I.J. Information Technology and Computer Science, **2015, 04, 42-49**
Published Online March 2015 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2015.04.04

Identity Management: Lightweight SAML for Less Processing Power

Mohammed Ali[1], **Tarek S. Sobh**[2,*], **Salwa El-Gamal**[3]
[1,3] Faculty of Computers and Information, Cairo University, Egypt
E-mail: Mohammed.ali.cs@gmail.com, s.elgamal@fci-cu.edu.eg

[2] Information Systems Department, Egyptian Armed Forces, Egypt
E-mail: tarekbox2000@yahoo.com

Abstract **—** Identity management has emerged as important issue for reducing complexity and improving user experience when accessing services. In addition to, recently authentication services added SAML to the range of authentication options to be available to cloud subscriber. This work mainly focused on SAML representation in the existing identity management frameworks and its suitability.

We introduced a new representation of SAML that makes it light in weight and easier in parsing and dealing with. This representation is demonstrated using JSON.

In our new SAML representation, we enhanced the performance of marshalling the SAML by 28.99%.

In this paper, we will go into these challenges to introduce a new representation for the identity and access management markup language. Our proposed representation is designed to match the small processing power devices for faster generation, parsing and communication.

Index Terms **—** SAML, SSO, XML, JSON, IdM, IdP, SP.

I. Introduction

Identity and Access Management (IdM) describes the management of people identifications that includes the authentication, authorization, and privileges or permissions within one or multiple enterprises and it aims to increase the security and decreasing the cost, downtime and redundant work.

Recent experimentations showed that mobile phones are capable of hosting rich web sites, web applications and web services while mobile devices are physically constrained devices due to the low processor speed, limited memory, limited battery, and slow wireless connection. So we must take into consideration these factors when implementing any communication system between mobile devices and another system or server.

While mobile hosting enables exciting and novel application scenarios, there are use cases that would benefit of more lightweight server transactions.

The rest of paper is organized as following; section 2 describes a related work, section 3 introduces the lightweight transportation and language framework, section 4 discusses the problem of SAML into XML for small processing devices and the current work in this area, section 5 describes the proposed solution and the experiential setup with demonstrating our proposed solution into implementation, section 8 discusses our results.

II. Related Work

KamalEldin Mohamed and Duminda Wijesekera proposed a lightweight Framework for Web Services Implementations on Mobile Devices [1] that is based on the existing web services and mobile devices technology standards. They deployed the RESTful web service on Jetty server deployed to the physical mobile web service host as well as SQLite as a lightweight database. They utilized the "HttpURLConnection" for making HTTP request calls as well as interpreting the HTTP response messages from the Andriod Web services host.

Yeon-Seok Kim and Kyong-Ho Lee proposed A Light-weight Framework for Hosting Web Services on Mobile Devices [2] that have several built-in modules such as the processing of SOAP messages, the execution and migration of services, and the publishing and discovery of services to be deployed on the mobile and tested to prove its matches with the mobile criteria [3].

Kevin C. Almeroth, Katia Obraczka and Dante De Lucia proposed a lightweight Pseudo-IP (PIP) protocol in "A Lightweight Protocol for Interconnecting Heterogeneous Devices in Dynamic Environments" research [4] which is a lightweight protocol to operate among devices in the farthest branches/leaves of an intranet while providing internetwork connectivity with other clouds and with the existing IP-based Internet infrastructure. Although this protocol is proposed to connect the heterogeneous devices in dynamic environment, it was careful enough to be a light protocol to match the criteria of the mobile devices.

In September 21, 2007, J. Hodges and S. Cantor introduced SAMLv2 Lightweight Web Browser SSO Profile draft [5]. This profile is similar to the Web Browser SSO Profile in SAMLv2 with some differences as it employs the HTTP POST-SimpleSign SAML binding rather than the other bindings specified in SAMLv2.

In this Lightweight SAML profile, some restrictions are made as in the <AuthRequest> message, AllowCreate is always true, <subject> and <Conditions> are disallowed, only a single assertion is returned and the

requirements for the IdP to validate the provider's assertion consumer service becomes more lighter.

III. Lightweight Transportation and Language Framework

A lightweight transportation layer protocols are also needed. Internet Content Adaptation Protocol (ICAP) [6] is a lightweight HTTP-like protocol which is used to extend transparent proxy servers, thereby freeing up resources and standardizing the way in which new features are implemented. ICAP is a lightweight protocol for executing a remote procedure call on HTTP messages.

A. Lightweight protocols

Lightweight protocols are communications protocols designed with less complexity in order to reduce overhead.

Lightweight communication between devices is a hot topic for more search area because of the widely use of the portable limit capabilities devices.

As any protocol is a set of rules defining communication between systems. It is an important part for any systems or services interactions. It considered being the language between entities. For the complexity of current designed protocols and the use of small devices in hosting applications uses these protocols, here is the invention of the lightweight protocols.

Usually when talking about small devices with limited processing power, it worth mention the strong relation between these small devices and cloud computing. Cloud computing has received large interest recently. Cloud offers a lot of services that widely used not only on small devices but also on PCs however, a shared problem between the mobile and the cloud computing is time and the cost of transferring a large amount of data. Also the generation of this data on the small devices that obviously will take more time or need reconstructed to match the small devices capabilities.

Here, with the widely use of the cloud, the identity of users or people are transferred into cloud. So, it requires an identity management for the identity on the cloud that grantee the trusted level of security for transferring theses sensitive data between servers and clients.

B. Security Assertion Markup language

One of the widely used markup language for exchanging the authentication and authorization data between security domains is the Security Assertion Markup language (SAML). SAML is an XML-based standard for exchanging authentication and authorization data between security domains, that is, between an identity provider (a producer of assertions) and a service provider (a consumer of assertions). SAML provides a standard way of expressing signed assertions about the identity and attributes of a system participant [7], [8].

The trend of hosting services on small devices, it required to think about the challenges of hosting the identity providers on small device that uses SAML as a communication language between providers.

Security Assertion Markup Language (SAML) Standard specifies a XML based framework for describing and exchanging security information between trusted partners in a closed IT-Federation [9]. For a SAML environment the following components are defined: Assertions, Protocols, Bindings and Profiles. SAML Assertions represent XML based messages which contain statements (e.g. authentication method, time of authentication, attributes as access rights) about a principal. SAML protocol is a simple request and response protocol that defines the way of each SAML element will be represented. Query is the most important part of any SAML protocol that may be Authentication query, Attribute query or Authorization decision query. Some of these protocols (in SAML 2.0) are Assertion Query and Request Protocol, Authentication Request Protocol, Artifact Resolution Protocol, Name Identifier Management Protocol, Single Logout Protocol and Name Identifier Mapping Protocol.

The way in which SAML protocol messages as shown in Fig. 1 are transmitted over underlying transport protocols is detailed by SAML Bindings. So, the SAML Bindings are the ways to inject or represent the SAML protocol into currently standard communication messages like HTTP and SOAP.

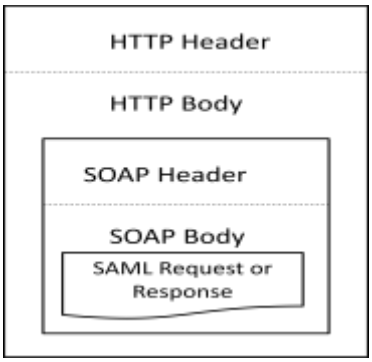

Fig. 1. SAML Message

Every use case (e.g. web SSO) is use a combination of the SAML assertions, protocols and bindings techniques. The description of these combination or groups to service different use cases is defined to be a SAML Profile.

In the web browser SSO profile, the user agent is either accessing a resource on a Service Provider (SP) or accessing the Identity Provider (IdP) and here is a sequence diagram for a scenario of a Web Browser SSO profile (see Fig. 2):

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human and machine readable. It is defined in the XML 1.0 Specification produced by the W3C [10] as shown in Fig. 3.

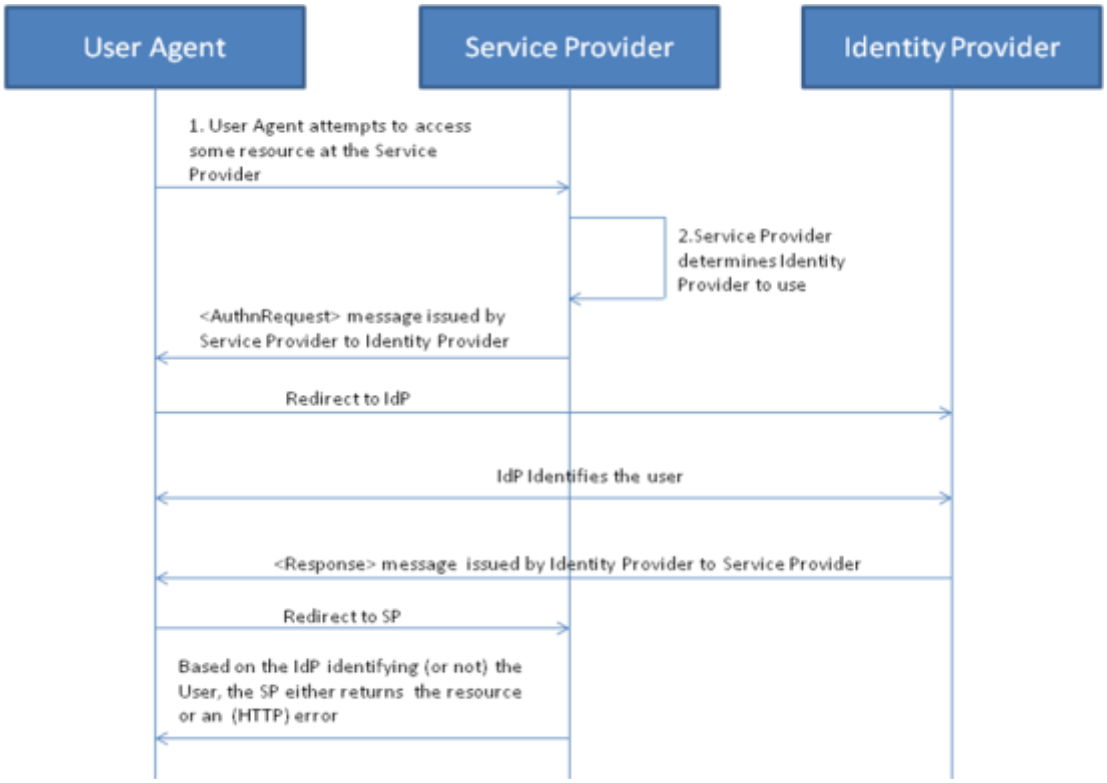

Fig. 2. Web Browser SSO Scenario

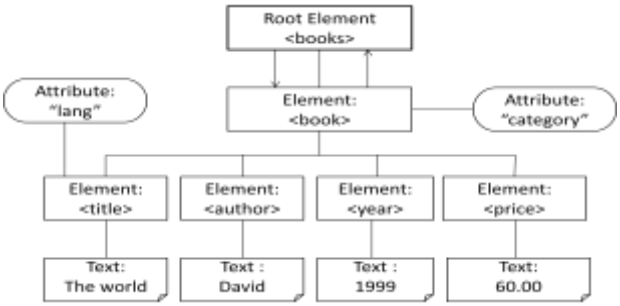

Fig. 3. XML Structure

C. XML

XML designed to achieve the simplicity, generality, and usability over the Internet through a structured document. Also it aimed to support multi-languages through its Unicode support. Although the design of XML focuses on documents, XML used in the representation of many technologies and communication protocols like web services [9], [10].

An XML element is everything from (including) the element's start tag to (including) the element's end tag.

An element can contain:

- *Text* the containing text between the start element tag and the end element tag (e.g. <title> The world</title>).
- *Attributes* provide some additional information about the element (e.g. <title lang="English">).
- *Other elements* that may be included into another element. (e.g. <book><author>David</author></book>).
- Or a mix of all of the above.

Elements may be self-closed, that have no element tag end but the element should end by "/" (e.g. <title lang="English" />).

```
<books>
        <book category="historical">
                <title lang="English">The World</title>
                <author>David</author>
                <year>1999</year>
                <price>60.00</price>
        </book>
</books>
```

D. JavaScript Object Notations (JSON)

The JavaScript Object Notation (JSON) is a text format for the serialization of data structure or object [11], [12], [13], [14]. The JSON can represent four primaries types (strings, numbers, boolean and zero) and two structured types (objects and vectors). The JSON was designed with the aim to be simple, portable, textual, and a subset of JavaScript.

JSON syntax is a subset of the JavaScript object notation syntax as the data is in name/value pairs and separated by comma. Also it curly brackets holds objects and square brackets holds arrays.

An example of JSON

```
{
    "employees": [
        { "firstName":"John" , "lastName":"Doe" },
        { "firstName":"Anna" , "lastName":"Smith" },
        { "firstName":"Peter" , "lastName":"Jones" }]
}
```


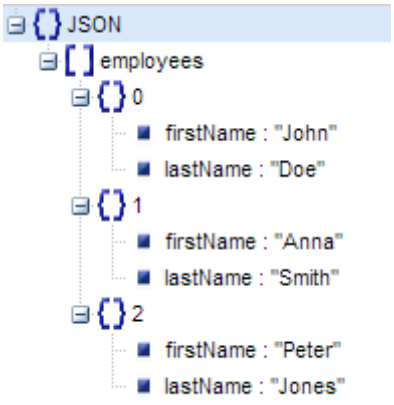

Fig. 4. JSON Example

JSON is much like XML as JSON is plain text, "self-describing" (human readable), hierarchical (values within values), can be parsed by JavaScript and can be transported using AJAX. However, there are some features in JSON that differ than XML as JSON have no end tag, shorter, quicker to read and write, can be parsed using built-in JavaScript eval(), uses arrays and no reserved words.

IV. Problem Statement And Current Cituation

As SAML specifies a XML based framework, SAML shows the performance problem for XML [8], [15], [16]:

- *Redundant syntax:* This can affect human reading; the efficiency of applications is obliged to higher storage costs.
- *The construction of parsers are not trivial as it seems*: Although theoretically allows XML parsers to construct simple and consistence, a good XML parser must be designed to process data arbitrarily nested and perform additional tests to detect data badly formatted or syntax errors (due in part to the problem described in the previous item). This can cause significant additional processing to use basic of XML.
- *Processing power:* The parsing and constructing operations for XML is not light. It costs a processing power that may be not applicable for small devices.
- *The syntax of XML is difficult for humans to use*: The sequence of keys that are used to wrote expressions in XML are hard to use. Generators or specialized editors are used to decrease the power weight of this problem.

SAML requests and responses generation is a performance weight to be generated from mobile devices or small processing power devices.

To let small devices with less processing power generate SAML request or get and parse a SAML response, it's too costly task. SAML is a XML based and working with XML need a much higher processing power. Also the size of transmitted data is much heavy because of the limited small devices bandwidth. So we need to make the SAML representation more light to be more suitable with less cost for small processing power devices.

Because of the weight of some processing needed to generate SAML requests and responses, SAML 2.0 introduced Enhanced Client or Proxy (ECP) Profile [8], [17] for non browser based clients. The ECP profile specifies interactions between enhanced clients or proxies and service providers and identity providers. It has, or knows how to obtain, information about the identity provider that the principal associated with the ECP wishes to use, in the context of an interaction with a service provider. This allows a service provider to make an authentication request to the ECP without the need to know or discover the appropriate identity provider and without need to know the client, at the end of the job, the client will be informed by the proxy and there is already a trusted relationship between the browser and the used proxy. It is a suitable solution case the client doesn't have a browser and the service provider have a trusted proxy that can delegate the job to, but still this solution away from representing the SAML as a server on small devices. In ECP, the identity provider or the service provider still deployed on strong machines not lightweight ones.

Another alternative approaches is the RB-SSO Proxy introduced by Thang Tran and Christian Wietfeld [18] that replaces the user agent (e.g. web browser) to communicate directly with the SP or the IdP considering the security concepts of the system have to be carefully considered.

Because this lightweight profile uses the simpleSign binding, the cryptographically-based security is optional. This shows the weakness of this mechanism because it is leaving itself explicitly open to Man in the Middle attacks.

All previous mentioned approaches are trying to lower the load on the client but not considering that mobile devices are now used to host mobility services to be provided. So, the SAML request and response need to be generated from small processing power.

V. Proposed Solution

We propose using the JSON (JavaScript Object Notation) a SAML representation to speed up response/request generation and parsing that will be definitely useful for small processing power devices. JSON is syntax for storing and exchanging text information much like XML. Most of mobile platforms that have small processing power supports JSON and already have the same data representation like JSON.

A. JSON vs. XML

As resulting by Ruben Fonseca and Alberto Simoes [11], [13], [14], [19], JSON is not to the scale as well as the XML when the size of the input increases significantly. The explanation may come to the fact that the specification of JSON to be less complex, so a format less flexible, but simpler and easier to process. The

memory consumption during parsing is better than XML. In the context of applications Web (where JSON the most used), we can conclude that a good bet because JSON Besides its functional advantages, it is slightly faster than the XML for inputs of small and medium dimension (even 1Mb input) as shown in Fig. 5.

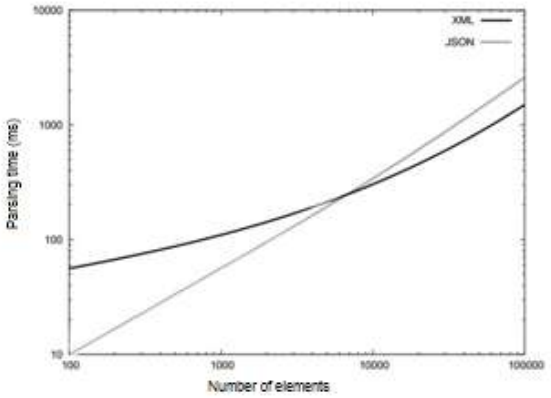

Fig. 5. XML and JSON

Using the JSON in SAML request and response representation is a good idea to speed up SAML request/response generation and be applicable to be deployed on small devices (quite processing power) because JSON is lightweight text-data interchange format, language independent, self-describing and easy to understand.

B. SAML based JSON

As JSON data is represented in name/value pairs, in our proposed solution we describe each XML element property into a pair of name/value.

The XML tag name is the name of the JSON element, and every JSON element may contain "attributes" element that contains a key and value for the attribute name and its value. Also the JSON element may contain "childs" element that contains all XML tag children included under the main tag element. Finally, the JSON element may contain "value" element that represents the tag value.

This is an example of "Assertion" tag element and its equivalent JSON element.

If the XML contains a self-closing tag, it will be represented normally as any XML tag and the value of the equivalent JSON element will take value "self-closing".

```
<saml:Assertion
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
IssueInstant="2012-07-31T23:57:16.472Z"
Version="2.0">
<saml:Issuer>http://synesty.com</saml:Issuer>
…
</saml:Assertion>
```

```
"saml:Assertion" :
{
        "attributes" :
        {
"xmlns:saml":"urn:oasis:names:tc:SAML:2.0:assertion",
"IssueInstant" : "2012-07-31T23:57:16.472Z",
"Version" : "2.0"
        },
        "childs" :
        {
                "saml:Issuer" :
                {
                        "value" : "http://synesty.com"
                }
        }
}
```

C. JSON schema

The schema used in JSON SAML representation according to IETF latest draft [6], [13] in November 22, 2010.

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "title": "SAML",
   "description": "A SAML Object",
   "type": "object"
    "properties": {"$ref": "#/definitions/element"},
    "patternProperties": "*:*",
    "additionalProperties": false
}
```

```
{
properties: {
"attributes" : {
        "patternProperties": "*:*"
        "additionalProperties": false
    },
    "childs" : {
        "$ref": "#/definitions/element"
    },
    "value" : {
     "type": "string",
     "additionalProperties": false
    }
}
}
```

To validate on the JSON schemas, we can use the schama validator [7], [14], [20].

D. SAML in JSON

We used an open source SAML implementation OpenSAML [7], [21], [22], [23] that build over three main factories

- Marshalling Factory.
- UnMarshalling Factory.
- Builder Factory.

A new JSON Marshalling and Unmarshalling factories are build and integrated into OpenSAML implementation.

Marshalling refers to the process of converting the data or the objects (SAML Object) into a byte-stream (XML or JSON), and Unmarshalling is the reverse process of converting the byte-stream (XML or JSON) back to their original data or object (SAML Object).

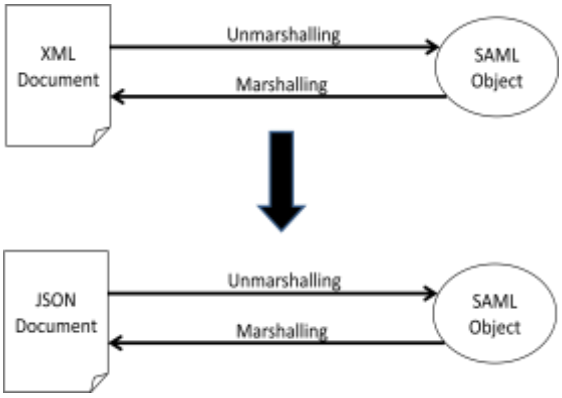

Fig. 6. Marshalling and Unmarshalling

The JSON library used is Jackson [13], [24] which is an open source library to read and write JSON objects.

E. Experimental Setup

Our experimental setup was done on Core2 Due machine with 4 GB RAM. Table (1) shows 100 run of the proposed solution when we represent SAML object in XML and JSON.

Table 1. run SAML object in XML and JSON

run no.	XML	JSON	run no.	XML	JSON
1	107	116	51	6	3
2	23	3	52	6	3
3	11	2	53	4	2
4	11	4	54	50	2
5	11	3	55	42	74
6	18	2	56	7	13
7	27	3	57	6	3
8	49	3	58	6	3
9	11	4	59	6	13
10	14	3	60	10	3
11	11	3	61	6	65
12	94	3	62	98	17
13	13	14	63	7	4
14	9	4	64	5	3
15	10	19	65	4	6
16	75	3	66	105	3
17	16	3	67	10	87
18	43	3	68	6	5
19	10	61	69	9	5
20	12	10	70	9	4
21	8	6	71	31	5
22	45	3	72	8	3
23	52	3	73	6	128
24	12	4	74	19	4
25	8	70	75	22	4
26	36	37	76	7	6
27	47	2	77	31	3
28	27	2	78	7	3
29	47	2	79	15	3
30	7	3	80	6	3
31	8	66	81	8	2
32	15	4	82	9	3
33	48	5	83	13	5
34	14	20	84	21	2
35	12	2	85	6	3
36	7	3	86	18	3
37	11	3	87	6	2
38	7	3	88	107	3
39	19	2	89	8	3
40	7	3	90	20	9
41	12	3	91	7	33
42	8	3	92	4	3
43	8	3	93	4	3
44	16	3	94	98	2
45	16	3	95	23	4
46	10	2	96	8	3
47	8	4	97	7	3
48	37	4	98	8	4
49	9	3	99	25	3
50	29	4	100	7	2

The average time of 100 run for representing SAML Object in XML takes 19.46 second while the average time of 100 run for representing the same SAML object in JSON takes 13.82 on the same machine as shown in Fig. 7.

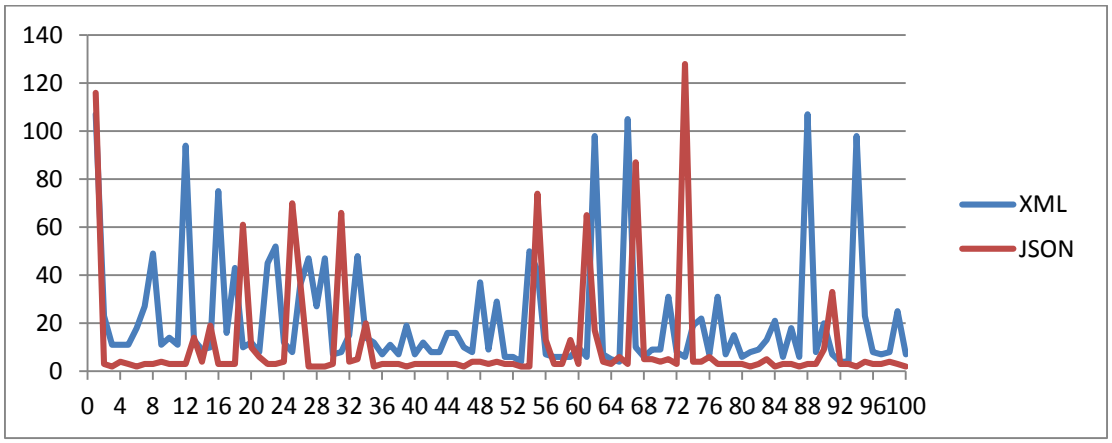

Fig. 7. XML and JSON marshalling for SAML object

F. SAML in JSON is faster

In this work, we argue that JSON representation for SAML requests and response enhancing the performance by 28.99% after calculating the average of 100 run for each representation for the same SAML object.

Away from performance enhancement, the JSON schema [11], [16], [25] validation libraries could be in JavaScript which is very suitable for the small and less processing power devices. These libraries are like tv4 [26], direct-schema [27] and schema.js [28].

VI. Conclusions

In this study, we have shown an enhancement in SAML performance by representing the SAML requests and response in JSON instead of XML. This performance enhancement may make the SAML eligible to work on small devices with small processing power. The best of SAML over JSON is not only the light representation but it also readable and not a data compression technique for XML like using GZip or the Efficient XML Interchange (EXI) to compress the XML although the compression is possible for JSON for small size but more processing power. EXI is an encoding format that allows efficient interchange of the XML information. The overhead here is the processing did for encoding the XML and overhead already did for writing the XML file or parsing it.

JSON also has schema like XML which a more research could done here to represent the SAML XML schemas into JSON schemas also removing the optional parts in JSON Schemas for customization and limited functionalities use for small processing devices would make it more light. So a future work goes here.

XML cryptography and signature are main features provided by XML for standard and defined way for these base security operations. The same in JSON, Michael B. Jones proposed the emerging JSON-Based identity protocol suite [15], [29], [30] that describes the JSON Web Token, Signature and Encryption. A future work goes here to define the JSON needed cryptography operations within the SAML-JSON schemas.

References

[1] KamalEldin Mohamed and Duminda Wijesekera, “A lightweight Framework for Web Services Implementations on Mobile Device”, IEEE First International Conference on Mobile Services, 2012

[2] Yeon-Seok Kim and Kyong-Ho Lee, “A Light-weight Framework for Hosting Web Services on Mobile Device”, Fifth European Conference on Web Services.

[3] Tarek S. Sobh and Medhat Fakhry “Evaluating Web Services Functionality and Performance”, International Journal of Information Technology and Computer Science (IJITCS), Vol. 6, No. 5, PP.18-27, April 2014

[4] Kevin C. Almeroth, Katia Obraczka and Dante De Lucia, “A Lightweight Protocol for Interconnecting Heterogeneous Devices in Dynamic Environments”, ICMCS '99 Proceedings of the IEEE International Conference on Multimedia Computing and Systems.

[5] J. Hodges and S. Cantor, SAMLv2 Lightweight Web Browser SSO Profile, available: http://identitymeme.org/doc/draft-hodges-saml-lsso-02.txt

[6] Internet Content Adaptation Protocol (ICAP), available: http://tools.ietf.org/html/rfc350

[7] OpenSAML v2.0. Available: https://wiki.shibboleth.net/confluence/display/OpenSAML/Home

[8] J. Hodges and S. Cantor, SAMLv2 Lightweight Web Browser SSO Profile, available: http://identitymeme.org/doc/draft-hodges-saml-lsso-02.txt

[9] Extensible Markup Language (XML), available: http://www.w3.org/ XML/.

[10] XML 1.0 Specification produced by the W3C, available: http://www.w3.org/TR/2008/REC-xml-20081126/

[11] JSON Schema Validator, available: https://github.com/fge/json-schema-validator.

[12] JSON Schema implementations, available: http://json-schema.org/

[13] JSON schema draft, available: http://tools.ietf.org/html/draft-zyp-json-schema-03.

[14] JSON Schema Validator, available: https://github.com/fge/json-schema-validator.

[15] Jackson, available: http://jackson.codehaus.org/

[16] JSON Schema implementations, available: http://json-schema.org/implementations.html

[17] Michael B. Jones, the Emerging JSON-Based Identity Protocol Suite, W3C Workshop on Identity in the Browser, April 27, 2011.

[18] Thang Tran and Christian Wietfeld, “Approaches for Optimizing the Performance of a Mobile SAML-based Emergency Response System”, Communication Networks Institute (CNI), Faculty of Electrical Engineering and Information Technology, Dortmund University of Technology, Germany.

[19] Jianneng Cao, Fang-Yu Rao, Mehmet Kuzu, Elisa Bertino and Murat Kantarcioglu, “Efficient Tree Pattern Queries On Encrypted XML Documents”, In the proceeding of EDBT/ICDT ’13, PP. 1-10, March 18-22 2013, Genoa, Italy.

[20] Kelly D. LEWIS and James E. LEWIS, “Web Single Sign-On Authentication using SAML”, IJCSI International Journal of Computer Science Issues, Vol. 2, pp. 41-48, 2009

[21] OASIS Standard, Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0, 15 March 2005.

[22] Patricia Arias Cabarcos, Florina Almen´arez Mendoza, Andr´es Mar´ın-L´opez, and Daniel D´ıaz-S´anchez, “Enabling SAML for Dynamic Identity Federation Management”, WMNC 2009, IFIP AICT 308, pp. 173–184, 2009.

[23] Md. Sadek Ferdous and Ron Poet, “Dynamic Identity Federation Using Security Assertion Markup Language (SAML)”, IDMAN 2013, IFIP AICT 396, pp. 131–146, 2013.

[24] Waldemar Hummer, Patrick Gaubatz, Mark Strembeck, Uwe Zdun, and Schahram Dustdar, An Integrated Approach for Identity and Access, The ACM Symposium on Access Control Models and Technologies (SACMAT), 2011.

[25] RAMALHO, Jos éCarlos ; SIMÕES, Alberto ; CARRIÇO, Lu ís, ed. lit. -“XATA2007 : XML : aplicações e tecnologias associadas : actas da Confer ência Nacional, 5, Lisboa, 2007.” [S.l. : s.n.], 2007, ISBN 978-972-99166-4-9. p. 33–46.

[26] Tv4, available: http://geraintluff.github.com/tv4/
[27] Direct-schema, available: https://github.com/IreneKnapp/direct-schema
[28] Schema.js, available: https://github.com/akidee/schema.js
[29] Hsu-Chun Hsiao, Tiffany Hyun-Jin Kim, Adrian Perrig, Akira Yamada, Samuel C. Nelson, Marco Gruteser, and Wei Meng, "LAP: Lightweight Anonymity and Privacy", IEEE Symposium on Security and Privacy, PP. 506-520, 2012.
[30] Nelson Gonzalez, Charles Miers, Fernando Red´ıgolo, Marcos Simpl´ıcio, Tereza Carvalho, Mats N¨aslund and Makan Pourzandi, "A quantitative analysis of current security concerns and solutions for cloud computing", Journal of Cloud Computing: Advances, Systems and Applications 2012, 1:11, Published by Springer

Authors' Profiles

Mohammed Ali received his B.Sc. degree in computer science from Faculty of Computers and Information at Cairo University. Currently, he works as a service information developer at HP. He is a highly experienced software engineer, results-oriented and technologies professional with particular expertise in Java Technologies, over 2 years of experience in developing, designing, architecture and integration of software, demonstrated ability to acquire technical knowledge and skills rapidly, innovative problem solver, able to see the business and technical sides of a problem with exceptional communication skills, both oral and written is seeking a position in the field of software engineering where these skills will add similar or greater value.

Tarek Salah Sobh received his B.Sc. degree in computer engineering from Military Technical College, Cairo, Egypt in 1987. Both M.Sc. and Ph.D. degrees from Computer and System Engineering Department, Faculty of Engineering, Al-Azhar University, Cairo, Egypt. He has managed, designed and developed several package for business applications and security systems. He has authored/co-authored of many refereed journal/conference papers and booklet. Some of the articles are available in the ScienceDirect Top 25 hottest articles. His research of interest includes computer networks, security systems, distributed systems, knowledge discovery, data mining, and software engineering**.**

Professor Salwa El-Gamal is currently the Vice Dean of Society and Environment Development in Faculty of Computers and Information at Cairo University. Years ago she took on her shoulders the responsibility of getting Cairo University involved in programming competitions, like ACM and IOI. Her most recent achievement is to push and succeed in adding courses to the syllabus that target enhancing students' knowledge of important aspects of CS like algorithms and advanced data structures.

How to cite this paper: Mohammed Ali, Tarek S. Sobh, Salwa El-Gamal,"Identity Management: Lightweight SAML for Less Processing Power", International Journal of Information Technology and Computer Science(IJITCS), vol.7, no.4, pp.42-49, 2015. DOI: 10.5815/ijitcs.2015.04.04