PROSPECTUS
ISSN 2674-8576 Gestão e Tecnologia

Artigo Original

A ANÁLISE DE DESEMPENHO DAS FUNÇÕES CRIPTOGRÁFICAS IMPLEMENTADAS NA API OPENDATAPLANE (ODP)

PERFORMANCE ANALYSIS OF THE CRYPTOGRAPHIC FUNCTIONS IMPLEMENTED IN THE OPENDATAPLANE (ODP) API

João Gabriel Rangel Gonçalves[1]; Thales de Társis Cezare[2]

1- Estudante regular do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da FATEC de Mogi Mirim; *2- Mestre em Engenharia e Ciência dos Materiais, pela Universidade São Francisco (USF), Itatiba-SP; docente da FATEC de Mogi Mirim; docente na USF.*

Contato: thales.cezare@fatec.sp.gov.br

RESUMO

A evolução constante da tecnologia é, atualmente, inevitável, presente de várias formas em nossa sociedade, porém o custo dessas tecnologias pode valer muito para uma organização, um exemplo é a de redes de comunicação usando servidores de alto desempenho, que necessitam estar sempre em funcionamento, como switches, firewall, armazenamento de dados, criptografia, etc., tudo sendo usado em várias máquinas pode se tornar inviável, por ter um alto custo de hardware sujeito a falhas inesperadas. Problemas de alto custo, manutenção de hardware e baixa escalabilidade na infraestrutura de rede, podem ser resolvidos em muitos casos com uma rede virtualizada, reduzindo a necessidade de compra de vários equipamentos para realizar suas tarefas. Este projeto visa realizar estudos nos mecanismos de segurança da informação, entender suas funcionalidades, realizar testes de desempenho nas funções criptográficas da API OpenDataPlane (ODP), visa ainda, avaliar vários modelos de criptografia implementados na API ODP em um Raspberry PI 4.
Palavras-chave: Plano de dados. Redes definidas por software. Raspberry. Criptografia.

ABSTRACT

The constant evolution of technology is currently inevitable, present in many ways in our society, but the cost of these technologies can be very valuable for an organization, an

example is that of communication networks using high performance servers, which need to be always in operation, such as switches, firewall, data storage, encryption, etc., all being used on multiple machines can become impracticable, due to the high cost of hardware subject to unexpected failures. Problems with high cost, hardware maintenance and low scalability in the network infrastructure can be solved in many cases with a virtualized network, reducing the need to purchase several pieces of equipment to perform your tasks. This project aims to carry out studies on information security mechanisms, understand its functionalities, perform performance tests on the cryptographic functions of the OpenDataPlane API (ODP), and also aims to evaluate various cryptography models implemented in the ODP API on a Raspberry PI 4.
Keywords: Data plan. Software defined networks. Raspberry. Cryptography.

INTRODUÇÃO

O OpenDataPlane (ODP) é uma API que permite que implementações forneçam independência da plataforma, aceleração automática de hardware e dimensionamento de CPU para aplicativos de rede, uma API que descreve um modelo funcional para aplicativos de plano de dados. Uma das especificações interessantes do ODP é a capacidade de receber, manipular e transmitir pacotes de dados, as APIs do ODP não possuem modalidade preferida, no qual permite inovar a forma como essas funções podem ser realizadas em várias plataformas que oferecem implementações do ODP.

O ODP também fornece APIs de criptografia exigidos pelos aplicativos, contendo vários modelos de criptografia, é possível fazer testes de desempenho desses modelos e mensurar a viabilidade de implementação.

Sabe-se que aplicações de redes construídas com o ODP pode se tornar uma ferramenta poderosa se implementada corretamente. O modelo de segurança da informação é essencial para qualquer aplicação, uma vez que é indispensável tendo em vista a tecnologia atual. Com o seu código aberto, a API ODP permite ser estudada e modelada de acordo com as necessidades, sendo assim, pode-se implementar novos modelos de criptografia, colaborando com o projeto que é Open Source. O ODP pode ser utilizado para implementar funções do plano de dados de uma rede definida por software.

O plano de dados é o componente de software de um roteador ou *switch* relacionado ao roteamento, encaminhamento de dados, ou tráfego de dados de usuário entre duas ou mais interfaces de rede. Os pacotes que passam entre roteadores e *switchs* para trafegar dados, usam o plano de dados.

O plano de dados precisa fornecer caminhos de alta velocidade, e, muita implementação está em hardware, isso não significa que redes virtualizadas não possuam um plano de dados, existem várias técnicas de software para fornecer esse caminho. Exemplo de Plano de Dados: Envio de Tráfego HTTP para outro dispositivo na rede.

A motivação dessa pesquisa é o estudo e análise de desempenho dos mecanismos e funções de criptografia implementados no ODP, com o objetivo de verificar a possibilidade de implementação desses em ambiente de capacidade computacional reduzido como, por exemplo SoCs.

O trabalho atual torna-se importante para aplicações do plano de dados de redes definidas por software que buscam melhorar o desempenho e garantir integridade, confidencialidade e autenticidade dos dados trafegados no plano de dados.

MATERIAIS E MÉTODOS

Coletou-se informações sobre desempenho de criptografias implementadas na API ODP, tanto em um computador com arquitetura x86, quanto em um Raspberry PI 4 com arquitetura ARM.

As informações coletadas foram filtradas, e seus dados foram usados para criação de gráficos para melhor análise.

As características de hardware e software dos ambientes preparados para medir desempenho das funções criptográficas da API podem ser observadas no Quadro 1:

Quadro 1. Arquitetura e organização de máquinas utilizadas nos testes.

COMPUTADOR X86	RASPBERRY
Processador em arquitetura x86: • Intel Core 2 Duo; • 2 Núcleos com frequência de 2.2Ghz; **Memória:** • 4GB RAM; **Sistema Operacional:** • Ubuntu 18.04; • Windows.	**Processador em arquitetura ARM:** • Broadcom BCM2711; • 4 Núcleos com frequência de 1.5Ghz; **Memória:** • 4GB RAM; **Sistema Operacional:** • Raspbian;

Fonte: organizado pelos autores.

Softwares Utilizados:

- ✓ OpenDataPlane e suas dependências;
- ✓ OriginLab (para análise de dados e gráficos);
- ✓ SSH (para envio de comandos para a Raspberry em remoto, o mesmo pode ser feito diretamente pelo Raspberry, eliminando a necessidade do SSH);
- ✓ Git (para baixar o ODP no Github de forma fácil e rápido).

Para a instalação da API:

1. Deve-se baixar a API localizado no github do ODP;
2. Entrar em modo root em seu terminal;
3. Instalar as dependências que o programa necessita para ser executado;
4. Entrar na pasta **odp/,** rode o comando **./bootstrap;**
5. Logo após, rode o comando **./configure**, caso esteja em um Raspberry PI 4, rode o comando **./configure CFLAGS="-mcpu=cortex-a72 -mfloat-abi=hard -mfpu=neon-fp-armv8 -mneon-for-64bits"**;
6. Configurado sem erros, utilize os comandos **make && make install;**
7. A API ODP já está instalada em sua máquina.

Para melhores práticas, é recomendado utilizar o passo a passo disponível no site OpenDataPlane. Após a instalação correta da API, para obter os dados que serão apresentados, o aplicativo necessário pode ser encontrado em /odp/test/performance/odp_crypto.

Rode o programa odp_crypto, se preferir salvar os dados em um documento de texto, insira em seu terminal o seguinte programa: odp_crypto >> (titulo de sua preferência.txt).

Com as informações salvas, será filtrado os dados das colunas payload (bytes) e elapsed (us) para serem analisadas em um software de análise de dados, como por exemplo, OriginLab, ou então pode-se utilizar um software alternativo de sua preferência.

FUNDAMENTAÇÃO TEÓRICA

Serviços de Criptografia

O ODP oferece APIs para executar operações de criptografia exigidos pelos aplicativos, essas APIs são baseadas em sessões que oferecem serviços de descarregamento de algoritmos de criptografia, é oferecido também serviços de transferência de protocolos para IPSec usando um conjunto diferente de APIs (OPENDATAPLANE, 2020).

Network Function Virtualization (NFV)

Com o aumento de redes de comunicação, aumentou-se também os custos de hardware e manutenção desses dispositivos, tornando-se algumas vezes caro demais para ser mantido, o Network Function Virtualization (NFV) pode ser considerada uma das soluções para esses problemas, sugerindo criação de softwares capazes de substituir equipamentos caros e de alto custo de funções de rede via virtualização, estes hardwares virtualizados podem ser encontrados em nuvem e em máquinas virtuais, deixando o custo

muito mais barato e de fácil manuseio, permitindo também escalabilidade mais simples em qualquer cenário (UFRJ, 2020).

A implantação do NFV é possível graças aos equipamentos de alta qualidade que vieram com a computação em nuvem, dentre eles temos *switches* Ethernet que possibilitam o tráfego de informação entre máquinas virtuais e interfaces físicas. Além disso, a computação em nuvem contribui com métodos de otimização de recursos computacionais, permitindo associar aplicações virtuais diretamente ao núcleo da CPU, memórias e interfaces corretas, além da reinicialização de máquinas virtuais defeituosas (UFRJ, 2020 / CIENA, 2020).

O ODP pode ser utilizado em conjunto com o NFV, pois uma rede virtual pode ser criada com o NFV para rodar uma API do OpenDataPlane com o objetivo de reduzir custos de hardware e manutenção e garantir que toda a infraestrutura funcione com eficiência e segurança.

Software Defined Network (SDN)

Redes Definidas por Software (SDNs) se baseiam na separação dos planos de dados e de controle da rede, sendo que este se refere ao conjunto de funções, logicamente centralizado em controladores de rede, que influencia em como os pacotes são encaminhados a destinos na rede por elementos que aquele define para realizar tal tarefa por meio de uma interface de comunicação bem definida. Dessa forma, a inteligência da rede se concentra em sua maior parte no plano de controle, o qual pode potencialmente abrigar qualquer aplicação de rede que possibilite a implementação de melhores estratégias para encaminhamento de tráfego por inúmeros atuadores em diferentes granularidades. Consequentemente, SDN pode estabelecer algoritmos eficientes no plano de dados para atuar no balanceamento de carga em enlaces, tendo como critérios políticas que contenham quaisquer critérios que sejam úteis a esta tarefa. (Rosa e colaboradores, 2014).

Sistemas Criptográficos

Criptografia Simétrica

Modificar mensagens até chegar ao estado indecifrável é o problema mais conhecido e abordado pela criptografia, a criptografia simétrica é uma das soluções propostas para esses desafios, decifrar uma mensagem só deve ser possível quem tem a posse de um segredo, chamado de chave. Toda criptografia simétrica oferece duas funções, cada uma com dois argumentos: uma para cifrar, e outra para decifrar. A mensagem, chamada de texto claro, passa por uma das funções de cifra que a torna em um texto cifrado (irreconhecível), que tem a necessidade de ser difícil de inverter sem a

chave para decifrar, caso contrário, qualquer pessoa poderia ter acesso à mensagem. (Pellegrini; Jerônimo, 2009, p. 7).

Algumas Criptografias Simétricas são: DES, 3DES, RC2, RC4 e AES.

Criptografia Assimétrica

O problema da criptografia simétrica está em que cada par que queira comunicar-se em sigilo, precisa compartilhar uma chave secreta, que em alguns casos é compartilhado em um canal de comunicação inseguro. A criptografia assimétrica permite que usuários se comuniquem em segredo sem este problema, neste modelo, dois usuários tem duas chaves, uma pública (que é divulgado na rede), e outra privada que fica guardado para si, quando um usuário precisa cifrar uma mensagem, ele usa a chave pública do usuário que irá receber, e apenas o usuário que recebe a mensagem decifra usando sua própria chave privada. (Pellegrini; Jerônimo, 2009, p. 8-9).

Algumas Criptografias Assimétricas são: RSA, DAS.

HASH

Funções de hash são usadas para identificar e autenticar arquivos, pois mapeiam sequências de bits em pequenas sequências de tamanho fixo. Assim, estas pequenas partes podem ser usadas para identificar arquivos e checar sua autenticação. Mas devemos levar em consideração que o número de bits nestes fragmentos é menor que o número de bits que representa os textos, estas funções são projetadas para evitar colisões entre dois hashes ou mais. (PELLEGRINI; JERÔNIMO, 2009, p. 9).

Algumas funções de HASH são: MD4, MD5, SHA1, SHA2.

HMAC

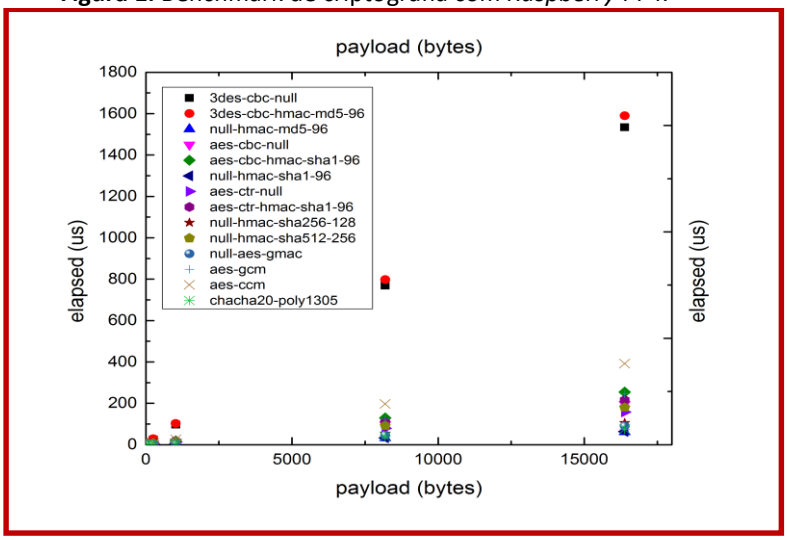
No mundo da computação e comunicação aberta, fornecer formas de verificar integridade de informações transmitidas através de meios não confiáveis é uma necessidade primordial. Mecanismos que fornecem essa verificação de integridade são chamados de códigos de autenticação de mensagens (MAC), estes usados entre duas partes que compartilham um segredo chave para validar informações transmitidas entre essas partes. O HMAC pode ser usado em combinação com qualquer hash criptográfico iterado função. Hashes MD5 e SHA1 são exemplos dessas funções hash. O HMAC também usa uma chave secreta para cálculo e verificação de valores de autenticação de mensagens (como visto nas funções HASH) (CANETTI, 1997).

RESULTADOS

Dados os métodos de obter as informações que serão apresentadas, usaremos todos os tipos de criptografia presentes no odp_crypto, vamos separar duas informações específicas (*payload (carga)* e *elapsed (tempo decorrido)*), e logo após, gerar um gráfico para a análise. Na Figura 1 observa-se os valores de tempo decorrido (*elapsed*) após uma

operação criptográfica em microssegundos (µs) para todas as possíveis operações criptográficas e de *hash* implementadas na API ODP, no ambiente *Raspberry* PI 4. O que pode ser observado é que a operação criptográfica mais demorada é o 3DES e a operação mais rápida é a autenticação hmac-md5-96. Percebe-se também que quanto mais aumenta o número de Bytes (*payload*) maior é o tempo decorrido.

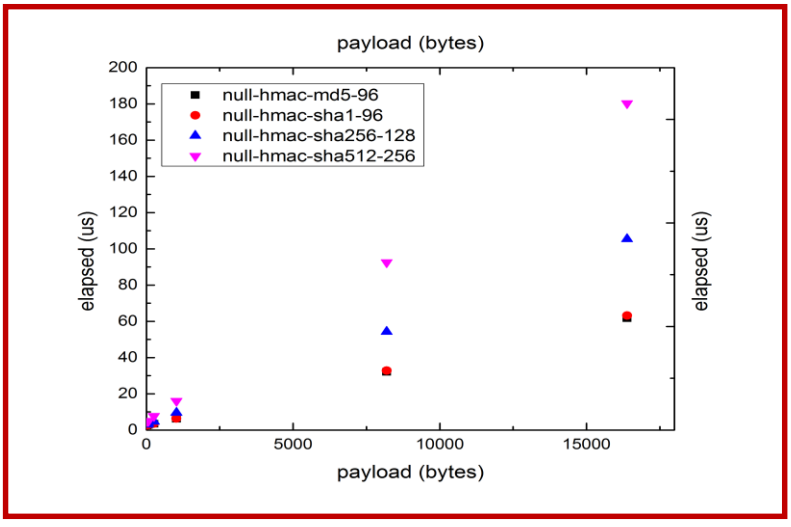
Figura 1. Benchmark de criptografia com *Raspberry* PI 4.



Fonte: adaptado pelos autores.

A seguir serão discutidos os resultados para as operações de autenticação, criptografia e criptografia + autenticação.

Figura 2. Comparativo das operações de autenticação em ambiente RPI4.

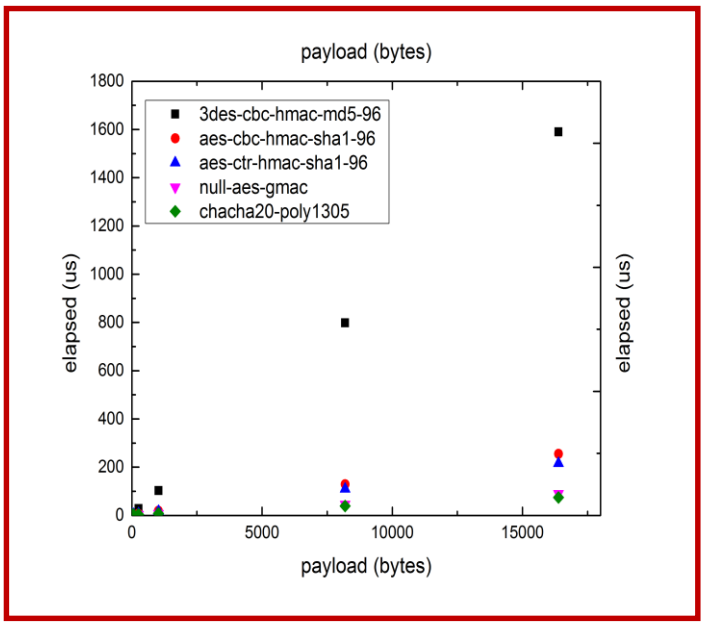


Fonte: adaptado pelos autores.

PROSPECTUS
ISSN 2674-8576 Gestão e Tecnologia
Artigo Original

Observa-se na Figura 2 os resultados de tempo decorrido (*elapsed*) para todas as operações de autenticação implementadas na API ODP. Nota-se que a operação mais lenta é hmac-sha512-256 e a mais rápida é a hmac-md5-96, nota-se um aumento significativo do *elapsed* a partir de aproximadamente 7.500 Bytes.

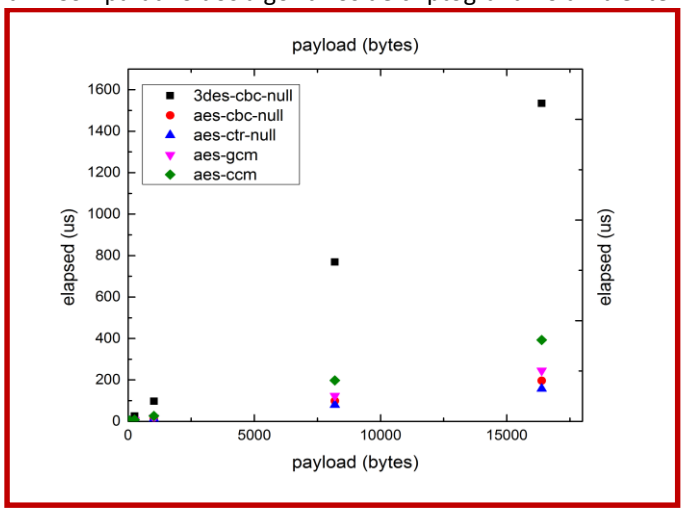
Figura 3. Comparativo das operações de criptografia + autenticação em ambiente RPI4.



Fonte: adaptado pelos autores.

Observa-se na Figura 3 que o algoritmo de criptografia com autenticação mais lento é o 3des-cbc-hmac-md5-96, esse modo de operação bem mais lento quando se tem um volume considerável de Bytes. O algoritmo criptográfico cm autenticação mais rápido é o chacha20-poly1305. O que se nota também é que o algoritmo AES fica bem próximo entre os mais rápidos.

Figura 4. Comparativo dos algoritmos de criptografia no ambiente RPI4.

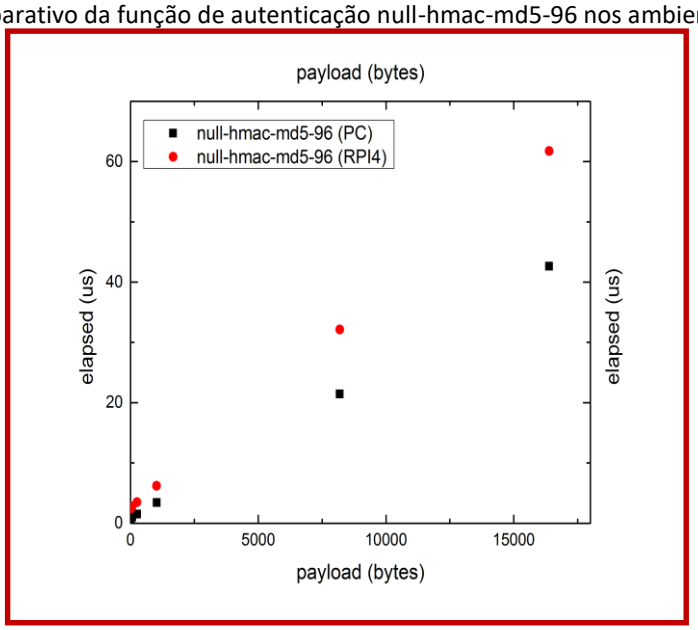


Fonte: adaptado pelos autores.

Na Figura 4 nota-se que o algoritmo criptográfico mais lento é o 3des-cbc-null e é o mais inseguro, porque já está obsoleto e sendo substituído pelo AES. O algoritmo que mostrou melhor desempenho foi o aes-ctr-null que não demonstra grande aumento no *elapsed* para grandes volumes de dados. O AES em todos os modos de operação se apresenta com alto desempenho.

Analisa-se a seguir o desempenho dos mecanismos e funções de autenticação em arquiteturas diferentes arquiteturas e organização de computadores, Computador x86 e o SoC *Raspberry* ARM.

Figura 5. Comparativo da função de autenticação null-hmac-md5-96 nos ambientes PC x RPI4.

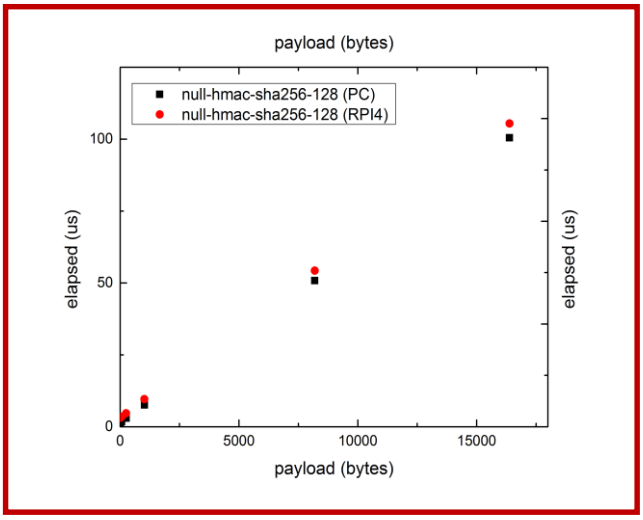


Fonte: adaptado pelos autores.

PROSPECTUS
ISSN 2674-8576 Gestão e Tecnologia
Artigo Original

Na Figura 5 observa-se o resultado para o desempenho da operação hmac-md5-96 para os ambientes PC e RPI4, nota-se que no PC obteve-se um melhor desempenho, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes, sobretudo em volumes pequenos de Bytes.

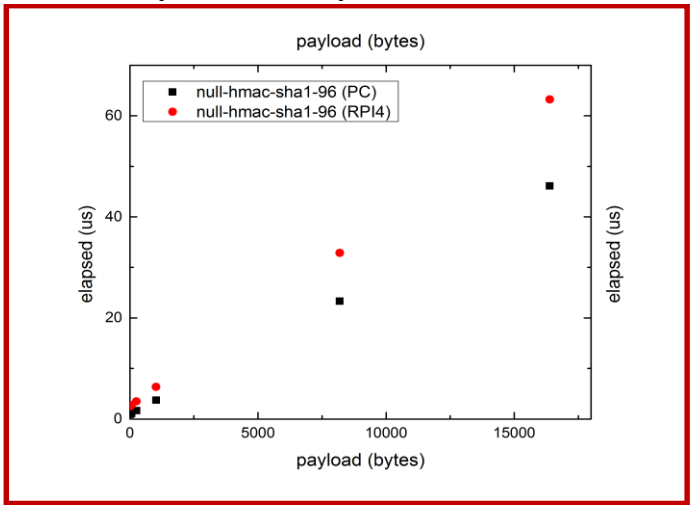
Figura 6. Comparativo da função de autenticação null-hmac-sha256-128 nos ambientes PC x RPI4



Fonte: adaptado pelos autores.

Na Figura 6 observa-se o resultado para o desempenho da operação hmac-sha256-128 para os ambientes PC e RPI4, nota-se que o desempenho para ambos os ambientes resultou num resultado de desempenho muito próximo, tanto para grandes volumes de Bytes, quanto para volumes pequenos de Bytes.

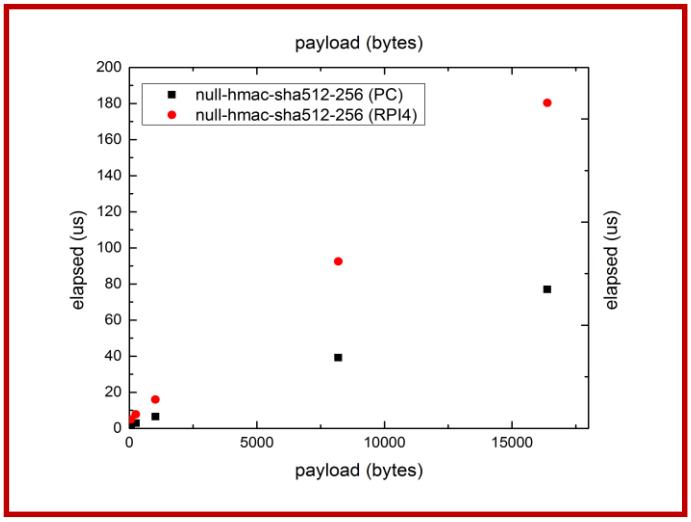
Figura 7. Comparativo da função de autenticação null-hmac-sha1-96 nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 7 observa-se o resultado para o desempenho da operação hmac-sha1-96 para os ambientes PC e RPI4, nota-se que no PC obteve-se um melhor desempenho, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes, sobretudo em volumes pequenos de Bytes.

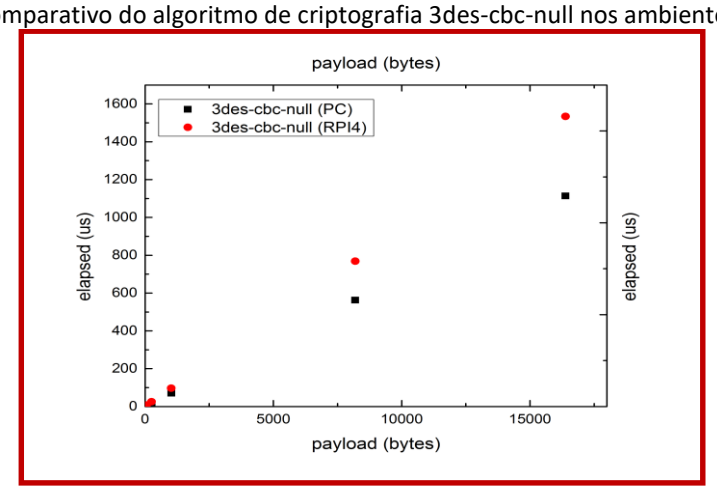
Figura 8. Comparativo da função de autenticação null-hmac-sha512-256 nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 8 observa-se o resultado para o desempenho da operação hmac-sha512-256 para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho muito melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes para volumes pequenos de Bytes.

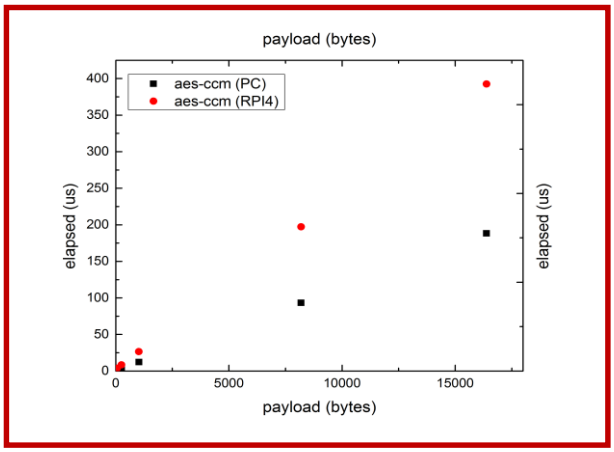
Figura 9. Comparativo do algoritmo de criptografia 3des-cbc-null nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 9 observa-se o resultado para o desempenho para o algoritmo criptográfico 3des-cbc-null para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

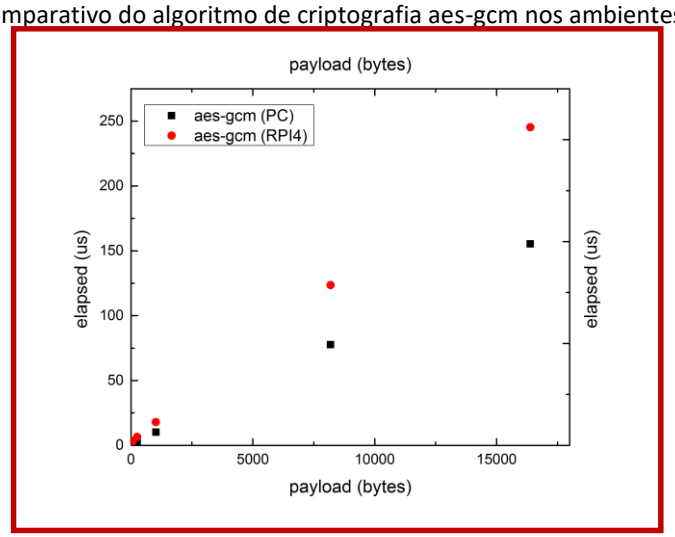
Figura 10. Comparativo do algoritmo de criptografia aes-ccm nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 10 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-ccm para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

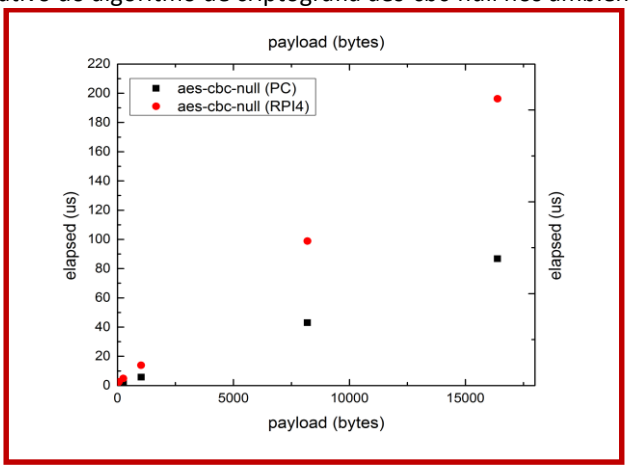
Figura 11. Comparativo do algoritmo de criptografia aes-gcm nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 11 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-gcm para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

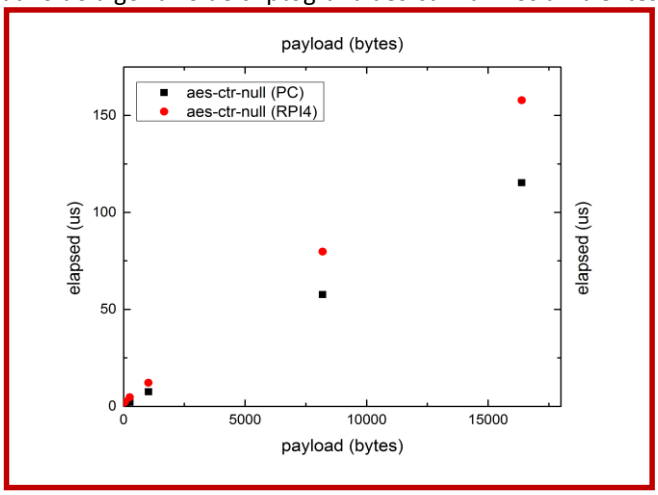
Figura 12. Comparativo do algoritmo de criptografia aes-cbc-null nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 12 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-cbc-null para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, esse modo cbc de operação do AES para grande volume de dados não tem bom desempenho em RPI4, entretanto para pequenos volumes de dados tem-se desempenhos próximos.

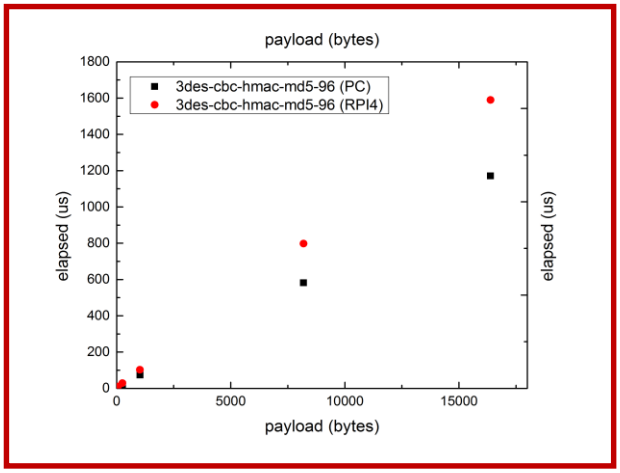
Figura 13. Comparativo do algoritmo de criptografia aes-ctr-null nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 13 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-ctr-null para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

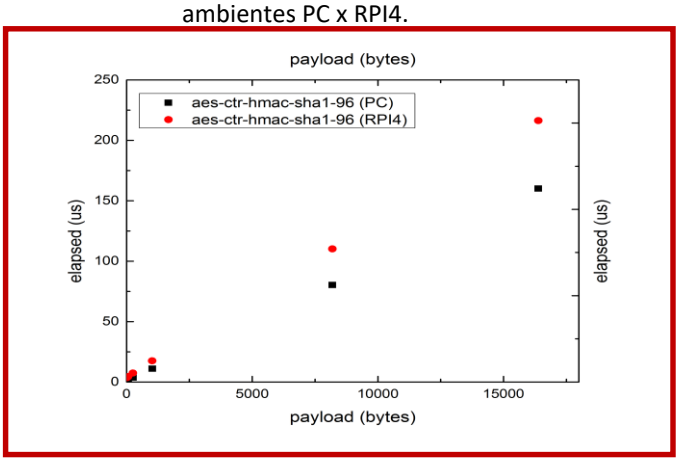
Figura 14. Comparativo do algoritmo de criptografia com autenticação 3des-cbc-hmac-md5-96 nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 14 observa-se o resultado para o desempenho para o algoritmo criptográfico 3des-cbc-hmac-md5-96 para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

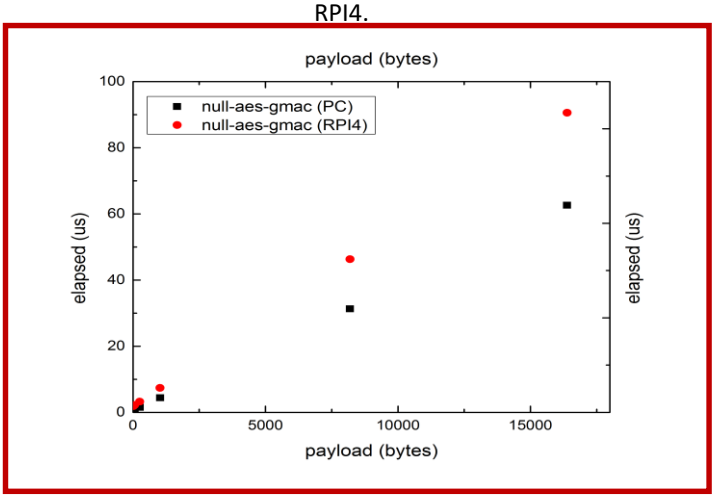
Figura 15. Comparativo do algoritmo de criptografia com autenticação aes-ctr-hmac-sha1-96 nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 15 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-ctr-hmac-sha1-96 para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

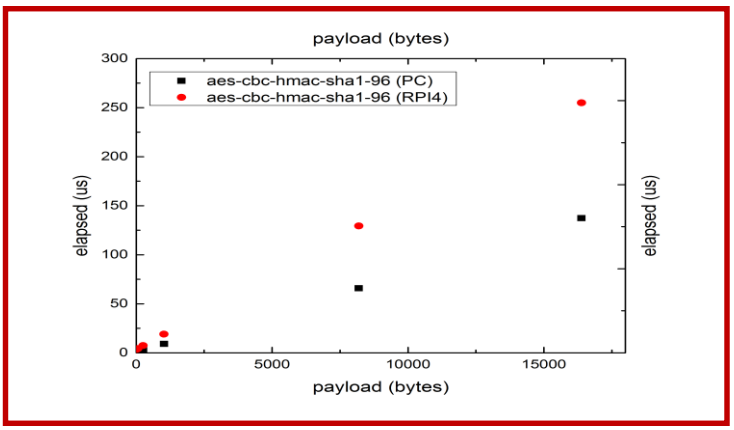
Figura 16. Comparativo do algoritmo de criptografia com autenticação null-aes-gmac nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 16 observa-se o resultado para o desempenho para o algoritmo criptográfico null-aes-gmac para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

Figura 17. Comparativo do algoritmo de criptografia com autenticação aes-cbc-hmac-sha1-96 nos ambientes PC x RPI4.

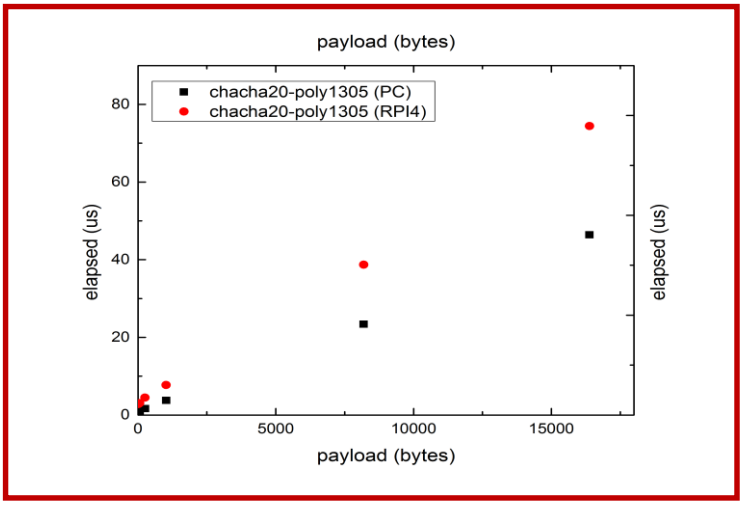


Fonte: adaptado pelos autores.

PROSPECTUS
ISSN 2674-8576 Gestão e Tecnologia
Artigo Original

Na Figura 17 observa-se o resultado para o desempenho para o algoritmo criptográfico aes-cbc-hmac-sha1-96 para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

Figura 18. Comparativo do algoritmo de criptografia com autenticação chacha20-poly1305 nos ambientes PC x RPI4.



Fonte: adaptado pelos autores.

Na Figura 18 observa-se o resultado para o desempenho para o algoritmo criptográfico chacha20-poly1305 (esse foi o algoritmo com autenticação que se obteve o melhor desempenho) para os ambientes PC e RPI4, nota-se que no PC obteve-se um desempenho melhor, sobretudo para grandes volumes de Bytes, entretanto os resultados do RPI4 também são interessantes e ficaram próximos, sobretudo para volumes pequenos de Bytes.

DISCUSSÃO

Dentre as operações de criptografia e autenticação, os mais rápidos para a API ODP são as de autenticação, nota-se que o Raspberry teve um desempenho satisfatório tanto para criptografia quanto para autenticação, oque leva a crer que a ralação custo benefício do Raspberry é muito boa, sobretudo para volume pequenos de Bytes. Nota-se ainda que o algotimo 3DES, quando comparado com outros tem um desempenho muito baixo, o que leva a crer que deva ser deixado de lado e outros como AES devem ser utilizado, a menos que se pretenda manter compatibilidade com sistemas legados. Se a

procura é pro melhor desempenho a criptografia chacha20-poly1305 é a mais indicada, quando se quer autenticação. Caso não seja necessária a autenticação o algoritmo criptográfico aes-ctr-null demonstrou ser o que tem melhor desempenho.

As vantagens dessa API de plano de dados está relacionada a facilidade de implementação para seu uso em redes definidas por software (SDN), como OpenFlow, OpenvSwitch entre outras, que permitem uma escalabilidade, custo, segurança e manutenção melhores do que as presente em redes por hardware.

A análise feita pode ser entendida como um ponto de partida para uma possível implementação de criptografia para esta API, é esperado novos estudos nesse mesmo assunto para melhores interpretações.

CONCLUSÃO

Nesta pesquisa foram abordados os conceitos e uma breve análise de criptografia em uma API de comunicação de rede que atua no plano de dados, os resultados coletados foram analisados e filtrados, gerados gráficos e revisados considerando a função de cada tipo de criptografia, é de se esperar que a segurança desse tipo de comunicação cresça consideravelmente nos proximos anos.

REFERÊNCIAS

CIENA. **O que é NFV?** – Ciena. Disponível em: <https://www.ciena.com.br/insights/what-is/What-is-Network-Functions-Virtualization_pt_BR.html/>. Acesso em 23 de fevereiro de 2020.

GRUPO BINÁRIO. **O que é IP Spoofing e como se proteger dele** – Grupo Binário. 22 de Julho de 2019. Disponível em: <https://www.binarionet.com.br/2019/07/22/o-que-e-ip-spoofing-e-como-se-proteger-dele//>. Acesso em 23 de fevereiro de 2020.

KRAWCZYK, H. BELLARE, M.; CANETTI R. **HMAC:** Keyed-Hashing for Message Authentication. Disponível em: https://tools.ietf.org/html/rfc2104 Acesso em: Abril de 2020.

OPENDATAPLANE. **OpenDataPlane (ODP) Users-Guide.** Disponível em: <https://opendataplane.github.io/odp/users-guide//>. Acesso em 18 de fevereiro de 2020.0

PELLEGRINI, J. **Introdução à Criptografia e seus Fundamentos: notas de aula - versão:** 2019.11.28.20.34. Disponível em: http://aleph0.info/cursos/ic/notas/cripto.pdf Acesso em: Abril de 2020.

ROSA, R.; SIQUEIRA, M.; BAREA, E.; MARCONDES, C.; ROTHENBERG, C. **Network Function Virtualization:** Perspectivas, Realidades e Desafios. Disponível em: http://www.dca.fee.unicamp.br/~chesteve/pubs/MC-SBRC14-NFV.pdf Acesso em: Abril de 2020.

UFRJ. **NFV.** Disponível em: https://www.gta.ufrj.br/ensino/eel878/redes1-2018-1/trabalhos-v1/nfv//. Acesso em 23 de Fevereiro de 2020.

Os autores declararam não haver qualquer potencial conflito de interesses referente a este artigo.