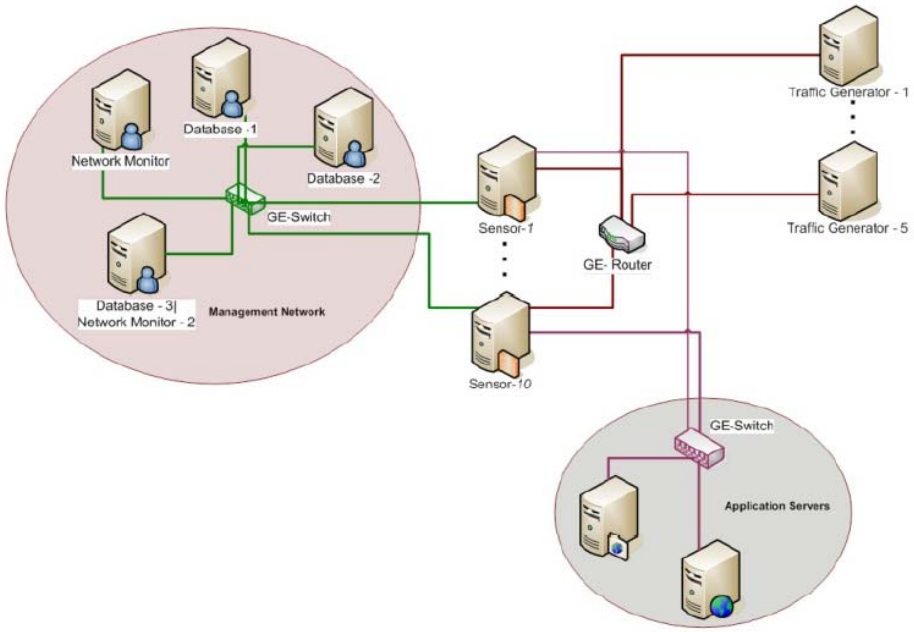


Then, in order to reduce the size of the JSON output string buffer, we have make an optimisation that consists in ignoring all extraneous formatting spaces and tabulations during the creation of JSON structure. Finally, the performance of this convertor and of the whole model in terms of buffer size and transmission time will be studied in next section.

5 Performance study

In order to generate 10 GB traffic, we have specified and realised the platform architecture depicted in Figure 7. Mainly, it consists of four components.

Figure 7 Test-platform architecture (see online version for colours)



The first one is the traffic generator which encompasses five servers. The traffic generator component uses some dedicated software like TCP replay (read and reply TCP traffic), Nmap (generates attacks and traffic), ITG (generates traffic with a law distribution) and Metasploit (for attack traffic definition).

The second component consists of the sensors and encompasses ten servers. Each server contains different IDSs like Snort (Martin, 1999) and BRO (Sommer, 2003).

The third component is the application server which includes two servers, one represent the HTTP server and the second is equipped with different server types like FTP, UDP. The application server represents the attacks targets.

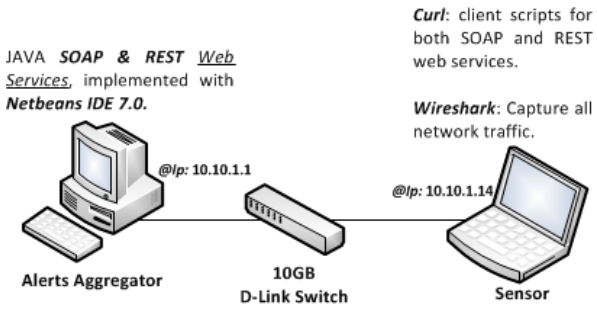
The last component is the network manager and encompasses four servers network monitoring and data storage.

In our experiments, we have used:

- Mellanox Vantage 6024 10 Gigabit Ethernet Switch to provide power-efficient and lowest latency capabilities on 10 Gigabit Ethernet.
- Mellanox Optical Cables to provide long range scalability for all network topologies and utilise innovative optical technologies to enable extremely high signal integrity and reliability.
- ConnectX EN – Single/Dual-Port 10 Gigabit Ethernet Adapters with PCI Express 2.0 to deliver high-bandwidth and industry-leading 10GBite connectivity with stateless offloads for converged fabrics in High-Performance Computing, Enterprise Data Centers, and Embedded environments.

To test the REStful communication model, as shown in Figure 8, two web services were implemented and launched on the alert aggregator side. In the sensor side, we have used curl to create scripts that send requests to the alert aggregator. Wireshark is already working in the sensor side and capture all network traffic.

Figure 8 Test-platform configuration for transmission time benchmarking (see online version for colours)

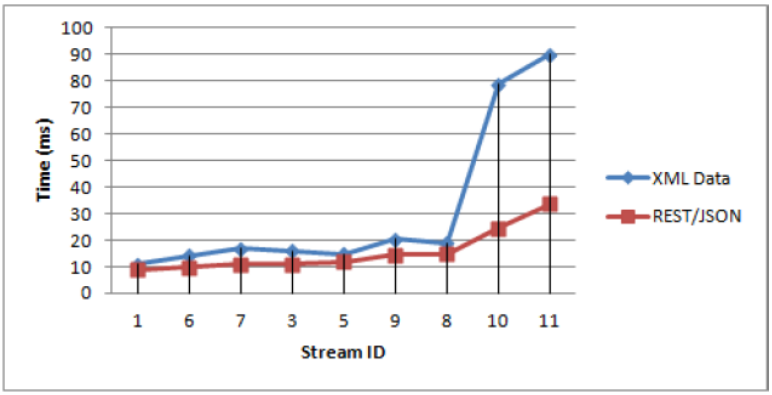


For the benchmarking we have considered two parameters, which are the Data size and the Transmission time. Figure 9 gives an idea about the gain in term of data buffer size. The gain in terms of transmission time is shown in Figure 10. The transmission time includes also the processing time, which is the time for parsing XML files to JSON ones.

Figure 9 Data buffer size gain (bytes)

XML	JSON	Gain
861	454	407
1240	978	262
1379	889	490
2133	1063	1070
2212	1146	1066
2394	1912	482
3417	1452	1965
12780	10300	2480
25474	20552	4922

Figure 10 Transmission time gain (see online version for colours)



We can observe a gain up to 60% in term of data transmission duration. It is clear that the reduction of the transmission time is more important by using REST instead of SOAP. Even for REST, with JSON format the results are better then with XML since XML is verbose.

6 Conclusions

The purpose of this paper is to describe an approach for overcoming limits of available solutions to ID problems in terms of scalability and heterogeneity. It has presented an RESTful communication model for high speed IDSs. The main features of this model are the adoption of the REST technology and the JSON data format. For more efficiency, we have based our protocol on two known standards, which are IDMEF for message format and CIDSS for rules definition. The proposed model was validated and tested. The results of measurements have showed an important gain in terms of data exchanged size, and transmission time.

One way to further improve the efficiency of our proposed architecture is to enhance the IDS entities behaviours with semantics, so that high levels of automation can be reached throughout the whole configuration and management tasks process, even though the conditions, resources or necessities vary over time. It will also be interesting to develop algorithms that would allow global distribution of various processing

components and ways to define the service composition processes. Once dynamically established and reconfigured, IDS compositions provide great flexibility to security administrators in security monitoring and response. The dynamic aspects of IDS compositions could be implemented using service orchestration.

Acknowledgements

This paper is a partial result of a research project granted by King Abdul Aziz City for Sciences and Technology (KACST), Riyadh, Kingdom of Saudi Arabia, under Grant Number INF 36-8-08. The authors would like to thank the reviewers for their constructive and helpful comments to prepare the final version of the paper.

References

Ammar, A. and Sallay, H. (2011) 'Measuring connection features' relevance to attack detection using neural networks', *Journal of Computer Research*, Vol. 10, No. 1, pp.968–975.

Bosin, A., Dessì, N. and Pes, B. (2004) 'Engineering knowledge discovery in network intrusion detection', in Yang, Z.R., Everson, R.M. and Yin, H. (Eds.): *'IDEAL', Lecture Notes in Computer Science*, pp.253–258, Springer.

Bosin, A., Dessì, N. and Pes, B. (2008) 'A service based approach to a new generation of intrusion detection systems', *Proceedings of the 2008 Sixth European Conference on Web Services', ECOWS '08*, IEEE Computer Society, Washington, DC, USA, pp 215–224.

Brandao, J., Mafra, P. and Fraga, J. (2006a) 'A new approach for IDS composition', *Proceedings of the IEEE International Conference on Communications', ICC '06, IEEE Computer Society.*

Brandao, J.E.M.S., da Silva Fraga, J., Mafra, M.M.P. and Obelheiro, R. (2006b) 'A WS-based infrastructure for integrating intrusion detection systems in large-scale environments', *Proceedings of the 2006 Confederated International Conference on the Move to Meaningful Internet Systems: CoopIS, DOA, GADA, and ODBASE – Volume Part I*, Springer-Verlag, Berlin, Heidelberg, pp.462–479.

Danyliw, R., Meijer, J. and Demchenko, Y. (2006) 'The Incident Object Description Exchange Format Data Model and XML Implementation', Internet draft draft-inch-ietf-iodef-08.txt, ietf.

Debar, H., Curry, D. and Feinstein, B. (2006) 'The intrusion detection message exchange format', internet draft draft-ietf-idwg-idmef-xml-16, ietf.

Feinstein, B., Matthews, G. and White, J. (2002) 'The Intrusion Detection Exchange Protocol (IDXP)', Internet draft draft-ietf-idwg-beep-idxp-07, ietf.

Foschini, L.A., Thapliyal, A.V., Cavallaro, L., Kruegel, C. and Vigna, G. (2008) 'A parallel architecture for stateful, high-speed intrusion detection', *Proceedings of the 4th International Conference on Information Systems Security', ICISS '08*, Springer-Verlag, Berlin, Heidelberg, pp. 203–220.

Grzegorz, K. and Agnieszka, P. (2012) 'Anomaly detection system based on service oriented architecture', *Proceedings of the 4th Asian conference on Intelligent Information and Database Systems – Volume Part III', ACIIDS'12*, Springer-Verlag, Berlin, Heidelberg, pp.376–385.

Keeni, G.M., Danyliw, R. and Demchenko, Y. (2006) 'Requirements for the format for incident information exchange (fine)', Internet draft draft-ietf-inch-requirements- 08.txt,ietf'.

Liu, M., Li, K. and Zhang, Z. (2010) 'One data preprocessing method in high-speed network intrusion detection', *InWireless, Mobile and Multimedia Networks (ICWMNN 2010), IET 3rd International Conference on.*

Martin, R. (1999) 'Snort – lightweight intrusion detection for networks', *Proceedings of the 13thUSENIXconference on System administration', LISA '99*, USENIX Association, Berkeley, CA, USA, pp.229–238.

Mauro, S., Denivaldo, L. and Zair, A. (2006) 'A remote ids based on multi-agent systems, web services and MDA', *Proceedings of the International Conference on Software Engineering Advances', ICSEA '06*, IEEE Computer Society, Washington, DC, USA, p.64.

Park, S., Kim, K., Jang, J. and Noh, B. (2003) 'Supporting interoperability to heterogeneous IDS in secure networking framework', *The 9th Asia-Pacific Conference on Communications, APCC '03*, pp.844–848.

Rao, R., Pal, A. and Patra, M.R. (2009) 'A service oriented architectural design for building intrusion detection systems', *International Journal of Recent Trends in Engineering*, Vol. 1, No. 3, pp.9–16.

Rouached, M. and Sallay, H. (2012) 'An efficient formal framework for intrusion detection systems', *ANT/MobiWIS 2012*, pp.968–975.

Sallay, H. (2011) 'Towards an integrated intrusion detection monitoring in high speed networks', *Journal of Computer Science*, Vol. 7, No. 7, pp.1094–1104.

Sallay, H., AlShalfan, K.A. and Fred, B.O. (2009) 'A scalable distributed ids architecture for high speed networks', *Int. J. Computer. Sciences and Network Security*, Vol. 9, No. 4, pp.9–16.

Sallay, H., Rouached, M., Ammar, A., Fredj, O.B., Al-Shalfan, K. and Saad, M.B. (2011) 'Wild-inspired intrusion detection system framework for high speed networks (f|p) IDS framework', *IJISP*, Vol. 5, No. 4, pp.47–58.

Sekar, R., Guang, Y., Verma, S. and Shanbhag, T. (1999) 'A high-performance network intrusion detection system', *Proceedings of the 6th ACM Conference on Computer and Communications Security, CCS '99*, ACM, New York, NY, USA, pp.8–17.

Sommer, R. (2003) 'Bro: an open source network intrusion detection system'.

Su, M.Y., Yu, G-J. and Lin, C-Y. (2009) 'A real-time network intrusion detection system for large-scale attacks based on an incremental mining approach', *Computers & Security*, Vol. 28, No. 5, pp.301–309.

Subbulakshmi, T., Mathew, G. and Shalinie, S.M. (2010) 'Real time classification and clustering of ids alerts using machine learning algorithms', *International Journal of Artificial Intelligence & Applications (IJAIA)*, Vol. 1, No. 1, pp.1–9.

Thomas, E. (2005) *Service-Oriented Architecture: Concepts, Technology, and Design*, Prentice Hall PTR, Upper Saddle River, NJ, USA.

Thomas, F.R. (2000) *Architectural Styles and the Design of Network-Based Software Architectures*, PhD thesis, AAI9980887.

Treinen, R.J. (2006) 'A framework for the application of association rule mining in large intrusion detection infrastructures', *9th International Symposium, RAID 2006*, Hamburg, Germany.

Wenbao, J. and Shuang, H.D.L. (2006) 'Load balancing algorithm for high-speed network intrusion detection systems', *Journal of Tsinghua Univ. (Sci. & Tech.)*, Vol. 46, No. 1, pp.106–110.

Wood, M. and Erlinger, M. (2002), 'Intrusion detection message exchange requirements', Internet draft draft-ietf-idwg-requirements-10, ietf.

Yu, F., Dai, X., Shen, Y. and Huang, H. (2005) 'Intrusion detection and simulation for high-speed networks', *International Conference on Services Systems and Services Management*, Vol. 2, No. 1, pp.835–840.