Gerenciamento, configuração e migração seguros de funções de rede virtualizadas utilizando corrente de blocos

[1]Anônimo

Abstract. *The integration of network function virtualization (NFV) and service function chaining (SFC) adds intelligence to the core of the network. However, the programmability of the core of the network brings new vulnerabilities and increases the range of attacks. Thus, it is necessary to ensure the security of the intermediate elements that constitute chains of virtual network functions (VNFs). This article proposes a blockchain architecture for secure management, configuration and migration of VNFs, which ensures: i) the immutability and traceability of the update history; ii) integrity and consistency of information; and iii) the anonymity of VNFs, tenants and settings. The proposed architecture ensures the secure update and migration of configurations at the network core. Simulations demonstrate that the proposal is robust to collusion attacks.*

Resumo. *A integração da virtualização de funções de rede (NFV) e do encadeamento de funções de serviço (SFC) agrega inteligência ao núcleo da rede. No entanto, a programabilidade do núcleo da rede traz novas vulnerabilidades e aumenta a abrangência de ataques. Assim, é necessário garantir a segurança dos elementos intermediários que constituem cadeias de funções de serviço de rede (VNFs). Este artigo propõe uma arquitetura de corrente de blocos para gerenciamento, configuração e migração seguros de VNFs, que assegura: i) a imutabilidade e a rastreabilidade do histórico de atualizações; ii) a integridade e a consistência da informação; e iii) o anonimato de VNFs, inquilinos e configurações. A arquitetura proposta garante a atualização e a migração seguras de configurações no núcleo da rede. Simulações demonstram que a proposta é robusta a ataques de conluio.*

1. Introdução

A virtualização de funções de rede (*Network Function Virtualization* – NFV) e o encadeamento de funções de serviço (*Service Function Chaining* – SFC) surgem como tecnologias alternativas em software para permitir que equipamentos de prateleira sejam capazes de desempenhar as funções antes delegadas a sistemas intermediários proprietários em hardware [Bhamare et al. 2016], ao mesmo tempo reduzindo os custos operacionais (OPEX) e custos de capital (CAPEX). No entanto, a utilização de NFV e SFC agrega inteligência ao núcleo da rede, trazendo novas vulnerabilidades e aumentando a abrangência de ataques [John et al. 2013, Bouras et al. 2017], pois agora o comprometimento de uma função virtualizada de rede (*Virtual Network Function* – VNF) no núcleo da rede põe em risco todo tráfego encaminhado por este elemento [Mijumbi et al. 2016]. Desta forma, são fundamentais a redução dos possíveis vetores de ataque a VNFs e o gerenciamento seguro e confiável de suas configurações [Massonet et al. 2016b].

Correntes de blocos (*blockchains*) dispensam autoridades centrais para o estabelecimento de consenso e de confiança [Christidis and Devetsikiotis 2016, Bozic et al. 2016], e fornecem mecanismos intrínsecos para assegurar autenticidade, integridade, não repúdio e disponibilidade. Além disso, sistemas baseados em correntes de blocos podem ser projetados de forma a também prover confidencialidade e privacidade [Boudguiga et al. 2017]. Essas características motivam o estudo de correntes de blocos e sua aplicação em redes de comunicação e sistemas distribuídos [Bozic et al. 2016].

Este artigo propõe uma arquitetura de correntes de blocos para gerenciamento seguro da configuração de funções virtualizadas de rede (VNF) pertencentes a cadeias de funções de serviço no núcleo da rede. Nesta arquitetura, a corrente de blocos atua como uma base de dados distribuída, capaz de atender com baixa latência requisições de configuração provenientes de VNFs e seus proprietários. Os mecanismos intrínsecos da corrente de blocos conferem segurança à arquitetura proposta, enquanto a adoção de um esquema de identificação por chaves de criptografia assimétricas e a encriptação de dados sigilosos conferem privacidade de identidade e confidencialidade de configuração, quando desejadas. A arquitetura proposta foi projetada de forma a não necessitar de alterações nas plataformas de orquestração de NFV e SFC, bem como não estar restrita a um subconjunto previsto de VNFs. A adoção desta arquitetura dispensa a manutenção de serviços em modo de escuta em uma VNF, reduzindo assim a exposição da VNF a possíveis vetores de ataque. Além disso, a arquitetura proposta busca possibilitar a transferência de configurações e estados de forma segura em uma migração de VNF, bem como definir um mecanismo de confiança entre diferentes provedores de serviço e fabricantes de VNF.

A Seção 2 indica os trabalhos relacionados. A Seção 3 apresenta os principais conceitos de corrente de blocos e consenso distribuído. A Seção 4 apresenta a arquitetura proposta. A Seção 5 avalia o desempenho da arquitetura proposta. Por fim, a Seção 6 descreve as principais conclusões.

2. Trabalhos Relacionados

Diversos trabalhos exploram o estado da arte e a aplicação de correntes de blocos em problemas de redes de comunicação [Mukhopadhyay et al. 2016, Bozic et al. 2016, Christidis and Devetsikiotis 2016]. Pontos predominantes nestes trabalhos são a utilização de chaves criptográficas assimétricas para assinatura e identificação, bem como o emprego da corrente de blocos como um repositório de dados incremental replicado, onde é feito o registro imutável de todas as transações passadas. [Xu et al. 2016] apresenta a corrente de blocos como mecanismo de comunicação e coordenação de serviços através de transações, demonstrando sua aplicabilidade como repositório de informação distribuído. [Boudguiga et al. 2017] apresenta uma solução para atualização de dispositivos IoT através de corrente de blocos. A arquitetura proposta no presente trabalho utiliza mecanismos semelhantes para configuração de VNFs, porém leva em consideração necessidades de confidencialidade, anonimato e auditoria que não são tratadas por Boudguiga *et al*.

A replicação da corrente de blocos recai em um problema de consenso distribuído [Saito and Yamada 2016], pois há necessidade de se garantir a consistência com transações trocadas em um ambiente público, portanto um ambiente inseguro e sem confiança entre estações pares.

[Bhamare et al. 2016, Bouras et al. 2017, Mijumbi et al. 2016] investigam o pro-

blema de vulnerabilidades de segurança devido a co-localização de funções virtualizadas de rede (VNF) de múltiplos inquilinos (*tenants*) usando as tecnologias de virtualização de funções de rede (NFV) e encadeamento de funções de serviço (SFC). Estabelecem que o comprometimento de uma única VNF no núcleo da rede compromete toda uma cadeia de funções de serviço e seus usuários. [Firoozjaei et al. 2017, Lal et al. 2017] investigam ameaças de segurança e propõem taxonomias para vulnerabilidades em NFV. Estes autores discutem também o desafio de transferir configurações e estados de forma segura na migração de uma VNF, uma vez que estes dados são expostos em claro para plataforma de nuvem durante a migração. [Reynaud et al. 2016] ressalta que VNFs são compostas de softwares de diferentes fornecedores, com diferentes vulnerabilidades, e faz menção a diferentes ataques que podem ser realizados contra VNFs através do acesso privilegiado ao seu terminal nas plataformas de nuvem. A arquitetura proposta é capaz de realizar a migração de configuração e estados de uma VNF através da corrente de blocos, preservando a confidencialidade das informações. Ao mesmo tempo, o mecanismo de atualização de configuração proposto dispensa a manutenção de serviços em escuta na VNF, mitigando ataques provenientes de terminais e da rede local ou remota.

Diversas soluções são propostas na literatura para o problema de configuração segura de VNFs. [Coughlin et al. 2017] propõe o uso de hardware seguro através do módulo de plataforma confiável (*Trusted Platform Module* – TPM) para proteger privacidade de configuração de VNFs. Apesar de atingir sua finalidade, esta abordagem depende de hardware específico e atestação remota centralizada, sofrendo impacto de até 50% na banda passante. [Massonet et al. 2016b] propõe uma arquitetura para configuração global de VNFs de segurança em redes federadas com implantação automatizada, porém esta arquitetura é dependente de controlador centralizado e acordo entre nuvens, resultando em ponto único de falha e atrasos de configuração em situações de grande concorrência. Uma variação dessa arquitetura é proposta em [Massonet et al. 2016a], onde a configuração de uma VNF de segurança é realizada pelo orquestrador de VNFs da plataforma de nuvem, sendo codificada no arquivo de descrição da função de serviço. Esta abordagem necessita da modificação de diversos componentes do orquestrador e é restrita a tipos de VNF pre-estabelecidos. Além disso, não é possível modificar a configuração de segurança da VNF após sua inicialização pelo mesmo mecanismo. [Pattaranantakul et al. 2017] propõe um mecanismo similar para controle de acesso a VNFs através dos arquivos de descrição das funções de serviço. No entanto, [Reynaud et al. 2016] verifica que abordagens desse tipo tornam as configurações das VNFs suscetíveis à vulnerabilidades de acesso à informação da plataforma de nuvem utilizada.

Este artigo propõe uma arquitetura que elimina a necessidade de uma entidade central, pois está disponível de forma distribuída, evitando os problemas de ponto único de falha e de concorrência no acesso à configuração. Além disso, a arquitetura proposta requer apenas conexão a um repositório da corrente de blocos e, portanto, não depende de hardware específico ou modificação da plataforma de nuvem, facilitando sua adoção e manutenção. As configurações de VNFs, se privadas, são encriptadas de forma que não são registradas pela plataforma de nuvem, evitando seus vetores de vulnerabilidade.

3. A Corrente de Blocos

Uma corrente de blocos é uma estrutura de dados replicada por membros de uma rede. Inicialmente proposta por [Nakamoto 2008], essa estrutura garante o funcionamento

correto da rede sem a necessidade de confiança em uma entidade central. Apesar do surgimento da cadeia de blocos estar associado a criptomoeda *Bitcoin*, a corrente de blocos não é dependente de nenhuma forma de ativo.

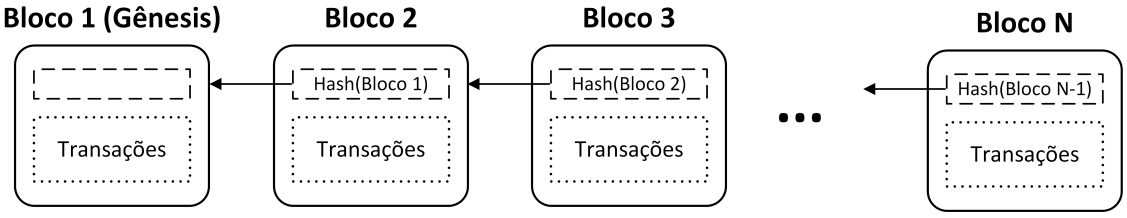


Figura 1. Topologia padrão de uma corrente de blocos na qual cada bloco é associado ao bloco seguinte e a integridade das transações de um bloco é garantida por um *hash*.

A corrente de blocos funciona como um livro-razão (*ledger*), ou registro permanente (*log*), cujas entradas são blocos de transações ordenados no tempo. Uma representação de uma topologia típica de corrente de blocos é mostrada na Figura 1. Cada bloco na corrente é identificado por um valor resultante de uma função resumo (*hash*) e possui tanto as transações realizadas em um determinado intervalo de tempo, como o identificador de seu antecessor. A exceção a esta regra é o bloco inicial, que por definição não possui bloco anterior. Nesta arquitetura, cada nó da rede possui uma cópia local da corrente onde pode verificar todas as transações desde o início. Dado que a cópia da corrente é a mesma em qualquer nó da rede, a propriedade de não repúdio entre membros é garantida. Todas as transações são assinadas através de pares de chaves assimétricas, onde cada membro usa sua chave pública como identificação.

A evolução da corrente de blocos de uma rede acontece de maneira a garantir consenso entre todos os pares. Uma nova transação é assinada por seu responsável e enviada a todos os seus vizinhos, que verificam sua validade. Os critérios de validação variam de acordo com a implementação, mas incluem, no mínimo, a autenticidade baseada em chave pública. Caso a transação seja aceita, é repassada aos próximos nós e eventualmente atinge toda a rede. Caso contrário, a transação é descartada. Ao fim de um intervalo de tempo acordado previamente, o conjunto de transações validadas pela rede é ordenado por um nó e abstraído em um novo bloco candidato. Este processo é comumente chamado de mineração de blocos. A escolha do bloco minerador é dependente do algoritmo de consenso utilizado. Por fim, o novo bloco é enviado pelo minerador em difusão (*broadcast*) para que cada participante verifique as transações e o *hash* proposto. Uma vez aceito, o bloco é inserido na corrente e o estado global é alterado.

Uma necessidade fundamental para a aplicação de qualquer estrutura de dados distribuída é o consenso entre os participantes sobre o conteúdo armazenado. No caso da corrente de blocos, isto significa dizer que todos os nós precisam, a cada novo bloco, concordar com todas as transações e com a ordem temporal em que são listadas. Do contrário, as cópias locais da corrente, e consequentemente o estado global da rede, podem divergir. Um mecanismo distribuído e eficiente de consenso se faz então necessário em qualquer sistema que utilize corrente de blocos.

4. A Arquitetura de Corrente de Blocos Proposta

Uma função virtualizada de rede (VNF) pode ser definida como uma máquina virtual ou contêiner responsável por executar um serviço de rede sobre um fluxo de pacotes que a atravessa. Vale ressaltar que nenhum pacote desse fluxo é endereçado à VNF [Bhamare et al. 2016]. Além disso, uma VNF pode ser separada em dois componentes principais: o serviço de rede executado e os componentes da plataforma de nuvem que determinam sua infraestrutura computacional. A arquitetura proposta neste artigo tem como escopo o gerenciamento de configuração do serviço de rede, deixando o gerenciamento da estrutura computacional subjacente a cargo da plataforma de nuvem utilizada pelo inquilino.

A arquitetura proposta tem como ideia básica um mecanismo proativo de configuração e atualização de função virtualizada de rede (VNF), ou seja, a solicitação de configuração e atualização é iniciada pela própria VNF. Todos os serviços de escuta na VNF são desabilitados. Até mesmo o terminal emulado pelas plataformas de nuvem para a VNF deve ser executado em modo passivo, apenas exibindo informações. Esta estratégia tem como objetivo eliminar possíveis vetores de ataque, pois todo *software* utilizado em uma VNF é sujeito a falhas de segurança. Assim, a própria VNF é responsável, através da execução de um módulo cliente, por se conectar a um repositório a fim de obter sua configuração e atualizações.

A arquitetura proposta é projetada de forma a prover também confidencialidade, privacidade e possibilidade de auditoria de todas as configurações e atualizações das funções virtualizadas de rede (VNF). Todas estas características são obtidas intrinsecamente com o uso da corrente de blocos formada como repositório de base histórica das configurações e atualizações das VNF, pois a utilização de corrente de blocos fornece prontamente mecanismos que asseguraram autenticidade, integridade, não repúdio e disponibilidade das informações.

A utilização de maioria simples para resolução de consenso não previne ataques de conluio [Schwartz et al. 2014]. Um ataque de conluio acontece quando diversos nós cooperam para obter a maioria dos votos e controlar a rede de maneira maliciosa. Assim, é também necessária a utilização de prova de trabalho (*Proof of Work* – PoW), na qual é requerido a solução de um problema com alto custo computacional e de tempo. Assume-se que o poder computacional dos usuários mal intencionados é menor do que o poder computacional dos usuários bem comportados [Saito and Yamada 2016, Tseng 2017]. Neste trabalho, é utilizado um protocolo de consenso [Anônimo et al.], baseado no protocolo RAFT [Ongaro and Ousterhout 2014], com definição automática de pares através de uma seleção aleatória uniforme entre os vizinhos conhecidos de um nó. Esta escolha de protocolo de consenso busca oferecer, ao mesmo tempo, robustez contra ataques de conluio e baixo tempo de efetivação para transações de configuração. A proposta elimina as fases de votação de líderes e o consenso por áreas. A ideia principal é definir uma ordem global entre todas as transações de um bloco. São considerados quatro passos durante a instalação de um novo bloco na corrente de blocos: i) os pedidos de transação dos módulos clientes são coletados durante um intervalo determinado; ii) as transações são difundidas na rede e ocorre a votação da aceitação do conjunto de transações, bem como a validação de cada transação; iii) as transações aceitas pela maioria dos pares são incluídas no próximo bloco e ocorre o estabelecimento da atualização; iv) ao ter uma transação rejei-

tada, o nó busca um novo conjunto aleatório e uniforme de nós na rede para votar a nova proposta de consenso.

4.1. O Modelo de Atacante

Esta arquitetura considera um modelo de atacante [Dolev and Yao 1983], ou seja, o atacante é capaz de *ler*, *enviar* e *descartar* uma transação endereçada à corrente de blocos, ou qualquer pacote de rede. O atacante pode agir de maneira passiva, se conectando na rede e capturando toda troca de mensagens, ou de maneira ativa, injetando, repetindo, filtrando ou trocando informações. Podem ser atacados inquilinos, VNFs, a corrente de blocos ou a rede.

Ataques à corrente de blocos consistem na tentativa de impedir que uma transação ou bloco legítimo seja incorporado à cadeia de blocos. Para que um ataque à corrente de blocos tenha sucesso, o atacante necessita controlar uma parcela significativa da rede para afetar o algoritmo de consenso utilizado. Esse tipo de ataque é mitigado pelo mecanismo de consenso utilizado, e será discutido com mais detalhes na Seção 5.

Ataques aos inquilinos ou às VNFs consistem na tentativa de obtenção de informação de configuração ou de personificação do alvo. Ataques de personificação não são possíveis pois toda transação enviada à corrente de blocos é assinada pelo seu emissor. Ataques que visam a obtenção de informações de configuração são mitigados utilizando criptografia das informações sigilosas, onde o atacante necessita obter a chave privada do destinatário da configuração. Este trabalho não trata o caso onde um inquilino ou VNF foi comprometido através da invasão de seu terminal ou roubo de suas chaves. No entanto, a arquitetura proposta prevê a ausência de qualquer serviço de escuta ativo em uma VNF, e a utilização de um terminal apenas em modo de leitura. Dessa forma, espera-se mitigar este vetor de ataque a VNFs. Além disso, a arquitetura proposta permite a auditoria das transações caso o atacante tente realizar alguma modificação na corrente de blocos utilizando pares de chaves roubados. Estes pares de chave podem ser facilmente substituídas pelo inquilino, impedindo maiores danos.

Ataques à rede representam a tentativa de isolamento de um inquilino ou VNF da rede, impedindo que este execute transações ou leia conteúdo da corrente de blocos. Essa categoria de ataque contempla ataques clássicos de rede, que podem ser mitigados através do estabelecimento de caminhos redundantes entre a corrente de blocos distribuída e VNFs ou inquilinos. O foco deste trabalho não são esses ataques, e sim os ataques à corrente de blocos e às transações previstas na arquitetura. No entanto, ao eliminar serviços de escuta na VNF, a arquitetura proposta elimina a possibilidade de ataques de negação de serviço baseados na camada de aplicação.

4.2. Os Componentes da Arquitetura Proposta

A arquitetura proposta é composta de três módulos principais, como pode ser verificado na Figura 2: o módulo de corrente de blocos, o módulo cliente de VNF e o módulo cliente de inquilino. O módulo de corrente de blocos é executado pelos provedores de telecomunicações em um número de instâncias suficiente para atender as requisições de módulos clientes de VNF em seus centros de dados. O módulo cliente de VNF é executado em cada VNF que deseja utilizar a arquitetura proposta para o seu gerenciamento de configuração. O módulo cliente de inquilino é executado por proprietários de VNFs, que

são inquilinos dos centros de dados e utilizam serviços de SFC fornecidos pelos provedores, por fabricantes de VNFs e por provedores de telecomunicações, que desejam oferecer conjuntos de configuração predefinidos.

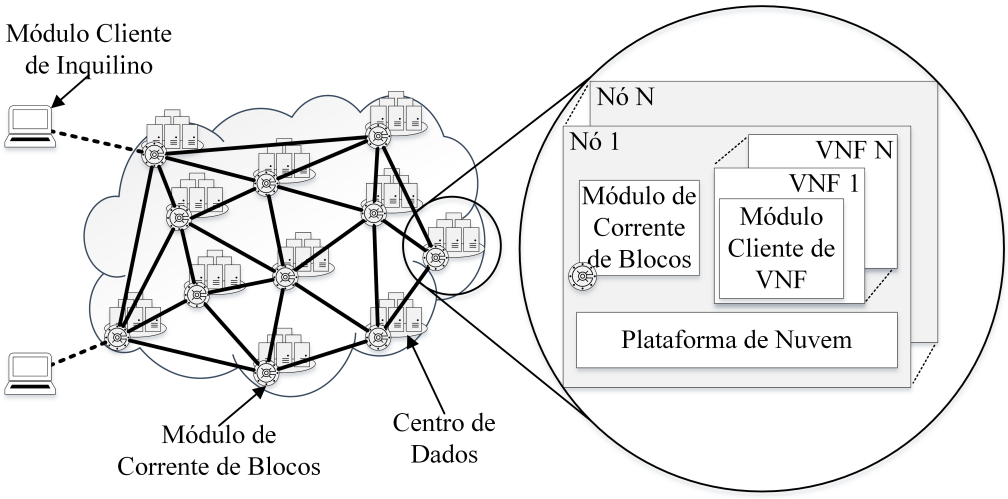


Figura 2. A arquitetura proposta. Os módulos clientes se comunicam indiretamente através dos módulos de corrente de blocos localizados nos centros de dados que oferecem serviços de NFV e SFC.

O módulo de corrente de blocos é o componente da arquitetura responsável por: i) hospedar a corrente de blocos; ii) receber, validar e propagar transações dos módulos clientes; e iii) responder a requisições dos módulos clientes. Este módulo se conecta a outros módulos de mesmo tipo a fim de realizar estas tarefas mantendo o consenso sobre o conteúdo da corrente de blocos, que é replicada em cada nó. Antes da inicialização deste módulo, é fornecida uma lista de endereços de módulos de corrente de blocos vizinhos. Durante a inicialização do nó, este irá solicitar a última versão da corrente de blocos de seus vizinhos e atualizar sua cópia local com os blocos aprovados pendentes. Ao completar essa atualização, é construído um banco de dados relacional local contendo índices para facilitar a busca de conteúdo na corrente de blocos. Esses índices permitem a pronta localização de informação relacionada a remetentes e destinatários de configurações.

Todo módulo de corrente de blocos, para cada transação recebida de outro módulo, tem que verificar três premissas: i) se o formato da transação corresponde ao tipo identificado, ii) se a assinatura da transação está correta e iii) se não há duplicação de transação. Se qualquer uma destas verificações falhar, a transação é descartada. Uma vez garantidas estas três premissas, a transação é enviada para os módulos de corrente de blocos vizinhos. A cada vez que é completada uma rodada do algoritmo de consenso, o novo bloco da corrente é fechado e inserido na corrente de blocos local, e são adicionadas ao banco de dados relacional local as informações referentes às transações contidas nesse bloco. Uma vez que esteja completamente inicializado, cada nó passa a responder requisições de informação de um módulo cliente baseado na sua cópia local da corrente de blocos, levando em consideração somente a informação já validada por consenso.

O módulo cliente de VNF é o componente responsável por: i) enviar transações originárias da VNF para um módulo de corrente de blocos; ii) realizar requisições periódicas de configuração da VNF que hospeda o módulo; iii) aplicar configurações recebidas

na VNF; e iv) gerenciar os pares de chave locais de VNF e de grupos lógicos de VNFs. Esse módulo se conecta a um ou mais módulos de corrente de blocos especificados em sua configuração.

Antes da inicialização do módulo, devem ser configuradas através da plataforma de orquestração de nuvem as seguintes variáveis: i) chaves públicas de inquilino; ii) endereços de módulos de corrente de blocos; e iii) tempo de atualização de configuração. Durante a inicialização deste módulo, é gerado um par de chaves assimétricas para a VNF. A chave pública gerada para a VNF é informada através do terminal da VNF, podendo ser verificada através do console de gerenciamento de VNF da plataforma de orquestração de nuvem utilizado no provedor de telecomunicações. Após a inicialização, o módulo cliente de VNF irá realizar uma requisição da última configuração determinada para VNF local a um módulo de corrente de blocos cadastrado. Essa requisição é realizada a cada intervalo de duração especificada em sua configuração. São possíveis três tipos de resposta: configuração não encontrada, transações de configuração, e transações de solicitação de configuração. Uma transação de configuração principal pode vir acompanhada de outras transações de configuração referenciadas por esta.

Ao receber um conjunto de transações como resposta, o módulo cliente de VNF irá realizar a validação das transações recebidas, que consiste em: i) verificar se a transações recebidas são endereçadas a uma chave pública de VNF ou de grupo lógico de VNF pertencente à VNF local; ii) verificar se os remetentes da transação principal correspondem a uma chave pública de inquilino cadastrada no módulo cliente de VNF local; iii) verificar se todas as assinaturas relacionadas às informações recebidas correspondem aos respectivos remetentes; iv) verificar se o campo *nonce* da transação é diferente da última transação recebida; e, caso seja uma resposta de configuração, v) verificar se a identificação de cada configuração relacionada às transações recebidas corresponde à que está listada na configuração principal. Se foi recebida como resposta uma transação de configuração, e esta foi validada corretamente, a configuração é aplicada; caso contrário, é descartada. Se for recebida como resposta uma transação de solicitação de configuração, e esta foi validada corretamente, o módulo cliente de VNF realiza o envio da configuração atual e dos estados de componentes relacionados para um módulo de corrente de blocos através de uma transação de configuração. A utilização do módulo cliente de VNF necessita de modificações na VNF para que este seja capaz instalar configurações, ler configurações e ler estados pertinentes aos componentes da VNF.

O módulo cliente de inquilino é o componente responsável por: i) enviar transações para o módulo de corrente de blocos; e ii) gerenciar os pares de chave de inquilino. Este módulo pode ser executado em qualquer estação, concomitantemente ou não com um módulo de corrente de blocos. Caso não possua um módulo de corrente de blocos local, o módulo cliente de inquilino deverá ser configurado com um ou mais endereços de módulos de corrente de blocos. Nenhuma transação é endereçada ao módulo cliente de inquilino, mas este pode solicitar informações referentes à corrente de blocos à um módulo de corrente de blocos conhecido.

Foi uma opção da arquitetura o não armazenamento da corrente de blocos em cada VNF, uma vez que isso resultaria em requisitos altos de espaço em disco para as VNFs. Essa decisão não implica em redução significativa de segurança, uma vez que as mensagens transmitidas entre módulos clientes e módulos de corrente de blocos contém

transações assinadas. Um possível ataque resultante dessa escolha é a negação de serviço de configuração de uma VNF por um módulo de corrente de blocos malicioso, através do não envio de respostas às requisições de módulos clientes. A configuração de múltiplos endereços de módulos de corrente de blocos nas VNFs é recomendada como forma de mitigar este ataque.

4.3. O Gerenciamento de Chaves e a Rastreabilidade de Transações

A arquitetura utiliza pares de chaves assimétricos. Existem três grupos lógicos de pares de chaves: i) pares de chave de VNF; ii) pares de chave de grupo lógico de VNFs; e iii) pares de chave de inquilinos de VNF. Os pares de chave de VNF têm como objetivo a identificação única da VNF como remetente ou destinatária de transações, bem como a assinatura de transações provenientes de um módulo cliente de VNF. Esses pares de chave são gerados pela VNF no momento em que é inicializado o módulo cliente de VNF local, e a chave pública do par é exibida no terminal da VNF. Pares de chave de grupo lógico de VNFs têm como objetivo o endereçamento simultâneo de transações a múltiplas VNFs, porém não são utilizadas para assinatura. Uma VNF pode possuir múltiplos pares de chave de grupo lógico de VNFs, que são recebidos como parte de uma transação de configuração. Apesar da funcionalidade desempenhada por pares de chave de grupo lógico de VNFs poder ser implementado utilizando uma chave simétrica de grupo, optou-se pela utilização de chave assimétrica para manter a uniformidade do campo de destinatário de transações como uma chave pública. Essa decisão possibilita a manutenção do anonimato de VNFs e não permite que um observador da corrente de blocos identifique quando uma transação é destinada a um grupo lógico de VNFs ou a uma VNF específica. Os pares de chave de inquilinos de VNF têm como objetivo a identificação do inquilino como remetentes em transações, bem como a assinatura de transações.

Como ocorre em outras implementações de corrente de blocos, não há uma infraestrutura de chaves públicas predefinida [Mukhopadhyay et al. 2016]. As chaves podem ser geradas por qualquer módulo cliente sob demanda, e podem ser prontamente substituídas através de uma transação de configuração. Não há mecanismo de revogação de par de chaves. Uma vez utilizados, permanecerão como identificadores de transação na corrente de blocos. Desta forma, não é possível a um observador da corrente de blocos rastrear as transações entre inquilinos e VNFs, a menos que este conheça previamente os pares de chaves relacionados ao inquilino e suas VNFs. Assim, a corrente de blocos anonimiza a identidade de inquilinos e as funções virtualizadas de rede (VNF) [Kosba et al. 2016], impedindo que estes sejam alvos de ataques direcionados baseados em informação contida na corrente de blocos. Porém, é conservada a transparência na execução de transações [Zhang and Wen 2015] e é possibilitada a auditoria do histórico de transações pelo inquilino ou uma terceira parte autorizada por este. Em alguns casos, pode haver interesse do inquilino na divulgação de sua chave pública. Os casos previstos na arquitetura são quando fabricantes de VNFs ou provedores de serviços de telecomunicação e nuvem desejam divulgar configurações predefinidas de VNFs para serem utilizadas por outros inquilinos como parte da configuração de suas VNFs. Deve ser observado que a utilização de configurações predefinidas implica em perda parcial do anonimato, uma vez que a funcionalidade parcial de uma VNF ou grupo lógico de VNFs fica exposta.

4.4. As Transações Adicionadas aos Blocos da Cadeia

Uma transação representa uma ação atômica a ser armazenada na corrente de blocos. São definidas duas classes de transação: i) a transação de configuração; e ii) a transação de requisição de configuração. Transações de configuração são emitidas por módulos clientes, tanto de inquilino quanto de VNF, e têm como objetivo a instalação de configurações em uma ou mais VNFs ou a definição de uma configuração predefinida de referência. Transações de requisição de configuração são emitidas somente por módulos cliente de inquilinos, e têm como objetivo a solicitação do estado de configuração de uma VNF.

Todas as transações possuem dois conjuntos principais de atributos: i) cabeçalho; e ii) conteúdo. Os campos do cabeçalho de uma transação são os mesmos para qualquer transação, e contêm o resumo (*hash*) da transação, para garantir a integridade da transação, e sua assinatura, realizada com a chave privada do remetente da transação, para garantir a autenticidade. Os campos do conteúdo de uma transação comuns a ambos os tipos de transação são: i) tipo, que define em qual dos dois tipos de transação a transação corrente se enquadra; ii) CID, de *Configuration IDentifier*, que define um identificador único para uma configuração; iii) VID, de *Version IDentifier*, que define a versão da configuração, é único para um mesmo CID; iv) compressão, que identifica se o conteúdo do campo configuração está comprimido, e qual o algoritmo de compressão utilizado; v) encriptação, que define se o conteúdo do campo configuração está encriptado; vi) descrição, um campo opcional que permite associar um texto descritivo à transação; vii) remetente, que identifica o remetente da transação por sua chave pública; viii) destinatário, que identifica o destinatário da transação por sua chave pública; ix) *timestamp*, que identifica a data e hora da transação; e x) *nonce*, que identifica a transação de forma única. Uma transação de configuração tem os seguintes campos adicionais: i) aplicar antes, que identifica quais configurações devem ser aplicadas antes da atual; ii) aplicar depois, que identifica quais configurações devem ser aplicadas depois da atual; e iii) configuração, que contém o conteúdo da configuração contida na transação. Os campos 'aplicar antes' e 'aplicar depois' são representados por uma lista de CIDs, para o caso onde é desejada a última versão de uma configuração, ou CIDs com VIDs, para o caso onde é desejada uma versão de configuração específica.

Uma transação de configuração pode ser emitida com ou sem criptografia a ser utilizada no campo configuração. Quando o campo deve ser criptografado, é utilizada a chave pública do destinatário para realizar a criptografia. O uso encriptado deste campo objetiva a confidencialidade das informações de configuração. Uma transação de configuração que tenha como objetivo definir uma configuração predefinida pública deve ser feita sem encriptação do campo configuração e sem especificação de destinatário.

Em uma transação de requisição de configuração, a semântica dos campos listados acima entre ii) e vi) é diferenciada, indicando os valores a serem utilizados pelo módulo cliente de VNF na transação correspondente. Uma transação de requisição de configuração de um módulo cliente de inquilino deve ser respondida por uma transação de configuração de um módulo cliente de VNF.

A validação de uma transação pelo módulo de corrente de blocos deve observar se todas as regras e semântica de campos acima descritas são obedecidas. Uma transação inválida é descartada imediatamente. A validação da transação é realizada localmente e é essencial para o funcionamento do algoritmo de consenso. Um nó vota pela aceitação de

uma transação se, e somente se, a transação é válida localmente e não entra em conflito com qualquer outra transação já aceita por aquele nó. Tal requisito é importante para a prova de corretude do consenso [Anônimo et al.].

4.5. A Migração Segura de Funções de Rede Virtualizadas

A migração de uma VNF consiste na cópia de sua configuração e estados, quando presentes, para outra máquina na rede. A utilização de mecanismos convencionais de migração, utilizados para migração de máquinas virtuais, implica na exposição do sistema às suas vulnerabilidades de segurança [Lal et al. 2017]. Dados não podem ser encriptados VNF-a-VNF pois precisam ser compreendidos pelo hipervisor [Firoozjaei et al. 2017]. No entanto, a migração de uma VNF não precisa ser tratada como a migração de uma máquina virtual tradicional, devido a seu escopo de uso mais simples, específico para prestação de serviço de rede.

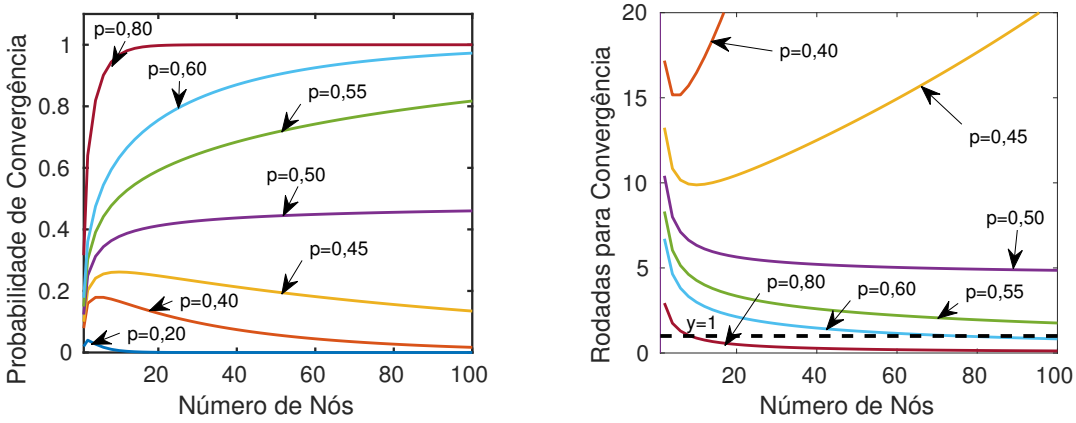
Este trabalho propõe que a migração do serviço desempenhado seja realizada através de transações na corrente de blocos, e que apenas a inicialização e a conexão da VNF fiquem a cargo da plataforma de nuvem utilizada pelo inquilino. Desta forma, após a inicialização de outra VNF contendo um módulo cliente de VNF da plataforma, o procedimento de migração do serviço desempenhado consiste em: i) uma transação de requisição de configuração para VNF de origem; e ii) uma transação de configuração para VNF de destino, incluindo a configuração obtida pela primeira transação no campo 'aplicar antes' da transação atual. Após este procedimento, a VNF de origem pode ser desligada. Por não depender da plataforma de nuvem utilizada, este mecanismo de migração pode ser realizado entre VNFs compatíveis executadas em plataformas de nuvem distintas, mesmo que a arquitetura de virtualização seja diferente.

5. A Avaliação da Corrente de Blocos Proposta

Análise da arquitetura proposta foi realizada através da formalização matemática do funcionamento do algoritmo de consenso e de aceitação dos blocos na corrente. O objetivo principal é verificar a robustez da arquitetura a ataques de conluio. No cenário avaliado, considera-se p_b a probabilidade de um nó ser bem comportado. Um nó bem comportado é aquele que não apresenta falhas bizantinas e não participa de conluio. A probabilidade de convergência do algoritmo de consenso é dada por

$$p^* = \sum_{i=0}^{i \leq \lfloor \frac{N}{2} \rfloor - 1} C_{N,i}\, p_b^i (1 - p_b)^{N-i}, \tag{1}$$

em que p^* é a probabilidade de convergência do consenso, N é o número total de nós na rede e i é o número de nós maliciosos, aqueles realizam conluio. Vale ressaltar que $C_{N,i}$ é combinação de nós maliciosos no conjunto dos N nós na rede. A partir da Equação1, pode-se definir a probabilidade de o consenso falhar k vezes consecutivas, dada por $p_{falha} = (1 - p^*)^k$, e pode-se definir um limiar de confiança arbitrário α que limite essa probabilidade. A convergência é obtida se o valor de k falhas consecutivas é menor do que α, indicando portanto alta garantia de consenso. O número de tentativas para se obter o consenso é $k = \frac{log(\alpha)}{log(1-p^*)}$, em que α é a probabilidade aceitável de não obter o consenso após k tentativas.



(a) Probabilidade de convergência.

(b) Convergência para $\alpha \leq 0,05$.

Figura 3. Avaliação da convergência do esquema de corrente de blocos. a) Probabilidade de um bloco ser aceito pelo protocolo de consenso e para probabilidade p dos nós serem bem comportados. b) Número de rodadas necessário para atingir uma probabilidade de falha de consenso $\alpha \leq 0,05$. Com $p = 0,80$, a partir de 10 nós, o consenso ocorre em apenas uma rodada. Para $p = 0,60$, o consenso em uma rodada ocorre para mais de 60 nós.

A Figura 3 mostra a convergência da arquitetura proposta para valores de p variando entre 0,20 até 0,80. Valor de $p = 0,80$ indica que os nós na rede são 80% bem comportados. O caso limite, mostrado na Figura 3(a) para $p = 0,50$, metade da rede pode se comportar de maneira maliciosa. Vale notar que quanto maior o número de nós na rede, maior a probabilidade de se chegar ao consenso e então adicionar o bloco à corrente. Paralelamente, ao observar o número de tentativas de se realizar o consenso, Figura 3(b), verifica-se que para um dado valor de $p \geq 0,50$, o número de tentativas até se obter uma probabilidade de falha no consenso $\alpha \leq 0,05$ permanece relativamente constante apesar do crescimento no número de nós na rede. A Figura 3(b) mostra ainda que para valores mais altos de p ($p \geq 0,60$) a inclusão do bloco na corrente pode ocorrer em apenas uma tentativa de consenso. Nesse caso, o tempo para inclusão do bloco na cadeia é limitado pela execução do algoritmo de consenso em duas vezes o tempo de ida e volta entre o nó iniciador do processo de inserção do bloco na cadeia e os demais nós na rede [Anônimo et al.]. Vale ainda ressaltar que a arquitetura proposta é resiliente a ataques de conluio de até $N - 1$ nós. Esse resultado é ressaltado na Figura 3 para $p = 0,50$, em que a probabilidade de se obter um consenso em uma roda é de aproximadamente 0,40 para 10 nós, mas ao executar entre 7 e 8 rodadas do algoritmo de consenso, a probabilidade de se atingir o consenso é de 0,95 (dado por $1 - \alpha$). Esse comportamento ocorre devido ao algoritmo de consenso adotado em que, ao ter a primeira falha no consenso, busca-se um novo conjunto de nós na rede. A busca de um novo conjunto aleatório e uniforme sobre todos os nós garante eventualmente a obtenção dos nós bem comportados.

6. Conclusão

Este artigo propõe uma arquitetura de correntes de blocos para gerenciamento seguro da configuração de funções virtualizadas de rede (VNF), de forma independente de *hardware* específico e da arquitetura da plataforma de nuvem. Além de eliminar a

existência de ponto único de falha e prover alta disponibilidade de informação de configuração, a arquitetura proposta garante a confidencialidade e anonimato de transações sem comprometer a capacidade de auditoria por uma entidade autorizada. Além disso, é proposto um esquema de migração que preserva a confidencialidade das informações de VNF baseando-se na utilização da arquitetura.

Os resultados demonstram que a arquitetura proposta é robusta a ataques de conluio de até aproximadamente 50% dos nós da rede, e que, mesmo sob ataques maiores, a convergência pode ser alcançada mediante um grande número de rodadas. No entanto, como é demonstrado no modelo de atacante, um ataque à corrente de blocos não é capaz de comprometer as configurações, apenas atrapalhar o registro de novas configurações na corrente. Em trabalhos futuros, será verificada a aplicabilidade de um mecanismo de reputação dos nós para ajuste de coeficientes de confiança durante a realização do algoritmo de consenso.

Referências

Anônimo et al.

Bhamare, D., Jain, R., Samaka, M., and Erbad, A. (2016). A survey on service function chaining. *J. Netw. Comput. Appl.*, 75(C):138–155.

Boudguiga, A., Bouzerna, N., Granboulan, L., Olivereau, A., Quesnel, F., Roger, A., and Sirdey, R. (2017). Towards better availability and accountability for IoT updates by means of a blockchain. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, pages 50–58.

Bouras, C., Kollia, A., and Papazois, A. (2017). SDN & NFV in 5G: Advancements and challenges. In *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, pages 107–111.

Bozic, N., Pujolle, G., and Secci, S. (2016). A tutorial on blockchain and applications to secure network control-planes. In *3rd Smart Cloud Networks Systems*, pages 1–8.

Christidis, K. and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *IEEE Access*, 4:2292–2303.

Coughlin, M., Keller, E., and Wustrow, E. (2017). Trusted click: Overcoming security issues of NFV in the cloud. In *Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, SDN-NFVSec '17, pages 31–36, New York, NY, USA. ACM.

Dolev, D. and Yao, A. (1983). On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208.

Firoozjaei, M. D., Jeong, J. P., Ko, H., and Kim, H. (2017). Security challenges with network functions virtualization. *Future Generation Computer Systems*, 67:315 – 324.

John, W., Pentikousis, K., Agapiou, G., Jacob, E., Kind, M., Manzalini, A., Risso, F., Staessens, D., Steinert, R., and Meirosu, C. (2013). Research directions in network service chaining. In *2013 IEEE SDN for Future Networks and Services*, pages 1–7.

Kosba, A., Miller, A., Shi, E., Wen, Z., and Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 839–858.

Lal, S., Taleb, T., and Dutta, A. (2017). Nfv: Security threats and best practices. *IEEE Communications Magazine*, PP(99):2–8.

Massonet, P., Dupont, S., Michot, A., Levin, A., and Villari, M. (2016a). An architecture for securing federated cloud networks with service function chaining. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pages 38–43.

Massonet, P., Dupont, S., Michot, A., Levin, A., and Villari, M. (2016b). Enforcement of global security policies in federated cloud networks with virtual network functions. In *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*, pages 81–84.

Mijumbi, R., Serrat, J., l. Gorricho, J., Latre, S., Charalambides, M., and Lopez, D. (2016). Management and orchestration challenges in network functions virtualization. *IEEE Communications Magazine*, 54(1):98–105.

Mukhopadhyay, U., Skjellum, A., Hambolu, O., Oakley, J., Yu, L., and Brooks, R. (2016). A brief survey of cryptocurrency systems. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 745–752.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf.

Ongaro, D. and Ousterhout, J. (2014). In search of an understandable consensus algorithm. In *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pages 305–319, Philadelphia, PA. USENIX Association.

Pattaranantakul, M., Tseng, Y., He, R., Zhang, Z., and Meddahi, A. (2017). A first step towards security extension for nfv orchestrator. In *Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, SDN-NFVSec '17, pages 25–30, New York, NY, USA. ACM.

Reynaud, F., Aguessy, F. X., Bettan, O., Bouet, M., and Conan, V. (2016). Attacks against network functions virtualization and software-defined networking: State-of-the-art. In *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pages 471–476.

Saito, K. and Yamada, H. (2016). What's so different about blockchain? blockchain is a probabilistic state machine. In *2016 IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 168–175.

Schwartz, D., Youngs, N., and Britto, A. (2014). The ripple protocol consensus algorithm. *Ripple Labs Inc White Paper*. https://ripple.com/files/ripple_consensus_whitepaper.pdf.

Tseng, L. (2017). Bitcoin's consistency property. In *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 219–220.

Xu, X., Pautasso, C., Zhu, L., Gramoli, V., Ponomarev, A., Tran, A. B., and Chen, S. (2016). The blockchain as a software connector. In *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, pages 182–191.

Zhang, Y. and Wen, J. (2015). An iot electric business model based on the protocol of bitcoin. In *2015 18th International Conference on Intelligence in Next Generation Networks*, pages 184–191.