An XML Firewall on Embedded Network Processor

Wei Wang[1,2] and Jun Li[2,3]
[1]*Department of Automation, Tsinghua University, China*
[2]*Research Institute of Information Technology, Tsinghua University, China*
[3]*National Lab for Information Science and Technology, China*
autotm@gmail.com

Abstract

XML-based services with flexible and intelligent structures for data expression and exchange are quickly gaining popularity. Enterprises are deploying XML-based services as a central component of the application integration. As the application data are crucial to the enterprises, the XML messages must be secured to ensure the reliability of these services. This paper presents the design of an embedded XML firewall with XML identification, XML validation, XML encryption and decryption, XML signature and signature verification, which is implemented on Intel IXP425, an embedded network processor for small and medium enterprise solutions. Suitable for enterprises to deploy XML security for their IT infrastructure, the XML firewall provides confidentiality, integrity and authenticity for XML-based services. Improvements are introduced and evaluated, including schema preprocessing and hardware acceleration for security processing. Ideas about future work of XML firewall based on this platform are also proposed.

1. Introduction

Data exchange and application integration are becoming more and more critical to business success. XML enables convenient data sharing, regardless of the platforms. XML also proliferates powerful intelligent search, a fine granularity structural search that goes to the interior of an XML document, because XML can leverage its tree structure to induce the complex semantic search to an intuitive and feasible tree search. Mainly due to these two attractive advantages, XML has quickly become a popular language to express data in electronic business for the construction of Web services and SOA (Service-Oriented Architecture). It is predicted that the XML application traffic will increase rapidly and take about 45% of overall network traffic in 2008 [1].

As XML-based services are becoming popular, various security threats emerge targeting XML applications. A survey of IT managers shows that 45.5% of them take security as the biggest obstacle of deploying XML-based Web services [3].

There are three major categories of XML-based security threats [4].

(1) Message Transport Security. During transportation through Internet, XML messages may be under attacks such as "man in the middle", or data compromise. SSL (Secure Sockets Layer) is useful to provide security for message transportation, but it is insufficient for many XML-based Web services, because these services require different secure levels of protection and extend beyond point to point topology addressed in SSL.

(2) XML-Based DoS (Denial of Service) Attacks. Attackers can send huge amount of XML messages to the server, which consume nearly all system resources so that the server will be unable to respond to valid users' requests. Besides, XDoS (XML-based DoS) attacks can be brought into effect in other ways, for example, recursive entity declarations in XML message can cause the parser of the server either to shut down with an out-of-memory error or to become irresponsible to legitimate requests by consuming great amount of processor cycles.

(3) Content-Based Attacks. Attackers can insert some special symbols or numbers into the XML messages. These symbols or numbers may form a section of malicious code, like invalid SQL sentence, to compromise the server.

In summary, an XML message, which is text-based and application-level processed, can be attached to HTTP protocol and transferred through Internet. It may pass through firewalls without getting checked, because most of the firewalls have their policies set to allow HTTP protocol. This makes it possible to use XML messages to invade the security of hosts via network.

This security need motivates the development of XML firewall, usually a network appliance with comprehensive XML processing functions built-in. It is predicted by Yankee Group that the revenue of XML firewall will be close to 100 million dollars in 2009 [2].

Our contributions in this paper are summarized as follows. We use DOM (Document Object Model) [12] processing scheme to implement an XML firewall on an embedded network processor with all four basic functions integrated into the XML security system. Based on the observation of XML-based services' characteristics, we introduce two optimizations for schema validation. Driven by the bottleneck analysis, we also develop hardware acceleration for XML security processing.

The remainder of the paper is organized as follows. We start by giving a system design and introduced improvements of our XML firewall in Section 2. We then provide the experimental data and corresponding analysis in Section 3. We bring some thoughts about future work of XML firewall based on this platform in Section 4. We then summarize our work in Section 5.

2. Design and Improvement

In this section, we describe the system functions and the hardware platform. We also present the improvements and processing scheme, and explain why they are introduced to the system.

2.1. XML Firewall Architecture

As shown in Figure 1, the XML firewall processes the messages over various protocols and makes policy-based decisions on the traffic by accessing the identity and policy stores managed by the policy server. Then the XML firewall routes the messages to the specific application server.

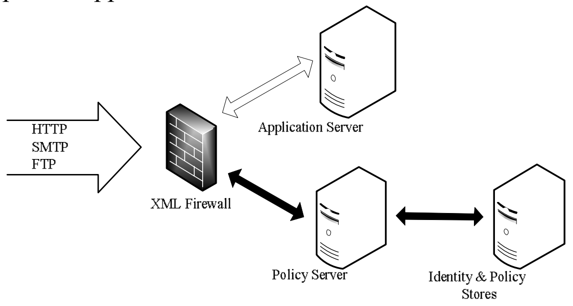

Figure 1. The XML firewall ecosystem

Generally speaking, an XML firewall contains four basic XML processing functions. They are identification, validation, encryption and decryption, signature and signature verification. We call the last two functions "security processing" in this paper. They are deployed against different threats, which are shown in Table 1.

Table 1. Threats resisted by different functions

	Message Transport Security	XML-Based DoS	Content-Based Attacks
Identification	N/A	√	√
Validation	N/A	√	√
Signature & Verification	√	N/A	√
Encryption & Decryption	√	N/A	N/A

In this paper we implemented all these four functions according to W3C Recommendations [7]-[10]. These functions can also process WS-Security [11] standard traffic for Web services with corresponding security policy and template. From industrial test report [5], it is obvious that security processing is the bottleneck in the whole XML processing, but once the embedded system with XScale core is used and the hardware acceleration is deployed, validation and identification will become the bottlenecks. Therefore, the XML firewall development presented in this paper is focused on the hardware acceleration in security processing and the optimizations in XML validation.

2.2. Embedded Network Processor and Hardware Acceleration

Intel IXP425 network processor is a highly integrated, versatile single-chip processor [6]. It contains an XScale core at 533MHz and three NPEs (Network Processing Engines), which can run their instruction streams in parallel. NPE B is capable of hardware acceleration for encryption (DES, 3DES and AES) and signature (HMAC-SHA1). It is used to accelerate the XML security processing in our system. Although XScale is not as powerful as best available standalone CPUs, IXP425 is a suitable embedded network processor with low cost and low power consumption for SME (Small and Medium Enterprise) solutions.

XML encryption ensures the confidentiality for data transportation. In addition, XML encryption can also provide a partial message encryption, which is more intelligent than SSL. As set forth, SSL is a point-to-point protocol used to ensure the confidentiality of messages. Although it is sufficient to meet the straightforward requirement, however, in XML-based services, like Web services, it will require that several pieces of information have different levels of confidentiality. Thus SSL is not adequate and XML encryption should be used.

Intrinsically XML encryption must have traditional encryption operation on the message, which is time consuming. We use 3DES algorithm to do encryption, which is supported by NPE B of IXP425. Performances of both hardware and software encryption are shown in Section 3. Hardware encryption means that, the sensitive message is parsed out by software, and then the plain text is converted to cipher text by hardware.

XML signature provides data integrity, authenticity, and non-repudiatability of XML messages like traditional signature in other applications. In addition, XML signature can provide a partial message signature, which is more flexible. We use HMAC-SHA1 algorithm to do signature, which is supported by NPE B of IXP425 and mentioned in W3C XML Signature [10]. This implementation is actually using a MAC (Message Authentication Codes) algorithm, which cannot provide non-repudiatability. Signature algorithm with non-repudiatability can be implemented by software. However, for the fair comparison to the hardware implementation, it is not used in our experiments. Performances of both hardware and software signature are shown in the following experiments in Section 3. This hardware acceleration is only used in conversion from plain text to HMAC value.

2.3. Performance Optimization of XML Validation

In most of XML-based services which require schema validation, there are lots of XML messages defined by one schema document. We don't have to process the schema document every time when validating the XML message. The following optimizations are based on the ideas of reusing the schema processing results.

(1) One optimization applied to XML validation is to avoid schema self-check. The XML schema documents, which describe the structures of XML messages, are written in XML grammar with some specific structures, thus they should be validated in XML firewall before they are used to check XML messages. However, this is a redundant work in most of cases when the schema documents are constructed by the administrators in advance and have been already checked. Thus the system does not need to check the validity of the schema document in the validation processing unless the schema document is changed or not trusted. Based on this observation, a simple TDL (Trusted Document List) is deployed for performance optimization.

(2) Schema documents can be stored in the server, some of which are used frequently. With this observation, we can optimize performance by preprocessing those most frequently used schema documents into corresponding tree structures in memory for the validation. These structures are named SGP (Schema Grammar Pool). In Section 3, we evaluate the performance enhanced by SGP in XML validation.

When SGP is not in place, TDL is introduced into this system in default; otherwise, TDL is unnecessary. In this paper, only impact of SGP is discussed in design and measured in experiments.

2.4. Choice of Processing Schemes

In this system, there are two schemes that can be applied to the whole process. One is using SAX (Simple API for XML) [13] to parse the XML message with schema validation and the other is using DOM (Document Object Model) [12]. We implement our XML firewall with DOM scheme.

The reason is that SAX is event-based and processing XML message like a pipeline. It validates XML message faster than DOM. But SAX doesn't construct any structure for the XML message in memory. So there are no results that can be used in security processing. DOM manages a tree structure in memory after XML validation. We can use this tree to manipulate XML message effectively for security processing.

3. Experiments and Analysis

Based on DOM scheme, we present some comparative experiments' results and corresponding analysis of XML firewall on the embedded network processor, IXP425.

3.1. Experiment Conditions

The hardware platform used in our experiments is IXP425 with Intel XScale core 533 MHz, and 64 MB memory.

The software kits used in this experiment are listed in Table 2.

Table 2. Software list

Function	Tools in IXP425
Platform	Snapgear-3.3.0
XML Identification	Xalan-c_1_9_0
XML Validation	Xerces-c _2_6_0
Hardware Signature	IPL_ixp400AccessLibraryWithCrypto-2_0
Hardware Encryption	IPL_ixp400AccessLibraryWithCrypto-2_0

The experiments are based on the following assumptions.

Firstly the XML messages in the experiments are generated manually according to a fixed structure in which the plain text to be signed and encrypted takes about more than 90% of the whole message in size. There are four different sizes of XML messages in the experiments, which are 1K, 5K, 20K and 100K bytes. There are two different sizes of schema documents in the experiments, which are 1K and 20K bytes.

Secondly the XML message has been processed for 100 times in each test. The performance is measured by processing time per message and is calculated by the average of three trails.

At last, crypto keys are allocated and known by both sender and receiver in advance. Actually the keys are stored in local hard disk and can be obtained when needed.

3.2. Experiment Results and Analysis

Experiments are carried out in such an XML security system. The impacts on processing speed of two optimizations, schema preprocessing and hardware acceleration, are evaluated with experimental data. We also present the performances of all four functions integrated in this system and illustrate corresponding analysis.

(1) Schema preprocessing

Figure 2 shows the performance improvement of XML schema validation with schema preprocessing mentioned in Section 2.3. The SGP created in the preprocessing is implemented by Xerces-c 2.6.0. The XML messages are 1KB in size. The schema documents vary in size. This paper presents two comparative experiments with 1KB and 20KB schema documents.

The X axis denotes the size of the schema document. The Y axis denotes the processing time in milliseconds per XML message.

From the figure we can see that the performance of XML schema validation is improved significantly by using SGP which is created in the schema preprocessing. The reason is that the schema processing parses the schema documents into tree structures in memory, which is a time-consuming part in XML validation.

The figure also shows that the processing time without schema preprocessing is increasing rapidly as the size of schema document increases from 1KB to 20KB. The processing time with schema preprocessing is increasing more slowly. The reason is that the increase of schema document' size has more influence on the schema preprocessing than that on XML checking.

As set forth, the schema documents can be stored in the firewall. So the conclusion from the experiments is that, schema preprocessing should be introduced into the XML firewall to enhance the performance of XML validation providing there is sufficient memory, and the schema documents are frequently reused.

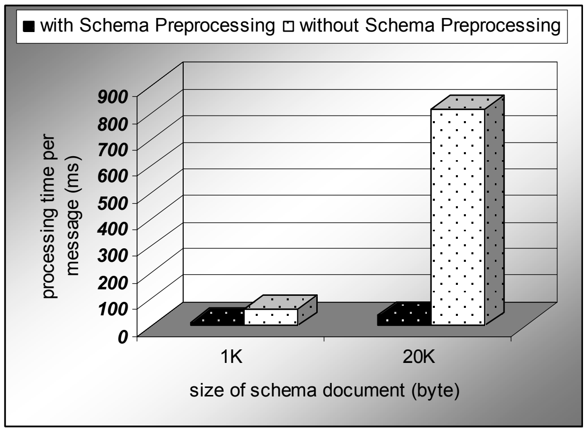

Figure 2. with vs. without schema preprocessing

(2) Hardware acceleration

The algorithms, parameters and develop kits used in the software and the hardware implementation are listed in Table 3.

Table 3. Parameters in signature and encryption

	Algorithm	Secret Key	IV	Develop Kit
Software sign	Hmac_Sha1	preshared	0	Openssl_0.9.8
Hardware sign	Hmac_Sha1	preshared	0	IPL_ixp400AccessLibraryWithCrypto-2_0
Software enc	Triple DES	preshared	Random	Freeswan1.92
Hardware enc	Triple DES	preshared	Random	IPL_ixp400AccessLibraryWithCrypto-2_0

The work flow of signature or encryption is shown in Figure 3. Hardware acceleration can be applied in shaded step "Sign/Enc", which is one of the five steps in XML security processing.

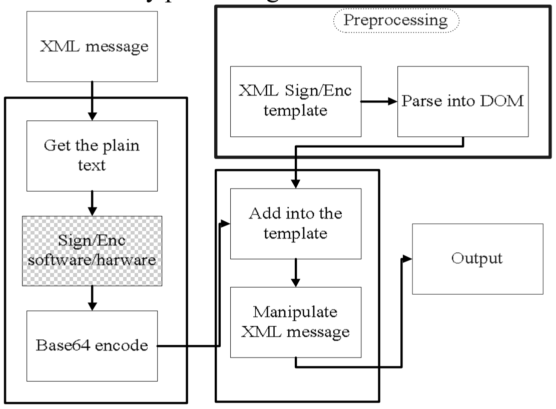

Figure 3. Work flow of signature or encryption

Figure 4 and Figure 5 show the performance improvement of hardware acceleration supported by NPE B of IXP425. The X axis denotes the size of the XML messages in the test. The Y axis denotes the processing time in milliseconds per XML message.

The plain text to be signed or encrypted is the root element's content, which takes most part of the XML message. So the X axis can also denote the size of the plain text approximately.

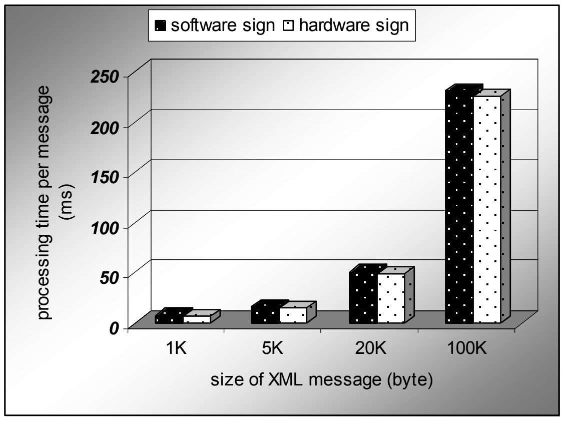

Figure 4. Signature: software vs. hardware

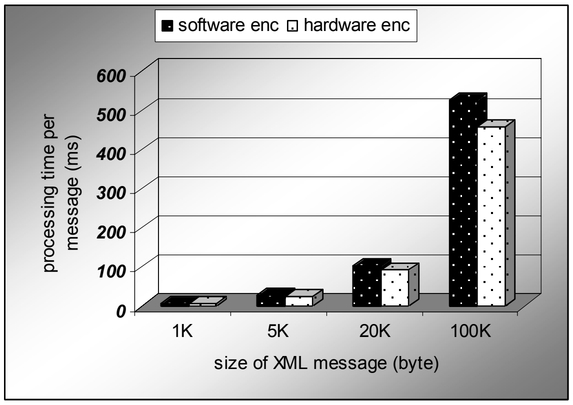

Figure 5. Encryption: software vs. hardware

From the figures we can see that the hardware acceleration can enhance the performances of XML signature and encryption. The processing time for signature is reduced by 3~10%, and the processing time for encryption is reduced by 7-15%.

There is a trend that the enhancement of performance is more and more significant as the size of XML message becomes larger. The first reason is that hardware acceleration is good at characters operation for security algorithm, such as rotation and shift, but the characters passing between processing engines adds overhead comparing with software operation. To be taken together, the improvement of hardware acceleration is better as the size of XML message, to be more exact, the size of plain text becomes larger. The second reason is that the step "Sign/Enc" shown in Figure 3 takes more and more proportion in time of the whole security processing, as the size of plain text increases. So the improvement of hardware acceleration is more significant for XML message with plain text in larger size.

(3) Overall Performance

In our system, XML identification has included XML message parsing. XML validation has used SGP. XML security processing has utilized hardware acceleration. The work flow of XML decryption and XML signature verification is shown in Figure 6, as also described in W3C Recommendations [9][10].

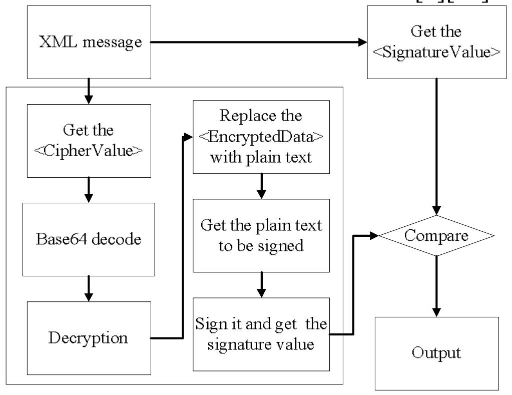
XML message
Get the <SignatureValue>
Get the <CipherValue>
Replace the <EncryptedData> with plain text
Base64 decode
Get the plain text to be signed
Compare
Decryption
Sign it and get the signature value
Output

Figure 6. Work flow of XML decryption and XML signature verification

Comparing XML signature with signature verification, we can see that there is one step in signature, which doesn't exist in signature verification. That is step "Manipulate XML message" in Figure 3. There are no more manipulations on XML message after the plaintext is signed in signature verification. The results of the experiments show that signature verification is faster than signature. But as the size of XML message increases, the performances of the two parts are similar. The gap between them is no more than 1% for 100K XML message. The reason is that, the processing time of "Manipulate XML message" step in signature doesn't increase so much as that of other steps, when the size of XML message becomes larger.

Comparing XML encryption with decryption, we can see that the two parts are opposite in procedure. But the performance shows that decryption is slower than encryption for XML message in any size. The reason is that the conversion from plain text to DOM tree in decryption is slower than the conversion from DOM tree to plain text in encryption.

Figure 7 shows the performance of all the XML processing functions. They have been optimized as presented above. The X axis denotes different functions in XML firewall. The Y axis denotes the processing time in milliseconds per XML message. Different shades denote different sizes of XML messages.

From the figure we can see that the most time consuming part is XML identification, which includes XML parsing and XPath query [7], after the other three parts are optimized. XML parsing, which is used in both identification and validation, is the focus of attention on the performance of XML firewall.

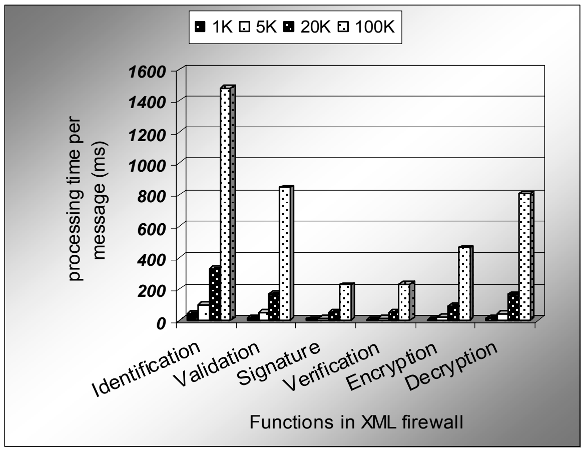

Figure 7. Performances of XML processing functions in XML firewall

4. Future Work

In terms of hardware, IXP425 has not been fully utilized by this system. The software design has few considerations for hardware features. We only deploy hardware acceleration in XML security processing. Optimizations based on hardware are important for the performance. For example, the NPE B and XScale core are running in parallel. Software can be designed to implement multi-thread program to hide latency in the NPE B and XScale.

In terms of processing scheme, global optimization can be applied considering the interactions of the four functions. The XML identification can merge with XML validation, for they both need XML parsing. At least the results of XML identification can be used in XML validation. From the experiments we can see that the XML parsing, which converts XML message into a specific structure in memory, is time-consuming. So the system should link different functions together in order to avoid unnecessary parsing. More optimizations based on processing scheme should be introduced according to different applications. For example, the conversion from plain text to DOM tree in XML decryption can be disabled if the XML signature is not necessary in some applications, because this DOM tree is used to find plain text to be signed in the XML message for XML signature verification.

To better evaluate the system, the XML firewall should be deployed on a network and tested in a real environment. Designing processing scheme according to statistical quantities can also improve the performance of the system.

Finally, the algorithms of XML parsing and XML schema validation will be investigated and some new algorithms are expected to enhance the performance of XML firewall.

5. Summary

Our XML firewall on embedded network processor is designed to secure XML-based services, like Web services. We use DOM scheme to implement it on Intel IXP425 and integrate four basic functions, XML identification, XML validation, XML encryption and decryption, XML signature and signature verification, to make an XML security system. Two main optimizations, schema preprocessing and hardware acceleration, have been introduced to the basic functions. The experimental results have shown the improvements brought from the two optimizations. The performances can be further improved by hardware utilization, other processing schemes, new algorithms for parsing and validation.

Acknowledgement

The author would like to thank Mr. Quan Huang for his help in the implementation work. Thanks also due to colleagues at Network Security Lab of Research Institute of Information Technology for their generous suggestions and encouragement.

References

[1] J. Bloomberg and R. Schmelzer, "A Guide to Securing XML and Web Services," White Paper of Zapthink, Jan 2004.

[2] A. Jaquith, "Application Assurance Platforms Arise from Web App Firewall Market's Ashes," Yankee Group, Dec 2005.

[3] "XML Application Firewalls," White Paper of Westbridge, Jan 2002.

[4] "Security within the XML Infrastructure," White Paper of Reactivity, Jan 2005.

[5] "Forum Sentry 1504 XML Security Appliance," Test Summary by Tolly Group, No. 203777, Sep 2003.

[6] "Intel IXP425 Network Processor," Product Brief, http://www.intel.com/design/network/prodbrf/27905104.pdf

[7] W3C XPath, http://www.w3.org/TR/xpath

[8] W3C XML Schema, http://www.w3.org/XML/Schema

[9] W3C XML Encryption, http://www.w3.org/Encryption/2001/

[10] W3C XML Signature, http://www.w3.org/Signature/

[11] OASIS WS-Security, http://www.oasis-open.org/committees/wss

[12] W3C DOM, http://www.w3.org/DOM/

[13] SAX Project, http://www.saxproject.org/