Avaliação de Desempenho de Algoritmos Criptográficos em Web Services Utilizando WS-Security

Douglas Rodrigues, Júlio C. Estrella, Kalinka R. L. J. C. Branco
Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo
São Carlos - SP - Brasil - 13560-970
{douglasr, jcezar, kalinka}@icmc.usp.br

ABSTRACT

In this paper we propose an evaluation and comparison of guidelines and techniques that allow not only the creation of secure Web services, but also the validation of the services used to determine whether the application has the desired characteristics related to performance and security. In this sense it is crucial evaluating the cryptographic algorithms and key length used. The results obtained allow to determine, based on specified objectives, the impact of security mechanisms used in application performance.

RESUMO

Neste artigo é proposta uma avaliação e comparação de diretrizes e técnicas que permitam não somente a criação de *Web services* seguros, mas também a validação dos serviços utilizados para determinar se a aplicação possui as características almejadas relacionadas ao desempenho e à segurança. Neste sentido é primordial avaliar os algoritmos criptográficos e o comprimento das chaves utilizadas. Os resultados obtidos permitem determinar, com base nos objetivos especificados, qual o impacto dos mecanismos de segurança utilizados no desempenho da aplicação.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General - Security and Protection

General Terms

Security, Performance

Keywords

SOA, Security, Performance

1. INTRODUÇÃO

A ideia básica da arquitetura orientada a serviço (*Service-Oriented Architecture* - SOA) tem recebido significante preocupação e atenção da comunidade de projeto e desenvolvimento de software. Desta forma, os *Web services* têm sido a implementação de SOA mais comumente utilizada [8].

A interoperabilidade proporcionada pelos *Web services* é fundamental para a integração de aplicações baseadas na *Web* e na Internet, sendo a mesma alcançada por meio da utilização de padrões baseados em XML (*eXtensible Markup Language*), tais como SOAP (*Simple Object Access Protocol*), WSDL (*Web Services Description Language*) e UDDI (*Universal Description, Discovery and Integration*). O SOAP é utilizado para a transferência de dados entre os serviços, enquanto que a WSDL define um esquema XML para descrever os serviços disponíveis e a especificação UDDI define um modo de publicar e descobrir informações sobre um serviço específico em um diretório ou registro de serviços [13]. Ainda em relação à interoperabilidade, esta permite a integração de aplicações, entretanto, não garante a confidencialidade das informações que trafegam pela Internet, as quais podem ser sigilosas. Deste modo, garantir a segurança das informações é uma necessidade primordial quando se utiliza os *Web services*, uma vez que seus fluxos de negócio, processos e arquiteturas internas podem ficar expostos.

Tecnologias como SSL/TLS (*Secure Sockets Layer/Transport Layer Security*) [9] [6] visam prover segurança ponto-a-ponto, porém não garantem segurança fim-a-fim. Tal segurança é necessária em um ambiente de *Web services*, pois as mensagens SOAP podem trafegar por diversos *Web services* intermediários antes de atingir o destinatário final. Portanto, se a criptografia for utilizada apenas na camada de transporte, as informações serão reveladas aos *Web services* intermediários pelos quais as mensagens passaram, de modo proposital ou por meio de lacunas existentes entre uma sessão segura e outra [18]. Nas Figuras 1 e 2 são ilustrados os contextos de segurança em uma configuração ponto-a-ponto e uma configuração fim-a-fim, respectivamente.

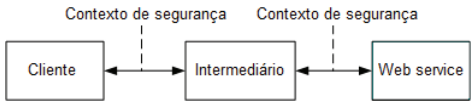

Figura 1: Segurança ponto-a-ponto [19].

Para que a segurança seja garantida no ambiente de *Web services* novos mecanismos de segurança devem ser considerados, como, por exemplo, o *WS-Security*. Este é um padrão

WebMedia'11: Proceedings of the 17th Brazilian Symposium on Multimedia and the Web. Full Papers.
October 3 -6, 2011, Florianópolis, SC, Brazil.
ISSN 2175-9642.
SBC - Brazilian Computer Society

Figura 2: Segurança fim-a-fim [19].

da OASIS (*Organization for the Advancement of Structured Information Standards*) que visa à segurança fim-a-fim das mensagens SOAP, provendo confidencialidade, integridade e autenticidade para as mesmas no nível de mensagem [20].

Entretanto, a utilização de tais padrões de segurança causa significativa sobrecarga no desempenho dos *Web services*. A preocupação com o desempenho de *Web services* seguros é legitimada pelo fato de que as especificações de segurança aumentam consideravelmente o tamanho da mensagem SOAP, principalmente o cabeçalho. Além disso, a adição dos elementos XML relacionados à segurança acarreta não apenas maior consumo de banda da rede para o transporte das mensagens SOAP, mas também consumo adicional de CPU para o processamento do documento XML e das operações necessárias à sua segurança [16] [7] [10].

O objetivo deste artigo é apresentar uma avaliação e comparação de diretrizes e a adequação de técnicas que permitam não somente a criação de *Web services* seguros, mas também a validação dos serviços utilizados para determinar se a aplicação final possui as características desejadas no que diz respeito ao desempenho e à segurança. Neste sentido é primordial avaliar os algoritmos criptográficos e o comprimento das chaves utilizadas. Adicionalmente, são determinados quais algoritmos criptográficos e suas chaves associadas impõem o menor impacto em termos de desempenho no contexto dos *Web services*. Tais algoritmos foram estudados e discutidos de modo a determinar quais deles, ou a combinação dos mesmos, garantem segurança fim-a-fim com um desempenho satisfatório.

O restante deste artigo está organizado da forma a seguir. A Seção 2 discute os trabalhos relacionados. A Seção 3 descreve a metodologia utilizada na avaliação de desempenho. Os resultados obtidos são analisados na Seção 4. E por fim, a Seção 5 conclui o artigo apresentando as considerações finais.

2. TRABALHOS RELACIONADOS

Em [5], é destacada uma avaliação de desempenho para *Web services* considerando aspectos de segurança. Os autores se concentram principalmente na variabilidade dos tamanhos de mensagens e na aplicação de algumas políticas de segurança, tais como políticas de criptografia, assinatura digital e criptografia de uma mensagem assinada, utilizando para tanto uma infraestrutura de chaves simétricas e assimétricas. Porém, eles não consideram as principais especificações de segurança para *Web services*, tampouco comparam os diversos algoritmos de criptografia existentes. A necessidade de se avaliar as especificações de segurança, frente aos principais tipos de ataques proferidos a este tipo de plataforma, e também a necessidade da utilização destas especificações como principais contramedidas a estes ataques, faz com que estudos mais específicos devam ser realizados.

Em [1], os autores comparam o desempenho de diversos mecanismos de segurança WSIT (*Web Services Interoperability Technologies*). O WSIT foi desenvolvido em um esforço conjunto entre a Sun e a Microsoft com o objetivo de permitir a interoperabilidade entre serviços Java e .Net e melhorar a qualidade de serviço. Estes mecanismos são aplicados em um *Web service* simples com a finalidade de testá-los com diferentes tamanhos de mensagens SOAP, de 1 *byte* a 1 *Mbyte*. Os resultados destes experimentos mostraram que mecanismos de segurança da camada de transporte, como o SSL, são consideravelmente mais velozes do que os mecanismos de segurança em nível de mensagem. Além disso, diferentemente dos mecanismos de segurança da camada de transporte, os protocolos de segurança em nível de mensagem possuem o problema de escalabilidade quando mensagens grandes são trocadas. Devido à necessidade de garantia de segurança fim-a-fim é que são apresentados estudos neste sentido.

Em [14] é realizada uma avaliação de desempenho comparando *Web services* e RMI (*Remote Method Invocation*), incluindo ainda suas variantes seguras, respectivamente, *WS-Security* e RMI-SSL. Com os resultados obtidos, pode-se observar que *Web services* e *WS-Security* possuem um desempenho inferior quando comparados ao RMI e RMI-SSL, devido principalmente ao tamanho das mensagens trocadas e à sobrecarga causada pelo processamento das mesmas. Entretanto, cabe ressaltar que o crescimento do número de serviços é notório, e se torna cada vez mais evidente a necessidade de se melhorar e avaliar a relação segurança/desempenho do *WS-Security*, entre outros padrões de *Web services*.

3. METODOLOGIA PARA A AVALIAÇÃO DE DESEMPENHO

Esta seção apresenta a metodologia utilizada visando avaliar o desempenho da criptografia e assinatura digital em mensagens SOAP, efetuando a variação dos algoritmos criptográficos, assim como o comprimento de suas chaves. A segurança foi provida em nível de mensagem por meio do *WS-Security*.

3.1 Configuração do Ambiente de Testes

Na Tabela 1 é listada a infraestrutura computacional utilizada nos experimentos executados. Neste caso o ambiente físico para execução dos experimentos é composto por quatro máquinas que possuem configurações idênticas. Uma destas máquinas é utilizada como provedor de serviços e as demais são utilizadas como clientes.

Tabela 1: Elementos de hardware.

Componente	Qtde.	Características
Provedor de serviços	1	Processador Intel Core 2 Quad Q6600 2,4 GHz, 2 GB de memória RAM, 120 GB de HD.
Clientes	3	Processador Intel Core 2 Quad Q6600 2,4 GHz, 2 GB de memória RAM, 120 GB de HD.
Switch	1	Gigabit 3Com Baseline 2916

A implementação dos serviços e dos clientes foi realizada utilizando a linguagem de programação Java. Além disso, utilizou-se o servidor de aplicações Apache Tomcat 7.0.6, o motor de processamento SOAP Apache Axis2 1.5.4 e o módulo de segurança Rampart 1.5.1.

3.2 Planejamento dos Experimentos

Nesta etapa são definidos os fatores ou as características que serão avaliadas, examinando quais delas podem influen-

ciar no desempenho de um determinado sistema computacional. Além disso, é necessário determinar a quantidade de dados coletados, a quantidade de replicações dos experimentos e a interação entre os fatores. A definição da quantidade de replicações é imprescindível para obter validação estatística dos resultados. Existe uma variedade de termos empregados durante a etapa de projeto e análise de experimentos, como variável de resposta, fatores, níveis e interação. Variável de resposta representa o resultado (saída) de um experimento. Normalmente, a variável de resposta é a medida de desempenho do sistema. Fatores são as variáveis que afetam a variável de resposta do sistema. Níveis são os valores que um determinado fator pode assumir. E a interação indica a dependência entre os fatores avaliados [12].

Existem alguns modos de realizar o planejamento de experimentos e o utilizado neste artigo é o planejamento fatorial completo [12]. Neste tipo de planejamento são utilizadas todas as combinações considerando todos os fatores e níveis. Assim, é possível avaliar todos os fatores, determinar o efeito de cada fator nos experimentos e verificar as interações entre eles. A principal desvantagem do fatorial completo é a grande quantidade de experimentos que devem ser realizados.

Tal planejamento foi escolhido devido à utilização de apenas três fatores com no máximo três níveis cada. O primeiro fator considerado é o algoritmo de chave simétrica, utilizado para criptografar os dados da mensagem, e possui três níveis: AES 192, AES 256 e 3DES. O segundo fator refere-se ao algoritmo de chave pública, empregado na criptografia da chave secreta (chave de sessão), o qual possui dois níveis: RSA-OAEP [11] e RSA 1.5 [15], ambos com chaves de 1024 *bits*. E por fim, o terceiro fator é definido como a quantidade de clientes, novamente com dois níveis: 1 e 3. Os experimentos realizados foram elaborados de acordo com a especificação dos fatores e níveis, e são apresentados na Tabela 2.

Tabela 2: Experimentos realizados.

Exp.	Tipo	Clientes
1	AES192 - RSA-OAEP	1
2	AES256 - RSA-OAEP	1
3	3DES - RSA-OAEP	1
4	AES192 - RSA1.5	1
5	AES256 - RSA1.5	1
6	3DES - RSA1.5	1
7	AES192 - RSA-OAEP	3
8	AES256 - RSA-OAEP	3
9	3DES - RSA-OAEP	3
10	AES192 - RSA1.5	3
11	AES256 - RSA1.5	3
12	3DES - RSA1.5	3

Neste ponto é necessário um esclarecimento sobre os algoritmos RSA 1.5 e RSA-OAEP. O primeiro é uma implementação do algoritmo RSA e o segundo requer uma melhor explicação. Basicamente, o RSA é vulnerável ao ataque de texto cifrado escolhido. Neste tipo de ataque, o criptoanalista possui uma grande quantidade de mensagens e seus equivalentes criptografados e, além disso, pode produzir uma mensagem criptografada específica para ser decriptada e obter o resultado gerado, com a finalidade de deduzir as chaves utilizadas. Visando impedir este tipo de ataque, a RSA Security Inc., um fornecedor de RSA e antigo mantenedor da sua patente, aconselha modificar o texto aberto por meio de um procedimento denominado preenchimento ideal de criptografia assimétrica (*Optimal Asymmetric Encryption Padding* - OAEP) [17] [4]. Uma explicação completa sobre o ataque citado e o OAEP pode ser encontrada em [3] e [21].

É importante destacar ainda a inclusão de um *timestamp* (um registro de data e hora, útil para prevenir ataques do tipo repetição em um servidor) na mensagem e a utilização do algoritmo RSA/SHA-1 em todos os experimentos para a realização da assinatura digital. Inicialmente, pretendia-se que a função de *hash* também fosse considerada um fator nesta avaliação de desempenho. Assim, os níveis deste fator seriam os algoritmos SHA-1 e SHA-256. Contudo, durante a pesquisa descobriu-se que a implementação do SHA-256 no módulo de segurança Rampart não funciona corretamente, conforme pode ser visto em [2]. A própria Apache considera este um defeito com alta prioridade, mas que desde 2009 está sem resolução, o que impediu a avaliação deste fator neste artigo.

Neste estudo a variável de resposta escolhida foi o RTT (*Round Trip Time*), ou seja, o tempo gasto desde a requisição do cliente até o recebimento da resposta. O RTT é bastante utilizado quando se pretende avaliar *Web services*, como em [14] e [1]. Cada experimento foi replicado 30 vezes a fim de garantir uma validação estatística. Esta quantidade de replicações foi suficiente para obter intervalos de confiança relativamente baixos, permitindo assim uma comparação adequada dos resultados.

4. ANÁLISE DOS RESULTADOS

A análise dos resultados consistiu na observação das médias dos RTTs obtidos na execução dos experimentos. Para que houvesse uma validação estatística, foram calculados os desvios padrão e os intervalos de confiança de 95%. Cabe ainda ressaltar que os gráficos de colunas apresentados nesta seção indicam valores em milissegundos no eixo Y.

4.1 Comparação entre Algoritmos de Chave Simétrica

Esta seção apresenta algumas análises sob o ponto de vista dos algoritmos de chave simétrica avaliados. Na Tabela 3 e no gráfico de Figura 3 são ilustrados os resultados coletados com apenas 1 cliente, comparando os algoritmos de chave simétrica combinados com o algoritmo de chave pública RSA-OAEP. Na Tabela 4 e no gráfico da Figura 4 são apresentados os resultados para comparação entre os algoritmos de chave simétrica em combinação com o algoritmo de chave pública RSA 1.5, também com 1 cliente. Nos dois casos, é possível notar que o *Web service* utilizando o algoritmo AES 192 obtém o melhor desempenho em termos de tempo, com uma ligeira vantagem em relação ao AES 256. Neste contexto, o pior desempenho pertence ao algoritmo 3DES. No caso do RSA-OAEP, o 3DES tem um acréscimo no RTT de 10,76% quando comparado ao AES 192, enquanto no caso do RSA 1.5 este acréscimo em relação ao AES 192 é de 10,92%. Os intervalos de confiança exibidos para os algoritmos AES 192 e AES 256 estão sobrepostos em ambos os casos, indicando que não há diferença estatística entre eles. Portanto, estes resultados levam a concluir que em termos de segurança e rapidez, a melhor escolha neste contexto é o AES 256.

Com relação aos resultados obtidos com 3 clientes, na Tabela 5 e no gráfico da Figura 5 são exibidos os valores para comparação entre os algoritmos de chave simétrica em conjunto com o algoritmo RSA-OAEP e na Tabela 6 e no gráfico

Tabela 3: Algoritmos de chave simétrica com RSA-OAEP (1 cliente).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA-OAEP	4986,67	117,87	42,18
AES256 - RSA-OAEP	5015,70	163,33	58,45
3DES - RSA-OAEP	5523,23	129,10	46,20

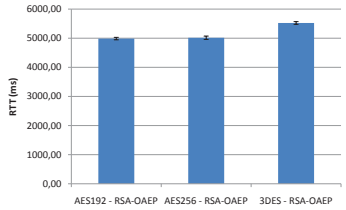

Figura 3: Algoritmos de chave simétrica com RSA-OAEP (1 cliente).

Tabela 4: Algoritmos de chave simétrica com RSA 1.5 (1 cliente).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA1.5	4998,93	147,43	52,76
AES256 - RSA1.5	5025,37	131,79	47,16
3DES - RSA1.5	5545,13	105,63	37,80

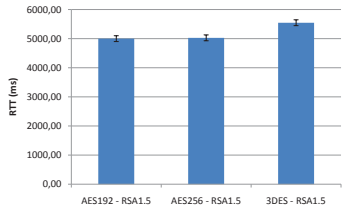

Figura 4: Algoritmos de chave simétrica com RSA 1.5 (1 cliente).

da Figura 6 são apresentados os valores comparando os algoritmos de chave simétrica combinados com o algoritmo RSA 1.5. Em ambos os casos, os resultados evidenciam que o menor RTT pertence ao *Web service* que emprega o algoritmo AES 192. O AES 256 apresenta o segundo menor RTT, com um aumento em relação ao AES 192 de 4,47% no caso do RSA-OAEP e de 4,26% no caso do RSA 1.5. O pior desempenho em ambos os casos está relacionado ao 3DES. Quando comparado ao AES 192, há uma piora no RTT de 14,59% e de 15,12% nos casos do RSA-OAEP e do RSA 1.5, respectivamente. Observa-se ainda nos gráficos de ambos os casos que os intervalos de confiança não se encontram sobrepostos, significando que os resultados são estatisticamente diferentes.

Tabela 5: Algoritmos de chave simétrica com RSA-OAEP (3 clientes).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA-OAEP	5072,09	54,05	19,34
AES256 - RSA-OAEP	5298,91	95,16	34,05
3DES - RSA-OAEP	5811,87	179,98	64,40

É possível notar alguns comportamentos similares nos casos apresentados. O 3DES sempre se apresenta como o algoritmo de chave simétrica mais ineficiente em todos os casos analisados. Tal ineficiência justifica-se pelo fato da execução do algoritmo DES por três vezes consecutivas, tornando-o lento em relação a outros algoritmos de chave simétrica.

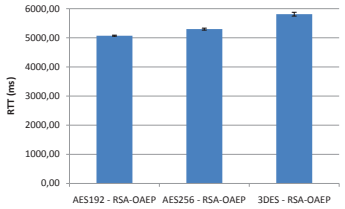

Figura 5: Algoritmos de chave simétrica com RSA-OAEP (3 clientes).

Tabela 6: Algoritmos de chave simétrica com RSA 1.5 (3 clientes).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA1.5	5109,30	108,20	38,72
AES256 - RSA1.5	5326,71	124,12	44,42
3DES - RSA1.5	5881,92	186,28	66,66

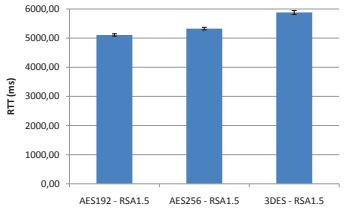

Figura 6: Algoritmos de chave simétrica com RSA 1.5 (3 clientes).

4.2 Comparação entre Algoritmos de Chave Pública

A análise contida nesta seção apresenta os resultados sob o ponto de vista dos algoritmos de chave pública. Com apenas 1 cliente, na Tabela 7 e no gráfico da Figura 7 podem ser vistos os resultados obtidos comparando os algoritmos de chave pública em conjunto com o AES 192, na Tabela 8 e no gráfico da Figura 8 estão os resultados comparando os algoritmos de chave pública combinados com o AES 256 e finalmente na Tabela 9 e no gráfico da Figura 9 encontram-se os valores obtidos para a comparação dos algoritmos de chave pública em conjunto com o 3DES. Nos três casos é possível notar um comportamento semelhante, onde a utilização do RSA-OAEP exibe o melhor desempenho, porém com pequena diferença em relação ao RSA 1.5. Com relação aos intervalos de confiança, pode-se observar a sobreposição dos mesmos, apontando que não existe diferença estatística entre eles.

Tabela 7: Algoritmos de chave pública com AES 192 (1 cliente).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA-OAEP	4986,67	117,87	42,18
AES192 - RSA1.5	4998,93	147,43	52,76

Os valores obtidos nos experimentos com 3 clientes estão

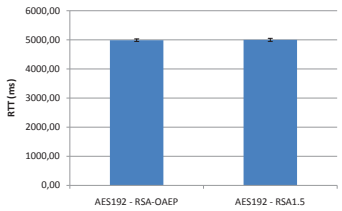

Figura 7: Algoritmos de chave pública com AES 192 (1 cliente).

Tabela 8: Algoritmos de chave pública com AES 256 (1 cliente).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES256 - RSA-OAEP	5015,70	163,33	58,45
AES256 - RSA1.5	5025,37	131,79	47,16

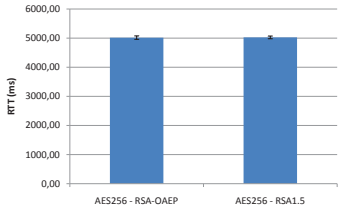

Figura 8: Algoritmos de chave pública com AES 256 (1 cliente).

Tabela 9: Algoritmos de chave pública com 3DES (1 cliente).

Tipo	RTT	Desv. Padrão	Int. Confiança
3DES - RSA-OAEP	5523,23	129,10	46,20
3DES - RSA1.5	5545,13	105,63	37,80

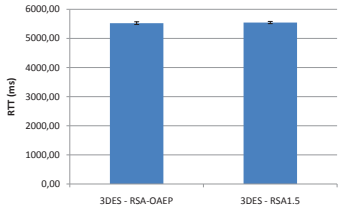

Figura 9: Algoritmos de chave pública com 3DES (1 cliente).

representados nas tabelas e figuras que seguem. Na Tabela 10 e no gráfico da Figura 10 são apresentados os resultados para comparação dos algoritmos de chave pública juntamente com o AES 192, na Tabela 11 e no gráfico da Figura 11 são apresentados os resultados comparando os algoritmos de chave pública com o AES 256 e, por fim, na Tabela 12 e no gráfico da Figura 12 estão os valores para comparação dos algoritmos de chave pública combinados com o 3DES. Novamente é possível encontrar um padrão no comportamento destes casos, onde o algoritmo RSA-OAEP possui o menor RTT. No entanto, com 3 clientes a diferença entre o RSA-OAEP e o RSA 1.5 aumenta em relação aos experimentos com 1 cliente. Pode-se observar pelas tabelas que os intervalos de confiança se sobrepõem apenas na comparação entre o RSA-OAEP e o RSA 1.5 combinados com o AES 256.

Tabela 10: Algoritmos de chave pública com AES 192 (3 clientes).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES192 - RSA-OAEP	5072,09	54,05	19,34
AES192 - RSA1.5	5109,30	108,20	38,72

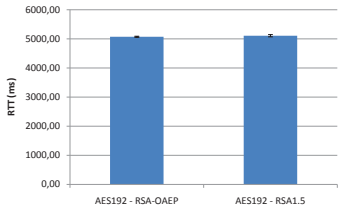

Figura 10: Algoritmos de chave pública com AES 192 (3 clientes).

Tabela 11: Algoritmos de chave pública com AES 256 (3 clientes).

Tipo	RTT	Desv. Padrão	Int. Confiança
AES256 - RSA-OAEP	5298,91	95,16	34,05
AES256 - RSA1.5	5326,71	124,12	44,42

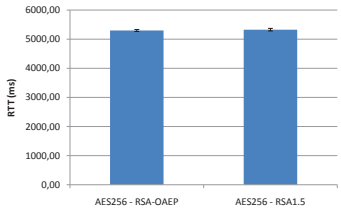

Figura 11: Algoritmos de chave pública com AES 256 (3 clientes).

Tabela 12: Algoritmos de chave pública com 3DES (3 clientes).

Tipo	RTT	Desv. Padrão	Int. Confiança
3DES - RSA-OAEP	5811,87	179,98	64,40
3DES - RSA1.5	5881,92	186,28	66,66

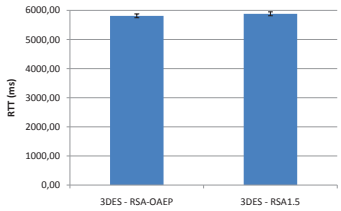

Figura 12: Algoritmos de chave pública com 3DES (3 clientes).

É importante destacar que em todos os casos apresentados nesta seção o algoritmo RSA-OAEP mostrou-se melhor em termos de desempenho, além de evitar o ataque de texto cifrado escolhido. Portanto, é possível concluir que o RSA-OAEP torna-se a melhor opção neste contexto.

4.3 Comparação entre Quantidade de Clientes

Nesta seção é realizada uma análise com foco na quantidade de clientes, ou seja, os resultados dos algoritmos avaliados são comparados com 1 e 3 clientes. Na Tabela 13 e no gráfico da Figura 13 são exibidos os resultados do AES 192 com RSA-OAEP. Na Tabela 14 e no gráfico da Figura 14 são comparados os resultados do AES 256 com RSA-OAEP. E, finalmente, na Tabela 15 e no gráfico da Figura 15 são apresentados os resultados do 3DES com RSA-OAEP. É possível perceber que o aumento do número de clientes acarreta degradação no desempenho de 1,71% para o AES 192 com RSA-OAEP. Para o AES 256 com RSA-OAEP esta queda no desempenho é de 5,65%. Enquanto para o 3DES com RSA-OAEP esta redução é de 5,23%. Em nenhuma destas comparações os intervalos de confiança se sobrepuseram, garantindo assim que os resultados são estatisticamente diferentes.

Tabela 13: RTT - AES192 - RSA-OAEP.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	4986,67	117,87	42,18
3	5072,09	54,05	19,34

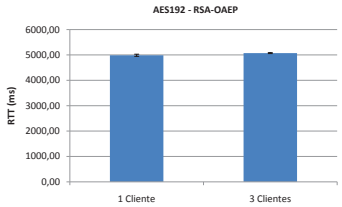

Figura 13: RTT - AES192 - RSA-OAEP.

Tabela 14: RTT - AES256 - RSA-OAEP.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	5015,70	163,33	58,45
3	5298,91	95,16	34,05

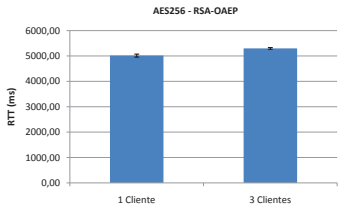

Figura 14: RTT - AES256 - RSA-OAEP.

Ainda com relação à comparação entre 1 e 3 clientes, na Tabela 16 e no gráfico da Figura 16 são apresentados os resultados do AES 192 com RSA 1.5. Os valores do AES 256 com RSA 1.5 são apresentados na Tabela 17 e no gráfico

Tabela 15: RTT - 3DES - RSA-OAEP.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	5523,23	129,10	46,20
3	5811,87	179,98	64,40

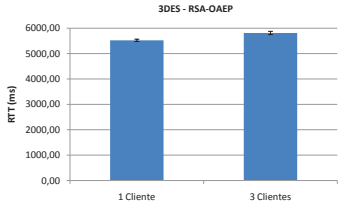

Figura 15: RTT - 3DES - RSA-OAEP.

da Figura 17. E por fim, os resultados do 3DES com RSA 1.5 são comparados na Tabela 18 e no gráfico da Figura 18. Com o aumento da quantidade de clientes, a redução no desempenho do AES 192 com RSA 1.5 é de 2,21%. A degradação do desempenho do AES 256 com RSA 1.5 é de 6% e do 3DES com RSA 1.5 é de 6,07%. Novamente, os resultados mostrados são estatisticamente diferentes, pois os intervalos de confiança não estão sobrepostos. Baseado nestes resultados, conclui-se que quando se pretende expandir a quantidade de clientes, tendo ainda a preocupação com o desempenho, deve-se dar preferência à utilização do AES 192. Pois, na medida em que o comprimento da chave aumenta, certamente haverá uma redução de desempenho que possivelmente causará um impacto negativo no funcionamento apropriado da aplicação.

Tabela 16: RTT - AES192 - RSA1.5.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	4998,93	147,43	52,76
3	5109,30	108,20	38,72

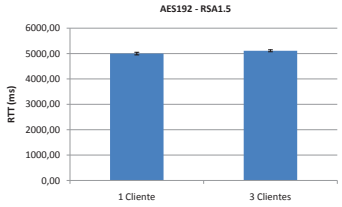

Figura 16: RTT - AES192 - RSA1.5.

Tabela 17: RTT - AES256 - RSA1.5.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	5025,37	131,79	47,16
3	5326,71	124,12	44,42

Tabela 18: RTT - 3DES - RSA1.5.

Qtde. Clientes	RTT	Desv. Padrão	Int. Confiança
1	5545,13	105,63	37,80
3	5881,92	186,28	66,66

Figura 17: RTT - AES256 - RSA1.5.

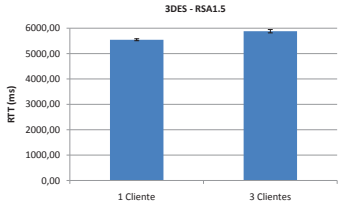

Figura 18: RTT - 3DES - RSA1.5.

4.4 Análise da Influência dos Fatores

Resumidamente, esta etapa é responsável por quantificar a influência de cada fator na execução dos experimentos, ou seja, determinar os fatores que mais influenciaram estes resultados, além de averiguar se a interação entre os fatores causou alguma influência significante.

A análise presente nesta seção realiza uma comparação da influência dos algoritmos de chave simétrica entre AES 192 e AES 256, AES 192 e 3DES e, por fim, AES 256 e 3DES em relação aos outros dois fatores considerados. Os gráficos apresentados a seguir ilustram a porcentagem de influência da cada um dos fatores, sendo estes representados pelas letras A, B e C, e as interações entre os fatores representadas pelas combinações das letras.

No gráfico da Figura 19 é ilustrada a influência dos fatores para os resultados obtidos com os algoritmos de chave simétrica AES 192 e AES 256. Pode-se notar que o fator clientes (C) exerce a maior influência, com 59,74%. Em seguida aparece o fator algoritmo de chave simétrica (A) com 24,5% de influência. A terceira maior influência pertence à interação entre os fatores algoritmo de chave simétrica e clientes (AC) com 14,83%. Os demais fatores e interações não possuem influências relevantes, não alcançando sequer 1%. Neste caso pode-se perceber que devido à pequena diferença entre os RTTs do AES 192 e do AES 256, a quantidade de clientes apresenta a maior influência nos resultados.

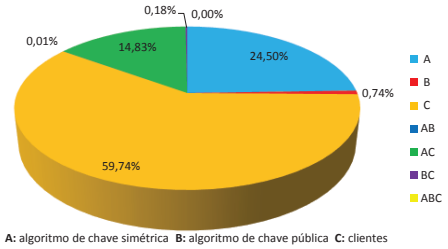

Figura 19: AES 192 vs. AES 256.

A influência dos fatores para os valores obtidos com os algoritmos AES 192 e 3DES é ilustrada na Figura 20. É possível observar que a maior influência pertence ao fator algoritmo de chave simétrica (A) com 88,37%, seguido pelo fator clientes (C) com 8,85%. A interação entre os fatores algoritmo de chave simétrica e clientes (AC) possui 2,42% de influência. Os fatores e interações restantes ficaram abaixo de 1% e não apresentam influência significativa. Devido ao RTT alto do 3DES e, consequentemente, a maior diferença entre os RTTs do AES 192 e do 3DES, o algoritmo de chave simétrica torna-se o fator mais influente.

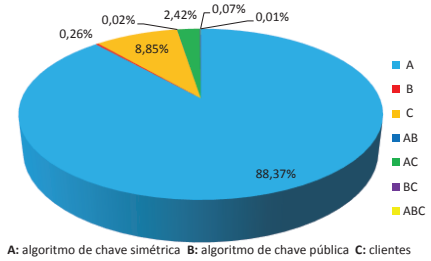

Figura 20: AES 192 vs. 3DES.

No gráfico da Figura 21 é exibida a influência de fatores para os valores coletados com os algoritmos AES 256 e 3DES. No gráfico é possível notar que o fator mais influente é o algoritmo de chave simétrica (A) com 74,65%. Logo após vem o fator clientes (C) com 24,89% de influência. Os demais fatores e interações não atingiram 1% e por isso não possuem influência significativa. Com o AES 256, a diferença entre o seu RTT e o do 3DES torna-se menor, com a consequente diminuição da influência do algoritmo de chave simétrica nos resultados, embora ainda permaneça como o fator mais influente.

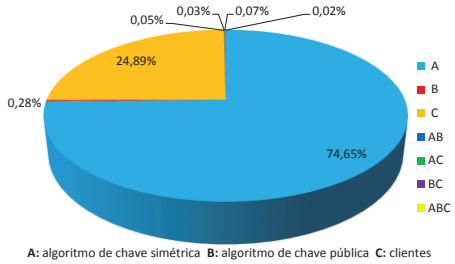

Figura 21: AES 256 vs. 3DES.

Com a análise da influência dos fatores, torna-se evidente que os fatores de maior influência são o algoritmo de chave simétrica e a quantidade de clientes. Diferentemente do algoritmo de chave simétrica, o fator algoritmo de chave pública causa pouca influência nos resultados, devido à sua utilização ser restrita à criptografia da chave secreta, ou seja, pequenas quantidades de dados.

5. CONCLUSÕES

Os problemas relacionados à adição de segurança em *Web services* já são bem conhecidos, tanto pelas operações de segurança e seu inerente custo de processamento, quanto pelo aumento de complexidade nas mensagens XML/SOAP devido à inserção de elementos de segurança, causando, consequentemente, maior consumo de banda da rede para transportar a mensagem e maior utilização de CPU para processamento da mesma. Além disso, é sabido que as técnicas de

segurança da camada de transporte, como o SSL, não são recomendadas para *Web services*, pois não garantem segurança fim-a-fim.

Desta forma, foram realizados experimentos com diferentes algoritmos de criptografia de chave simétrica e de chave pública, variando também o comprimento das chaves utilizadas, com o propósito de analisar os algoritmos avaliados e inferir quais deles causam a menor degradação em termos de desempenho, quais oferecem um maior nível de segurança e, principalmente, quais são os mais indicados quando se pretende balancear os níveis de desempenho e de segurança em determinadas situações. Adicionalmente, foram realizadas análises de influência dos fatores avaliados, as quais apontam os fatores que mais exerceram influência nos resultados dos experimentos realizados. Desta forma, os objetivos da avaliação de desempenho das técnicas e dos algoritmos de segurança também foram alcançados, uma vez que a análise dos resultados obtidos permitiu apontar a degradação no desempenho causada pelos mesmos.

Fundamentado nas análises dos resultados obtidos, é possível afirmar que dentre os algoritmos de chave simétrica avaliados, o AES 192 exibe o melhor desempenho em termos de velocidade. No entanto, em alguns casos apresentados, seu desempenho não possui diferença estatística em relação ao AES 256, o qual provê um maior nível de segurança a um custo relativamente semelhante no desempenho. Portanto, nestes casos, o algoritmo AES 256 pode ser considerado a melhor escolha. O 3DES, por sua vez, é o mais lento, devido ao seu modo de funcionamento. Em relação aos algoritmos de chave pública, deve-se preferir o uso do RSA-OAEP em detrimento do RSA 1.5, pois além de apresentar os menores RTTs, o RSA-OAEP ainda possui a função de prevenir contra ataques de texto cifrado escolhido. Ainda de acordo com os resultados obtidos, quando se eleva o número de clientes de um *Web service*, o algoritmo de chave simétrica que menos sofre degradação em seu desempenho é o AES 192, principalmente quando combinado com o algoritmo de chave pública RSA-OAEP. Para finalizar, quando se deseja um equilíbrio entre segurança e desempenho, as escolhas mais indicadas são o AES 192 ou AES 256 (dependendo da quantidade de clientes) como algoritmos de chave simétrica e o RSA-OAEP como algoritmo de chave pública.

Agradecimentos

Os autores agradecem o apoio do CNPq e da FAPESP.

6. REFERÊNCIAS

[1] B. Alrouh and G. Ghinea. A performance evaluation of security mechanisms for web services. In *Proceedings of the 2009 Fifth International Conference on Information Assurance and Security - Volume 02*, pages 715–718, Washington, DC, USA, 2009. IEEE Computer Society.

[2] Apache. Wrong signaturemethod and digestmethod generated in request in case of algoritm suite having sha256 hashing algorithm (example: Basic256sha256), 2009. The Apache Software Foundation. `https://issues.apache.org/jira/browse/RAMPART-216`.

[3] M. Bellare and P. Rogaway. Optimal asymmetric encryption - how to encrypt with rsa. In *Advances in Cryptology - EUROCRYPT'94*, pages 92–111. Springer-Verlag, 1995.

[4] A. Boldyreva, H. Imai, and K. Kobara. How to strengthen the security of rsa-oaep. *IEEE Transactions on Information Theory*, 56(11):5876–5886, 2010.

[5] S. Chen, J. Zic, K. Tang, and D. Levy. Performance evaluation and modeling of web services security. *IEEE International Conference on Web Services*, 0:431–438, 2007.

[6] T. Dierks and C. Allen. The tls protocol version 1.0, 1999. RFC 2246. `http://www.ietf.org/rfc/rfc2246.txt`.

[7] R. Engelen and W. Zhang. An overview and evaluation of web services security performance optimizations. In *IEEE International Conference on Web Services, 2008. ICWS '08*, pages 137–144, 2008.

[8] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[9] A. O. Freier, P. Karlton, and P. C. Kocher. The ssl protocol version 3.0, 1996. Internet Draft. `http://www.mozilla.org/projects/security/pki/nss/ssl/draft302.txt`.

[10] N. Gruschka, M. Jensen, L. Iacono, and N. Luttenberger. Server-side streaming processing of ws-security. *IEEE Transactions on Services Computing*, PP(99):1–14, 2011.

[11] R. Housley. Use of the rsaes-oaep key transport algorithm in the cryptographic message syntax (cms), 2003. RFC 3560. `http://www.ietf.org/rfc/rfc3560.txt`.

[12] R. K. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, April 1991.

[13] N. Josuttis. *SOA in Practice: The Art of Distributed System Design*. O'Reilly Media, Inc., 2007.

[14] M. B. Juric, I. Rozman, B. Brumen, M. Colnaric, and M. Hericko. Comparison of performance of web services, ws-security, rmi, and rmi-ssl. *The Journal of Systems and Software*, 79:689–700, May 2006.

[15] B. Kaliski. Pkcs #1: Rsa encryption version 1.5, 1998. RFC 2313. `http://www.ietf.org/rfc/rfc2313.txt`.

[16] H. Liu, S. Pallickara, and G. Fox. Performance of web service security. In *Proceedings of 13th Annual Mardi Gras Conference*, pages 1–8, Baton Rouge, Louisiana, 2005.

[17] J. Liu and J. Li. A novel key exchange protocol based on rsa-oaep. In *10th International Conference on Advanced Communication Technology, 2008. ICACT 2008*, volume 3, pages 1641–1643, 2008.

[18] M. Mashood and G. Wikramanayake. Architecting secure web services through policies. In *International Conference on Industrial and Information Systems, 2007. ICIIS 2007*, pages 5–10, 2007.

[19] Microsoft. Security in a web services world: A proposed architecture and roadmap, 2002. `http://msdn.microsoft.com/en-us/library/ms977312.aspx`.

[20] OASIS. Web services security (wss) tc, 2006. `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss`.

[21] D. Pointcheval. How to encrypt properly with rsa. *CryptoBytes*, 5(1):10–19, 2002.