Plataforma para Análise de Tráfego e Optimização de Infraestruturas de Comunicação

An Application for Traffic Analysis and Optimization of Network Infrastructures

Marco Pereira
Department of Informatics
University of Minho
Braga, Portugal
marcoavpereira@gmail.com

Pedro Sousa
Centro Algoritmi, Department of Informatics
University of Minho
Braga, Portugal
pns@di.uminho.pt

Resumo — **Uma infraestrutura de comunicação nem sempre é fácil de gerir e o nível de dificuldade aumenta com a dimensão e o número de dispositivos presentes na infraestrutura de rede. A monitorização de uma rede é um processo fundamental na medida em que previne e detecta eventuais problemas e dispõe de diversas ferramentas auxiliares do trabalho de um administrador de redes de computadores. Neste contexto, este trabalho apresenta uma plataforma inovadora de monitorização e optimização de uma infraestrutura de rede, sendo o seu interface de fácil análise mesmo para quem não possui elevadas competências na área das redes de computadores. De igual modo, e ao contrário dos sistemas de monitorização tradicionais, a plataforma desenvolvida possui ainda um módulo de optimização que possibilita a recomendação de possíveis alterações/configurações por forma a optimizar os recursos da rede.**

Palavras Chave – Monitorização de Redes; Traffic Flows; Redes de Computadores; Qualidade de Serviço (QoS); Segurança.

Abstract — **A communication infrastructure is not always easy to manage and the level of difficulty increases with the size and number of devices present in the network. Thus, the monitoring of a network is a fundamental process in that it may prevent or detect possible problems, also offering several auxiliary tools for the work of a computer network administrator. In this context, this work presents a novel platform for the monitoring and optimization of network infrastructures, which interface is easy and intuitive to analyze even for those who do not have high skills in the area of computer networks. Furthermore, and unlike most of traditional monitoring systems, the developed platform has an additional optimization module allowing the recommendation of possible changes/configurations in order to optimize the network resources.**

Keywords – Network Monitoring; Traffic Flows; Computer Networks; Quality of Service (QoS); Security.

I. INTRODUÇÃO

As redes de computadores tornaram-se numa ferramenta imprescindível no quotidiano de grande parte da população mundial. A Internet é um bom exemplo de que estas redes estão cada vez mais presentes no nosso dia-a-dia, sendo demonstrado pelo seu crescimento exponencial [1]. Para além da utilização da Internet e respetivos serviços (e.g. correio eletrónico, partilha de ficheiros, pagamentos online, entre outras), as empresas dependem das suas redes internas, conhecidas como Intranets, que possibilitam a utilização de aplicações/serviços por parte dos seus funcionários. Esta dependência representa um aumento de rentabilidade e uma diminuição de custos, contudo, grande parte das empresas não estão aptas a executarem as suas atividades se as respetivas Intranets não estiverem operacionais.

Apesar da importância e dos benefícios da utilização das redes de computadores, estas não fazem sentido quando não existem aplicações que as utilizem, como por exemplo, o *File Transfer Protocol* (FTP), o correio eletrónico, os clientes *web*, bem como diversos serviços de voz. Estas, apresentam-se como ferramentas imprescindíveis no nosso dia-a-dia, tanto para uso pessoal como profissional. O desempenho das aplicações é comprometido pela estabilidade e desempenho da rede. De forma a perceber a existência de anomalias, e a probabilidade destas ocorrerem, é fundamental a monitorização da rede. Neste sentido é possível detetar erros de configuração e dispositivos na rede que estão com mau funcionamento (e.g. *routers* e *switchs*). Em alguns casos as redes encontram-se corretamente configuradas mas, mesmo assim, apresentam um baixo desempenho. Estas situações podem ser verificadas quando o tráfego gerado pelas aplicações excede a capacidade da rede, que poderão ser retificadas através de um aumento de capacidade. Contudo, certas aplicações podem não funcionar corretamente e é importante perceber quais são. É ainda possível que o tráfego excessivo seja propositadamente gerado, por utilizadores com intenções duvidosas. Através da monitorização é possível identificar o tráfego por origem, destino e, em alguns casos, por aplicação, de forma a perceber qual a fonte do tráfego excessivo.

Para monitorizar as redes de computadores existem várias aplicações, pagas ou gratuitas, capazes de identificar o tráfego por origem, destino e aplicação (e.g. o *Nagios* [2] e o *Multi Router Traffic Grapher* (MRTG) [3]). No entanto, ainda não existe uma aplicação totalmente adequada para utilizadores

com escassos conhecimentos técnicos na área de redes de computadores. Ao nível da grande parte das aplicações existentes, as métricas monitorizadas são capturadas através do protocolo *Simple Network Management Protocol* (SNMP), implicando que os dispositivos tenham este protocolo ativo [9], ou através de *plugins* instalados nos dispositivos, fazendo com que a operação de monitorização seja pouco escalável.

Este trabalho surge de um caso real de uma empresa que necessitava de uma ferramenta para monitorizar a sua rede, utilizada por mais de 500 funcionários. Para além de monitorizar a rede, também existia a necessidade de monitorizar o tráfego produzido pelas aplicações desenvolvidas na empresa e por cada departamento. Esta ferramenta deveria ser simples e intuitiva para que qualquer técnico do departamento de informática conseguisse entender o estado da rede. Para complementar os processos de monitorização, constituiu-se como objetivo a criação de um módulo de alertas e otimização da infraestrutura de rede da empresa.

Este documento é composto por 5 seções organizados da seguinte forma: a secção I apresentou o trabalho a desenvolver e os seus objetivos, a secção II apresenta a arquitetura da plataforma e a interação dos módulos, a secção III descreve detalhadamente os módulos principais da plataforma, a secção IV demonstra casos práticos da utilização da plataforma e, por fim, a secção V apresenta as principais conclusões deste trabalho e o trabalho planeado para o futuro da plataforma.

II. ARQUITETURA

Nesta secção é apresentada a arquitetura da plataforma assim como a comunicação entre os diferentes módulos que a constituem. A Fig. 1 ilustra os módulos da plataforma e as respetivas comunicações.

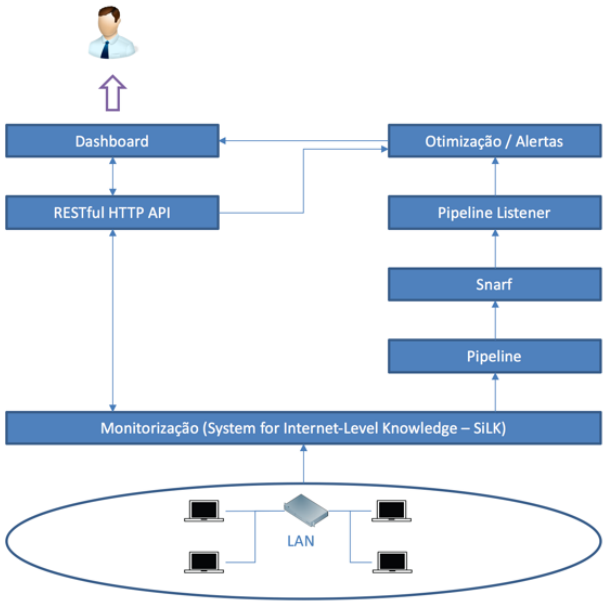

Figure 1. Arquitetura da plataforma desenvolvida

O módulo de Monitorização é a base desta arquitetura, pois o seu objetivo é capturar, extrair e guardar a informação necessária do tráfego da rede. Todas as decisões da plataforma dependem do resultado da monitorização, que é enviado para os módulos da *Application Programming Interface* (API), sempre que requisitado, e do Pipeline, sempre que seja capturado tráfego. O módulo da API, *RESTfull HTTP API*, processa os pedidos do módulo do *Dashboard*, funcionando como uma *interface* que facilita o acesso às informações capturadas na rede. Outra função da API é alterar os parâmetros do módulo de Otimização/Alertas sempre que receber um pedido do *Dashboard* com essa finalidade. O *Dashboard* funciona como *interface* para o utilizador, que pode ser executado através de uma aplicação móvel ou como um *website*, onde é possível visualizar a informação da rede, em tempo real, receber alertas do estado da rede e configurar os parâmetros do módulo de Otimização.

Quando existem novas informações sobre o tráfego da rede, estas são enviadas ao módulo *Pipeline* [11] que é responsável por processá-las e decidir se notifica o módulo de Otimização/Alertas, por intermédio do módulo *Pipeline Listenner*, sobre as novas informações. O *Snarf* [10] é a ferramenta utilizada pelo *Pipeline* para enviar alertas que posteriormente são capturados pelo *Pipeline Listenner*. O *Pipeline Listenner* funciona como um intermediário que tem a simples tarefa de receber informação do *Pipeline* e entregar ao módulo *Otimização/Alertas*. Este último módulo é responsável por alertar o *Dashboard* de novas ocorrências na rede e configurar os dispositivos da *Local area network* (LAN) de acordo com os tipos dessas mesmas ocorrências. Exemplos de configurações poderão ser a sugestão de rotas alternativas para tráfego de certas aplicações e definições de horários para execução de operações que geram elevada carga na rede (e.g. processos de *backups* associados e determinados servidores).

III. MÓDULOS DESENVOLVIDOS

Nesta secção são apresentados os principais módulos da plataforma descrevendo a construção e os objetivos dos mesmos. O primeiro módulo apresentado é o módulo de Monitorização, a base da arquitetura da plataforma, seguindo-se os módulos *Dashboard* e API, responsáveis pela apresentação das informações da rede ao utilizador. Por fim, será apresentado o módulo de Otimização/Alertas, responsável por alertar o utilizador de possíveis recomendações/otimizações e configuração das mesmas nos dispositivos da rede.

A. *Módulo de Monitorização*

Para monitorizar o tráfego da rede optou-se pela técnica denominada de monitorização por *flows*. Um *flow* é definido como um conjunto de pacotes, com um determinado conjunto de atributos comuns, que passam por um ponto de observação na rede durante um certo intervalo de tempo [4]. Alguns exemplos de atributos comuns são o tuplo de cinco atributos (IPs e portas de origem e destino e protocolo de transporte), pacotes de uma rede e pacotes com a mesma porta de origem para um determinado IP de destino [5]. Este tipo de monitorização captura a informação presente nos *headers* dos pacotes que circulam na rede e pode ser aplicada em diferentes áreas, como por exemplo, contabilização [6], otimização [7] e segurança [8].

A ferramenta *System for Internet-Level Knowledge* (SILK), desenvolvida pela *CERT Network Situational Awareness Team* [12], foi utilizada neste módulo devido à sua compatibilidade com os principais protocolos de exportação de *flows* (*IPFIX*, *NetFlow* e *SFlow*), às constantes atualizações e à sua completa documentação. Esta ferramenta divide-se em dois módulos [13], o *Packing tools* (recebe os *flows* de um *flow generator*, converte-os para um formato próprio da ferramenta, designado por *Silk Flow*, e guarda a informação em ficheiros) e o *Analysis tools* (permite fazer a análise do tráfego da rede, sendo responsável pela leitura dos ficheiros, criados pelo *Packing tools*, e por filtrar, agrupar, ordenar e contar os *flows* de acordo com os comandos do utilizador). A plataforma apenas necessita de aceder ao módulo *Analysis tools*.

Em [14] são apresentados e detalhados todos os campos dos *flows* armazenados e os comandos SILK que permitem a análise do tráfego. É possível criar novos campos, possíveis de interagir com os comandos, através de *scripts Python*, dos *Prefix Maps* (Pmaps) e *IPSets*. Os Pmaps permitem criar associações entre um nome e uma porta/protocolo de transporte (e.g. protocolo HTTP - 80/TCP), um IP ou um intervalo de IPs (e.g. gama de IPs de uma rede). Com estes campos podemos criar as associações entre os nomes de aplicações e as respetivas portas/protocolos e perceber o impacto de cada uma na rede. Os *IPSets* permitem criar conjuntos de IPs e filtrar o tráfego tendo em conta estes conjuntos. Exemplos de aplicação dos *IPSets* são as *blacklist*, as *whitelist* e os IPs dos servidores de uma empresa.

Para identificação do tráfego das aplicações desenvolvidas e dos departamentos da empresa, a plataforma proposta foi desenvolvida tendo em conta os seguintes cinco componentes:

- *Servidores*: pretende-se monitorizar o tráfego que ocorre nos servidores da empresa para analisar a carga a que são submetidos e identificar as aplicações mais requisitadas. No SILK, foi criado um *IPSet* com os IPs dos servidores para ser possível, facilmente, obter o tráfego dos servidores. Também foi criado um *Pmap* que faz a correspondência entre o IP e o nome do servidor.
- *Serviços*: os serviços correspondem aos protocolos que atuam na última camada da pilha protocolar *TCP/IP*: a camada de aplicação. Exemplos destes serviços são o *Hypertext Transfer Protocol* (HTTP) e o *Hyper Text Transfer Protocol Secure* (HTTPS). Os serviços presentes na plataforma foram obtidos através da lista de serviços gerida pela *Internet Assigned Numbers Authority (*IANA) [15] que identifica o protocolo de transporte e a porta por omissão utilizada pelos serviços. Para utilizar os serviços no SILK, existe um Pmap que associa o protocolo de transporte e a porta ao nome do serviço.
- *Categorias*: são grupos de serviços que têm o objetivo de simplificar a procura e identificação do tráfego de serviços que atuam na mesma área. Por exemplo, para procurar o tráfego de email é necessário identificar o tráfego relativo aos protocolos *Simple Mail Transfer Protocol* (SMTP), *Internet Message Access Protocol* (IMAP) e *Post Office Protocol* (POP). Ao criar a categoria *Email* é possível associar estes serviços à categoria e identificar todo o tráfego relativo a email. No SILK, foi criado um Pmap que associa o protocolo e a porta dos serviços ao nome da respetiva categoria.
- *Departamentos*: a identificação do tráfego por departamento da empresa é realizada através da segmentação da rede. Para cada um deles que se pretenda identificar, é criado um novo segmento da rede e atribuído ao departamento. Deste modo, os dispositivos de cada departamento têm um IP associado ao respetivo segmento. Para identificação dos departamentos no SILK, foi criado um Pmap que associa o segmento da rede ao nome do respetivo departamento.
- *Aplicações*: correspondem às aplicações desenvolvidas internamente na empresa. Cada aplicação tem o seu próprio servidor e utiliza um serviço da rede. Recorrendo à integração existente de *scripts python* com o SILK, foram criadas *scripts* para filtrar e identificar o tráfego das aplicações.

B. *Módulos Dashboard e API*

Para a construção deste módulo é utilizado o Ionic [16]. O Ionic é uma *framework* Javascript, baseada no *AngularJS* e no *Cordova*, permitindo o *Dashboard* funcionar em *browser*, independentemente do sistema operativo, e como aplicação móvel para *smartphones* e *tablets*. Este módulo é constituído por três páginas que podem ser acedidas através do menu, localizado no canto superior direito da aplicação: *Dashboard*, *Alertas* e *Configurações*.

A página *Dashboard* é a interface que permite ao utilizador consultar o tráfego capturado pelo módulo de monitorização. Esta simples interface facilita a interpretação do estado da rede por parte de um utilizador com escassos conhecimentos em redes de computadores. A Fig. 2 ilustra esta interface.

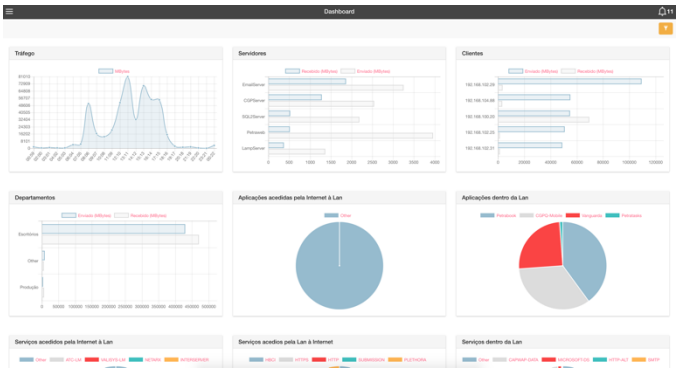

Figure 2. *Dashboard* – Página com os resultados

Esta interface é constituída por 10 componentes, sendo o valor *Other* correspondente ao tráfego que não pertence a nenhuma das designações dos presentes, que são:

- Variação do tráfego: este componente expressa a quantidade de informação (em *Megabytes*) que circulou na rede. A análise a este componente permite detetar oscilações na rede que possam corresponder a

uma anomalia (e.g. um ataque ou mau funcionamento de uma aplicação).

- Top de servidores: indica os servidores da LAN que mais tráfego recebem dos clientes percebendo, assim, quais os mais sobrecarregados.
- Top de clientes: indica os clientes que mais tráfego geram. Com este componente é possível descobrir situações anormais, como por exemplo, vírus nos dispositivos dos clientes.
- Top departamentos: este componente ilustra os 5 departamentos que mais tráfego geram, disponível através de um gráfico de barras horizontal.
- Top das aplicações acedidas pela Internet: permite compreender quais as aplicações da empresa mais requisitadas pelos utilizadores da Internet.
- Top dos aplicações acedidas pela LAN: este componente apresenta o top das aplicações da empresa mais utilizadas pela LAN. O valor *Other* é omitido neste componente para obter, apenas, informação das aplicações.
- Top dos serviços acedidos pela Internet à LAN: este top indica os serviços mais requisitados pela Internet à LAN.
- Top dos serviços acedidos pela LAN à Internet: análogo ao último componente descrito, a informação apresentada corresponde aos serviços mais requisitadas pelos clientes da LAN à Internet.
- Top dos serviços acedidos da LAN à LAN: são apresentados os serviços mais utilizadas dentro da LAN.
- Lista de *flows*: exibe a informação dos *flows* capturados.

Todos os componentes acima descritos apresentam a informação em tempo real ou, recorrendo a filtros, num período de tempo especifico, sendo os valores de tráfego apresentados em *Megabytes*. Para obter informação mais pormenorizada é possível filtrar a informação pelos 5 componentes descritos na no módulo anterior e, também, pelos sensores que capturam e exportam os *flows*. Outra funcionalidade do *Dashboard* são os alertas. Os alertas são mensagens que avisam sobre um evento anormal que está a ocorrer na rede. Estas mensagens são enviadas pelo módulo de Otimização/Alertas e apresentadas na página *Dashboard*. No canto superior direito do *Dashboard* (Fig. 2) encontra-se o número de alertas recebidos e, após clicar nesse número, surgem os alertas. A página *Alertas*, acessível através do menu, disponibiliza a listagem dos alertas e das ações executadas pela plataforma. O último item do menu, *Configurações*, encaminha para a página de gestão da aplicação, permitindo gerir os 5 componentes apresentados no módulo de Monitorização. Permite ainda realizar a gestão dos alertas e ações, que serão apresentadas no módulo de Otimização/Alertas.

A comunicação entre o *Dashboard* e o módulo de Monitorização é realizada através de uma *REST API* [17]. Esta API, desenvolvida em *JAVA* e utiliza como servidor a aplicação *Glassfish* [19], converte os pedidos do *Dashboard* em comandos SILK e envia o resultado no formato JavaScript Object Notation (JSON) [18].

C. Módulo de Otimização/Alertas

O módulo de Otimização/Alertas é responsável por executar alertas e/ou ações quando são detetadas anomalias na rede. O objetivo é fazer com que a plataforma execute medidas que atenuem os impactos destas anomalias. Neste contexto, considera-se uma anomalia como sendo um comportamento anormal da rede (e.g. picos de tráfego por um dispositivo). Este conceito é subjetivo na medida em que um comportamento pode ou não ser considerado uma anomalia dependendo da rede em questão. O lado direito da Fig. 1, que representa a arquitetura da plataforma, ilustra os processos envolvidos neste módulo: *Pipeline*, *Pipeline-Listenner* e Otimizações/Alertas.

O *Pipeline* é uma ferramenta, desenvolvida pelos mesmos responsáveis do SILK (CERT NetSA), que inspeciona os *flows* no momento em que estes são criados no SILK, gerando alertas quando necessário. O modo de operação desta ferramenta consiste em três etapas:

- Filtros: são definidos critérios para filtrar os *flows* que, possivelmente, fazem parte de uma anomalia. Aqueles que corresponderem aos critérios serão enviados para a próxima etapa. Exemplo de um filtro é o tráfego HTTP que utiliza, por omissão, a porta 80 e o protocolo de transporte TCP.
- Processamento: nesta etapa, os *flows* filtrados, são processados por *evaluations* ou por *statistics*. As *evaluations* comparam os seus estados internos (valores acumulados) com valores previamente definidos. As *statistics* exportam os seus estados (valores acumulados) em intervalos de tempo previamente definidos.
- Alertas: são gerados quando os estados das *evaluations* forem iguais ou superiores aos valores definidos ou quando forem atingidos os intervalos de tempo das *statistics*.

Os filtros, *evaluations* e *statistics* são definidos no ficheiro de configuração do Pipeline, recorrendo à *syntax* desta ferramenta que pode ser consultada em [11]. Os alertas, enviados pelo Pipeline, podem ser capturados através da ferramenta *Snarf* [10] (opção utilizada na plataforma) ou podem ser escritos num ficheiro para posterior leitura. O *Pipeline-Listenner* é o módulo da plataforma que recebe os alertas do Pipeline, através do *Snarf,* converte-os para o formato JSON e entrega-os ao próximo módulo, Otimizações/Alertas.

O módulo Otimizações/Alertas é responsável por alertar os utilizadores das anomalias da rede e executar ações para mitigar o impacto das mesmas. No contexto da plataforma desenvolvida, as ações são comandos executados nos dispositivos da rede (e.g. *routers* e *firewalls*) através de protocolos de acesso remoto, como por exemplo, SSH e Telnet. Recorrendo aos comandos nativos dos sistemas operativos dos dispositivos, é possível realizar qualquer operação, desde que

seja implementada pelo sistema operativo (e.g. bloqueio de tráfego associado a um IP e alteração de rotas nas tabelas de *routing*). O *workflow*, representado na Fig. 3, é constituído por cinco tarefas que resumem os passos necessários para alcançar o objetivo do módulo, detetar anomalias na rede e mitigar os seus impactos. Estas tarefas são:

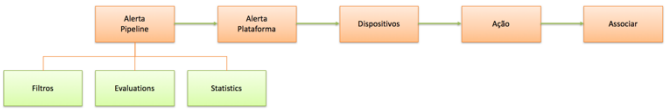

Figure 3. *Otimização/Alertas* - Workflow do módulo

- Definir alerta no *Pipeline*: a primeira tarefa é definir o alerta no *Pipeline* recorrendo aos filtros, *evaluations* e/ou *statistics*.
- Registo do alerta na *Plataforma*: recorrendo à secção de gestão de alertas do *Dashboard*, adicionar um alerta com o número e descrição do alerta criado no *Pipeline* na tarefa anterior.
- Definir dispositivos: definir os dispositivos da rede onde se pretende executar a ação resultante do alerta definido.
- Criar ações: nesta tarefa é criada a ação e os comandos a executar no sistema operativo instalado nos dispositivos definidos na tarefa anterior.
- Associar o alerta à ação e ao dispositivo: por último, associa-se o alerta à ação e aos dispositivos acima definidos.

Concluindo todas as tarefas do *workflow*, a plataforma fica preparada para alertar e mitigar os impactos da anomalia adicionada.

IV. Ensaios Ilustrativos da Plataforma

Nesta secção são apresentados dois exemplos práticos da utilização da plataforma para deteção de anomalias e otimização tendo por base o tráfego capturado na empresa. O primeiro exemplo demonstra o bloqueio de um dispositivo na *firewall* da empresa e o segundo demonstra a possibilidade de efetuar *backups* de um servidor em períodos de tempo de pouca ocupação.

A. *Bloqueio de um dispositivo*

O objetivo deste exemplo é bloquear um dispositivo externo à rede da empresa quando é detetado um *scan* às portas da rede pelo *pipeline*. Nesta demonstração foi utilizada a aplicação *Zenmap* [20], interface gráfica do utilitário *Nmap* [21] que permite descobrir os serviços e dispositivos disponíveis existentes numa rede de computadores, criando assim, um mapa da rede. A Fig. 4 ilustra a arquitetura utilizada na demonstração. O computador, que executou a aplicação *Zenmap*, estava fora da rede da empresa e tinha uma ligação à Internet através de um *router* doméstico, da operadora Vodafone, com o IP público 178.166.70.52. A empresa tem ligação à Internet através do IP público 88.157.192.160. Na mesma figura, a entidade *FlowCollector* é o servidor que executa os módulos de *Monitorização*, da API e da *Otimização/Alertas*, significando, que é este servidor que recebe os *flows* exportados e executa o *Pipeline*. Por fim, o computador, dentro da LAN executa o módulo de *Dashboard*.

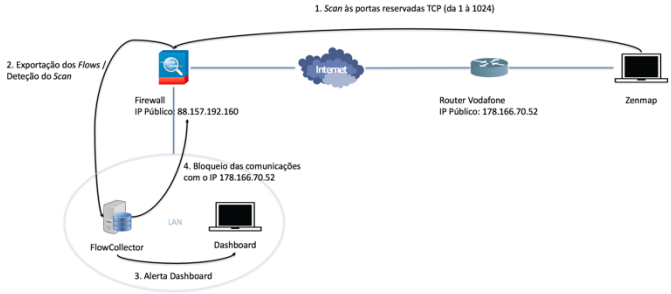

Figure 4. Bloquear Acesso à Rede: Arquitetura da demonstração

A demonstração consistiu em quatro passos. No primeiro passo foi executado um *vertical scan* ao IP 88.157.192.160 utilizando as principais portas TCP reservadas (1 à 1024), através da aplicação *Zenmap*. A Fig. 5 apresenta o resultado da aplicação *Zenmap*.

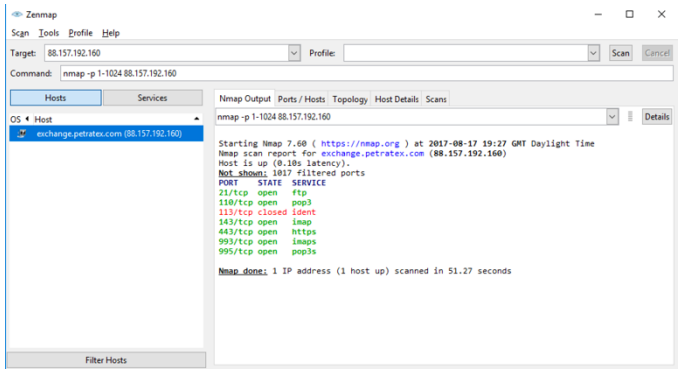

Figure 5. Bloquear Acesso à Rede: Execução do *vertical scan*

O segundo passo consistiu na exportação dos *flows* da *firewall* para o servidor *FlowCollector* e na deteção do *vertical scan* pelo módulo de *Otimização/Alertas*. Após deteção do *scan*, e da execução das operações do módulo *Otimização/Alertas*, são executados os passos 3 e 4.

No passo 3, o servidor *FlowCollector* notifica o módulo de *Dashboard* sobre o *vertical scan* detetado. A Fig. 6 ilustra a notificação recebida.

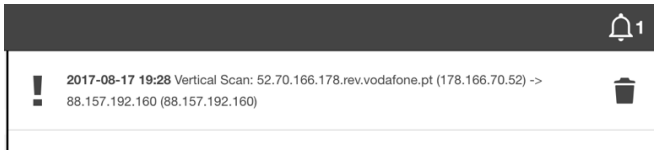

Figure 6. Bloquear Acesso à Rede: Alerta do Dashboard

Por fim, no passo 4 o servidor *FlowCollector* executa os comandos associados à ação em demonstração na *firewall* para bloquear o IP público 178.166.70.52 de comunicar com a rede da empresa. Após este passo, executou-se um segundo *scan* à rede mas sem sucesso devido às novas configurações introduzidas na *firewall*. Nesta demonstração os passos 2, 3 e 4 fazem parte das operações da plataforma desenvolvida, ou seja,

apenas foram executados os *scans* de forma a provocar uma anomalia e verificar que a plataforma tem o comportamento esperado. O tempo que a plataforma necessita para executar as ações é variável devido a vários fatores: os *timeouts* de exportação dos *flows*, o tempo de deteção da anomalia pelo *pipeline*, o poder de processamento do servidor e o tempo de execução dos comandos nos dispositivos da rede.

B. *Backup de um servidor*

Este segundo exemplo demonstra a possibilidade de executar *backups* da informação de um servidor quando este está com pouca carga, não afetando o desempenho das aplicações executadas no servidor. Para este caso considerou-se que o servidor está com pouca carga quando recebe menos de um 1 *Gigabyte* de informação da rede durante 1 hora. A Fig. 7 ilustra o tráfego do servidor em causa no período das 19h às 20h do dia 4 de Outubro de 2017. Neste período o servidor encontrava-se disponível para efetuar *backups* da informação, tendo em conta o tráfego recebido e envido pelo servidor.

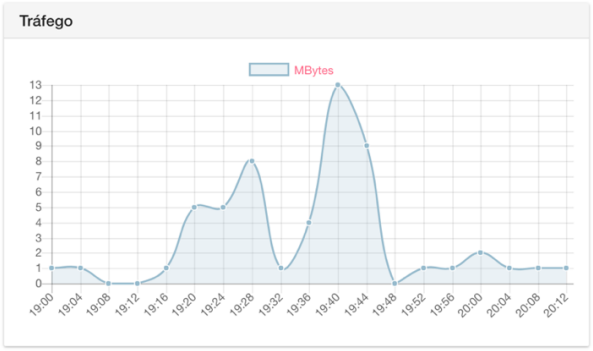

Figure 7. Período em que o servidor *CGPServer* estava com pouca carga

Ás 20h02 foi recebido um alerta, ilustrado pela Fig. 8, que indicava que o servidor se encontrava com pouca carga (recebeu e enviou menos de 1 *Gigabyte* de tráfego no período de 1 hora).

Figure 8. Servidor CGPServer com pouca carga: Alerta *Dashboard*

Recorrendo à lista de alertas e ações executadas, disponível no *Dashboard*, foi possível verificar a ordem de execução, pela plataforma, do *backup* do servidor. Para realização do *backup* foram pré-programados comandos da *shell* do *Linux* através de uma ligação SSH. Neste exemplo, a otimização que a plataforma proporcionou consistiu na possibilidade de executar esta operação de *backup* num período em que o servidor estava mais disponível.

V. CONCLUSÕES

Neste documento foi apresentada uma plataforma de monitorização de rede criada de acordo com as necessidades de uma empresa. A plataforma desenvolvida encontra-se em produção na rede de computadores da empresa, o que prova a sua funcionalidade. Contudo, e como todas as aplicações, precisa de melhorias contínuas para satisfazer as necessidades que vão surgindo. Até ao momento, as melhorias que se destacam são o enriquecimento do módulo de otimização com mais anomalias e as respetivas ações de otimização/recomendação, a possibilidade de realizar as configuração do *pipeline* no *Dashboard* e a otimização do desempenho do *Dashboard* quando são apresentados os resultados de vários dias.

AGRADECIMENTOS

Este trabalho foi apoiado pelos projecto COMPETE: POCI-01-0145-FEDER-007043 e pela Fundação para a Ciência e Tecnologia no âmbito do projecto UID/CEC/00319/2013.

REFERÊNCIAS BIBLIOGRÁFICA

[1] World internet users statistics and 2014 world population. http://www.internetworldstats.com/stats.htm, 2014.

[2] Nagios. https://www.nagios.org, 2017.

[3] Multi router traffic grapher (mrtg). https://oss.oetiker.ch/mrtg/doc/mrtg.en.html, 2017.

[4] B. Claise, B. Trammell, and P. Aitken. Specification of the ip flow information export (ipfix) protocol for the exchange of flow information. RFC 7011, RFC Editor, September 2013. URL http://www.rfc-editor.org/rfc/rfc7011.txt.

[5] Brian Trammell and Elisa Boschi. An introduction to ip flow information export (ipfix). *IEEE communications magazine*, 49(4):89 – 95, 2011-04. ISSN 0163-6804.

[6] Bingdong Li, Jeff Springer, George Bebis, and Mehmet Hadi Gunes. A survey of network flow applications. *Journal of Network and Computer Applications*, pages 567–581, 2013.

[7] T. Zseby, E. Boschi, N. Brownlee, and B. Claise. Ip flow information export (ipfix) applicability. RFC 5472, RFC Editor, March 2009. URL http://www.rfc-editor.org/rfc/rfc5655.txt.

[8] Anna Sperotto, Gregor Schaffrath, Ramin Sadre, Cristian Morariu, Aiko Pras, and B. Stiller. An overview of IP flow-based intrusion detection. *IEEE Communications Surveys and Tutorials*, pages 343–356, 2010.

[9] Ming-Han Wan e Mong-Fong Horng. An intelligent monitoring system for local-area network traffic. *Eighth International Conference on Intelligent Systems Design and Applications*, pages 657–660, 2008.

[10] CERT Network Situational Awareness Group. snarf documentation release 0.3.0. 2017.

[11] CERT Network Situational Awareness Group. Analysis pipeline handbook. 2017.

[12] System for internet-level knowledge. URL https://tools.netsa.cert.org/silk/index.html, 2017.

[13] CERT Coordination Center. Silk installation handbook. 2017.

[14] Ron Bandes e Timothy Shimeall e Matt Heckathorn e Sidney Faber. Using silk for network traffic analysis. analyst's handbook for silk versions 3.8.3 and later. 2014.

[15] Service name and transport protocol port number registry. URL https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml, 2017.

[16] Ionic. URL https://ionicframework.com, 2017.

[17] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD dissertation, Irvine University of California, 2000.

[18] Javascript object notation. URL http://www.json.org, 2017.

[19] Glassfish. URL https://javaee.github.io/glassfish, 2017.

[20] Zenmap. URL https://nmap.org/zenmap/, 2017.

[21] Nmap. URL https://nmap.org/, 2017.