UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Azure2SOC: Integração de Tecnologias de Segurança MS AZURE no Ecossistema do CyberSOC da Altice Portugal

Pedro Martins Gomes Valsassina Galveias

Mestrado em Segurança Informática

Trabalho de Projeto orientado por:
Prof. Doutor Bernardo Ferreira

2021

Agradecimentos

Agradeço ao Professor Doutor Bernardo Ferreira por orientar este projeto, pela sua constante disponibilidade e pelos seus conselhos e anotações, o que contribuiu em muito para o melhor desenvolvimento possível deste trabalho.

Agradeço, também, ao Engenheiro José Alegria, pela coorientação na Altice Portugal, pela confiança, oportunidade de ter ingressado na sua equipa e poder ter concluído o meu mestrado neste ambiente e na companhia de ótimos profissionais, recursos e condições.

A todos os colaboradores da DCY, que, desde o primeiro dia, foram incansáveis na forma como me acolheram – numa altura de novos desafios e rotinas enquanto vivíamos um clima de pandemia – e por todo o apoio e suporte que me ofereceram no decorrer deste projeto. Em particular, agradeço ao Engenheiro Alberto Bruno, que esteve, desde sempre, muito próximo de mim, do trabalho e de todos percalços que ocorreram, tendo sempre a disponibilidade e amabilidade de ajudar em tudo o que lhe era possível.

Aos meus avós

Resumo

No âmbito da conclusão do mestrado em Segurança Informática, foi proposto, pela Altice Portugal, o seguinte projeto. Este consistiu na integração dos alertas de segurança referentes às plataformas *cloud* da *Microsoft – Azure* e *Office 365* – no *IBM QRadar*, o *SIEM* (*Security Information Management System*) do *CyberSOC* (o *Security Operations Center* da Altice Portugal).

Para este efeito, o primeiro passo foi o estudo das tecnologias a utilizar e os seus registos (*logs*). Com isto, foi possível definir estratégias diferentes para a implementação da solução. A seguida foi o uso das potencialidades dos sistemas e produtos *Azure*, com grande ênfase no uso do seu *SIEM* – o *Azure Sentinel* – que teve o papel "cérebro" da operação, visto que era o responsável pela configuração e deteção de incidentes dos quais emitia um alerta. Os restantes produtos *Azure* foram utilizados com o intuito do desenvolvimento de uma *pipeline* robusta e rápida para o tratamento e encaminhamento dos alertas na ligação *Azure* – Altice.

Pode ser dito que a solução foi implementada com sucesso até certo ponto, visto que permite que os alertas sejam visualizados no *IBM QRadar* de forma legível e intuitiva; isto aliado à implementação de um encaminhamento rápido, algo essencial neste tipo de soluções de segurança. Em contrário, por uma questão temporal, não foi possível desenvolver a correlação entre os novos alertas e as fontes de dados já configuradas no *QRadar*, aproveitando mais potencialidades de um SIEM além da visualização dos alertas, sendo que foi um objetivo não alcançado e faz parte do trabalho futuro. Outro ponto importante era a rapidez de todo o processo de encaminhamento.

Este trabalho permitiu um maior panorama de monitorização e análise por parte do *CyberSOC* e a possibilidade de exploração das capacidades da *cloud Microsoft* e dos seus recursos *cloud*, de uma forma mais segura.

Palavras-chave: Cibersegurança, *SIEM*, Integração, *Azure*, *QRadar*

Abstract

Following the conclusion of a master's degree in Information Security and its final dissertation, Altice Portugal proposed the present project. It consisted in the integration of Microsoft cloud platforms security alerts – Azure and Office 365 – in the IBM QRadar, CyberSOC's (Altice Portugal's Security Operations Center) SIEM (Security Information Event Management) system.

The first step was to study the technologies to use and its logs. With this, it was possible to define a series of different approaches to implement the final solution. Secondly, was the use of the Azure system's potentialities and products, with great emphasis on its SIEM – Azure Sentinel – that turned out to be the brain of the operation, being responsible for the configuration and detection of incidents for which it would then emit an alert. The other Azure solutions were used with the intent of developing a solid, and fast pipeline for processing and then sending these alerts from Microsoft Azure to Altice's intranet.

One might say that the solution was, to a certain level, successfully enforced, considering that it allows the alerts to be viewed on the QRadar Console in an easy and comprehensive way, provided by a fast execution pipeline – an important aspect in this type of solutions. However, due to lack of time, it was not possible to develop correlation between the new alerts and the data sources previously configured in QRadar, that would allow to take advantage of more potential of a SIEM system beyond, the visualization of these alerts. Since this objective was not achieved, it is part of this project's future work.

This work allowed, therefore, CyberSOC to have an enhanced monitoring and analysis panorama and the possibility of exploring the capabilities of the Microsoft cloud and its computing resources, now in a safer way.

The remaining document is written in Portuguese.

Keywords: Cybersecurity, SIEM, Integration, Azure, QRadar

Conteúdo

Lista de Figuras

Acrónimos e siglas

AD Active Directory

B2B Business-to-business

CSIRT Computer Security Incident Response Team

CSV Comma Separated Values

DCY Direção de Cybersecurity e Privacidade

DSM Device Support Module

FTP File Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

IaaS Infrastructure as a Service

IP Internet Protocol

ISO International Organization for Standardization

KQL Kusto Query Language

MS Microsoft

PaaS Plataform as a Service

QID QRadar Identifier

SIEM Security Information and Event Management

SFTP Secure File Transfer Protocol

SOAR Security Orchestration Automated Response

SOC Cyber Security Operations Center

TCP Transmission Control Protocol

TLS Transport Layer Security

UDP User Datagram Protocol

URL Uniform Resource Locator

Capítulo 1

Introdução

1.1 Contexto

O projeto foi realizado com o apoio da Altice Portugal, antiga Portugal Telecom, com a orientação do Eng. José Alegria, acompanhado do Eng. Alberto Bruno no local e pelo Prof. Doutor Bernardo Ferreira da parte da Faculdade de Ciências da Universidade de Lisboa; serve, também, como dissertação final do mestrado em Segurança Informática do ano letivo de 2020/2021.

A Altice Portugal é uma gigante tecnológica nacional, pioneira numa série de tecnologias e soluções, que fornece serviços de televisão, comunicação, redes e sistemas de informação, entre outros. Devido à sua dimensão, esta tem ao seu dispor uma equipa de cibersegurança, na qual está inserida um *Security Operation Center* (*SOC*), no qual têm um *software* de *SIEM* (*Security Information Event Management*) – o *IBM QRadar*. O trabalho foi desenvolvido no âmbito de enriquecer e melhorar este sistema ao integrar e tratar as ocorrências detetadas nos sistemas *cloud* da *Microsoft – Azure* e *Office 365* – neste *SIEM*.

Por questões de privacidade e sensibilidade dos dados que foram tratados durante este projeto, muitos alertas e, consequentemente as suas especificações, serão omitidos. No entanto, por motivos de enriquecimento do documento e explicação do que foi feito no trabalho, serão dados alguns exemplos.

1.2 Motivação

Com o passar do tempo, e em especial nos últimos anos, a complexidade dos sistemas, soluções empresariais e produtos para o cliente comum – todos estes de componente informática – têm aumentado exponencialmente. Desde maiores capacidades e velocidades de comunicação, à integração e automatização do mais simples utensílio

caseiro com a rede doméstica. Este aumento drástico e rápido parece até desafiar leis que tentam projetar e quantificar o desenvolvimento do ramo, como a de Moore ou Koomey: "o número de transístores tem um aumento de 100%, pelo mesmo custo, a cada 18 meses." [1] e "o número de operações por joule de energia aumenta para o dobro a cada 1,57 anos." [2], respetivamente.

Tendo em conta todo este paradigma de evolução, os ataques informáticos e os seus vetores de ataque também têm acompanhado este crescimento. Com isto, as organizações tiveram de se adaptar, de modo a proteger os seus ativos. As grandes empresas, como é o caso da Altice Portugal, têm, nos seus quadros, uma equipa qualificada para tratar do tópico da cibersegurança. Dentro dessa estrutura, a equipa de segurança tem, ao seu dispor, um *Security Operation Center* (*SOC*) – "componente responsável pela monitorização constante e acompanhamento de incidentes em tempo real. Entre as tarefas mais comuns do *SOC*, encontram-se a reação rápida a incidentes, gestão de crises, coordenação com a equipa de IT do cliente, coordenação com autoridades e arquivo de *logs* relevantes" [3]. De forma à equipa *SOC* poder desemprenhar da melhor forma as suas funções, este dispõe de um *Security Information Event Management* (*SIEM*) – um *software* que agrega e analisa a atividade de diferentes pontos e estruturas espalhadas pela rede da organização. Armazena, normaliza, agrega e analisa esses dados para descobrir tendências, detetar ameaças e permitir que as organizações investiguem quaisquer possíveis incidentes [4].

Para melhorar este sistema, o presente trabalho vai-se focar na integração da informação dos sistemas *cloud* da Altice Portugal, que estão hospedados em *Microsoft Azure*, no seu *SIEM IBM QRadar*. Para isso foi desenhada e produzida uma solução que, automaticamente, processa as informações do sistema *Azure* de modo a que estas apareçam e possam ser analisadas diretamente no sistema *QRadar*, tendo aí o acesso ao panorama geral, mas completo, das incidências de segurança.

1.3 Objetivos

Este trabalho focou-se no desenho e criação de uma solução que permita uma automatização da replicação dos *logs* de incidências e alertas de segurança dos sistemas *Microsoft Azure* e *Office 365* da Altice Portugal para o seu sistema de *SIEM*, o *IBM QRadar*. Como exemplos dos recursos *cloud* temos o seu *Active Directory* ou os registos

referentes à plataforma do *Office 365*. Procurou-se desenvolver uma solução que permita a deteção de incidentes e levantamento dos respetivos alertas; que faça um encaminhamento fiável e rápido para a rede interna da Altice e, de seguida para o *IBM QRadar*; e que apresente estes alertas com todos os seus campos, informações relevantes e severidade associada, tudo isto de forma legível e interativa. Continuando na vertente do *IBM QRadar*, outro objetivo do trabalho é o uso destes novos alertas em novas regras de correlação com outras fontes de dados, consequentemente criando novos *use cases* e enriquecendo, de forma mais completa, a experiência de monitorização e resposta do *CyberSOC.*

Neste contexto, a primeira fase consistiu na familiarização com as tecnologias da *Microsoft*, das suas potencialidades, não exclusivamente a nível de segurança, e dos registos – *logs* – das mesmas; no estudo e familiarização com o *IBM QRadar*, o *SIEM* já usado pela Altice Portugal, e no estudo das melhores estratégias abordagens e decisões a tomar. Depois disto, prosseguiu-se para um desenvolvimento mais detalhado da *pipeline Azure* – Altice e à implementação desta solução, tal como esta passou por uma série de testes para confirmar o seu bom funcionamento. Houve, ainda, uma última fase que contemplou a sua produção e integração no *SIEM* do *SOC* da Altice Portugal, apresentado neste os alertas já tratados e de forma facilmente legível e compreensível. Esta foi a última fase por não ter sido possível, temporalmente, avançar para a etapa de configuração da correlação com as fontes de dados já existentes no *IBM QRadar* da organização, incluindo a criação de *use cases* tendo por base os novos alertas.

1.4 Contribuições

As contribuições deste trabalho focam-se, principalmente, no melhoramento e no aumento da capacidade de *awareness* do *SOC* da Altice, o *CyberSOC*, enriquecendo o seu *SIEM*, o *IBM QRadar*, com informações referentes às plataformas *cloud* que utiliza, o *Microsoft Azure* e o *Office 365*. Tendo isto em conta, podemos destacar as seguintes contribuições oferecidas:

- A adição de uma nova camada de segurança e monitorização ao *SIEM*, na medida em que agora é possível supervisionar e detetar ataques e outros incidentes de segurança que ocorram num ambiente *cloud Microsoft.*

- A centralização das informações no *SIEM* já utilizado pela equipa do *SOC*, não sendo necessária um estudo ou formação adicional noutro *software*, ou até alterações nos processos de funcionamento do *SOC*.
- A criação de um processo e de uma *pipeline* fiável entre a plataforma *Azure* e a rede interna da Altice que permita o tratamento e, de seguida, o encaminhamento das informações, neste caso o alerta de segurança e os seus detalhes.
- A deteção e o levantamento dos alertas pretendidos, seguida da apresentação dos mesmos na interface do *IBM QRadar*. Esta apresentação é feita de uma forma legível e com todos os campos e informações relevantes do alerta; é associado e mapeado, para cada alerta, um nome, categoria e severidade própria do *SIEM* da *IBM*.
- A possibilidade de potencialização de todas as capacidades dos sistemas *Azure*, visto que agora os recursos da empresa que utilizem estes serviços são vigiados pelo *IBM QRadar* e pela equipa do *CyberSOC*. Permite uma futura exploração de todas as potencialidades das plataformas *cloud* e da computação em nuvem com muito menos desconhecimento e exposição a nível de segurança.

Resumidamente, estre trabalho resulta num enriquecimento no que toca às informações que são trabalhadas pelo *SIEM* do *CyberSOC* e contribui para a evolução do seu panorama de monitorização, resultante da adição dos alertas de segurança das plataformas e produtos *cloud* da *Microsoft*. Noutro ponto, permite um abrir de porta à exploração das potencialidades destes produtos e da computação em nuvem de uma forma mais segura.

1.5 Organização do documento

Este documento está organizado da seguinte forma:

- Capítulo 2 – Contexto e trabalho relacionado
    - Neste capítulo é dado um contexto relativo ao ambiente de trabalho, ao tipo de tecnologias que vão ser utilizadas, é feita uma análise mais

detalhada referente a *software* ou serviço usado e às estratégias a delinear para o desenrolar do processo.

- Capítulo 3 – Implementação da solução
    - Este capítulo foca-se em detalhar o que foi feito neste trabalho para a implementação da solução. Em suma, é uma descrição detalhada, passo-a-passo, de todo o processo de implementação.
- Capítulo 4 – Avaliação da solução
    - Tendo por base o que foi proposto fazer neste projeto, neste capítulo procura-se avaliar se o resultado final do mesmo satisfaz, e de que forma, os objetivos iniciais do trabalho.
- Capítulo 5 – Possíveis melhoramentos e alterações
    - Neste capítulo, tal como o nome indica, vai ser abordado um conjunto de medidas e alterações que podem vir a melhorar do projeto.
- Capítulo 6 – Conclusão e discussão
    - Este capítulo foca-se num balanço geral de todo o trabalho. Desde os objetivos, ao trabalho feito, ao que correu bem e menos bem.

Capítulo 2

Contexto e trabalho relacionado

2.1 *SOC*

Como já explicado anteriormente - mas de forma menos exaustiva - um centro de operações de segurança, ou *Security Operations Center* em inglês (*SOC*), é uma componente que tem ao seu dispor uma equipa responsável por monitorizar e analisar, continuamente, a vertente de cibersegurança de uma organização. A equipa do *SOC*, além da componente de monitorização, pode ter nos seus quadros uma equipa de resposta a incidentes, de modo a garantir que os problemas sejam resolvidos rapidamente após a sua descoberta.

Os *Security OperationsCcenter* monitorizam e analisam atividades na rede, servidores, *endpoints*, bases de dados, aplicações, *websites* e outros sistemas, procurando atividades anómalas que possam ser indicativas de um incidente ou comprometimento no que toca à segurança do sistema. O *SOC* é responsável por garantir que possíveis incidentes de segurança sejam corretamente identificados, analisados, defendidos, investigados e reportados. O principal benefício de ter um centro de operações de segurança é o aprimoramento da deteção de incidentes de segurança por meio de uma monitorização e análise contínua da atividade na generalidade do sistema.

Essa função de análise, vinte e quatro horas por dia e sete dias por semana, da atividade que ocorre na infraestrutura informática da organização, torna estas equipas *SOC* essenciais para garantir a deteção e resposta adequada aos incidentes de segurança encontrados. É esta monitorização constante, fornecido por um *SOC*, é o que oferece às organizações a vantagem de terem a oportunidade de se defender, em tempo real, contra eventos maliciosos, independentemente da origem, hora do dia ou tipo de ataque [5].

A *time gap* entre o tempo que um agente malicioso leva para comprometer o sistema e o momento em que esta ação é detetada está bem documentada no relatório anual de investigações de violação de dados da *Verizon* [6]. Ter um centro de operações de segurança, funcional e ativo, é essencial para minimizar esse intervalo e mitigar quaisquer consequências que o ataque possa ter.

2.1.1 *SOC* Altice – *CyberSOC*

O *SOC* da Altice Portugal, o *CyberSOC*, tem presença física em Lisboa e Covilhã por motivos de redundância em caso de desastre, e tem a capacidade de ter uma visão panorâmica de toda a organização, como uma visão incisiva e específica sobre certo evento, utilizador ou infraestrutura. O *Security Operations Center* é constituído por vários técnicos de segurança – desde responsáveis pela rápida resposta a incidentes, pelo *threat hunting* e por uma equipa de engenharia, que é responsável pela configuração, desenvolvimento e manutenção dos sistemas (do *SIEM*, por exemplo). A Altice e o seu *SOC* trabalham, também, para terceiros, numa relação de produto *business to business* (*B2B*).

As principais atribuições e papéis do *SOC* da Altice são os seguintes:

- Conceção e gestão de processos de segurança definidos no âmbito das melhores práticas ISO 20000 e ISO 27001, numa perspetiva de certificação e melhoria continua;
- Apoiar as áreas de Gestão de Produto, Consultoria e Engenharia, na conceção de Produtos e Serviços de Segurança e elaboração de projetos complexos de segurança para clientes, incluindo a participação em reuniões com o cliente, apresentação e defesa das soluções e serviços propostos;
- Contribuir para o esforço nacional de cibersegurança no âmbito da Rede Nacional de *CSIRT'*s, na produção de alertas, com recomendações de mitigação associadas, e na promoção em geral de uma cultura de segurança na Internet em Portugal com particular ênfase nos sectores empresariais;
- Garantir a definição, recolha e divulgação de indicadores que demonstrem a visão global sobre o estado da segurança das infraestruturas e serviços geridos e promovam o controlo e melhoria continua da qualidade da segurança associada a esses serviços;
- Assegurar resposta a pedidos técnicos relativos a investigações criminais, sendo originados pelas Autoridades, através do Departamento Jurídico;
- Auxiliar investigações do Departamento de Fraude, acompanhando a pesquisa, identificando fontes, analisando e correlacionando informação

técnica relevante à investigação e garantindo a recolha de evidências seguindo os procedimentos adequados para garantir a sua validade em Tribunal;

- Detetar, analisar e coordenar a resposta a incidentes de Segurança em Sistemas de Informação nas Infraestruturas da Altice Portugal e clientes, em alinhamento com as diversas áreas do grupo e *stakeholders* do processo.

2.2 *SIEM*

O software de *SIEM – Security Information and Event Management* – oferece aos profissionais da equipa de segurança do *SOC*, uma visão e um histórico das atividades de toda a infraestrutura informática da organização, incluindo, por exemplo, desde servidores, a aplicações, passando por *firewalls* e bases de dados.

O *SIEM* identifica, categoriza e correlaciona eventos. Podendo, de seguida, considerar um incidente de segurança, consoante a sua análise [7].

Um evento é qualquer ocorrência observável no nosso ambiente ou infraestrutura informática. Um evento pode ser algo tão banal e inofensivo como aceder à rede da organização ou a receção de um *e-mail*. No entanto, nem todos os eventos são incidentes, mas todos os incidentes são eventos, ou conjuntos de eventos. Um incidente é algo que afeta, de forma negativa, os sistemas informáticos e, consequentemente, tem impacto na vertente do negócio. Pode ser uma diminuição não planeada da disponibilidade ou uma redução da qualidade de um, ou mais, serviços informáticos [8].

Resumidamente o *software* procura satisfazer, principalmente, um objetivo principal. É consumir uma série de eventos de uma, ou mais fontes de dados, analisá-los, correlacioná-los e alertar no caso de um ou mais desses eventos corresponderem a alguma regra de um conjunto de regras predeterminadas referentes às fontes de dados que são monitorizadas e, portanto, indicar um possível incidente e consequente problema de segurança.

Por exemplo, um utilizador fazer cinco tentativas de acesso à sua conta em dez minutos é considerado aceitável. No entanto, um utilizador fazer cem tentativas de acesso à sua conta num intervalo de dez minutos seria sinalizado como uma potencial tentativa de ataque e, consequentemente, um incidente de segurança [9, 10].

Assim sendo, e usando os seus *dashboards*, o *SIEM* ainda nos permite visualizar todos estes eventos e alertas, além de outras métricas e estatísticas da nossa infraestrutura informática, permitindo um aumento da eficiência das investigações e respostas e redução do tempo perdido em análise [11].

2.2.1 *IBM QRadar*

O *QRadar* é uma solução de *SIEM* desenvolvida pela *IBM*. Como já explicado anteriormente, um *SIEM* é o sistema responsável pela recolha, processamento dos eventos e ocorrências que considere importantes e pelo alerta consequente. Neste ponto vai ser feita uma introdução à estrutura e funcionamento do *QRadar*.

Este sistema é uma solução modular que permite uma monitorização em tempo real de toda a infraestrutura informática da organização, e, com isto, detetar e responder a possíveis ameaças. Sendo um sistema bastante flexível e costumizável, podemos, facilmente, configurar a plataforma à medida das nossas necessidades – dos nossos *logs*, *flows* ou do tipo de análise que se pretende fazer [12].

Este sistema apresenta muitas vantagens no que toca a essa deteção, análise e resposta. Um dos pontos fortes e proveitosos do *QRadar* é a capacidade de correlacionar, automaticamente, eventos referentes ao incidente em questão, oferecendo uma visão *end-to-end* de toda a cadeia de acontecimentos, facilitando a análise, resposta e resolução do problema. Outro é a sua aptidão para lidar com um volume de dados e tráfego muito elevado, de várias fontes diferentes, e, de seguida, a sua normalização para análise. Isto acaba por facilitar, e muito, o trabalho de um analista, ao ter os registos e informação na forma mais compreensível possível – seja para análise imediata, seja para arquivo e análise posterior. Isto tudo sem sobrecarregar o utilizador ou administrador com problemas e preocupações no que toca a gestão espaço de armazenamento ou de performance computacional, já que o *QRadar* tem um sistema que é facilmente escalável e que se ajusta, automaticamente, aos recursos disponíveis/necessários [12, 13].

2.2.1.1 *DSM* e *DSM Editor*

Os dados que o *QRadar* tem a capacidade de receber podem ser, como já dito anteriormente, provenientes de várias fontes. Este processo é agilizado por algo que é

denominado por *Device Support Module* (*DSM*), um ficheiro *plug-in*, que é responsável por encaminhar eventos de uma fonte externa, convertendo-os para um formato adequado e normalizando-os para poderem ser processados e visualizados da forma mais apropriada. Por exemplo, o *DSM* do 3Com Switch 8800 encaminha e normaliza os dados dos eventos proveniente de um 3Com Switch 8800 [14].

O *DSM Editor* é a plataforma responsável pela criação e edição destes *DSM's*. Podemos ter *DSM's* já feitos pela *IBM* em parceria com outras entidades, como este da *3Com*, como no exemplo dado anteriormente, e podemos criar nós os nossos *DSM's* consoante as nossas necessidades e o que queremos adicionar e normalizar no *SIEM*. Este permite o mapeamento dos novos eventos e alertas – a sua adição à base de dados do *QRadar*, extraindo as suas propriedades recorrendo a *regular expressions* e outras táticas [15].

Além de todas estas definições e funcionalidades, podemos, ainda, aprofundar e melhorar a experiência de *SIEM* recorrendo a uma customização mais detalhada e profunda, consoante as nossas necessidades. Para isto podemos recorrendo a módulos *QRadar*, que podemos ativar, usar e destacar consoante as nossas intenções – tendo, por exemplo, um módulo de gestão de risco ou de gestão de vulnerabilidades.

O funcionamento do *QRadar* pode-se dividir em três etapas: recolha de dados, processamento de dados e uso/análise dos dados.

2.2.1.2 Recolha de dados

Esta é a primeira etapa – consiste na recolha de dados presentes na rede, que podem ser considerados como eventos ou *flows*. O *QRadar* usa *collectors*, estruturas desenhadas para recolher dados das *log sources* ou diretamente da rede. Estes são, mais exatamente, um *event collector* e um *flow collector*, respetivamente [13, 16]. Um *Device Support Module* pode, também, funcionar como *collector* exclusivamente e diretamente para o tipo de dados que recolhe. Após esta recolha, os dados são normalizados, a sua informação convertida em formatos usáveis pelo sistema e passados à camada de processamento de dados [16].

Numa explicação mais detalhada, os eventos referem-se às ocorrências que sucedem num ambiente de utilizador, ou seja, *logs* de utilização de aplicações, *logins*, uso de plataformas de *e-mail*, registos de *firewall*, etc. [17].

Explicando, agora, os *flows* – são registos de atividade entre dois nós da rede. Isto é, o *QRadar* normaliza e extrai, desses dados, informações como endereço *IP*, portos usados, tamanho e número de pacotes, entre outros [17].

2.2.1.3 Processamento de dados

Depois da recolha de dados ocorre o processamento dos mesmos. Ou seja, depois dos dados serem normalizados para um formato usável, através de instâncias denominadas de *flow processors* e *event processors* estes são corridos num *Custom Rules Engine* (*CRE*), componente no sistema responsável por, consoante os dados que recebe e as regras pré-definidas que tem, gerar alertas e reportar incidentes. De seguida, estas informações são armazenadas para um futuro tratamento [13].

É, também, nesta fase que podem entrar os módulos adicionais do *QRadar*, como o *QRadar Risk Manager* (*QRM*), *QRadar Vulnerability Manager* (*QVM*), ou o *QRadar Incident Forensics*, que, consoante as suas intenções, podem captar outros dados e fazer outras análises mais específicas no que toca à sua função.

Dando um exemplo de funcionalidade de cada um – o *QRadar Risk Manager* permite simular várias tipologias e organizações de rede, como alterar configurações e regras do *Custom Rule Engine*, de modo a planear uma disposição e configuração mais adequada e segura. O *QRadar Vulnerability Manager* pode cruzar os dados recolhidos e processados pelo sistema com as informações de *softwares* de *scanning* de vulnerabilidades, como o *Nessus* ou o *Rapid7*, de modo a ter uma análise de ameaças e alertas mais sólida e apropriada à severidade de cada ocorrência. O *QRadar Incident Forensics* permite-nos, por exemplo, fazer um *replay* de toda uma sessão de rede para uma análise mais cuidada da mesma [13].

2.2.1.4 Uso/análise dos dados

A terceira e última fase do processo é bastante simples. Foca-se exclusivamente no uso e análise dos dados por parte dos utilizadores ou administradores, podendo estes pesquisar e analisar os dados já previamente recolhidos, normalizados e processados, reconfigurar definições e atuar sobre ocorrências, tudo isto diretamente pela customizável e flexível interface do *QRadar Console* [13]. Um resumo do funcionamento e organização do QRadar pode ser ilustrado pela Figura 1.

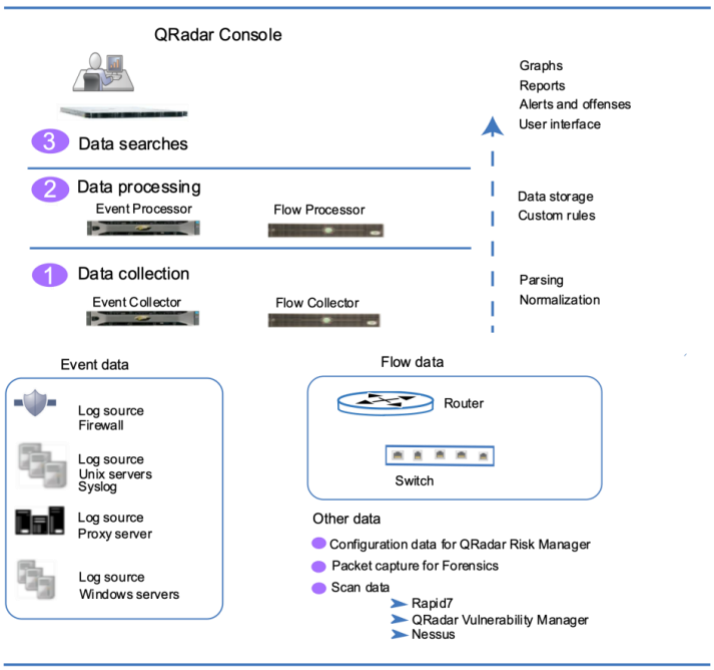

Figura 1 - Organização do funcionamento do *IBM QRadar*

Fonte: https://sensei-infosec.netlify.app/siem/ibm-qradar/2020/04/14/qradar-architecture.html

2.2.2 *Azure Sentinel*

O *Azure Sentinel* é uma solução de *Security Information Event Management* (*SIEM*) e *Security Orchestration Automated Response* (*SOAR*) que pertence à plataforma *cloud* da *Microsoft*, a *Azure*. Por isso mesmo, este software de segurança tem o seu funcionamento baseado em computação em nuvem. Visto isto, o *Azure Sentinel* oferece uma análise inteligente com métricas adequadas e *threat intelligence* em toda a organização cliente, fornecendo, assim, uma centralização dos serviços de deteção de incidente e levantamento de alertas, análise de ameaças, *hunting* e resposta rápida a incidentes [18], sendo este *overview* de capacidades representadas na Figura 2.

O *Sentinel*, ao trabalhar numa infraestrutura *cloud*, permite captar todos os dados de utilizadores, aplicações, dispositivos e redes hospedadas numa ou em mais plataformas *cloud*, além das suas capacidades para fazer o mesmo numa infraestrutura *on-premise*. A *threat intelligence* adjacente ao software da *Microsoft* – isto é, o conhecimento que temos de modo a prevenir e mitigar ataques ou outros incidentes – permite uma deteção rápida e uma minimização de falsos positivos. Tem a seu dispor, também, uma possibilidade de integração com outras soluções – uma *Logic App*, por exemplo, sendo essa solução chamada de *playbook* – que permite o *Sentinel* adquirir o estatuto de *SOAR*, abrindo

portas a automatizações na resposta a incidentes. Por fim, o *Sentinel* destaca-se, também, pelo seu constante contacto e apoio em algoritmos de inteligência artificial na investigação de ameaças e incidentes, como na procura de atividades suspeitas em conjuntos de grandes quantidades de dados [18].

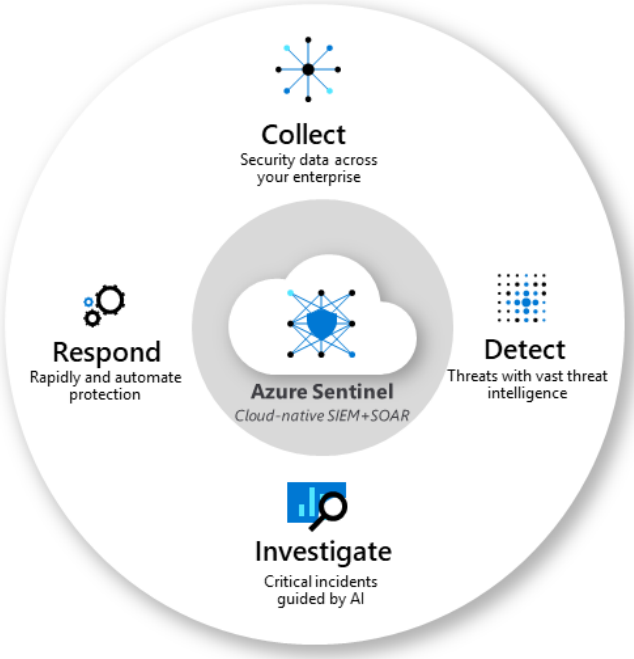

Figura 2 - Overview da estrutura e das capacidades do *Azure Sentinel*

Fonte: https://docs.microsoft.com/pt-pt/azure/sentinel/overview

No que toca a ligação e fontes de dados, o *Azure Sentinel* fornece uma série de conectores para produtos *Microsoft*, sendo estas muito fáceis de configurar (muitas vezes não mais do que um simples carregar de um botão) e de disponibilidade imediata, sendo que os dados dessas mesmas soluções são prontamente encaminhados para o *Sentinel*, ficando prontos para serem analisados. Entre estes conectores está o *Office 365*, o *Azure Active Directory* e o *Microsoft 365 Defender*. Além dos conectores para produtos *Microsoft*, o *Sentinel*, obviamente, fornece soluções para produtos de terceiros. Podemos ligar, também de maneira muito fácil, fontes de dados como *Syslog* ou *REST-API*, permitindo a integração mais personalizada e adequada às nossas necessidades e às especificações dos nossos produtos e sistemas [18].

De modo a aprimorar a deteção de alertas e incidentes, o *SIEM* da *Microsoft* tem ao seu dispor uma solução de seu nome "*Analytics"*. Trata-se de *queries* agendados para correr, sucessivamente, a cada intervalo de tempo definido pelo utilizador, podendo

percorrer uma série de dados e eventos à escolha e retornam-nos, se estes corresponderem aos nossos parâmetros de deteção. Estas regras de alarmísticas são configuradas diretamente no portal do *Microsoft Azure*, escrevendo o *query* num editor próprio e utilizando a linguagem *Kusto Query Language*; são conjuntos de *queries* às nossas fontes de dados – os nossos conectores – que podemos criar ou usar alguns já feitos, e que permitem detetar, alertar e no fim levantar um incidente, que pode ser constituído por um ou mais eventos ou alertas – correlacionando-os, se for esse o caso. É apresentado um exemplo de uma destas queries na Figura 3. Isto permite uma melhor organização e higiene no que toca ao número de alertas e incidentes que teremos de rever ou investigar. Além desta vertente, que, como já dito, é bastante comum neste tipo de *software*, o *Azure Sentinel* alia *machine learning* de modo a mapear os comportamentos das entidades na rede e procurar anomalias em todos os seus recursos e infraestruturas [18].

```
1  VMComputer
2  | distinct Computer, PhysicalMemoryMB
3  | join kind=inner (
4      InsightsMetrics
5      | where Namespace == "Memory" and Name == "AvailableMB"
6      | project TimeGenerated, Computer, AvailableMemoryMB = Val
7  ) on Computer
8  | project TimeGenerated, Computer, AvailableMemoryMB, PhysicalMemoryMB
```

Figura 3 - Exemplo de uma regra de alarmística usando a *Kusto Query Language*

No que toca ao processo de investigação – como costume neste tipo de soluções – o *Sentinel* coloca, ao nosso dispor, uma interface bastante intuitiva, que permite uma análise simples, mas ao mesmo tempo cuidada e profunda, dos incidentes e possíveis ameaças à segurança. Além desta investigação reativa, permite, também, o desenvolvimento de um processo de *hunting*. Isto é, a procura proactiva, usando os dados reais que fornecemos ao *Sentinel*, de outras ameaças antes que o alerta referente às mesmas seja desencadeado. Dá, assim, a possibilidade de uma maior prevenção, *threat intelligence* melhorada e permite estar um passo à frente do possível agressor [18].

Outro aspeto interessante deste *software* é a forte relação de proximidade com os desenvolvedores e analistas, na área da segurança, da *Microsoft*. Estes vão criando com regularidade novos *playbooks*, *hunting queries*, regras de alarmística e outros itens que, posteriormente, são publicados no repositório *GitHub* próprio do *Azure Sentinel*, de modo a que os seus utilizadores os possam utilizar [18] – como pode ser observado a sua vasta oferta de conteúdos na Figura 4.

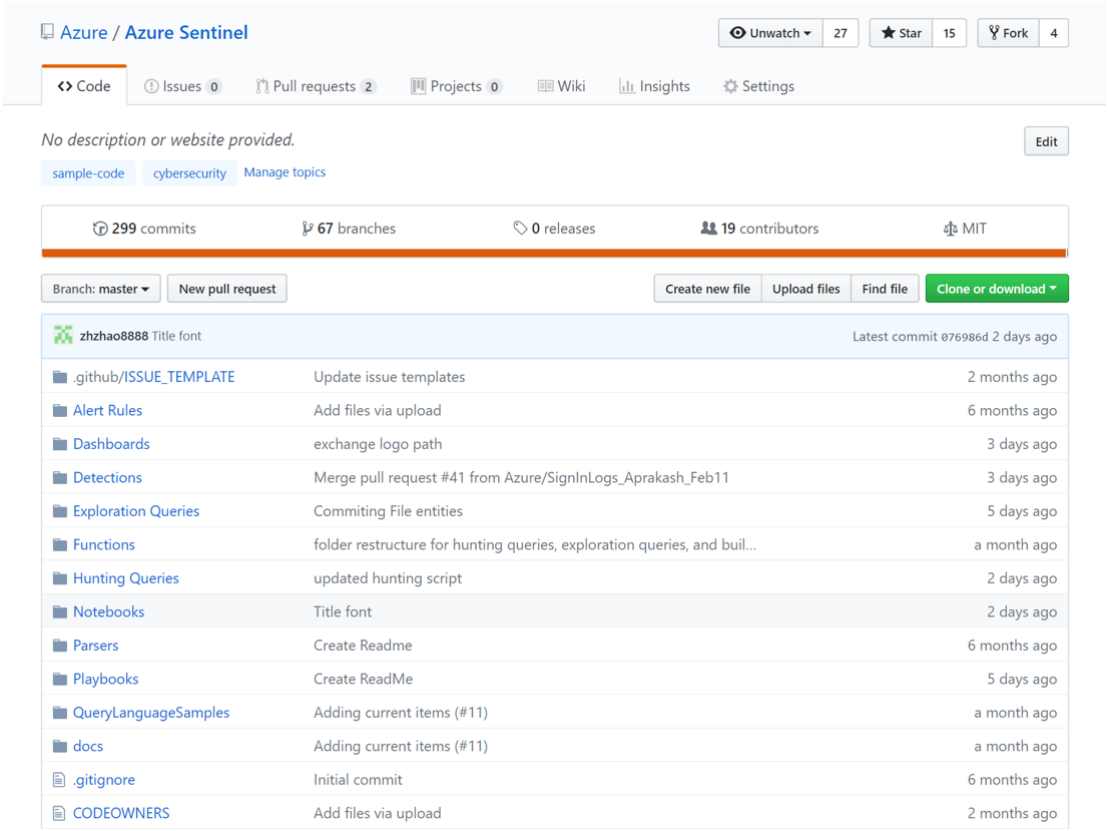

Figura 4 - Captura de ecrã da primeira página do repositório *GitHub* do *Azure Sentinel*

2.3 *Microsoft Azure*

O *Microsoft Azure* é uma plataforma computacional baseada em *cloud*, que permite hospedar sistemas, ferramentas e recursos de uma organização. Esta solução, no fundo, serve para fornecer um apoio tecnológico à organização, minorando os custos e complicações que podem existir nestes processos. (Como por exemplo, a necessidade de ter e gerir um grande *data center* próprio.) Com este cenário, a organização confia sistemas como, por exemplo, faturação, *business intelligence*, armazenamento de dados ou ambientes de desenvolvimento de software à *cloud* do *Azure*.

Mais concretamente, no que toca a serviços computacionais, oferece a possibilidade de correr máquinas virtuais – *Infrastructure as a Service* (*IaaS*) – dando a possibilidade de iniciar máquinas virtuais Windows ou Linux; e serviços de aplicação – *Plataform as a Service* (*PaaS*) – facilitando toda a componente de desenvolvimento e execução de aplicações [19]. Quanto a serviços de gestão de identidade, o *Azure* oferece o *Active Directory*. Este permite um início de sessão único, facilitando o acesso aos recursos por parte do utilizador, além de permitir a gestão de privilégios e acessos aos

mesmos, como a criação e gestão desses mesmos recursos – domínios, utilizadores e objetos – dentro da rede, como seria espectável num produto de *Active Directory* [20]. Como dito anteriormente, a plataforma da *Microsoft* também disponibiliza serviços de armazenamento e gestão de dados. Entre estes serviços, além do serviço padrão de base de dados *SQL*, apresenta, também, soluções *NoSQL* [19]. Oferece, adicionalmente às bases de dados, o armazenamento e transferência de ficheiros num formato *file system* [21]. Estas são as funcionalidades e aspetos mais conhecidos e mais utilizados da plataforma *Microsoft Azure* e, como tal, também entre os mais relevantes para este projeto. Além destes, tem ainda outras soluções, tal como uma vertente de inteligência artificial e *machine learning*, *internet of things*, *blockchain*, media, *PaaS* para dispositivos móveis, etc. [19].

2.3.1 Segurança no *Microsoft Azure*

Num tópico mais relevante a este trabalho, o *Microsoft Azure* também dispõe de uma série de soluções no que toca à cibersegurança. Entre estas, está o *Azure Sentinel*, o *SIEM*, e o *Azure Security Center*, ferramenta que fornece uma gestão de segurança centralizada, que permite, por exemplo, identificar, alertar e corrigir vulnerabilidades existentes ou configurações incorretas [22].

Contrastando estes dois sistemas, o *Azure Sentinel* tem um papel mais virado para a monitorização, análise contínua e investigação de toda a infraestrutura *IT* (*Azure*, *third-party* e/ou *on-premise*), como *SIEM* que é; enquanto o *Azure Security Center* foca-se mais na prevenção e resolução de vulnerabilidades e *misconfigurations* que possam dar, ou ter dado, origem a incidentes de segurança [23].

2.3.2 *Azure Event Hubs*

Nesta secção faz-se um pequeno enquadramento do que é a solução dos *Event Hubs* disponibilizados pelo *Microsoft Azure*, visto que nos vai ser essencial para a execução deste trabalho na sua estratégia e plano principal.

O *Azure Event Hubs* é uma plataforma de *streaming* de dados e serviço de ingestão de eventos. Pode receber e processar milhões de eventos por segundo. Os dados enviados

para um *hub* de eventos podem ser transformados, armazenados e/ou enviados para plataformas *third-party*. Estes têm uma série de panoramas de utilização possíveis, tais como a deteção de anomalias (fraude/valores atípicos), registo de aplicação, arquivo de dados, processamento de transações, entre outras [24].

Os *Event Hubs* representam a "porta de entrada" para uma *pipeline* de eventos, denominado frequentemente como *ingestor* de eventos. Um *ingestor* de eventos é uma componente ou serviço que se encontra entre os agentes que produzem os eventos e os agentes que consomem/tratam esses eventos, funcionando como intermediário. Isto de modo a separar a produção do *stream* de eventos do seu consumo e, consequentemente, separar as duas entidades [24].

2.3.2 *Azure Logic Apps*

As *Azure Logic Apps* – aplicações lógicas – são um serviço da *cloud* da *Microsoft* que permite agendar e automatizar tarefas e *workflows*, sendo possível conciliar outras *apps*, dados, sistemas e serviços. Por outras palavras, este serviço simplifica a forma como se concebe e constrói soluções escaláveis para integração de aplicações, de dados, de sistemas, de aplicações empresariais e de comunicações *business-to-business* [25]. A Figura 5 ilustra a facilidade de uso, simplicidade e intuição no uso desta ferramenta.

Sabendo que esta descrição muito teórica pode não ser a mais esclarecedora, apresento, à frente, alguns exemplos de casos de uso das *Logic Apps* e de alguns cenários que se podem automatizar:

- Enviar notificações de *e-mail* com o *Office 365* quando ocorrem certo tipo de eventos em vários sistemas, aplicações ou serviços.
- Mover ficheiros carregados de um servidor *SFTP* ou *FTP* para o armazenamento do *Azure*.
- Monitorizar tweets sobre um assunto específico, armazená-los e alertar caso haja uma palavra ou expressão específica.

Cada *workflow* de uma *Logic App* começa com um *trigger*, que é acionado quando um evento específico ocorre ou quando novos dados disponíveis cumprem critérios específicos. Sempre que o *trigger* é acionado, é criada uma instância, dessa mesma aplicação lógica, que executa as ações descritas no *workflow*. Estas ações podem incluir

conversões de dados, declarações e manuseamento de variáveis e controlos de fluxo, tais como *if statements*, *switch statements* e *loops* [25].

Figura 5 - Exemplo de um *workflow* de uma aplicação lógica de exemplo (*template* disponibilizado pela Microsoft): "*Send me an email when a new item is added to a SharePoint Online list*".

2.3.2.1 *Nested Logic Apps*

No fundo, uma *Nested Logic App* é uma aplicação lógica que é inicializada a partir de outra, uma *Logic App* "pai" - ou principal. Esta estratégia de programação é especialmente útil para uma organização modular e de mais fácil compreensão, como, também, permite a reutilização de vários blocos de código; além destas vantagens, permite, ainda, ultrapassar algumas limitações das *Logic Apps* (por exemplo: número máximo de variáveis) [26].

Pondo este conceito numa vertente mais prática para uma melhor compreensão: A *Logic App* X, no decorrer da sua execução, chama, por uma ligação *HTTPS*, a *Logic App* Y, que é iniciada por um trigger *HTTPS*, é executada e retorna o seu resultado à *Logic App* X, que continua a sua execução.

2.3.3 *Azure Functions*

As *Azure Functions* são um serviço oferecido pela plataforma *Azure* que permite o desenvolvimento de código num ambiente independente de configuração de servidores, de ambientes de desenvolvimento ou outras complicações adjacentes à programação convencional, tal como problemas de escalabilidade – sendo que com mais execuções,

mais recursos são alocados a essa aplicação – aproveitando os recursos da computação em *cloud* da *Azure*. Estes blocos de código são denominados de *Functions* [27].

Estas implementações podem ser construídas em várias linguagens, a *Azure* suporta *Node.js, Java, Python, C#* e *Powershell Core*, sendo que o seu funcionamento e lógica não difere por este desenvolvimento estar a ter por base os serviços computacionais da *Azure* – como podemos ver pela estrutura de código apresentada na Figura 6.

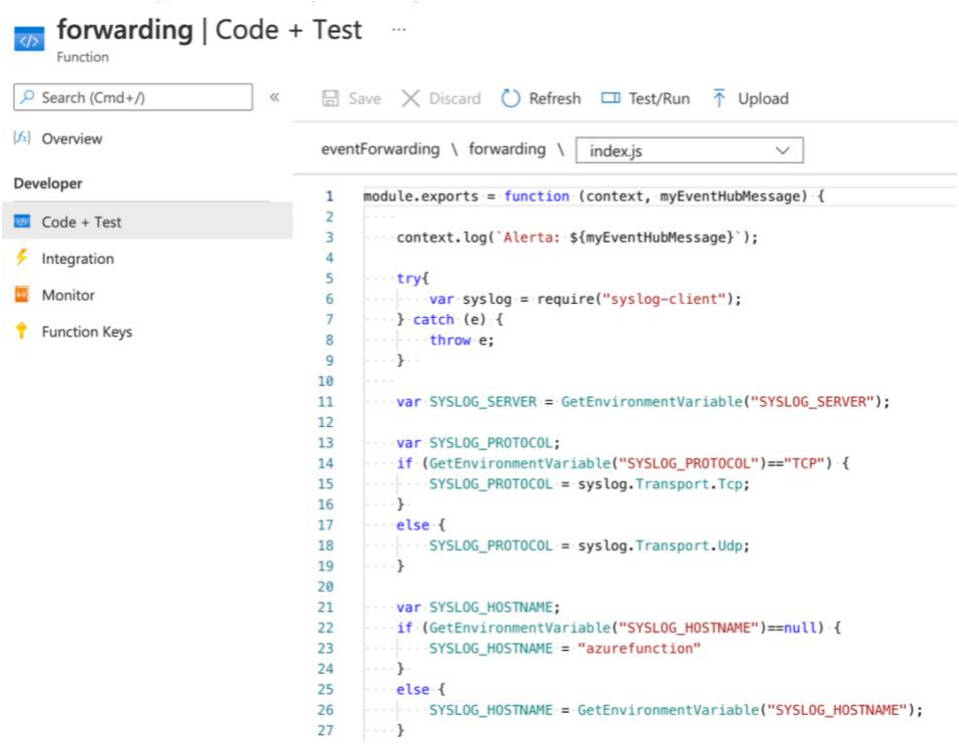

Figura 6 - Exemplo do ambiente de desenvolvimento de uma *Azure Function*

Estas funções, tal como as *Logic Apps*, são inicializadas e executadas em resposta a um acontecimento – um *trigger*. Por ser uma solução tão simples, eficiente, flexível e que funciona à base de *triggers*, esta é uma solução muito popular e procurada para proceder ao desenvolvimento de automatizações, de ferramentas de processamento de dados, métodos de respostas a alterações no estado do sistema em questão e até *API's*. Outra das suas principais *flagships* é a sua capacidade de integração e enlace com uma enorme quantidade de outros serviços *Azure* através dos seus *triggers* e *bindings* próprios já pré-configurados, além das bibliotecas próprias das linguagens em questão que já permitem uma grande variedade no que toca a possibilidades de ligações [27], sendo um exemplo desse esquema de enlaces apresentado na Figura 7.

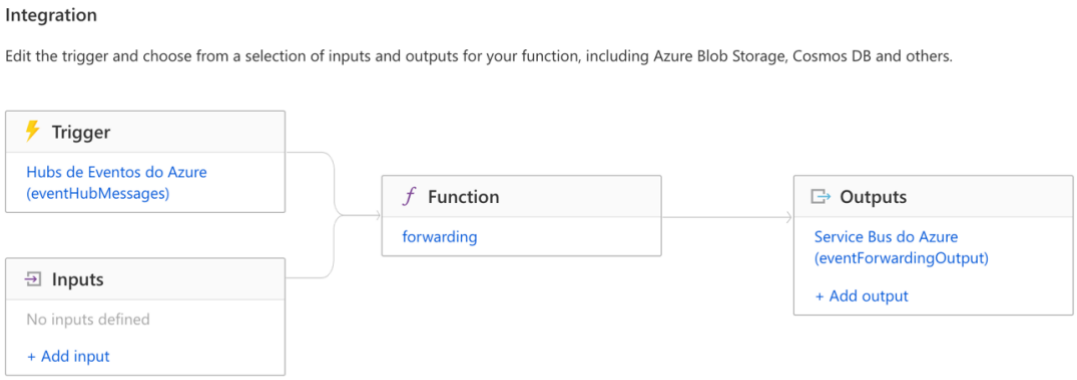

Figura 7 – Ilustração do esquema de integrações disponível nas *Azure Functions*

2.3.4 *Azure Functions* vs *Azure Logic Apps*

Tendo em conta os dois últimos subcapítulos, é normal que surja uma dúvida: em que difere uma *Azure Function* de uma *Azure Logic App* e quando é mais adequado usar cada uma delas?

A *Logic App* é orientada para a automatização de processos, organização de tarefas, processos de negócio e *workflows* quando estes tendem a envolver outras aplicações, informações, sistemas e serviços de outras entidades, sendo que a *Logic App* tem conectores próprios que facilitam muito essa tarefa de integração. A *Azure Function* é algo mais abstrato e menos elaborado, numa linguagem que abranja os termos de programação, é uma solução mais baixo-nível – é um serviço que nos dá um ambiente de desenvolvimento do código que quisermos, desde que este possa ser executado com base num *trigger*.

No que toca ao processo de desenvolvimento, a *Logic App* apresenta-nos uma interface com um *workflow* referente a todos os passos e operações efetuadas no decorrer da execução da aplicação, sendo que pode ser editada diretamente no *workflow* e fazendo desta solução algo incrivelmente intuitivo e acessível até a quem pouca formação tem na área das tecnologias. A *Azure Function*, como dito anteriormente, é puramente um panorama de código, podendo – por um lado – ser mais flexível e eficiente no que nos permite fazer, mas, também, menos intuitivo e fácil de lidar. A nível de integração, uma grande diferença está na facilidade com que a *Logic App* consegue comunicar com outros serviços *Azure* ou *Microsoft* (por exemplo: *Outlook*) e até outros serviços *third-party* já estabelecidos no meio tecnológico (por exemplo: serviços *Google*, *Adobe*, *Amazon*,

Dropbox, etc.), enquanto a *Azure Function* permite outro tipo de ligações, ligações mais "cruas" e menos restritas, como a conexão e comunicação com qualquer tipo de servidor, em qualquer tipo endereço *IP*, utilizando *sockets* com recurso às soluções da linguagem escolhida, por exemplo.

Em suma, são duas ferramentas bastante completas e com as suas valências, ainda que distintas, apesar de ambas poderem dar a entender que são soluções semelhantes. No entanto, ambas têm funções para as quais são mais eficientes e adequadas, sendo por isso que usámos as duas neste projeto.

2.3.5 *Azure Service Bus*

O *Service Bus* da *Azure* é um serviço que permite a intermediação de mensagens e informações com recurso a filas e outras estruturas de dados, sendo que a fila é a estrutura de dados usada e relevante para este projeto. Uma mensagem pode ser um objeto em que se encontre *metadata* e o próprio corpo da mensagem – as informações que se pretendem passar, podendo ser desde *plain text* a mensagens estruturadas como *JSON* ou *XML* [28].

Este serviço e estas mensagens podem ter muitos usos úteis em cenários distintos, como por exemplo:

- O simples encaminhamento e transferência de mensagens entre duas entidades, não tendo a necessidade de configurações complexas à base de protocolos de transporte ou *sockets*.
- A independência entre aplicações e estruturas. Tendo esta intermediação, não existe necessidade de nenhum interveniente estar constantemente disponível para receber, enviar ou aceder às mensagens.
- Permite algum balanceamento de carga na medida em que permite que vários consumidores acedam às suas estruturas de dados em simultâneo, sendo que é feita uma gestão, pela parte do *Service Bus*, em relação à *ownership* e destino de cada mensagem.

Além destes aspetos e vantagens, concilia-se todos os benefícios oferecidos pela computação em *cloud* oferecidos pela *Azure*, como a auto-escalabilidade consoante as

necessidades, e a pouca manutenção necessária para manter o correto funcionamento do serviço [28].

2.3.6 Altice e *Microsoft Azure*

A Altice Portugal tem, ao seu dispor, uma série destes recursos e soluções hospedadas na plataforma de *cloud* da *Microsoft*. Entre estes, está o *Active Directory* (*Azure AD*), essencial para o funcionamento da organização. No mês de Janeiro de 2021 registou-se 1.720.760 inícios de sessão, tendo uma média de 73.182 acessos por dia útil de semana (segunda-feira a sexta-feira), que justifica, exatamente, essa mesma importância extrema para o correto desenrolar das atividades. Ao *Azure AD* junta-se a plataforma *365*, que não sendo "oficialmente" parte da plataforma *Azure*, faz parte dos serviços *cloud* da *Microsoft*, integrando as ferramentas do *Microsoft Office*, incluindo *Microsoft SharePoint, Microsoft Teams* e *Microsoft Outlook*, essenciais no clima de pandemia e, consequente, nos regimes teletrabalho em que vivemos. Tal como o *Azure AD*, esta plataforma apresenta uma carga de utilização igualmente muito grande; podendo dar como exemplo uma média diária de 603.809 e-mails processados (recebidos e enviados; dados da semana útil de 25 a 29 de Janeiro de 2021) revelando, também, a sua extrema importância para a Altice Portugal. Esta grandeza número pode ser ilustrada pelos valores do *dashboard* apresentado na Figura 8.

Serão estas as soluções – e, consequentemente, os alertas provenientes das mesmas – que serão integrados no *IBM QRadar*; dando, também, ímpeto e a abertura da possibilidade de uma futura exploração e aproveitamento mais exaustivo dos recursos que o *Microsoft Azure* tem a oferecer, passando a assegurar, em todos os cenários, a melhor segurança possível.

Foram deixados de fora deste descritivo todos os produtos *Azure* que foram criados e desenvolvidos no âmbito deste projeto, visto que já foram apresentados anteriormente e não faziam parte da infraestrutura da Altice num momento prévio a este projeto.

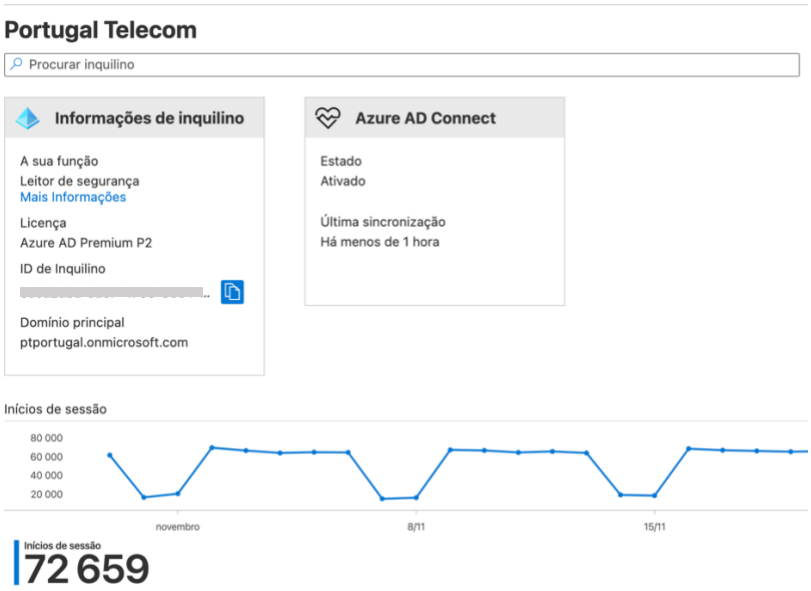

Figura 8 - Exemplo de *dashboard* do *Microsoft Azur*e da Altice Portugal: Descrição Geral do *Azure Active Directory*

2.4 *Syslog*

O termo *Syslog* significa *System Logging Protocol* e é um protocolo muito utilizado mundialmente para enviar registos – *logs* – de sistema ou mensagens relativas a outros eventos para um servidor específico, um servidor *Syslog*. Em muitos dos casos é utilizado para armazenar e disponibilizar centralmente os vários *logs* recebidos de várias máquinas diferentes. Tipicamente, o protocolo usa um transporte por *UDP* no porto 514, apesar de poder ser configurado em qualquer outro porto ou qualquer outro protocolo de transporte [29].

A mensagem *Syslog*, atualmente, usa o standard RFC5424, que tem o seu esquema ilustrado na Figura 9. O protocolo, na sua mensagem, tem, principalmente, um cabeçalho e só depois a mensagem em si. Normalmente, este cabeçalho apresenta – de relevante para o projeto – um *timestamp*, *hostname*, um valor de severidade e outro de *facility* – um valor que define onde foi produzido o *log* em questão (se no *kernel*, se por um serviço de *e-mail*, etc.). A mensagem em si não tem qualquer tipo de complexidade, sendo que é apenas texto corrido no campo a seguir ao cabeçalho [30].

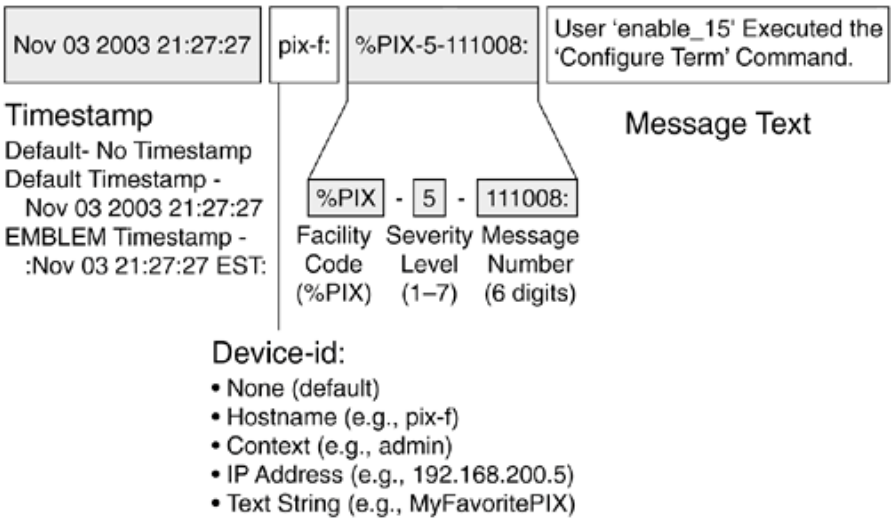

Figura 9 - Exemplo de uma mensagem *Syslog*

Fonte: https://stackify.com/syslog-101/

Capítulo 3

Implementação da solução

3.1 Estudo e familiarização das ferramentas

Para a realização deste trabalho foi necessária uma familiarização mais detalhada e profunda das ferramentas a utilizar, o *IBM QRadar* e as plataformas do *Microsoft Azure*. Quanto ao *QRadar* foi feito um estudo mais teórico, focado mais na pesquisa das funcionalidades, funcionamento e resultados produzidos. Quanto às plataformas *Microsoft*, esta aprendizagem foi mais prática e intuitiva. Com isto, foi possível explorar as plataformas, tanto como o seu comportamento, as suas funcionalidades, informações e, por fim e mais importante para este trabalho, os seus *logs* e alertas.

3.2 Estudo e delineação das estratégias a explorar

De forma a iniciarmos uma componente mais prática do projeto, em primeiro lugar, foi necessário estudar, explorar e definir as possíveis estratégias de execução que tínhamos ao nosso dispor.

De início, ao falar com a *Microsoft* - mais propriamente com o Eng. José Cabreira – a sugestão foi a utilização do *Azure Sentinel*, solução de segurança da *Microsoft* que funciona como *SIEM*, para a intermediação dos registos para o *QRadar*. Esta seria a solução mais eficiente a nível de organização, sendo menos *error-prone*. Em contraste, seria, também, a que acarretaria mais custos, visto que necessita tanto da subscrição do *Azure Sentinel* como da dos restantes produtos da *Azure*. Esta opção funcionaria com a configuração do *Azure Sentinel* de modo a este agregar os dados de todas as fontes (*Azure, Office 365*, entre outros) e tratar da alarmística e da centralização da informação, sendo que o transporte até à rede Altice é assegurada por outros produtos *Azure*.

No entanto, pela nossa mão, foram encontradas outras possíveis soluções, se bem que semelhantes. Outra alternativa passa por não usar o *Sentinel*, mas sim o *DSM* do

Azure Security Center e o *DSM* do *Office 365*. Isto assemelha-se à solução do *Sentinel* na componente *Azure*. Ou seja, os registos e alertas do *Azure* são tratados pelo *Azure Security Center* e pela sua alarmística, enquanto os dados referentes ao *Office 365* teriam de ter a sua alarmística implementada por parte do *QRadar*. Assim, para uma melhor abstração, teríamos dois *Event Hubs* – um para os alertas Azure já tratados e um para os registos *Office 365* ainda por tratar [31, 32]. No entanto, apesar de parecido e de ter especificações interessantes, o *Azure Security Center* não é um software de *SIEM*, sendo que pecaria no detalhe e mesmo, em parte, no objetivo final deste projeto. A meu ver, o *Azure Security Center* não poderia ser parte da solução principal, mas sim um complemento muito interessante para o leque de soluções de segurança e um ativo importante no seu desenvolvimento.

Outra opção seria trabalharmos diretamente com os dados não tratados nas duas vertentes: *Azure* e *365*. Neste caso, seria usado o *DSM Microsoft Azure Platform*, que está encarregue de passar os *activity logs* da plataforma, e o *DSM Office 365*. Sendo assim, o *IBM QRadar* receberia os dados sem estes terem sido tratados a nível de alarmística e classificação como incidente de segurança, sendo que essa parte teria de ser configurada e desenvolvida do lado da plataforma da *IBM* [32, 33]. A nível de completude e abrangência a nível de segurança, esta solução é a menos favorável e a que procuraremos evitar. Isto porque, a maior complexidade que requer a nível de implementação, leva a uma maior probabilidade de erro ou falha de configuração. Noutro ponto desfavorável a esta opção, foi desaconselhado e desautorizado o uso desta estratégia, isto devido a consistir numa ligação direta entre uma parte muito sensível da rede interna Altice e a *internet*.

Visto isto, no meu entendimento, concluí que a solução mais adequada passa pelo uso do *Azure Sentinel*. Seguidamente, no processo, o encaminhamento do respetivo alerta até ao *Event Hub* é efetuado por uma *Logic App.* Proximamente, estes eventos têm de ser encaminhados para a rede da Altice. Para isto, a solução escolhida é o recurso a uma *Azure Function*. Esta vai estar encarregue de aceder ao novo alerta que chegou ao nosso *Event Hub* e enviá-lo para um servidor *Syslog* da Altice Portugal, dando por terminado o processo de transporte *Azure* – Altice. De seguida, o processo geral de transporte *Azure Sentinel – IBM QRadar* fica facilmente concluído, sendo que os alertas já estando no servidor *Syslog* na *intranet* da empresa, a ligação torna-se bastante simples – o servidor *Syslog* é configurado como uma nova *log source* do *IBM QRadar*.

3.3 *Logs*

Com o decorrer da familiarização referida anteriormente, deparei-me com vários tipos de *logs* diferentes: alertas do *365* e registos do *Azure Active Directory*, a principal e mais relevante solução que a Altice Portugal tem hospedado no *Microsoft Azure*.

Quanto aos alertas do *365*, estes são isso mesmo, alertas, como o apresentado na Figura 10. Ou seja, registos já tratados e passados por regras de alarmística já definidas de modo a produzir esses mesmos alertas. Além deste aspeto, os alertas não tinham a quantidade de informação e o detalhe que desejávamos. Com isto, foi pedido à *Microsoft* alguma forma de aceder a estes registos, sendo que tivemos acesso a uma plataforma gráfica, apresentada na Figura 11, de boa usabilidade com os alertas e com os respetivos eventos, tal como opção exportação desses mesmos registos num ficheiro *.csv*. Quanto aos registos de auditoria do *Azure Active Directory*, estes são bastante mais complexos e "crus", tanto na plataforma online como no ficheiro exportado como *.csv*.

Figura 10 - Detalhes de um evento reportado pelo *Office 365* ("*FileDeleted*")

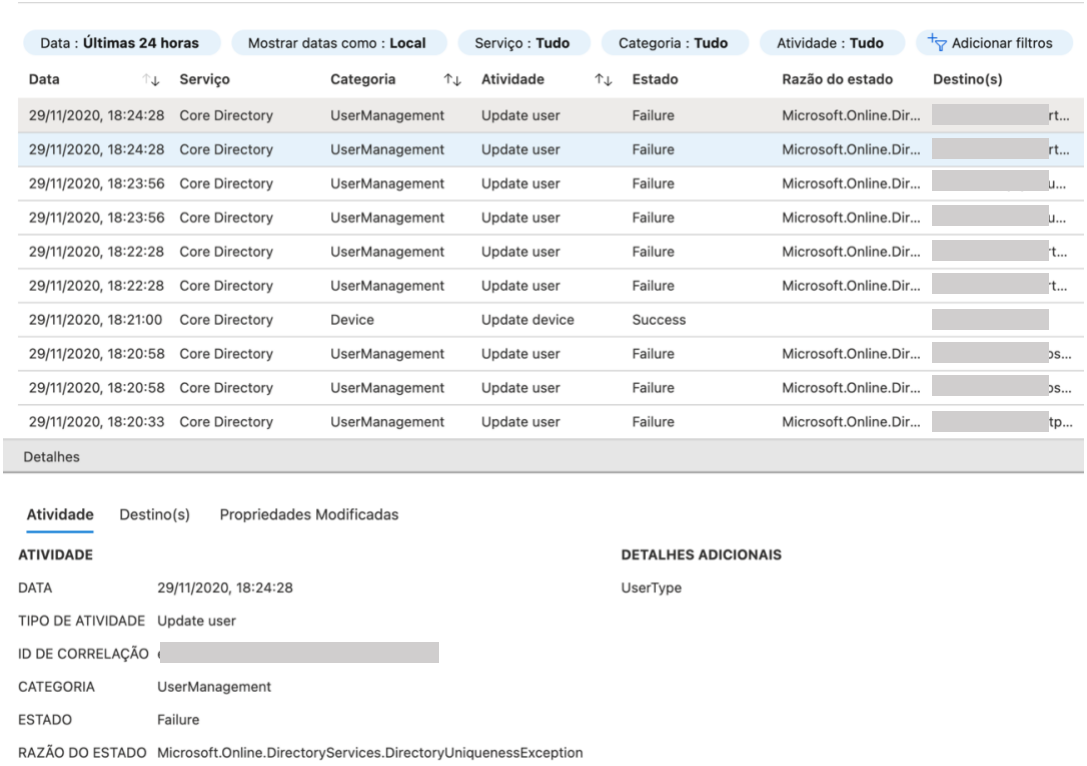

Figura 11 - Interface gráfica do *Azure* para visualizar registos de auditoria

Para poder explorar mais detalhes, foi feita a descarga de um ficheiro *.csv* desses mesmos registos para serem analisados de forma mais livre. E, para essa análise, foi desenvolvida uma pequena e simples aplicação *JAVA* (com as funcionalidades e *output* representados nas Figuras 12 e 13, respetivamente) que sanitiza, retira as informações mais importantes e separa esses mesmos registos consoante alguns dos seus campos (por exemplo: todos os registos que tenham como categoria "*UserManagement*").

```
//////////////////////////////////////////
/// Logs Azure - Choose your command: ///
//////////////////////////////////////////

* Print log by Correlation ID - press 1
* Print logs by Service - press 2
* Print logs by Actor Type - press 3
* Print logs by Activity - press 4
* Print logs by Category - press 5
* Print logs by Actor Display Name - press 6
* Print all logs - press 7
* Print statistics - press 8
* Quit - press 9
```

Figura 12 - Opções da aplicação *JAVA* para analisar os registos de auditoria do *Azure*

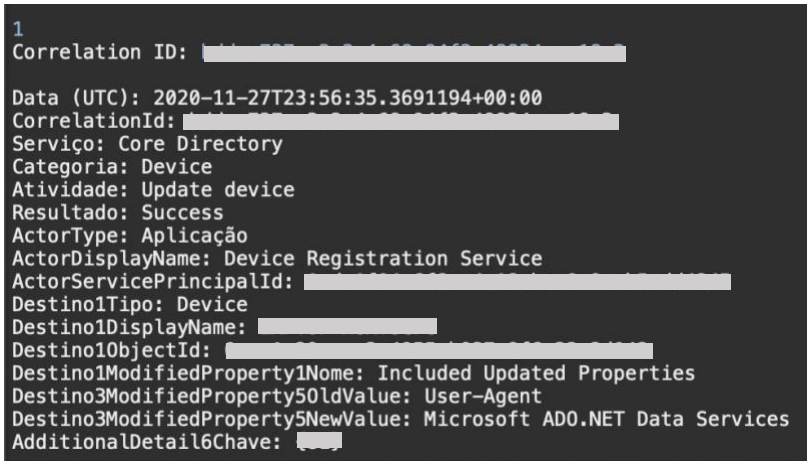

Figura 13 - Exemplo de execução da aplicação e exemplo de um registo

No entanto, com o uso do *Azure Sentinel* e com a sua ingestão automática dos eventos, apesar de não deixar de ter feito parte do desenrolar do projeto, esta aplicação deixa de ter utilidade. Esta aplicação foi desenvolvida apenas com o intuito de providenciar uma melhor orientação e pré-análise dos registos, principalmente um estudo ligado à sua estrutura (que campos dispõe, que valores podem ter, entre outros), enquanto não nos era disponibilizado o acesso à plataforma de *SIEM*.

3.4 Configuração da *Azure*

Neste subcapítulo será abordada a forma como foi feita a configuração dos produtos da plataforma *cloud Azure;* seja no que toca aos alertas gerados, como aos produtos usados na criação da pipeline *Azure* - Altice.

3.4.1 *Azure Sentinel*

Esta configuração tem como objetivo pôr o *SIEM* da *Microsoft* a funcionar com todos os dados e informações disponíveis que a Altice Portugal tem hospedadas na plataforma *cloud Azure*. Isto garantindo uma análise abrangente e minuciosa de todos os eventos que ocorrem, alertando o *SOC* quando necessário.

Mais concretamente, os objetivos principais desta configuração são dois. Começa por uma aproximação às políticas de segurança *on-premise* em produtos *Microsoft* e em

produtos “parentes”. Ou seja, para clarificar este conceito de produtos “parentes”, um exemplo seriam os dois *Active Directory*, o *on-premise* e o *Azure*. Há, assim, a necessidade e a intenção de mapear as políticas de segurança já existentes, de modo a atingir congruência entre as duas soluções, a *on-premise* e a *cloud*. Dando um exemplo mais concreto: se a nível local classificamos um ataque *brute force* qualquer conjunto de dez, ou mais, tentativas de login falhadas num período de três, ou menos, minutos, a nível do *Azure Active Directory* também teremos de ter esses mesmos parâmetros. O segundo objetivo é expandir estas políticas, pesquisar e identificar o que é mais relevante monitorizar e alertar, que comportamentos e eventos, além do que já temos definido nas políticas *on-prem*, seriam importantes passar para o *SIEM* principal da Altice, o *IBM QRadar*.

3.4.1.1 Conectores

Os conectores são, no fundo, o que disponibiliza ao *Azure Sentinel* os dados e informações necessárias para este ter um funcionamento adequado e completo. São encarregues de fazer a ligação entre as fontes de dados e o *SIEM*, alimentando-o.

Foram ligados cinco conectores, tantos quanto possível, tendo em conta os recursos *Microsoft* que a Altice Portugal utiliza. Estes são designados de “*Azure Active Directory”, “Azure Active Directory Identity Protection”, “Azure Activity”, “Office 365”* e *“Office 365 Advanced Threat Protection”.*

O “*Azure Active Directory*” é o conector responsável pela ligação, tal como o nome indica, dos eventos provenientes do *Active Directory* da *Azure*. Este disponibiliza a informação de “*Sign-in logs*”, referentes a inícios de sessão dos utilizadores, e “*Audit logs*”, referentes às restantes operações do *Active Directory*, como a atividade de utilizadores, dos seus grupos, *roles*, aplicações associadas, entre outros. O conector “*Azure Active Directory Identity Protection*” é a ligação ao produto do mesmo nome. O *Identity Protection* é um produto de segurança do próprio *Active Directory*, capaz de nos fornecer informações como utilizadores de risco, eventos de risco, possíveis vulnerabilidades, entre outras. Ainda referente à plataforma *Azure* temos o conector “*Azure Activity*”. Este conector é responsável pela transmissão de dados sobre eventos mais gerais e não focados completamente em utilizadores ou grupos, como o conector

"*Azure Active Directory*". Este tipo de informações inclui, como exemplo, a modificação de um recurso ou a iniciação de uma nova máquina virtual.

Entrando, agora, na plataforma *365*, temos os restantes conectores. O de nome "*Office 365*" é, como o nome indica, responsável por passar os dados referentes das atividades dos utilizadores no *Office 365* para o *Azure Sentinel*. Nisto incluindo operações como *download* de ficheiros, pedidos de acesso, alterações às caixas de *e-mail* e os detalhes dos utilizadores que praticarem essas ações. O "*Office 365 Advanced Threat Protection*", tal como o "*Azure Active Directory Identity Protection*", é uma ferramenta de segurança. Esta oferece alguma proteção contra ameaças provenientes de *e-mails* ou ligações presentes nos mesmo . Ambos, sendo conectores que têm presente uma vertente de segurança, são capazes de, independentemente, levantar incidentes que sucederam nos dados que representam – neste caso, cabe-nos apenas associar esse incidente a uma regra de alarmística de modo a que o sistema origine, paralelamente ao incidente, um alerta. Cada conector tem um conjunto de incidentes que é capaz de sinalizar.

3.4.1.2 Analytics Rules

Como já dito anteriormente, as *Analytics Rules* – regras de alarmística – é o que nos vai permitir a deteção e criação dos nossos alertas. Também como já discutido no capítulo passado, destaca-se sua capacidade de flexibilidade e facilidade de edição. Assim, permite que a criação de regras que nos seja facilitada e que estas sejam proveitosas para os nossos objetivos de uma forma eficaz e eficiente.

Esta flexibilidade e facilidade de manuseamento desta ferramenta do *Azure Sentinel*, permitiu alcançar um dos principais objetivos deste projeto: o mapeamento e consequente congruência entre tudo o que é monitorizado e reportado na rede interna da Altice Portugal e o que, semelhantemente, ocorre na *cloud Azure* – grande parte destes eventos são referentes aos *Active Directories*, ora o hospedado *on-premise* e o *Azure*. Neste lote estão presentes os alertas referentes a incidentes de ataques de *brute-force* relativos a inícios de sessão ou a operações como "*Account added to privileged group*" ou "*Login from an unfamiliar location"*.

No entanto, consideramos que este projeto tem muito mais potencial para o deixarmos apenas com este objetivo de congruência *cloud – on-prem*. Tendo em conta esse pensamento, procedeu-se a um momento de pesquisa sobre as formas de como

poderíamos enriquecer esta implementação e a experiência de segurança do *SOC*. Neste processo, optámos por adicionar algumas regras de alarmística feitas pelas equipas *Microsoft* do *Azure Sentinel*. A escolha destas regras baseou-se em vários critérios, na sua popularidade entre a comunidade *Sentinel* aquando da pesquisa feita, a variedade de diferentes níveis de severidade das regras escolhidas, e, obviamente, a conferência com a equipa de *SOC* e, mais especificamente, com o Eng. Alberto Bruno, para se detalhar quais as mais interessantes e as que podiam ser mais relevantes para a organização. Alguns exemplos de alertas originados por esta abordagem são relativos a tentativas de início de sessão a contas desativadas, várias alterações de *password* por parte de um utilizador num dado intervalo de tempo e a operação de conceder permissões a outro utilizador. Além desta abordagem, achámos pertinente continuar a nossa pesquisa e alargar horizontes além do oferecido pelo *Azure Sentinel* e pela sua equipa. Com isto, deparámo-nos com o livro *Top 10 Security Events to Monitor in Azure AD and Office 365* da empresa de *software Quest* [34]. Deste adicionámos os alertas referentes a criação de recursos (*deploy* de aplicações, por exemplo), controlo de acesso de utilizadores externos (se um utilizador externo é adicionado ao *Teams* e prontamente removido, o que pode ser suspeito) e a operação de apagar várias equipas no *Teams*.

Como foi explicado no capítulo introdutório, apenas são dados poucos exemplos do conjunto de alertas que passámos a ter a capacidade de captar.

3.4.1.3 Alertas emitidos pelo *Azure Sentinel*

Como dito anteriormente, os alertas são originados de várias maneiras diferentes. Dividindo-os por essas fontes e abordagens diferentes, os alertas que iremos retratar neste documento são os seguintes:

Azure Active Directory Identity Protection:

- *Anonymous IP*
- *Malware linked IP address*
- *Password spray*

Office 365 Advanced Threat Protection:

- *A potentially malicious URL click was detected*
- *Email reported by user as malware or phish*
- *Email messages containing malicious URL removed after delivery*

Adicionados manualmente:

- *Azure Portal brute force sign-in attack*
- *User added to Azure Active Directory Privileged Groups*
- *Sign-ins from IPs that attempt sign-ins to disabled accounts*
- *Multiple password reset by user*
- *Suspicious granting of permissions to an account*
- *Suspicious resource deployment*
- *External user added and removed in short timeframe*
- *Multiple Teams deleted by a single user*

3.4.2 Ligação *Azure Sentinel – Azure Event Hub*

3.4.2.1 Logic App – Get-SentinelAlerts2

A ligação entre o *Azure Sentinel* e o *Event Hub* usado é feita através da aplicação lógica *Get-SentinelAlerts2*. Esta é baseada na aplicação *Get-SentinelAlertsEvidence* de Yaniv Shasha, presente no repositório *GitHub* da *Azure*, sob a pasta "*Sentinel*" e a subpasta "*Playbooks*" [35].

A aplicação é iniciada quando o *Azure Sentinel* levanta um incidente e emite o respetivo alerta – é esse o seu *trigger*. Nesse alerta é-nos dada uma série de informações, tais como o nome do alerta, severidade, descrição, hora e data e um campo de "*Extended Properties*", que pode ter, ou não, mais alguns detalhes, dependente da configuração da sua regra de alarmística – como demonstrado na Figura 14. Esta configuração é referente ao *query* em si, onde pode estar especificado, em linguagem *Kusto*, campos que se referem às entidades participantes no evento ou os seus endereços, como os *usernames* ou os seus endereços *IP*.

Figura 14 - Exemplo de informações dadas pelo alerta levantado pelo *Azure Sentinel*.

No entanto, a decisão foi de ignorar essas mesmas informações, que estariam no campo "*Extended Properties*". Isto de forma a estarmos o mais independentes possível de uma correta configuração do *query* da regra de alarmística. Assim, apesar de reconhecermos e termos em conta a importância desta regra ter uma configuração completa, achámos a decisão mais segura ter essa organização arquitetural. Visto isto, de seguida, é feito um *query* de modo a capturar o evento, ou os eventos, que originaram o alerta inicial do *Azure Sentinel*, tendo um exemplo deste na Figura 15. O objetivo deste método é a captura dos eventos "em bruto", de maneira a conseguirmos ter acesso a todos os detalhes possíveis, e não tratados, sobre o sucedido. Após este passo, corre-se um *script* de *Javascript* para transformar o resultado do *query* anterior numa variável usável na nossa *Logic App*. A esta variável vamos chamar de *event string*, sendo que a Figura 16 é apresentado um exemplo de uma destas estruturas.

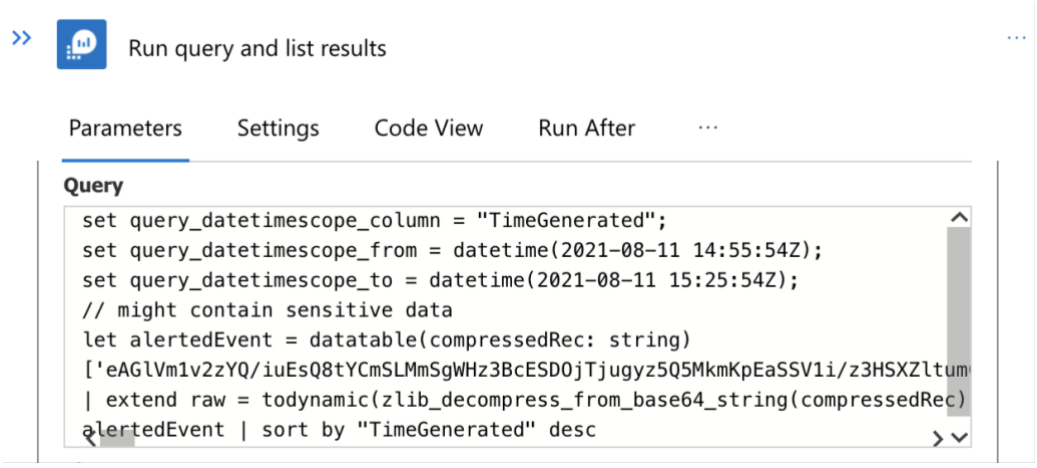

Figura 15 - Exemplo de um *query* para recolha de informação em bruto de um evento

{"AccountCustomEntity":"[redacted]@telecom.pt","AppDisplayName":"[redacted]","EndTimeUtc":"2021-08-11T15:38:46.334Z","FailureCount":25,"set_Device":"[\"Windows 10;Chrome 92.0.4515\"]","set_IPAddress":"[\"[redacted]\"]","set_Location":"[\"Lisboa, Lisboa, PT\"]","set_ResultType":"[\"[redacted]\"]","StartTimeUtc":"2021-08-11T15:38:44.948Z","SuccessCount":0,"TimeGenerated":"2021-08-11T15:00:00Z","timestamp":"2021-08-11T15:38:44.948Z","UserDisplayName":"[redacted]","UserPrincipalName":"[redacted]@telecom.pt"}

Figura 16 - Exemplo de uma *event string* (de outro tipo de evento devido às suas menores dimensões, de modo a facilitar a leitura)

Posteriormente, esta *event string* é encaminhada para outra *Logic App* de nome *Parser*, encarregue de tratar a *event string* e retornar uma *string* que contém os campos do evento de forma mais legível.

Por fim, esta *string* já tratada, como pode ser vista Figura 17, retorna à *Get-SentinelAlerts2* e é encaminhada para o *Event Hub*, sendo que esta antes é enviada por *e-mail* para uma conta da Altice, que vai armazenar a totalidade das execuções da aplicação, fugindo ao período de retenção do histórico de execução de 90 dias das *Logic Apps*. Após este passo temos um mecanismo de deteção de falhas, de modo a estarmos informados caso algo na *pipeline Azure Sentinel – Event Hub* não tenha corrido bem e o alerta não ter sido encaminhado. Para isto, temos mais um passo a seguir ao encaminhamento final para o *Event Hub*, é um envio de um *e-mail* caso não se concretize o envio da *string* já tratada pelo processo de *parsing* para o *Event Hub*, para uma conta de *e-mail* à escolha, de modo a passar essa informação e termos a noção que o processo não decorreu como esperado.

Figura 17 - Exemplo da *event string* após o *parsing*.

3.4.2.2 Parsing

O processo de *parsing* é feito, também, logo na plataforma *Azure*, usando o conceito das *Nested Apps*, já explicitado anteriormente. Esta solução divide-se em três partes, ou camadas, cada uma composta por uma aplicação lógica.

A primeira destas aplicações, a primeira camada, – a aplicação lógica *Parser* – é chamada pela aplicação principal – a *Get-SentinelAlerts2* – e, usando a *event string* aliada a *scripts* de *Javascript*, determina quem é a entidade responsável pelo alerta, o qual podemos chamar de *alert provider*. Um exemplo deste *script* pode ser analisado na Figura 19.

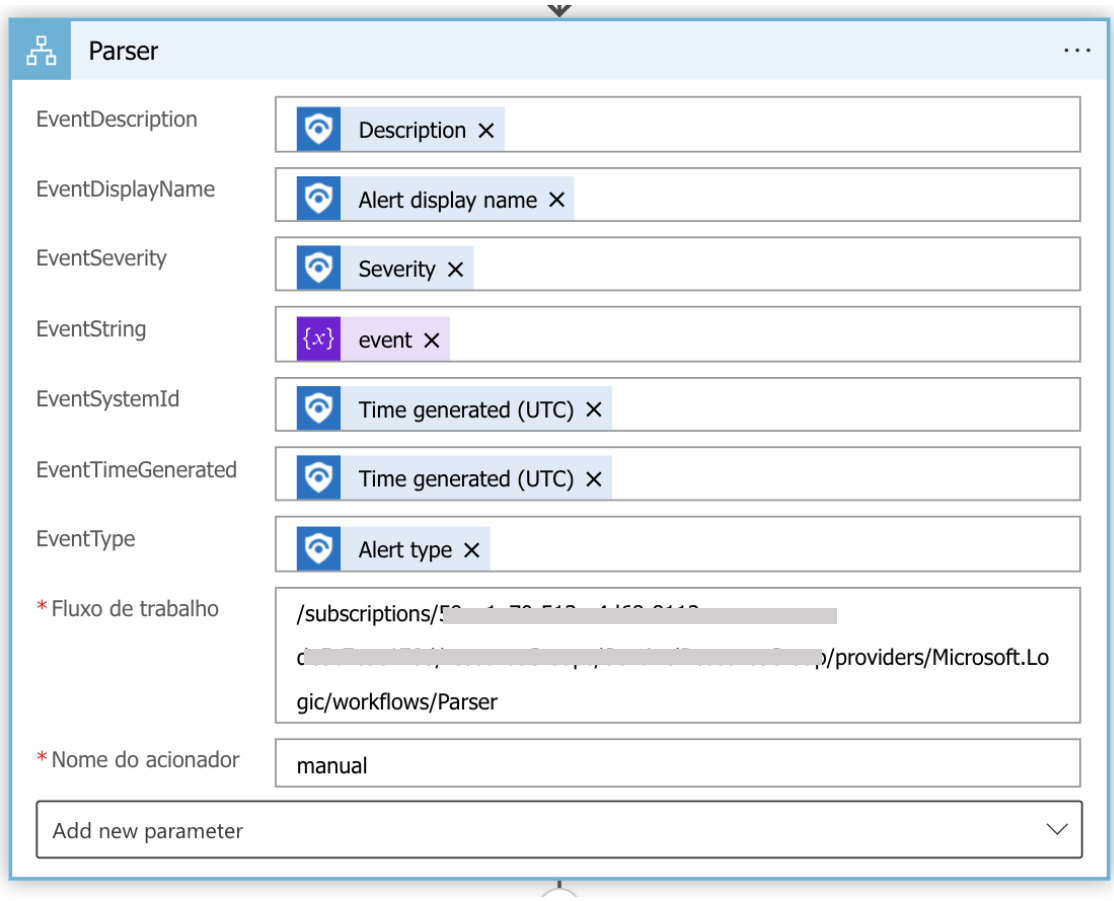

Figura 18 - Chamada da aplicação *Parser* por parte da *Get-SentinelAlerts2*

Ou seja, se foi acionado pelo *Azure Active Directory Identity Protection*, pelo *Office 365 Advanced Threat Protection*, ou pelas *Analytics Rules* que criámos diretamente no *Azure Sentinel*. Além deste processo são, também, guardadas algumas das informações que são logo providenciadas pelo alerta original do *Azure Sentinel*, como falado no início do ponto 3.4.2.1 e que nos são relevantes – por exemplo, o nível de severidade. (Estas informações são enviadas no pedido *HTTPS* que a *Get-SentinelAlerts2* faz à aplicação *Parser*, como ilustrado na Figura 18, sendo que esta as guarda em variáveis).

Figura 19 - Processo de extração do campo do *alert provider*

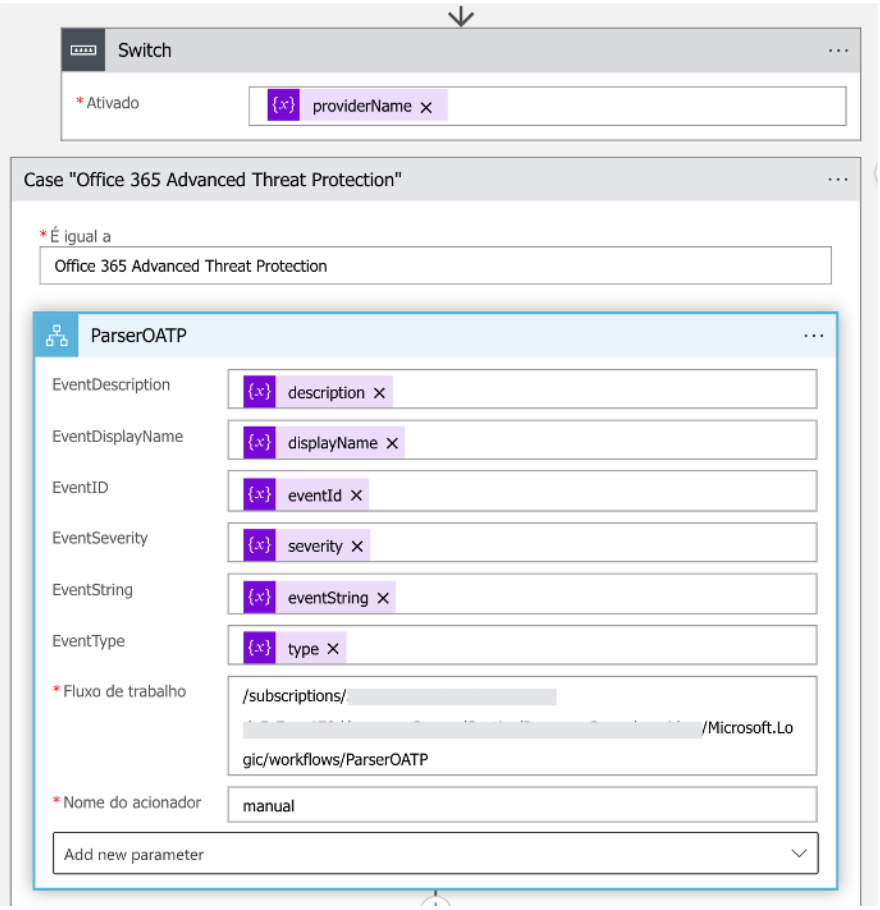

Figura 20 - *Switch-case* referente ao *alert provider* e pedido da aplicação *Parser* à aplicação *ParserOATP* (1ª camada de *parsing* para a 2ª)

Seguidamente, é chamada a segunda camada, a segunda *Nested App*, que é focada apenas nos alertas de cada um dos *alert providers* listados acima. Por exemplo, pode ser chamada a aplicação *ParserOATP*, construída para os eventos do *Office 365 Advanced Threat Protection.* Esta aplicação, usando, também, os dados enviados no pedido *HTTPS* proveniente da camada superior (descrição do alerta, nome do alerta, *event string* e severidade, como apresentado na Figura 20) e *scripts* de *javascript* (neste caso para normalizar o formato da data e hora do alerta, como para ajustar alguma ocasião em que o conteúdo, e não o formato, possa estar desajustado – no caso de um alerta de *brute force sign-in*, por exemplo, temos de decidir se utilizamos a hora da primeira tentativa, da última ou, até, de quando o alerta foi emitido) vai emitir um pedido à terceira e última camada, consoante o nome do alerta. Um esquema semelhante aos anteriores, podendo, igualmente, ser analisado na Figura 21. Ou seja, vai chamar uma *Nested App* de *parsing* focada, apenas, num único tipo de alerta. A título de exemplo, caso seja um alerta do tipo *Email messages containing phish URLs removed after delivery*, a aplicação *ParserOATP* vai chamar a aplicação de terceira camada no nosso processo de *parsing*, de seu nome *ParserOATP_EmailWithPhishUrlRemoved*.

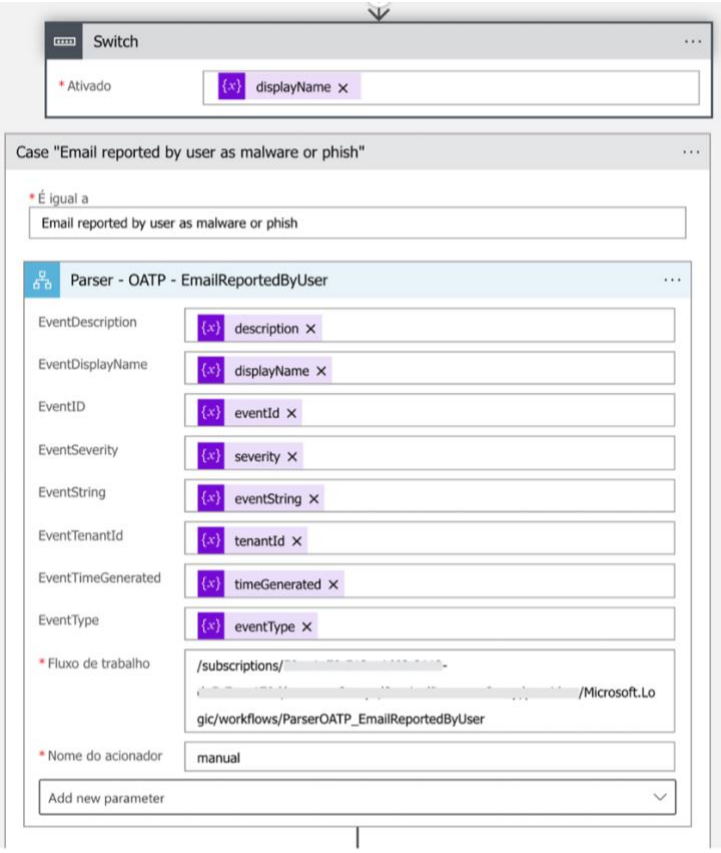

Figura 21 - *Switch-case* referente ao nome do alerta e pedido da aplicação *ParserOATP* à aplicação *ParserOATP_EmailReportedByUser* (2ª camada de *parsing* para a 3ª)

Esta terceira aplicação é, finalmente, a verdadeira responsável pelo *parsing* do alerta. Vai juntar todos os dados já tratados anteriormente, como os dados que são disponibilizados pelo *Sentinel* quando o alerta é acionado e a data e hora do alerta com o formato e conteúdo adequado, aos dados relevantes processados a partir da *event string* (a aplicação recebe, na mesma lógica das camadas superiores, todos essas informações através do pedido *HTTPS*). O processo de extração de informações relevantes tendo por base a *event string* é levada a cabo, como usada em outros processos em etapas anteriores, por *scripts* de *Javascript*. Estes *scripts*, ilustrados alguns exemplos na Figura 22, vão procurar certas palavras, ou expressões, chave que remetam a campos e informações relevantes sobre o alerta, como *usernames* ou endereços *IP*; isto recorrendo à função *contains* (que retorna um valor booleano, caso a palavra ou caracter indicado esteja presente na *string* dada). De seguida, usando funções *split* (divide uma *string* na palavra ou caracter que for indicada como separador), vai capturar os valores correspondentes a esses campos, que se encontram em frente das expressões já identificadas no passo anterior. Posteriormente, todos esses valores são associados a variáveis locais e é composta uma *string* com todos os campos considerados relevantes para o alerta, separados por uma quebra de linha; além desta organização é apresentada, também, a *event string*, estando esta estrutura apresentada na Figura 23. Isto, também, para o caso de haver intenção ou necessidade de se fazer uma análise mais detalhada com recurso a dados menos tratados.

Esta nova *string* é o que vai ser encaminhado para o nosso *Event Hub*, sendo que vai ser passada para as camadas aplicacionais superiores, das quais já falámos anteriormente, até à nossa aplicação *Get-SentinelAlerts2*, sendo esta a responsável pelo referido encaminhamento.

Resumindo, isto significa que teremos uma aplicação lógica para cada tipo de alerta, acrescendo uma para cada tipo de *alert provider* e outra como a fachada de todo o processo. Isto pode parecer uma opção um pouco confusa ou até rudimentar, mas tem as suas razões. Em primeiro lugar, por termos alertas de vários *providers* (e não só), temos uma *event string* quase única para cada alerta, sendo que cada um tem os seus campos e tipos de informação diferentes. Isto, além de permitir destacar todos os campos e informações relevantes relativas ao alerta, permite uma normalização dos campos que são parecidos entre os diferentes alertas – ou mesmo iguais, mas com designações diferentes (por exemplo: campos com a designação "*Client IP Address*", "*Client IP*", "*User IP*" ... entre outros.). Facilita, também, o processo posterior de *parsing* no *IBM QRadar*, já que

não é necessária toda essa adaptação a tantos campos e informações diferentes, mas ao mesmo tempo semelhantes. Aliado a este ponto, há todo a aspeto da abstração, facilmente chegamos ao processo de *parsing* de cada alerta, sendo muito fácil detetar algum erro ou proceder a alguma alteração. Em segundo lugar, é útil para qualquer processo de auditoria aos alertas, devido à sua leitura muito mais facilitada. Este cenário pode ser útil na possibilidade de uma falha no *IBM QRadar*, ou num processo de *quality assurance* entre o que sai do *Azure Sentinel* e o que acaba por ser apresentado no *SIEM* da *IBM*.

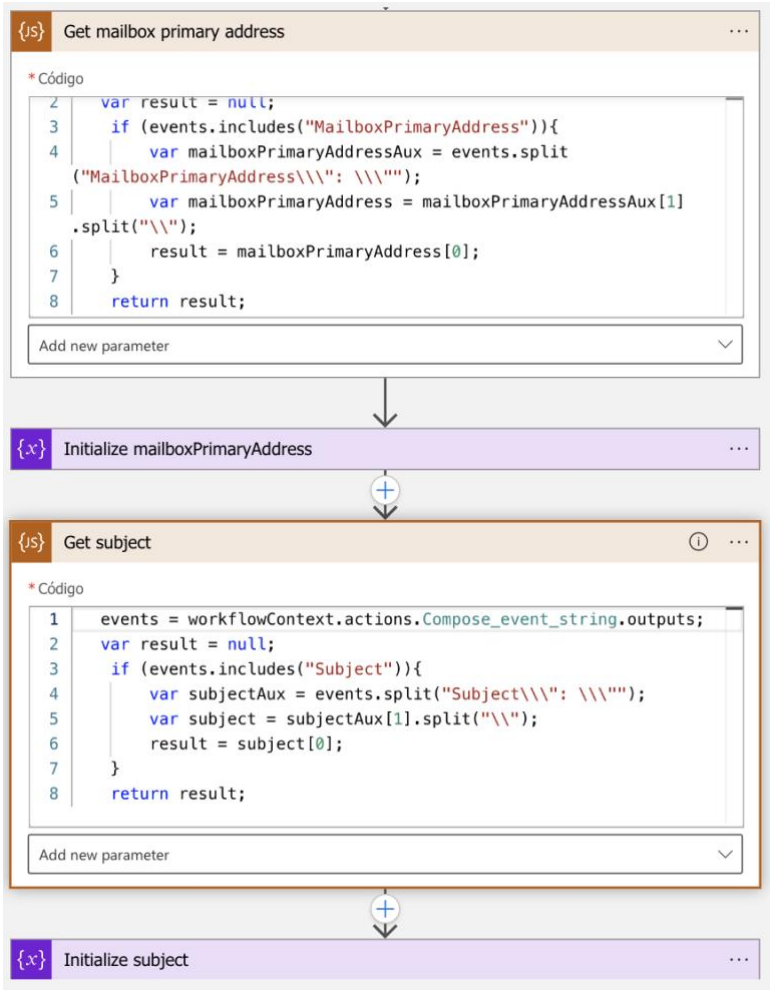

Figura 22 - Exemplo de processos de extração de campos específicos do alerta

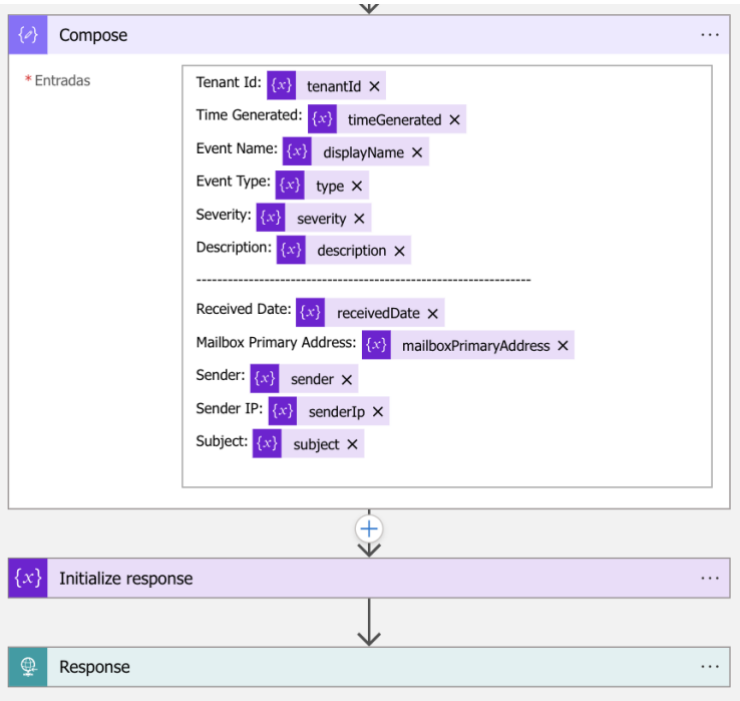

Figura 23 - Montagem da mensagem que vai ser enviada de volta à *Get-SentinelAlerts2*

3.4.3 Ligação *Azure Event Hub – IBM QRadar* Altice

A ligação entre o *Event Hub* e o *QRadar* da Altice é feita com recurso a uma *Azure Function*. Esta aplicação tem o nome de *eventForwarding* e tem por base *Node.js 14.* Esta aplicação foi baseada no projeto *AzureMonitor2Syslog* de Miguel Ângelo Pereira, presente num repositório público de *GitHub* do autor [36]. A esta foram feitas algumas alterações, visto até que a ideia e a lógica era o que desejávamos, mas não culminou numa implementação funcional para o nosso caso.

Para auxiliar e, consequentemente intermediar esta ligação, foi utilizado um servidor *Syslog* da Altice Portugal, que serviu como ponto de contacto entre a *internet* e a própria *intranet*. Isto de modo a acomodar e ir ao encontro das políticas internas de segurança da instituição.

Antes da execução, no painel de configurações da aplicação, foram criadas várias variáveis estáticas que acomodavam os valores referentes ao servidor *Syslog*: o seu *hostname* (“*AzureSentinel”),* o seu endereço *IP*, o seu porto, a sua *facility* (informação desprezável para o efeito) e o protocolo de transporte a ser utilizado (*TCP*).

A execução da aplicação é despoletada pela entrada de um objeto no *Event Hub*, neste caso um dos nossos alertas – é esse o seu *trigger*. Após este *trigger*, é inicializada

a biblioteca "*syslog-client*", responsável em *Javascript* pelo funcionamento e implementação de um cliente ou servidor *Syslog* – no nosso caso de um cliente. São, também, iniciadas as variáveis correspondentes aos valores guardados nas configurações da aplicação e já explicitadas acima – as especificações do servidor *Syslog*.

Após estas considerações iniciais e ações preliminares, corre um ciclo que vai iterar nas novas mensagens do *Event Hub*. (Não temos volume de operações suficientemente elevado que justifique esta abordagem, não existem mensagens – alertas – a entrar no *Event Hub* a uma velocidade mais elevada do que aquela a que são consumidas. No entanto, não achamos que faça sentido desenvolver uma solução que não se adeque evoluções futuras e tenha problemas a escalar as suas operações). Neste ciclo vai ocorrer uma verificação para garantir que a mensagem é do tipo *string* (e não *object*, como estava na implementação inicial e originava erros) e de seguida vai ser enviada para o servidor *Syslog*. Ao contrário da implementação do Miguel Ângelo Pereira, não necessitamos de tratar a mensagem com tantas transformações e operações *JSON*, tornando o processo mais simples, direto, fácil de entender e menos propenso a erros ou falhas.

A aplicação tem, no seu fim, um enlace – *binding* – com o *Service Bus* da *Azure*. Este *Service Bus* vai armazenar o estado de execução e vai funcionar como *trigger* de outra *Logic App*, que vai ler esta mensagem e aferir se a execução ocorreu como esperado e se o evento foi encaminhado com sucesso para o servidor *Syslog* da Altice. Caso isto não ocorra, esta mesma *Logic App*, vai enviar um *e-mail* para uma conta à escolha a comunicar que o evento não foi encaminhado, acompanhado com as especificações do mesmo. Funciona, também, como um mecanismo de deteção de falha.

Todo o resumo do processo Azure – Altice pode ser representado pela figura abaixo, a Figura 24.

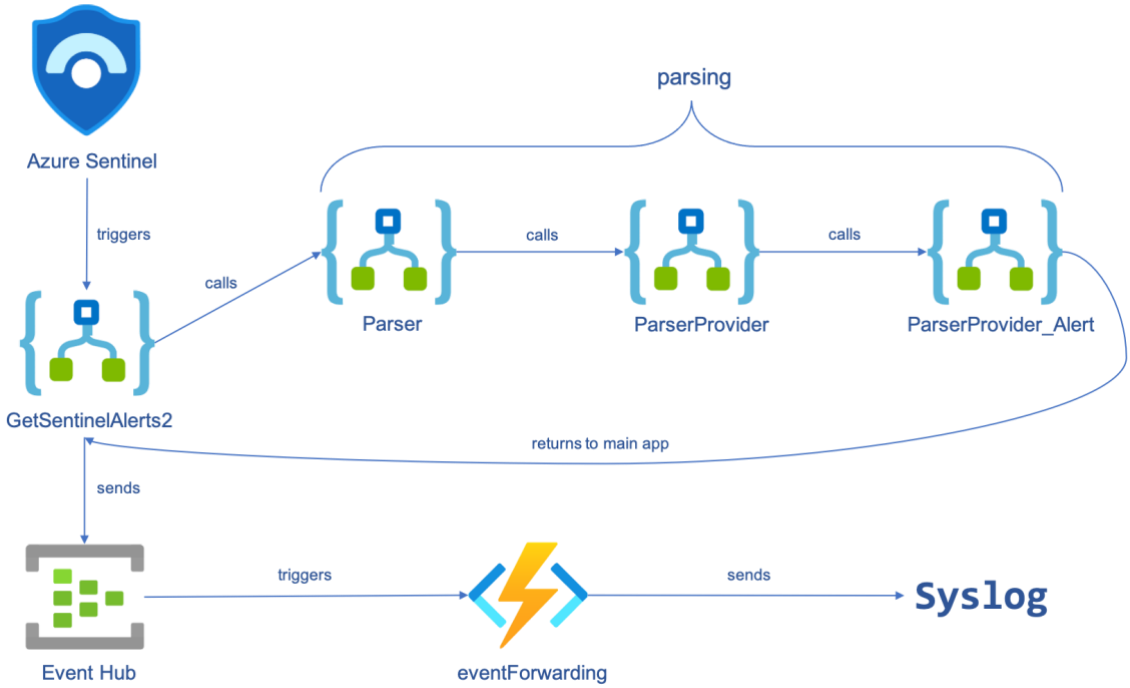

Figura 24 - Esquema, em resumo, do processo do lado do *Azure*

3.5 Configuração do *IBM QRadar*

Este subcapítulo retrata o tópico da configuração do *SIEM* da *IBM*. No que toca a este processo neste *software*, não foi necessário, de longe, tanto trabalho e configurações distintas e complexas – de modo a alcançar o funcionamento esperado – como na plataforma *Azure*.

Nesta fase foi utilizado o *IBM QRadar* de *quality assurance* da Altice Portugal. Este, numa "hierarquia" de validade e completude da informação, encontra-se entre a instância do *IBM QRadar* de desenvolvimento e a do *QRadar* de produção; sendo que o de desenvolvimento é a etapa mais precoce e, como o nome indica, mais virado para testes, o de produção é a "etapa final", é onde o *SOC* tem o seu sistema de *SIEM* real hospedado.

3.5.1 Servidor Syslog – IBM QRadar

As mensagens que chegam da *Azure* para o servidor *Syslog* da Altice são, imediatamente, encaminhadas para o *QRadar*. Isto porque, tal como explicado

anteriormente, é forma de ir ao encontro das políticas de segurança da organização, no que toca à abertura de ligações e às comunicações entre a *internet* e a *intranet* da empresa, sendo que seria desaconselhado ter uma ligação direta entre o *SIEM* e, consequentemente, informações sensíveis e privadas da organização e dos seus cliente e trabalhadores, e a rede exterior.

Após o encaminhamento dos alertas, estes ficam "à responsabilidade" de um *collector*. Visto que não há nenhuma *log source* associada, os alertas são apresentados no separador de *Log Activity* sem qualquer tipo de *parsing* ou mapeamento dos alertas, sendo estes apenas associados ao seu *collector*.

Devido a este aspeto de não haver uma *log source* associada, o próximo passo é, precisamente, a configuração e associação de uma ao nosso fluxo de alertas que chega do *collector* e, anteriormente, do servidor *Syslog*. O processo de criação de uma *log source* é bastante simples e intuitivo; a interface do *IBM QRadar*, apesar de considerar inferior – a nível de intuição e facilidade de aprendizagem – à do *Azure Sentinel*, não deixa de ser de fácil utilização. Indo pelo separador *Admin* e entrando na secção *Log Sources*, temos a opção de adicionar uma nova. Aí é nos pedido uma série de informações elementares para a correta configuração da nova fonte de dados, tal como o protocolo referente às mensagens (*Syslog*, no nosso caso), qual o *collector* que tem essas mesmas mensagens, qual o *hostname* ou endereço *IP* de onde essas mensagens são originárias (*Log Source Identifier*), uma métrica de credibilidade (procura representar a validade e integridade dos *logs* recebidos. Posteriormente, este valor será usado para o cálculo da magnitude e severidade/gravidade dos incidentes do *IBM QRadar*), entre outros – não tão interessantes ou relevantes a nível de uma correta configuração do processo lógico, tendo por exemplo o nome da nova fonte de dados.

Após estes progressos, já teremos a nossa nova *log source* e os alertas já são apresentados no *Log Activity* sob a sua designação – "*Microsoft Azure Sentinel @ AzureSentinel*", visto que "*AzureSentinel*" é *hostname* no envio das mensagens *Syslog* da *Azure* para o servidor de *Syslog* da Altice e, por isso, é – também – o nosso *Log Source Identifier*. Pode parecer redundante, porém é uma forma de manter uma boa prática de higiene e organização das fontes de dados, que muitas vezes são apresentadas num formato tipo_de_dados@endereço_ou_hostname (por exemplo: AlticeSignIns@10.1.10.1).

3.5.2 Mapeamento dos alertas

Como dito no subcapítulo anterior, os registos dos alertas já são apresentados sob a nossa nova fonte de dados. No entanto, estes aparecem como "*Unknown*", sem valor de severidade (às vezes representado como "gravidade" em vez de "severidade") ou sem qualquer entidade associada, tendo apenas acesso à *event string* proveniente da *Azure*, sem qualquer tipo de mapeamento, como pode ser visto na Figura 25.

Para solucionar este problema, procedeu-se ao mapeamento dos alertas e, posteriormente, ao *parsing* e mapeamento das informações relevantes dos mesmos. Para isto recorreu-se à ferramenta *DSM Editor* do *QRadar*. Como já referido anteriormente, o *DSM Editor* é a ferramenta que permite extrair as informações relevantes, categorizar e mapear eventos, definir campos de informações próprios e específicos a cada alerta, e permite criar novos *QID* (*QRadar Identifier*) – uma representação numérica específica e única referente a um tipo de evento, ao qual se associa nome, severidade, descrição e categorias; no fundo, o adicionar de eventos ao *QRadar*.

Event Name	Log Source	Event Count	Time
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 10:20:19 AM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 11:37:27 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	11	Aug 23, 2021, 11:37:14 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 11:37:12 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 11:37:12 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 11:37:12 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 22, 2021, 1:50:29 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 24, 2021, 6:19:19 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 23, 2021, 10:45:46 AM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	5	Aug 24, 2021, 12:01:46 AM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	2	Aug 24, 2021, 12:02:34 AM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	6	Aug 24, 2021, 12:02:08 AM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	15	Aug 23, 2021, 11:41:56 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 24, 2021, 6:22:56 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	1	Aug 24, 2021, 6:22:43 PM
Unknown	Microsoft Azure Sentinel @ AzureSentinel	2	Aug 24, 2021, 12:02:53 AM

Low Level Category	Source IP	Source Port	Destination IP	Destina Port	Username	Magnitude
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	
Unknown	10.	0	0.0.0.1	0	N/A	

Figura 25 - Estado dos alertas quando chegam ao *IBM QRadar* sem o *DSM* definido

(A Figura 25 foi alterada em relação à captura de ecrã original. Houve um corte entre a coluna *Time* e *Low Level Category*, de modo a ter melhor legibilidade.)

No *DSM Editor*, para cada alerta *unknown* que tínhamos, foi necessária a criação do novo *QID*, seguindo o mesmo nome e descrição que já dispúnhamos no *Azure Sentinel*, sendo que as categorias eram escolhidas, de uma lista disponibilizada, as mais adequadas. Quanto à severidade, o *Sentinel* e o *QRadar* têm formas distintas de classificação. Enquanto o *Sentinel* classifica um alerta com os níveis de severidade (*Informational*, *Low*, *Medium* e *High*, o *QRadar* classifica-os numa escala numérica de 1 a 10, sendo 10 o valor de severidade mais alto. A escolha de equivalências entre os níveis de severidade foi o seguinte: *Informational* – 1, *Low* – 3, *Medium* – 6 e *High* – 9.

A etapa seguinte era, pegando na *event string* enviada pela *Azure*, extrair as informações e campos necessários para cada alerta. Para isto, usava-se o processo de *parsing* presente no *DSM Editor*, que com recurso a *regular expressions* ia permitir o isolamento da informação. De seguida, essa informação era ou associada a propriedades já existentes no *DSM*, como *IP* de origem, *username*, etc. ou a propriedades que foram criadas no decorrer deste processo, de modo a adequarem-se com o conteúdo do alerta. Por exemplo, a criação de uma propriedade "*e-mail sender*" ou "*subject*" em alertas relacionados com *e-mails*, visto que o *DSM Editor* não continha essas propriedades. No entanto, de modo a uma melhor higiene e simplicidade, procurou-se ao máximo não criar muitas novas propriedades e usar as já existentes, desde que adequadas. Este processo foi relativamente fácil, muito pelo processo de *parsing* já efetuado na plataforma *Azure*, que propiciou a chegada dos alertas com uma estrutura muito mais compreensível, que permitiu a fácil criação de *regular expressions* e, consequentemente, uma extração de informações muito mais simples.

No fim deste processo, já se pode observar, no nosso separador de *Log Activity*, todos os nossos alertas apresentados pelo seu nome, categorias, *IP* de origem, *username* e muitas outras informações., sendo que uma parte do projeto pode ser dada como terminada, tal como apresentado na Figura 26.

Event Name	Log Source	Event Count	Time ▾
Email containing malicious URL removed a…	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 9, 2021, 9:25:35 AM
Sign-ins from IPs that attempt sign-ins to di…	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 9, 2021, 9:24:03 AM
Azure Portal brute force sign-in attack	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 9, 2021, 6:53:44 AM
Anonymous IP address	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 9, 2021, 4:31:22 AM
Anonymous IP address	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 9, 2021, 4:01:34 AM
Malware linked IP address	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 7, 2021, 7:17:56 PM
Email reported by user as malware or phish	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 7, 2021, 5:22:04 PM
Suspicious Resource deployment	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 6, 2021, 7:52:19 PM
Suspicious granting of permissions to an a…	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 6, 2021, 7:52:16 PM
Multiple Teams deleted by a single user	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 6, 2021, 7:52:14 PM
Multiple Password Reset by user	Microsoft Azure Sentinel @ AzureSentinel	1	Sep 6, 2021, 7:52:12 PM

Low Level Category	Source IP	Source Port	Destination IP	Destina Port	Username	Magnitude
Mail Denied	194.	0	0.0.0.1	0	-m-mo…	
User Login Success	188.	0	0.0.0.1	0	o-j-sou…	
Brute force login	188.	0	0.0.0.1	0	-m-pin…	
Suspicious IP Address	192.	0	0.0.0.1	0	-marin…	
Suspicious IP Address	192.	0	0.0.0.1	0	-marin…	
Known Offender IP	85.	0	0.0.0.1	0	m.jorda…	
Mail Denied	194.	0	0.0.0.1	0	-r-faria…	
User Activity	194.	0	0.0.0.1	0	o-g-bru…	
User Activity	194.	0	0.0.0.1	0	o-g-bru…	
User Activity	194.	0	0.0.0.1	0	a-c-viei…	
Password Change Succeeded	194.	0	0.0.0.1	0	)57681…	

Figura 26 - Captura do *Log Activity* após os alertas passarem pelo respetivo *DSM*

3.6 Estratégias de simulação de alertas

Para se conseguir concluir esta etapa, e algumas das anteriores, com sucesso, foi necessário recorrer a estratégias de injeção e geração dos alertas nos sistemas. Ou seja, como faríamos para simular os eventos que levariam à produção do alerta referente. Para nossa "sorte", todos os alertas que nos comprometemos a configurar e migrar para o *QRadar* foram gerados passivamente ao longo dos últimos meses – foram todos detetados pelas nossas *Analytics Rules* e conectores de segurança em ocorrências reais, com os dados e registos reais da empresa. Tendo por exemplo, sido sinalizados possíveis ataques de *brute-force* ou cliques num possível *URL* malicioso num *e-mail* por parte de utilizadores reais em situações reais.

De modo a testar e desenvolver o processo de *parsing* da *Azure*, recorremos ao histórico de execuções da nossa *Get-SentinelAlerts2* para ir testando e recriando os hipotéticos resultados da execução do *parsing*. Visto que a nossa aplicação só guarda registos durante 90 dias, adicionámos um passo à mesma que envia a *event string* do nosso alerta via *e-mail* para uma conta Altice, sendo que esta serviu de armazenamento e registo de todas as execuções. Mas como e porque é que isto nos pode ser útil? No que toca ao primeiro processo de *parsing*, o feito pelas *Logic Apps* na *Azure*, nunca houve problema, já que o íamos desenvolvendo à medida que os incidentes eram detetados e os alertas gerados. Como dito anteriormente, todos os alertas que nos comprometemos a

configurar foram detetados, *à priori*, pelo *Azure Sentinel*; no entanto, numa fase mais avançada deste projeto, na configuração, mapeamento e *parsing* dos alertas já no *QRadar*, alguns destes, temporalmente não se encaixavam nesta janela de 90 dias desde a sua deteção. É aqui que entra a solução do encaminhamento via *e-mail* dos alertas e das suas *event strings* para uma conta Altice, funcionando como armazenamento dos mesmos. As *Azure Functions* permitem o teste das suas aplicações e do seu código através do *input* textual que o utilizador considere mais adequado. Ora, para nós o mais adequado é a injeção dos nossos alertas – e das respetivas *event strings* – levantados há mais de 90 dias; sendo que para a *eventForwarding*, receber as informações pelo seu *trigger* de *Event Hub* ou através deste *input* manual em ambiente de teste, é igual para o seu funcionamento. Ou seja, irá processá-los da mesma forma e, assim, conseguimos enviar estes alertas mais antigos para o *QRadar*. Ambos os processos tiveram de ser continuados até cada alerta ter o seu *parsing* da forma que considerámos mais pertinente, podendo considerar um pouco como um processo de tentativa e erro.

Capítulo 4

Avaliação da solução

No presente capítulo vai ser desenvolvida e explicada a avaliação da solução implementada e o balanço do processo.

Quanto aos resultados finais, resumidamente, pode ser concluído que estes, até certo ponto, foram positivos. O objetivo do projeto, no que toca à visualização dos alertas, foi alcançado, na medida em que os eventos das infraestruturas *cloud* da *Microsoft* – o *Azure* e o *Office 365* – culminavam em alertas que eram apresentados, com todos os detalhes e informações relevantes, no *SIEM* do *SOC* da Altice, o *IBM QRadar*. Na outra face da moeda, por questões de tempo, não foi possível ir ao encontro de outro objetivo deste trabalho – a criação de regras de correlação e *use cases* utilizando os novos alertas e outros eventos, alertas e fontes de dados no *IBM QRadar*, de modo a aproveitar todas as potencialidades deste *SIEM*. Este objetivo faz parte do trabalho previsto para o futuro e, por isso, está mais desenvolvido no capítulo seguinte.

No que toca aos objetivos singulares do tipo e fonte dos alertas, foi concluída a congruência entre os sistemas *on-premise* e as plataformas *cloud*. Além desta coerência, a solução foi enriquecida com alertas de conectores já disponíveis, alertas aconselhados pela *Microsoft* e por bibliografia especializada; podendo, assim, depreender o sucesso nesse aspeto.

Em relação à estratégia escolhida para o encaminhamento dos alertas – o uso do *Azure Sentinel* e dos restantes produtos *Azure* – esta permitiu uma fácil gestão e monitorização de todo o processo, tal como a implementação de automatizações, incluindo mecanismos de deteção de falha no transporte dos alertas até à rede interna da Altice; além disso, o processo de *parsing* e a possibilidade de armazenamento dos alertas num formato textual legível, acaba por ser relevante em situações de indisponibilidade do *IBM QRadar* ou para auditoria ao processo de encaminhamento do alerta. Isto tudo, claro, aliado ao correto funcionamento da ligação entre a *Azure* e o servidor *Syslog* da empresa.

Outro ponto importante é o tamanho da janela temporal referente a todo o processo. Ou seja, se neste ambiente de cibersegurança, o tempo desde o levantamento

do alerta no *Azure Sentinel* até este aparecer, já com o *parsing* efetuado, na interface do *IBM QRadar*. Estes valores, conjugando os tempos de execução da vertente *Logic App* e da *Azure Function*, são, respetivamente, no máximo 51,74s e 6,85s, o que resulta num tempo máximo de execução do encaminhamento de 58,59s; os valores temporais médios são 18,04s e 0,84s, o que resulta num tempo médio de execução de 18,84s; no cenário mais em que o encaminhamento é mais rápido, os valores são de 8,96s e 0,02s, o que resulta num tempo total de 8,98s. Num âmbito de uma solução *SIEM*, a curta velocidade a que os eventos são processados e os alertas levantados é fundamental para uma resposta pronta ao problema. Com estes baixos valores referentes ao tamanho da janela temporal no processamento e encaminhamento na vertente *Azure*, podemos inferir que não é posta em causa a segurança e solidez do *SIEM*, nem a relevância dos alertas que lá chegam, sendo estes tempos bastante satisfatórios, ainda mais quando temos em conta a complexidade do processo.

Quanto à parte do *IBM QRadar*, os alertas que lhe chegam são apresentados, na sua interface, com o seu mapeamento e *parsing* feito após passarem pelo *DSM* criado, de forma intuitiva e muito facilmente legível. Esta interface do *SIEM* da *IBM* está construída de forma a apresentarem, por defeito, a sua *log source*, nome, categoria, *IP* originário, *username* correspondente e severidade, ordenados por data e hora. Estes, quando são selecionados, mostram todos os seus outros campos relevantes e a *event string* correspondente, para o caso de haver a necessidade de mais especificidade. No que toca a este aspeto, a conclusão é, tal como nos pontos referidos acima, de sucesso em relação ao objetivo. A organização, o mapeamento do alerta, extração das suas informações relevantes e a sua apresentação de forma facilmente legível e compreensível, eram pontos essenciais para este trabalho e para a performance do *CyberSOC*, sendo que ficou, apenas, a pecar pela incapacidade do desenvolvimento atempado dos *use cases* e das regras de correlação dos novos alertas com os dados já existentes no *SIEM*, tal como referido anteriormente.

Capítulo 5

Possíveis melhoramentos e alterações

Neste capítulo vamos abordar quais os possíveis melhoramentos e abordagens a ter para aperfeiçoar e ter em conta um processo de busca da maior completude possível deste projeto, de modo a estar constantemente na vanguarda no que toca à vertente da segurança e da performance do *SOC* da Altice Portugal. O primeiro passo é a passagem dos dados do *QRadar* de *quality assurance* para o *QRadar* de produção, a principal e mais importante instância de *SIEM* do *SOC*.

Tendo em conta algumas complicações a nível temporal, não foi possível aproveitar todas as capacidades de um SIEM, mais propriamente do IBM QRadar. Isto na medida em que não foi possível criar ofensas, correlacionar esta nova fonte de dados com outras fontes de eventos já monitorizadas pelo *CyberSOC* e, com recurso a esta correlação, o desenho de novos *use cases*. Para uma compreensão mais intuitiva em relação a este ponto, a título de exemplo, poderíamos ter a correlação de um alerta *Azure* de um utilizador ligado, em simultâneo, à rede interna da organização, com outros eventos e alertas levantados por outras fontes de dados referentes a esse mesmo utilizador ou máquina. Este ponto é bastante importante para permitir ao *SOC* da Altice um maior aproveitamento da integração deste novo ecossistema e está no topo da lista referente ao trabalho futuro a desenvolver. Concluído esse ponto, é possibilitada a exploração ao máximo das novas possibilidades oferecidas pela junção e, principalmente, correlação dos novos alertas com os dados já existentes.

Seguidamente, a abordagem mais óbvia é a procura contínua de novos alertas adequados às nossas necessidades, com a investigação de novos conectores e regras de alarmística apropriadas e oportunas. Dentro destas novas fontes de dados e, consequentemente, novos alertas, um produto *Azure* que nos chamou à atenção foi o *Azure Security Center*. Esta plataforma é um sistema de gestão de segurança; providencia uma análise de possíveis ameaças e avalia o nosso ambiente e recursos, de modo a promover recomendações – monitoriza as configurações de segurança e a "saúde" dos recursos e infraestruturas, recomendando alguma alteração com o intuito de alcançar uma

maior robustez e garantir que tudo está dentro das melhores práticas da segurança informática. Dando alguns exemplos, esta ferramenta pode alertar para a ausência de cifragem de dados armazenados e/ou em transporte, ou relembrar a verificação de existência de atualizações dos nossos sistemas. Podendo ser integrado no *Azure Sentinel*, o *Security Center* deve ser visto mais como uma ferramenta mais preventiva e proactiva no que toca ao nosso ambiente de segurança [20]. Este pode ser, também, bastante útil para o concretizar da sugestão do Eng. José Alegria em relação à integração de alertas sobre aplicações Altice e *third-party, virtual machines e virtual networks* hospedadas na plataforma *Azure*.

No que toca ao processo de ingestão de informação da *Azure* para o *QRadar* é, também, pertinente a exploração das restantes estratégias referidas no subcapítulo 3.2. É proveitoso perceber se acaba por ser mais apropriado para a organização ter a ingestão dos eventos em cru de todas as fontes de dados (por exemplo: *Azure Active Directory* e *Office 365*) no *IBM QRadar*, sendo que as regras de alarmística e as restantes configurações dos eventos e alertas seriam feitas nesse *SIEM*, o que resultava numa certa independência do *Azure Sentinel*.

Assumindo esta opção da intermediação pelo *Azure* e pelos seus serviços, a solução apresentada neste documento, é importante ter em consideração a contínua monitorização dos sistemas – do *Azure Sentinel*, do estado das *Logic Apps* e *Azure Functions*, por exemplo – a procura de *updates* e de novas *features* dessas mesmas soluções e o acompanhamento da comunidade de técnicos e *developers* da *Microsoft*. No que toca aos produtos *Azure* utilizados neste projeto, é muito oportuno explorar e desenvolver o encaminhamento dos alertas pela *Azure Function eventForwarding* usando um transporte *TLS*, em vez do transporte *TCP* utilizado. No decorrer deste projeto, por questões temporais e burocráticas, não foi possível desenvolver esta solução dessa forma a tempo da redação deste documento. No entanto, consideramos relevante o uso desse protocolo de transporte mais seguro e é algo que fica no topo das prioridades no que toca a melhoramentos futuros. No que toca aos mecanismos de deteção de falha, mais propriamente na *pipeline Azure Sentinel – Event Hub* e *Event Hub – Syslog*, é igualmente pertinente averiguar se a solução do aviso de falha ser processado via um *e-mail* ou se há uma solução mais adequada a um funcionamento mais fluído do *SOC*.

Em relação a outras ferramentas e iniciativas do Direção de *Cyber Security* e Privacidade (DCY) da Altice, o facto deste projeto ter uma grande variedade e abrangência no tipo de dados que é tratado (vários tipos de alertas diferentes, de diferentes

tipos de ataques e ocorrências de segurança), é esperado o aproveitamento dos resultados do mesmo para outros projetos do departamento – podemos dar como exemplo programas de combate a *phishing* e os alertas deste trabalho referentes a esse tópico.

Em suma, apesar de certa conclusão e da redação deste documento, este é um projeto para qual se irá sempre procurar possíveis melhoramentos e formas de completar e enriquecer a sua experiência e competência a nível de segurança, assim como o consequente melhoramento do *SOC* da Altice Portugal.

Capítulo 6

Conclusão e discussão

Este trabalho culmina com o enriquecimento da informação, da capacidade, do *awareness* e da performance de segurança do *SIEM* da Altice Portugal e, consequentemente, do seu *SOC*.

Com a enorme e crescente de popularidade e uso de plataformas de computação em *cloud*, a integração dos alertas detetados na plataforma *cloud* da Altice – o *Microsoft Azure* e *Office 365* – é essencial e de extrema importância no que toca à questão de segurança, oferecendo a possibilidade e capacidade de monitorizar essas ocorrências através do *SIEM* – o *IBM QRadar* – já usado pelo *SOC* da organização. No entanto, como dito no capítulo anterior, não foi possível o desenvolvimento de regras de correlação entre os alertas e com outras fontes de dados, o que nos melhoraria muito a performance de segurança e permitiria explorar todas as demais potencialidades do IBM QRadar, não sendo apenas um ambiente de visualização desses mesmo alertas.

Além do cerne do trabalho, é com alguma felicidade que vemos o resultado final do mesmo resultar numa solução – em geral – constituída por processos simples, intuitivos e de fácil compreensão; além de promover alertas explícitos, completos e de fácil leitura. Consideramos, e foi feito com esse intuito, que qualquer utilizador recém-chegado ao *SOC*, à DCY ou mesmo um cliente *B2B* da Altice, não terá dificuldades em entender o projeto, todas as suas envolvências e mesmo este documento.

Outra questão relevante e indicadora da importância que este projeto pode trazer, é a forma como este trabalho, de integração dos registos e alertas das plataformas *cloud*, abre a porta a uma utilização mais exaustiva e segura destes recursos, permitindo possíveis migrações de sistemas *on-prem* para uma hospedagem em nuvem, ou para o uso da plataforma *Azure* e dos seus recursos da forma mais eficiente, eficaz e menos onerosa possível; tudo sem preocupações referentes ao desconhecimento do que se passaa a nível de ocorrências de segurança nesse mesmo ambiente, que era o cenário até à execução e implementação deste trabalho. Toda a previsão futura de crescimento destas plataformas de computação em nuvem e a enorme quantidade de recursos e soluções que

oferecem e virão a oferecer, dá um ênfase maior à segurança informática especializada e centrada nesta vertente, sendo esta muito importante para garantirmos soluções e sistemas seguros, tendo em conta serem facilmente escaláveis e com capacidade de acompanhar todos esses desenvolvimentos e inovações sem pôr em causa essa mesma segurança.

No que toca às decisões tomadas, mais propriamente em relação à estratégia a usar no trabalho (referido no subcapítulo 3.2 e no capítulo 4), consideramos existir muitas vantagens no uso da intermediação por parte do *Azure Sentinel* e dos restantes produtos *Azure* utilizados no processo de encaminhamento dos alertas; no fim do trabalho e no seu balanço, consideramos que, provavelmente, foi a melhor opção a tomar. Permitiu, ainda que pareça o contrário com o uso de vários produtos da *Azure*, um processo muito simples, intuitivo e de fácil manuseamento. No que toca à vertente de cibersegurança, a possível integração – e a facilidade da mesma – com ferramentas como o *Azure Security Center*, outras ferramentas de segurança *Microsoft* e *third-party*, e a centralização inicial de todos os dados provenientes dessas fontes, oferece uma primeira linha de segurança. Esta, sendo tanto rica em usabilidade, como igualmente rica em detalhe, torna-se um ponto importante e que deve pesar na decisão de uma futura alteração da estratégia desta ligação *Azure* – Altice. Além disto, outros aspetos como a possibilidade futura de maior aproveitamento dos seus recursos *Azure* por parte da Altice, a enorme variedade disponível desses mesmos recursos e as vantagens que podem oferecer ou, até, a existência de uma comunidade ativa de *developers* dos sistemas *Microsoft*, fazem-nos crer que o aproveitamento de todas estas especificações e vantagens, facilmente é a melhor estratégia a abraçar.

Devido a uma série de complicações de teor mais burocrático com a Altice, como os processos de pedido de autorização para aberturas de conectividades e a concretização do mesmo, houve alguns atrasos no desenvolvimento do trabalho. Apesar de estas situações serem algo esperadas num ambiente empresarial desta dimensão e tendo em conta a área sensível que é a cibersegurança, estas não nos permitiram explorar outras estratégias e melhoramentos, mas tal como indicado no capítulo anterior, é algo que estará sempre em mente e em processo de teste e implementação. No entanto, por outro lado, a Altice e os seus colaboradores da DCY, sempre mostraram a sua disponibilidade para ajudar a concretizar este trabalho em tudo o que podiam, sendo que dificilmente poderia ter encontrado uma equipa e ambiente melhor para me auxiliar na condução deste trabalho.

Bibliografia

[1] Moore, Gordon E. (1965). "Cramming more components onto integrated circuits".

[2] Koomey, Jonathan; Berard, Stephen; Sanchez, Marla; Wong, Henry (2010), "Implications of Historical Trends in the Electrical Efficiency of Computing", IEEE Annals of the History of Computing, 33 (3): 46–54.

[3] "SOC as a service". Claranet PT. https://www.claranet.pt/cybersecurity/soc-as-a-service. Acedido a 27 de Outubro de 2020.

[4] "What is SIEM? A Complete Beginner's Guide". Varonis - Inside Out Security. https://www.varonis.com/blog/what-is-SIEM. Acedido a 28 de Outubro de 2020.

[5] "What is a Security Operations Center (SOC)?". Digital Guardian. https://digitalguardian.com/blog/what-security-operations-center-soc. Acedido a 27 de Outubro de 2020.

[6] Verizon. (2020). "2020 Data Breach Investigations Report". https://enterprise.verizon.com/resources/reports/2020-data-breach-investigations-report.pdf. Acedido a 25 de Janeiro de 2021.

[7] K. Pratt M. "What is SIEM software? How it works and how to choose the right tool". CSO Online. https://www.csoonline.com/article/2124604/what-is-SIEM-software-how-it-works-and-how-to-choose-the-right-tool.html. Acedido a 4 de Novembro de 2020.

[8] "What's the difference between an event and an incident?". Databarracks. https://www.databarracks.com/blog/whats-the-difference-between-an-event-and-an-incident. Acedido a 5 de Novembro de 2020.

[9] S. Bhatt, P. K. Manadhata and L. Zomlot, "The Operational Role of Security Information and Event Management Systems," in IEEE Security & Privacy, vol. 12, no. 5, pp. 35-41, Set.-Oct. 2014.

[10] A. Osório, “Threat detection in SIEM considering risk assessment,” Tese de Mestrado, Faculdade de Ciências da Universidade de Lisboa, 2018.

[11] “What is SIEM and how does it work?”. FireEye. https://www.fireeye.com/products/helix/what-is-SIEM-and-how-does-it-work.html. Acedido a 5 de Novembro de 2020.

[12] “IBM Knowledge Center | IBM QRadar SIEM Solution Brief”. IBM. https://www.ibm.com/downloads/cas/RLXJNX2G . Acedido a 7 de Outubro de 2020.

[13] “IBM Knowledge Center | QRadar architecture overview”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_7.3.3/com.ibm.qradar.doc/c_qradar_deployment_guide_arch.html. Acedido a 9 de Outubro de 2020.

[14] “IBM Knowledge Center | QRadar supported DSMs”. IBM. https://www.ibm.com/docs/en/dsm?topic=configuration-qradar-supported-dsms. Acedido a 21 de Dezembro de 2020.

[15] “IBM Knowledge Center | DSM Editor overview”. IBM. https://www.ibm.com/docs/en/qsip/7.3.2?topic=qradar-dsm-editor-overview. Acedido a Acedido a 21 de Dezembro de 2020.

[16] “IBM Knowledge Center | QRadar components”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_7.3.3/com.ibm.qradar.doc/c_qradar_comps2_deployment_guide.html. Acedido a 9 de Outubro de 2020.

[17] “IBM Knowledge Center | QRadar events and flows”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_7.3.3/com.ibm.qradar.doc/c_qradar_deploy_event_and_flow_pipeline.html. Acedido a 11 de Outubro de 2020.

[18] “What is Azure Sentinel?”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/sentinel/overview. Acedido a 26 de Outubro de 2020.

[19] “Directory of Azure Cloud Services”. Microsoft Azure. https://azure.microsoft.com/en-us/services Acedido a 26 de Outubro de 2020.

[20] “Azure Identity and Access Management Solutions”. Microsoft Azure. https://azure.microsoft.com/en-us/product-categories/identity. Acedido a 27 de Outubro de 2020.

[21] “Azure Files”. Microsoft Azure. https://azure.microsoft.com/en-us/services/storage/files. Acedido a 29 de Outubro de 2020.

[22] “What is Azure Security Center?”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/security-center/security-center-introduction. Acedido a 29 de Novembro de 2020.

[23] “What is the difference between Azure Security Center and Azure Sentinel?”. Medium. https://medium.com/the-cloud-builders-guild/what-is-the-difference-between-azure-security-center-and-azure-sentinel-9d91eb801cd2. Acedido a 30 de Novembro de 2020.

[24] “What is Azure Event Hubs? - a Big Data ingestion service - Azure Event Hubs”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-about. Acedido a 4 de Dezembro de 2020.

[25] “What is Azure Logic Apps?”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-overview. Acedido a 30 de Janeiro de 2021.

[26] “Create and Call Nested Logic Apps”. Serverless Notes. https://www.serverlessnotes.com/docs/nested-logic-apps. Acedido a 13 de Fevereiro de 2021.

[27] “Introduction to Azure Functions”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-overview. Acedido a 3 de Junho de 2021.

[28] “What is Azure Service Bus?”. Microsoft Docs. https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-messaging-overview. Acedido a 18 de Agosto de 2021.

[29] “What is Syslog?”. Paessler. https://www.paessler.com/it-explained/syslog#:~:text=Syslog%20stands%20for%20System%20Logging,location%20for%20monitoring%20and%20review. Acedido a 29 de Junho de 2021.

[30] “The Syslog Protocol – RFC 5424”. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc5424. Acedido a 03 de Julho de 2021.

[31] “IBM Knowledge Center | Microsoft Azure Security Center”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_DSM/com.ibm.dsm.doc/c_dsm_guide_ms_azure_security_center_overview.html. Acedido a 13 de Novembro de 2020.

[32] “IBM Knowledge Center | Microsoft Office 365”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_DSM/com.ibm.dsm.doc/c_dsm_guide_microsoft_office_365_overview.html. Acedido a 29 de Novembro de 2020.

[33] “IBM Knowledge Center | Microsoft Azure Platform”. IBM. https://www.ibm.com/support/knowledgecenter/SS42VS_DSM/com.ibm.dsm.doc/c_dsm_guide_microsoft_azure_overview.html. Acedido a 17 de Novembro de 2020.

[34] Quest Software Inc. *Top 10 Security Events to Monitor in Azure AD and Office 365.* Aliso Viejo, 2020.

[35] Shasha, Yaniv. *Get-SentinelAlertsEvidence*. GitHub. 2020. https://github.com/Azure/Azure-Sentinel/tree/master/Playbooks/Get-SentinelAlertsEvidence

[36] Ângelo Pereira, Miguel. *AzureMonitor2Syslog*. GitHub. 2020. https://github.com/miguelangelopereira/azuremonitor2syslog

Anexos

Anexo A – **Regra de alarmística *Azure Portal Brute Force Sign-in Attack***

```
let failureCountThreshold = ;
let authenticationWindow = ;
SigninLogs
| extend Device = strcat(DeviceDetail.operatingSystem, ";",
DeviceDetail.browser)
| extend StatusCode = tostring(Status.errorCode), StatusDetails =
tostring(Status.additionalDetails)
| extend Location = strcat(LocationDetails.city, ", ", LocationDetails.state,
", ", LocationDetails.countryOrRegion)
// Split out failure versus non-failure types
| extend FailureOrSuccess = iff(ResultType in (      "      ", "      ",
"      ", "      "), "Success", "Failure")
| summarize StartTimeUtc = min(TimeGenerated), EndTimeUtc =
max(TimeGenerated), makeset(IPAddress), makeset(Device), makeset(Location),
    makeset(ResultType), FailureCount = countif(FailureOrSuccess ==
"Failure”)
    by bin(TimeGenerated, authenticationWindow), UserDisplayName,
UserPrincipalName, AppDisplayName
| where FailureCount >= failureCountThreshold
| extend timestamp = StartTimeUtc, AccountCustomEntity = UserPrincipalName
```

Anexo B – **Regra de alarmística** ***User added to Azure Active Directory Privileged Groups***

```
let timeframe = 1d;
let OperationList = dynamic(["Add member to role", "Add member to role in PIM completed (permanent)"]);
let PrivilegedGroups = dynamic(["Global Administrator", "Privileged Role Administrator", "Exchange Administrator"]);
AuditLogs
| where TimeGenerated >= ago(timeframe)
| where LoggedByService =~ "Core Directory"
| where Category =~ "RoleManagement"
| where OperationName in~ (OperationList)
| extend updatedUserName = parse_json(TargetResources[0]).userPrincipalName
| extend updatedUserId = parse_json(TargetResources[0]).id
| extend UpdatedUser = pack("UpdatedUserId", updatedUserId, "UpdatedUserName", updatedUserName)
| extend newRoleId = split(tostring(parse_json(TargetResources[0]).modifiedProperties[0].newValue), "\"")[1]
| extend newRoleDisplayName = split(tostring(parse_json(TargetResources[0]).modifiedProperties[1].newValue), "\"")[1]
| extend newRoleWellKnownObject = split(tostring(parse_json(TargetResources[0]).modifiedProperties[3].newValue), "\"")[1]
| extend NewRole = pack("RoleId", newRoleId, "RoleDisplayName", newRoleDisplayName, "RoleWellKnownObject", newRoleWellKnownObject)
| extend initiatedByUserId = InitiatedBy.user.id
| extend initiatedByUserName = InitiatedBy.user.userPrincipalName
| extend InitiatedByUser = pack("InitiatedId", initiatedByUserId, "InitiatedUserName", initiatedByUserName)
| extend InitiatedByApp = Identity
| project
    TenantId,
    TimeGenerated,
    SourceSystem,
    OperationName,
    UpdatedUser,
    NewRole,
    InitiatedByUser,
    InitiatedByApp
```

Anexo C – **Regra de alarmística *Sign-ins from IPs that attempt sign-ins to disabled accounts***

```
let lookBack =  ;
SigninLogs
| where TimeGenerated >= ago(lookBack)
| where ResultType == "     "
| where ResultDescription == "User account is disabled. The account has been
disabled by an administrator."
| summarize StartTimeUtc = min(TimeGenerated), EndTimeUtc =
max(TimeGenerated), disabledAccountLoginAttempts = count(),
    disabledAccountsTargeted = dcount(UserPrincipalName),
applicationsTargeted = dcount(AppDisplayName), disabledAccountSet =
makeset(UserPrincipalName),
    applicationSet = makeset(AppDisplayName)
    by IPAddress
| order by disabledAccountLoginAttempts desc
| join kind= leftouter (
    // Consider these IPs suspicious - and alert any related  successful
sign-ins
    SigninLogs
    | where TimeGenerated >= ago(lookBack)
    | where ResultType == 0
    | summarize
        successfulAccountSigninCount = dcount(UserPrincipalName),
        successfulAccountSigninSet = makeset(UserPrincipalName, 15)
        by IPAddress
    // Assume IPs associated with sign-ins from 100+ distinct user accounts
are safe
    | where successfulAccountSigninCount < 100
    )
    on IPAddress
// IPs from which attempts to authenticate as disabled user accounts
originated, and had a non-zero success rate for some other account
| where successfulAccountSigninCount != 0
| project StartTimeUtc, EndTimeUtc, IPAddress, disabledAccountLoginAttempts,
disabledAccountsTargeted, disabledAccountSet, applicationSet,
    successfulAccountSigninCount, successfulAccountSigninSet
| order by disabledAccountLoginAttempts
| extend timestamp = StartTimeUtc, IPCustomEntity = IPAddress
```

Anexo D – **Regra de alarmística *Multiple password reset by user***

```
let timeframe = 1d;
let PerUserThreshold = 5;
let TotalThreshold = 10;
let action = dynamic(["change", "changed", "reset"]);
let pWord = dynamic(["password", "credentials"]);
let PasswordResetMultiDataSource =
    (union isfuzzy=true
        (//Password reset events
        //4723: An attempt was made to change an account's password
        //4724: An attempt was made to reset an accounts password
        SecurityEvent
        | where TimeGenerated >= ago(timeframe)
        | where EventID in ("4723", "4724")
        | project TimeGenerated, Computer, AccountType, Account, Type),
        (//Azure Active Directory Password reset events
        AuditLogs
        | where TimeGenerated >= ago(timeframe)
        | where OperationName has_any (pWord) and OperationName has_any
(action)
        | extend AccountType = tostring(TargetResources[0].type), Account =
tostring(TargetResources[0].userPrincipalName),
            TargetResourceName =
tolower(tostring(TargetResources[0].displayName))
        | project TimeGenerated, AccountType, Account, Computer =
TargetResourceName, Type),
        (//OfficeActive ActiveDirectory Password reset events
        OfficeActivity
        | where TimeGenerated >= ago(timeframe)
        | where OfficeWorkload == "AzureActiveDirectory"
        | where (ExtendedProperties has_any (pWord) or ModifiedProperties
has_any (pWord)) and (ExtendedProperties has_any (action) or
ModifiedProperties has_any (action))
        | extend AccountType = UserType, Account = OfficeObjectId
        | project TimeGenerated, AccountType, Account, Type, Computer = ""),
        (SigninLogs
        | where TimeGenerated >= ago(timeframe)
        | where OperationName =~ "Sign-in activity" and ResultType has_any
("50125", "50140")
        | project
            TimeGenerated,
            AccountType = AppDisplayName,
            Computer = IPAddress,
            Account = UserPrincipalName,
            Type
        )
    );
```

```
let pwrmd = PasswordResetMultiDataSource
    | project TimeGenerated, Computer, AccountType, Account, Type;
(union isfuzzy=true
    (pwrmd
    | summarize
        StartTimeUtc = min(TimeGenerated),
        EndTimeUtc = max(TimeGenerated),
        Computer = makeset(Computer),
        AccountType = makeset(AccountType),
        Total=count()
        by Account, Type
    | where Total > PerUserThreshold
    | extend ResetPivot = "PerUserReset"),
    (pwrmd
    | summarize
        StartTimeUtc = min(TimeGenerated),
        EndTimeUtc = max(TimeGenerated),
        Computer = makeset(Computer),
        Account = tostring(makeset(Account)),
        AccountType = makeset(AccountType),
        Total=count()
        by Type
    | where Total > TotalThreshold
    | extend ResetPivot = "TotalUserReset")
)
| extend
    timestamp = StartTimeUtc,
    AccountCustomEntity = Account,
    HostCustomEntity = tostring(Computer)
```

Anexo E – **Regra de alarmística** ***Suspicious granting of permissions to an account***

```
let starttime = 1d;
let endtime = 1d;
// The number of operations below which an IP address is considered an
unusual source of role assignment operations
let alertOperationThreshold = 5
let createRoleAssignmentActivity = AzureActivity
    | where OperationName == "Create role assignment"
    | where ActivityStatus == "Succeeded";
createRoleAssignmentActivity
| where TimeGenerated between (ago(starttime)..ago(endtime))
| summarize count() by Caller
| where count_ >= alertOperationThreshold
| join kind = rightanti (
    createRoleAssignmentActivity
    | where TimeGenerated > ago(endtime)
    | summarize StartTimeUtc = min(TimeGenerated), EndTimeUtc =
max(TimeGenerated), ActivityTimeStamp = makelist(TimeGenerated),
ActivityStatus = makelist(ActivityStatus),
        OperationIds = makelist(OperationId), CorrelationId =
makelist(CorrelationId), ActivityCountByCallerIPAddress = count()
        by ResourceId, CallerIpAddress, Caller, OperationName, Resource,
ResourceGroup
    )
    on Caller
| extend timestamp = StartTimeUtc, AccountCustomEntity = Caller,
IPCustomEntity = CallerIpAddress
```

Anexo F – **Regra de alarmística *Suspicious resource deployment***

```
let szOperationNames = dynamic(["Create or Update Virtual Machine", "Create
Deployment"]);
let starttime = 1d;
let endtime = 1h;
let RareCaller = AzureActivity
    | where TimeGenerated between (ago(starttime) .. ago(endtime))
    | where OperationName in~ (szOperationNames)
    | project ResourceGroup, Caller, OperationName, CallerIpAddress
    | join kind=rightantisemi (
        AzureActivity
        | where TimeGenerated > ago(endtime)
        | where OperationName in~ (szOperationNames)
        | summarize
            StartTimeUtc = min(TimeGenerated),
            EndTimeUtc = max(TimeGenerated),
            ActivityStatus = makeset(ActivityStatus),
            OperationIds = makeset(OperationId),
            CallerIpAddress = makeset(CallerIpAddress)
            by ResourceId, Caller, OperationName, Resource, ResourceGroup
        )
        on Caller, ResourceGroup
    | mvexpand CallerIpAddress
    | where isnotempty(CallerIpAddress);
let Counts = RareCaller
    | summarize ActivityCountByCaller = count() by Caller;
RareCaller
| join kind= inner (Counts) on Caller
| project-away Caller1
| extend
    timestamp = StartTimeUtc,
    AccountCustomEntity = Caller,
    IPCustomEntity = tostring(CallerIpAddress)
| sort by ActivityCountByCaller desc nulls last
```

Anexo G – **Regra de alarmística *External user added and removed in short timeframe***

```
OfficeActivity
| where OfficeWorkload =~ "MicrosoftTeams"
| where Operation =~ "MemberAdded"
| extend UPN = tostring(parse_json(Members)[0].UPN)
| where UPN contains ("#EXT#")
| project TimeAdded=TimeGenerated, Operation, UPN, UserWhoAdded = UserId,
TeamName
| join (
    OfficeActivity
    | where OfficeWorkload =~ "MicrosoftTeams"
    | where Operation =~ "MemberRemoved"
    | extend UPN = tostring(parse_json(Members)[0].UPN)
    | where UPN contains ("#EXT#")
    | project TimeDeleted=TimeGenerated, Operation, UPN, UserWhoDeleted =
UserId, TeamName, TeamGuid
    )
    on UPN
| where TimeDeleted > TimeAdded
| project TimeAdded, TimeDeleted, UPN, UserWhoAdded, UserWhoDeleted,
TeamName, TeamGuid
| extend timestamp = TimeAdded, AccountCustomEntity = UPN
```

Anexo H – **Regra de alarmística *Multiple Teams deleted by a single user***

```
// Adjust this value to change how many Teams should be deleted before
including
let max_delete_count =
// Adjust this value to change the timewindow the query runs over
OfficeActivity
| where OfficeWorkload =~ "MicrosoftTeams"
| where Operation =~ "TeamDeleted"
| summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated),
DeletedTeams = make_set(TeamName) by UserId
| where array_length(DeletedTeams) > max_delete_count
| extend timestamp = StartTime, AccountCustomEntity = UserId
```

Anexo I – **Código da aplicação *Get-SentinelAlerts2***


```
{
    "definition": {
        "$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-
06-01/workflowdefinition.json#",
        "actions": {
            "For_each": {
                "actions": {
                    "Compose_count": {
                        "inputs": "@variables('count')",
                        "runAfter": {},
                        "type": "Compose"
                    },
                    "Enviar_um_e-mail_(V2)": {
                        "inputs": {
                            "body": {
                                "Body": "<p>@{variables('event')}</p>",
                                "Subject": "@triggerBody()?['AlertDisplayName']",
                                "To": censored
                            },
                            "host": {
                                "connection": {
                                    "name":
"@parameters('$connections')['office365_1']['connectionId']"
                                }
                            },
                            "method": "post",
                            "path": "/v2/Mail"
                        },
                        "runAfter": {
                            "Set_Event_String_2": [
                                "Succeeded"
                            ]
                        },
                        "type": "ApiConnection"
                    },
                    "Get_event_from_query": {
                        "inputs": {
                            "code": "var text =
workflowContext.actions.Run_query_and_list_results.outputs.body.value;\r\nvar count =
workflowContext.actions.Compose_count.outputs;\r\n\r\nreturn text[count]"
                        },
                        "runAfter": {
                            "Compose_count": [
                                "Succeeded"
                            ]
                        },
                        "type": "JavaScriptCode"
                    },
                    "Increment_count": {
                        "inputs": {
                            "name": "count",
                            "value": 1
                        },
                        "runAfter": {
                            "Get_event_from_query": [
                                "Succeeded"
                            ]
                        },
                        "type": "IncrementVariable"
                    },
                    "Parse_JSON_3": {
                        "inputs": {
                            "content": "@outputs('Get_event_from_query')?['body']",
                            "schema": {
                                "properties": {
                                    "body": {
                                        "properties": {
                                            "value": {
                                                "items": {
```

```
"properties": {
    "AccessList": {
        "type": "string"
    },
    "AccessMask": {
        "type": "string"
    },
    "AccessReason": {
        "type": "string"
    },
    "Account": {
        "type": "string"
    },
    "AccountCustomEntity": {
        "type": "string"
    },
    "AccountDomain": {
        "type": "string"
    },
    "AccountExpires": {
        "type": "string"
    },
    "AccountName": {
        "type": "string"
    },
    "AccountSessionIdentifier": {
        "type": "string"
    },
    "AccountType": {
        "type": "string"
    },
    "Activity": {
        "type": "string"
    },
    "AdditionalInfo": {
        "type": "string"
    },
    "AdditionalInfo2": {
        "type": "string"
    },
    "AllowedToDelegateTo": {
        "type": "string"
    },
    "Attributes": {
        "type": "string"
    },
    "AuditPolicyChanges": {
        "type": "string"
    },
    "AuditsDiscarded": {},
    "AuthenticationLevel": {},
    "AuthenticationPackageName": {
        "type": "string"
    },
    "AuthenticationProvider": {
        "type": "string"
    },
    "AuthenticationServer": {
        "type": "string"
    },
    "AuthenticationService": {},
    "AuthenticationType": {
        "type": "string"
    },
    "AzureDeploymentID": {
        "type": "string"
    },
    "AzureTableName": {
        "type": "string"
    },
    "CACertificateHash": {
        "type": "string"
    },
    "CAPublicKeyHash": {
```

```
        "type": "string"
    },
    "CalledStationID": {
        "type": "string"
    },
    "CallerProcessId": {
        "type": "string"
    },
    "CallerProcessName": {
        "type": "string"
    },
    "CallingStationID": {
        "type": "string"
    },
    "CategoryId": {
        "type": "string"
    },
    "CertificateDatabaseHash": {
        "type": "string"
    },
    "Channel": {
        "type": "string"
    },
    "ClassId": {
        "type": "string"
    },
    "ClassName": {
        "type": "string"
    },
    "ClientAddress": {
        "type": "string"
    },
    "ClientIPAddress": {
        "type": "string"
    },
    "ClientName": {
        "type": "string"
    },
    "CommandLine": {
        "type": "string"
    },
    "CompatibleIds": {
        "type": "string"
    },
    "Computer": {
        "type": "string"
    },
    "DCDNSName": {
        "type": "string"
    },
    "DeviceDescription": {
        "type": "string"
    },
    "DeviceId": {
        "type": "string"
    },
    "DisplayName": {
        "type": "string"
    },
    "Disposition": {
        "type": "string"
    },
    "DomainBehaviorVersion": {
        "type": "string"
    },
    "DomainName": {
        "type": "string"
    },
    "DomainPolicyChanged": {
        "type": "string"
    },
    "DomainSid": {
        "type": "string"
    },
```

```
"EAPType": {
    "type": "string"
},
"ElevatedToken": {
    "type": "string"
},
"ErrorCode": {},
"EventData": {
    "type": "string"
},
"EventID": {
    "type": "integer"
},
"EventOriginId": {
    "type": "string"
},
"EventSourceName": {
    "type": "string"
},
"ExtendedQuarantineState": {
    "type": "string"
},
"FailureReason": {
    "type": "string"
},
"FileHash": {
    "type": "string"
},
"FilePath": {
    "type": "string"
},
"FilePathNoUser": {
    "type": "string"
},
"Filter": {
    "type": "string"
},
"ForceLogoff": {
    "type": "string"
},
"Fqbn": {
    "type": "string"
},
"FullyQualifiedSubjectMachineName": {
    "type": "string"
},
"FullyQualifiedSubjectUserName": {
    "type": "string"
},
"GroupMembership": {
    "type": "string"
},
"HandleId": {
    "type": "string"
},
"HardwareIds": {
    "type": "string"
},
"HomeDirectory": {
    "type": "string"
},
"HomePath": {
    "type": "string"
},
"HostCustomEntity": {
    "type": "string"
},
"IPCustomEntity": {
    "type": "string"
},
"ImpersonationLevel": {
    "type": "string"
},
"InterfaceUuid": {
```

```
        "type": "string"
    },
    "IpAddress": {
        "type": "string"
    },
    "IpPort": {
        "type": "string"
    },
    "KeyLength": {
        "type": "integer"
    },
    "Level": {
        "type": "string"
    },
    "LmPackageName": {
        "type": "string"
    },
    "LocationInformation": {
        "type": "string"
    },
    "LockoutDuration": {
        "type": "string"
    },
    "LockoutObservationWindow": {
        "type": "string"
    },
    "LockoutThreshold": {
        "type": "string"
    },
    "LoggingResult": {
        "type": "string"
    },
    "LogonGuid": {
        "type": "string"
    },
    "LogonHours": {
        "type": "string"
    },
    "LogonID": {
        "type": "string"
    },
    "LogonProcessName": {
        "type": "string"
    },
    "LogonType": {
        "type": "integer"
    },
    "LogonTypeName": {
        "type": "string"
    },
    "MG": {
        "type": "string"
    },
    "MachineAccountQuota": {
        "type": "string"
    },
    "MachineInventory": {
        "type": "string"
    },
    "MachineLogon": {
        "type": "string"
    },
    "ManagementGroupName": {
        "type": "string"
    },
    "MandatoryLabel": {
        "type": "string"
    },
    "MaxPasswordAge": {
        "type": "string"
    },
    "MemberName": {
        "type": "string"
    },
```

```
"MemberSid": {
    "type": "string"
},
"MinPasswordAge": {
    "type": "string"
},
"MinPasswordLength": {
    "type": "string"
},
"MixedDomainMode": {
    "type": "string"
},
"NASIPv4Address": {
    "type": "string"
},
"NASIPv6Address": {
    "type": "string"
},
"NASIdentifier": {
    "type": "string"
},
"NASPort": {
    "type": "string"
},
"NASPortType": {
    "type": "string"
},
"NetworkPolicyName": {
    "type": "string"
},
"NewDate": {
    "type": "string"
},
"NewMaxUsers": {
    "type": "string"
},
"NewProcessId": {
    "type": "string"
},
"NewProcessName": {
    "type": "string"
},
"NewRemark": {
    "type": "string"
},
"NewShareFlags": {
    "type": "string"
},
"NewTime": {
    "type": "string"
},
"NewUacValue": {
    "type": "string"
},
"NewValue": {
    "type": "string"
},
"NewValueType": {
    "type": "string"
},
"ObjectName": {
    "type": "string"
},
"ObjectServer": {
    "type": "string"
},
"ObjectType": {
    "type": "string"
},
"ObjectValueName": {
    "type": "string"
},
"OemInformation": {
    "type": "string"
```

```
},
"OldMaxUsers": {
    "type": "string"
},
"OldRemark": {
    "type": "string"
},
"OldShareFlags": {
    "type": "string"
},
"OldUacValue": {
    "type": "string"
},
"OldValue": {
    "type": "string"
},
"OldValueType": {
    "type": "string"
},
"OperationType": {
    "type": "string"
},
"PackageName": {
    "type": "string"
},
"ParentProcessName": {
    "type": "string"
},
"PartitionKey": {
    "type": "string"
},
"PasswordHistoryLength": {
    "type": "string"
},
"PasswordLastSet": {
    "type": "string"
},
"PasswordProperties": {
    "type": "string"
},
"PreviousDate": {
    "type": "string"
},
"PreviousTime": {
    "type": "string"
},
"PrimaryGroupId": {
    "type": "string"
},
"PrivateKeyUsageCount": {
    "type": "string"
},
"PrivilegeList": {
    "type": "string"
},
"Process": {
    "type": "string"
},
"ProcessId": {
    "type": "string"
},
"ProcessName": {
    "type": "string"
},
"ProfilePath": {
    "type": "string"
},
"Properties": {
    "type": "string"
},
"ProtocolSequence": {
    "type": "string"
},
"ProxyPolicyName": {
```

```
        "type": "string"
    },
    "QuarantineHelpURL": {
        "type": "string"
    },
    "QuarantineSessionID": {
        "type": "string"
    },
    "QuarantineSessionIdentifier": {
        "type": "string"
    },
    "QuarantineState": {
        "type": "string"
    },
    "QuarantineSystemHealthResult": {
        "type": "string"
    },
    "RelativeTargetName": {
        "type": "string"
    },
    "RemoteIpAddress": {
        "type": "string"
    },
    "RemotePort": {
        "type": "string"
    },
    "RequestId": {
        "type": "string"
    },
    "Requester": {
        "type": "string"
    },
    "RestrictedAdminMode": {
        "type": "string"
    },
    "RowKey": {
        "type": "string"
    },
    "RowsDeleted": {
        "type": "string"
    },
    "SamAccountName": {
        "type": "string"
    },
    "ScriptPath": {
        "type": "string"
    },
    "SecurityDescriptor": {
        "type": "string"
    },
    "ServiceAccount": {
        "type": "string"
    },
    "ServiceFileName": {
        "type": "string"
    },
    "ServiceName": {
        "type": "string"
    },
    "ServiceStartType": {},
    "ServiceType": {
        "type": "string"
    },
    "SessionName": {
        "type": "string"
    },
    "ShareLocalPath": {
        "type": "string"
    },
    "ShareName": {
        "type": "string"
    },
    "SidHistory": {
        "type": "string"
```

```
},
"SourceComputerId": {
    "type": "string"
},
"SourceSystem": {
    "type": "string"
},
"Status": {
    "type": "string"
},
"StorageAccount": {
    "type": "string"
},
"SubStatus": {
    "type": "string"
},
"SubcategoryGuid": {
    "type": "string"
},
"SubcategoryId": {
    "type": "string"
},
"Subject": {
    "type": "string"
},
"SubjectAccount": {
    "type": "string"
},
"SubjectDomainName": {
    "type": "string"
},
"SubjectKeyIdentifier": {
    "type": "string"
},
"SubjectLogonId": {
    "type": "string"
},
"SubjectMachineName": {
    "type": "string"
},
"SubjectMachineSID": {
    "type": "string"
},
"SubjectUserName": {
    "type": "string"
},
"SubjectUserSid": {
    "type": "string"
},
"TableId": {
    "type": "string"
},
"TargetAccount": {
    "type": "string"
},
"TargetDomainName": {
    "type": "string"
},
"TargetInfo": {
    "type": "string"
},
"TargetLinkedLogonId": {
    "type": "string"
},
"TargetLogonGuid": {
    "type": "string"
},
"TargetLogonId": {
    "type": "string"
},
"TargetOutboundDomainName": {
    "type": "string"
},
"TargetOutboundUserName": {
```

```
        "type": "string"
    },
    "TargetServerName": {
        "type": "string"
    },
    "TargetSid": {
        "type": "string"
    },
    "TargetUser": {
        "type": "string"
    },
    "TargetUserName": {
        "type": "string"
    },
    "TargetUserSid": {
        "type": "string"
    },
    "Task": {
        "type": "integer"
    },
    "TemplateContent": {
        "type": "string"
    },
    "TemplateDSObjectFQDN": {
        "type": "string"
    },
    "TemplateInternalName": {
        "type": "string"
    },
    "TemplateOID": {
        "type": "string"
    },
    "TemplateSchemaVersion": {
        "type": "string"
    },
    "TemplateVersion": {
        "type": "string"
    },
    "TenantId": {
        "type": "string"
    },
    "TimeCollected": {
        "type": "string"
    },
    "TimeGenerated": {
        "type": "string"
    },
    "TokenElevationType": {
        "type": "string"
    },
    "TransmittedServices": {
        "type": "string"
    },
    "Type": {
        "type": "string"
    },
    "UserAccountControl": {
        "type": "string"
    },
    "UserParameters": {
        "type": "string"
    },
    "UserPrincipalName": {
        "type": "string"
    },
    "UserWorkstations": {
        "type": "string"
    },
    "VendorIds": {
        "type": "string"
    },
    "VirtualAccount": {
        "type": "string"
    },
```

```
        "Workstation": {
            "type": "string"
        },
        "WorkstationName": {
            "type": "string"
        },
        "_ResourceId": {
            "type": "string"
        },
        "timestamp": {
            "type": "string"
        }
    },
    "required": [
        "TenantId",
        "TimeGenerated",
        "SourceSystem",
        "Account",
        "AccountType",
        "Computer",
        "EventSourceName",
        "Channel",
        "Task",
        "Level",
        "EventData",
        "EventID",
        "Activity",
        "PartitionKey",
        "RowKey",
        "StorageAccount",
        "AzureDeploymentID",
        "AzureTableName",
        "AccessList",
        "AccessMask",
        "AccessReason",
        "AccountDomain",
        "AccountExpires",
        "AccountName",
        "AccountSessionIdentifier",
        "AdditionalInfo",
        "AdditionalInfo2",
        "AllowedToDelegateTo",
        "Attributes",
        "AuditPolicyChanges",
        "AuditsDiscarded",
        "AuthenticationLevel",
        "AuthenticationPackageName",
        "AuthenticationProvider",
        "AuthenticationServer",
        "AuthenticationService",
        "AuthenticationType",
        "CACertificateHash",
        "CalledStationID",
        "CallerProcessId",
        "CallerProcessName",
        "CallingStationID",
        "CAPublicKeyHash",
        "CategoryId",
        "CertificateDatabaseHash",
        "ClassId",
        "ClassName",
        "ClientAddress",
        "ClientIPAddress",
        "ClientName",
        "CommandLine",
        "CompatibleIds",
        "DCDNSName",
        "DeviceDescription",
        "DeviceId",
        "DisplayName",
        "Disposition",
        "DomainBehaviorVersion",
        "DomainName",
        "DomainPolicyChanged",
```

```
"DomainSid",
"EAPType",
"ElevatedToken",
"ErrorCode",
"ExtendedQuarantineState",
"FailureReason",
"FileHash",
"FilePath",
"FilePathNoUser",
"Filter",
"ForceLogoff",
"Fqbn",
"FullyQualifiedSubjectMachineName",
"FullyQualifiedSubjectUserName",
"GroupMembership",
"HandleId",
"HardwareIds",
"HomeDirectory",
"HomePath",
"ImpersonationLevel",
"InterfaceUuid",
"IpAddress",
"IpPort",
"KeyLength",
"LmPackageName",
"LocationInformation",
"LockoutDuration",
"LockoutObservationWindow",
"LockoutThreshold",
"LoggingResult",
"LogonGuid",
"LogonHours",
"LogonID",
"LogonProcessName",
"LogonType",
"LogonTypeName",
"MachineAccountQuota",
"MachineInventory",
"MachineLogon",
"MandatoryLabel",
"MaxPasswordAge",
"MemberName",
"MemberSid",
"MinPasswordAge",
"MinPasswordLength",
"MixedDomainMode",
"NASIdentifier",
"NASIPv4Address",
"NASIPv6Address",
"NASPort",
"NASPortType",
"NetworkPolicyName",
"NewDate",
"NewMaxUsers",
"NewProcessId",
"NewProcessName",
"NewRemark",
"NewShareFlags",
"NewTime",
"NewUacValue",
"NewValue",
"NewValueType",
"ObjectName",
"ObjectServer",
"ObjectType",
"ObjectValueName",
"OemInformation",
"OldMaxUsers",
"OldRemark",
"OldShareFlags",
"OldUacValue",
"OldValue",
"OldValueType",
"OperationType",
```

```
"PackageName",
"ParentProcessName",
"PasswordHistoryLength",
"PasswordLastSet",
"PasswordProperties",
"PreviousDate",
"PreviousTime",
"PrimaryGroupId",
"PrivateKeyUsageCount",
"PrivilegeList",
"Process",
"ProcessId",
"ProcessName",
"Properties",
"ProfilePath",
"ProtocolSequence",
"ProxyPolicyName",
"QuarantineHelpURL",
"QuarantineSessionID",
"QuarantineSessionIdentifier",
"QuarantineState",
"QuarantineSystemHealthResult",
"RelativeTargetName",
"RemoteIpAddress",
"RemotePort",
"Requester",
"RequestId",
"RestrictedAdminMode",
"RowsDeleted",
"SamAccountName",
"ScriptPath",
"SecurityDescriptor",
"ServiceAccount",
"ServiceFileName",
"ServiceName",
"ServiceStartType",
"ServiceType",
"SessionName",
"ShareLocalPath",
"ShareName",
"SidHistory",
"Status",
"SubjectAccount",
"SubcategoryGuid",
"SubcategoryId",
"Subject",
"SubjectDomainName",
"SubjectKeyIdentifier",
"SubjectLogonId",
"SubjectMachineName",
"SubjectMachineSID",
"SubjectUserName",
"SubjectUserSid",
"SubStatus",
"TableId",
"TargetAccount",
"TargetDomainName",
"TargetInfo",
"TargetLinkedLogonId",
"TargetLogonGuid",
"TargetLogonId",
"TargetOutboundDomainName",
"TargetOutboundUserName",
"TargetServerName",
"TargetSid",
"TargetUser",
"TargetUserName",
"TargetUserSid",
"TemplateContent",
"TemplateDSObjectFQDN",
"TemplateInternalName",
"TemplateOID",
"TemplateSchemaVersion",
"TemplateVersion",
```

```
                            "TokenElevationType",
                            "TransmittedServices",
                            "UserAccountControl",
                            "UserParameters",
                            "UserPrincipalName",
                            "UserWorkstations",
                            "VirtualAccount",
                            "VendorIds",
                            "Workstation",
                            "WorkstationName",
                            "SourceComputerId",
                            "EventOriginId",
                            "MG",
                            "TimeCollected",
                            "ManagementGroupName",
                            "Type",
                            "_ResourceId",
                            "timestamp",
                            "AccountCustomEntity",
                            "HostCustomEntity",
                            "IPCustomEntity"
                        ],
                        "type": "object"
                    },
                    "type": "array"
                }
            },
            "type": "object"
        },
        "headers": {
            "properties": {
                "Cache-Control": {
                    "type": "string"
                },
                "Content-Length": {
                    "type": "string"
                },
                "Content-Type": {
                    "type": "string"
                },
                "Date": {
                    "type": "string"
                },
                "Expires": {
                    "type": "string"
                },
                "Pragma": {
                    "type": "string"
                },
                "Set-Cookie": {
                    "type": "string"
                },
                "Strict-Transport-Security": {
                    "type": "string"
                },
                "Timing-Allow-Origin": {
                    "type": "string"
                },
                "Transfer-Encoding": {
                    "type": "string"
                },
                "Vary": {
                    "type": "string"
                },
                "X-Content-Type-Options": {
                    "type": "string"
                },
                "X-Frame-Options": {
                    "type": "string"
                },
                "x-ms-apihub-cached-response": {
                    "type": "string"
                },
                "x-ms-request-id": {
```

```
                                        "type": "string"
                                    }
                                },
                                "type": "object"
                            },
                            "statusCode": {
                                "type": "integer"
                            }
                        }
                    }
                },
                "runAfter": {
                    "Increment_count": [
                        "Succeeded"
                    ]
                },
                "type": "ParseJson"
            },
            "Parser": {
                "inputs": {
                    "body": {
                        "Request": {
                            "EventDescription": "@triggerBody()?['Description']",
                            "EventDisplayName": "@triggerBody()?['AlertDisplayName']",
                            "EventSeverity": "@triggerBody()?['Severity']",
                            "EventString": "@variables('event')",
                            "EventSystemId": "@triggerBody()?['TimeGenerated']",
                            "EventTimeGenerated": "@triggerBody()?['TimeGenerated']",
                            "EventType": "@triggerBody()?['AlertType']"
                        }
                    },
                    "host": {
                        "triggerName": "manual",
                        "workflow": {
                            "id":
"/subscriptions/censored/Microsoft.Logic/workflows/Parser"
                        }
                    }
                },
                "runAfter": {
                    "Set_Event_String_": [
                        "Succeeded"
                    ]
                },
                "type": "Workflow"
            },
            "Set_Event_String_": {
                "inputs": {
                    "name": "event",
                    "value": "@{json(string(body('Parse_JSON_3')))}"
                },
                "runAfter": {
                    "Parse_JSON_3": [
                        "Succeeded"
                    ]
                },
                "type": "SetVariable"
            },
            "Set_Event_String_2": {
                "inputs": {
                    "name": "event",
                    "value": "@{body('Parser')}"
                },
                "runAfter": {
                    "Parser": [
                        "Succeeded"
                    ]
                },
                "type": "SetVariable"
            }
        },
        "foreach": "@body('Run_query_and_list_results')?['value']",
        "runAfter": {
            "Initialize_Event_String": [
```

```
                "Succeeded"
            ]
        },
        "runtimeConfiguration": {
            "concurrency": {
                "repetitions": 1
            }
        },
        "type": "Foreach"
    },
    "Initialize_Event_String": {
        "inputs": {
            "variables": [
                {
                    "name": "event",
                    "type": "string"
                }
            ]
        },
        "runAfter": {
            "Initialize_count": [
                "Succeeded"
            ]
        },
        "type": "InitializeVariable"
    },
    "Initialize_count": {
        "inputs": {
            "variables": [
                {
                    "name": "count",
                    "type": "integer",
                    "value": 0
                }
            ]
        },
        "runAfter": {
            "Run_query_and_list_results": [
                "Succeeded"
            ]
        },
        "type": "InitializeVariable"
    },
    "Parse_JSON": {
        "inputs": {
            "content": "@triggerBody()?['ExtendedProperties']",
            "schema": {
                "Query": {
                    "type": "string"
                }
            }
        },
        "runAfter": {},
        "type": "ParseJson"
    },
    "Parse_JSON_2": {
        "inputs": {
            "content": "@body('Parse_JSON')",
            "schema": {
                "properties": {
                    "body": {
                        "properties": {
                            "Query": {
                                "type": "string"
                            },
                            "Query End Time UTC": {
                                "type": "string"
                            },
                            "Query Period": {
                                "type": "string"
                            },
                            "Query Results Aggregation Kind": {
                                "type": "string"
                            },
```

```
                                    "Query Start Time UTC": {
                                        "type": "string"
                                    },
                                    "Search Query Results Overall Count": {
                                        "type": "string"
                                    },
                                    "Trigger Operator": {
                                        "type": "string"
                                    },
                                    "Trigger Threshold": {
                                        "type": "string"
                                    }
                                },
                                "type": "object"
                            }
                        },
                        "type": "object"
                    }
                },
                "runAfter": {
                    "Parse_JSON": [
                        "Succeeded"
                    ]
                },
                "type": "ParseJson"
            },
            "Run_query_and_list_results": {
                "inputs": {
                    "body": "set query_datetimescope_column = \"TimeGenerated\";\nset
query_datetimescope_from = datetime(@{body('Parse_JSON_2')?['Query Start Time UTC']});\nset
query_datetimescope_to = datetime(@{body('Parse_JSON_2')?['Query End Time
UTC']});\n@{body('Parse_JSON_2')?['Query']} | sort by \"TimeGenerated\" desc ",
                    "host": {
                        "connection": {
                            "name":
"@parameters('$connections')['azuremonitorlogs_1']['connectionId']"
                        }
                    },
                    "method": "post",
                    "path": "/queryData",
                    "queries": {
                        "resourcegroups": censored
                        "resourcename": censored,
                        "resourcetype": censored,
                        "subscriptions": censored,
                        "timerange": "@{body('Parse_JSON_2')?['Query Period']}"
                    }
                },
                "runAfter": {
                    "Parse_JSON_2": [
                        "Succeeded"
                    ]
                },
                "type": "ApiConnection"
            },
            "Send_an_email_(V2)": {
                "inputs": {
                    "body": {
                        "Body": "<p>Erro no encaminhamento do Azure Sentinel para o IBM
QRadar.<br>\n<br>\nAlerta:<br>\n<br>\n@{variables('event')}</p>",
                        "Subject": "Erro: Alerta Sentinel não encaminhado -
@{triggerBody()?['AlertDisplayName']}",
                        "To": censored
                    },
                    "host": {
                        "connection": {
                            "name": "@parameters('$connections')['office365_1']['connectionId']"
                        }
                    },
                    "method": "post",
                    "path": "/v2/Mail"
                },
                "runAfter": {
                    "Send_event": [
```

```
                        "TimedOut",
                        "Skipped",
                        "Failed"
                    ]
                },
                "type": "ApiConnection"
            },
            "Send_event": {
                "inputs": {
                    "body": {
                        "ContentData": "@{base64(variables('event'))}"
                    },
                    "host": {
                        "connection": {
                            "name": "@parameters('$connections')['eventhubs']['connectionId']"
                        }
                    },
                    "method": "post",
                    "path": "/@{encodeURIComponent('sentinel')}/events"
                },
                "runAfter": {
                    "For_each": [
                        "Succeeded",
                        "TimedOut",
                        "Skipped",
                        "Failed"
                    ]
                },
                "type": "ApiConnection"
            }
        },
        "contentVersion": "1.0.0.0",
        "outputs": {},
        "parameters": {
            "$connections": {
                "defaultValue": {},
                "type": "Object"
            }
        },
        "triggers": {
            "When_a_response_to_an_Azure_Sentinel_alert_is_triggered": {
                "inputs": {
                    "body": {
                        "callback_url": "@{listCallbackUrl()}"
                    },
                    "host": {
                        "connection": {
                            "name":
"@parameters('$connections')['azuresentinel_1']['connectionId']"
                        }
                    },
                    "path": "/subscribe"
                },
                "type": "ApiConnectionWebhook"
            }
        }
    },
    "parameters": {
        "$connections": {
            "value": {
                "azuremonitorlogs_1": {
                    "connectionId":
"/subscriptions/censored/Microsoft.Web/connections/azuremonitorlogs-1",
                    "connectionName": "azuremonitorlogs-1",
                    "id": "/subscriptions/censored/azuremonitorlogs"
                },
                "azuresentinel_1": {
                    "connectionId": "/subscriptions/censored/azuresentinel-1",
                    "connectionName": "azuresentinel-1",
                    "id": "/subscriptions/censored/azuresentinel"
                },
                "eventhubs": {
                    "connectionId": "/subscriptions/censored/eventhubs",
                    "connectionName": "eventhubs",
```

```
                    "id": "/subscriptions/censored/eventhubs"
                },
                "office365_1": {
                    "connectionId": "/subscriptions/censored/office365-2",
                    "connectionName": "office365-2",
                    "id": "/subscriptions/censored/office365"
                }
            }
        }
    }
}
```

Anexo J – **Código da aplicação *Parser***

```
{
    "definition": {
        "$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-
06-01/workflowdefinition.json#",
        "actions": {
            "Compose_description_string": {
                "inputs": "@triggerBody()?['Request']?['EventDescription']",
                "runAfter": {},
                "type": "Compose"
            },
            "Compose_event_string": {
                "inputs": "@variables('eventString')",
                "runAfter": {
                    "Initialize_description": [
                        "Succeeded"
                    ],
                    "Initialize_displayName": [
                        "Succeeded"
                    ],
                    "Initialize_eventId": [
                        "Succeeded"
                    ],
                    "Initialize_eventString": [
                        "Succeeded"
                    ],
                    "Initialize_severity": [
                        "Succeeded"
                    ],
                    "Initialize_type": [
                        "Succeeded"
                    ]
                },
                "type": "Compose"
            },
            "Get_ProviderName": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n    var result = null;\r\n    if
(events.includes(\"ProviderName\")){\r\n        var providerNameAux =
events.split(\"ProviderName\\\":\\\"\");\r\n        var providerName =
providerNameAux[1].split(\"\\\"\");\r\n        if (providerName[0] == \"IPC\"){\r\n
result = \"Azure Active Directory Identity Protection\";\r\n        }\r\n        else if
(providerName[0] == \"OATP\"){\r\n            result = \"Office 365 Advanced Threat
Protection\";\r\n        }\r\n    }\r\n    return result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_VendorName": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"VendorName\")){\r\n        var vendorNameAux =
events.split(\"VendorName\\\":\\\"\");\r\n        var vendorName =
vendorNameAux[1].split(\"\\\"\");\r\n        result = vendorName[0]\r\n    }\r\n    return
result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Initialize_description": {
```

```
                "inputs": {
                    "variables": [
                        {
                            "name": "description",
                            "type": "string",
                            "value":
"@{substring(outputs('Compose_description_string'),0,sub(length(outputs('Compose_description_str
ing')),1))}"
                        }
                    ]
                },
                "runAfter": {
                    "Compose_description_string": [
                        "Succeeded"
                    ]
                },
                "type": "InitializeVariable"
            },
            "Initialize_displayName": {
                "inputs": {
                    "variables": [
                        {
                            "name": "displayName",
                            "type": "string",
                            "value": "@triggerBody()?['Request']?['EventDisplayName']"
                        }
                    ]
                },
                "runAfter": {},
                "type": "InitializeVariable"
            },
            "Initialize_eventId": {
                "inputs": {
                    "variables": [
                        {
                            "name": "eventId",
                            "type": "string",
                            "value": "@triggerBody()?['Request']?['EventSystemId']"
                        }
                    ]
                },
                "runAfter": {},
                "type": "InitializeVariable"
            },
            "Initialize_eventString": {
                "inputs": {
                    "variables": [
                        {
                            "name": "eventString",
                            "type": "string",
                            "value": "@triggerBody()?['Request']?['EventString']"
                        }
                    ]
                },
                "runAfter": {},
                "type": "InitializeVariable"
            },
            "Initialize_providerName": {
                "inputs": {
                    "variables": [
                        {
                            "name": "providerName",
                            "type": "string",
                            "value": "@{outputs('Get_ProviderName')?['body']}"
                        }
                    ]
                },
                "runAfter": {
                    "Get_ProviderName": [
                        "Succeeded"
                    ]
                },
                "type": "InitializeVariable"
            },
```

```
"Initialize_response": {
    "inputs": {
        "variables": [
            {
                "name": "response",
                "type": "string"
            }
        ]
    },
    "runAfter": {
        "Initialize_providerName": [
            "Succeeded"
        ],
        "Initialize_vendorName": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_severity": {
    "inputs": {
        "variables": [
            {
                "name": "severity",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventSeverity']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_type": {
    "inputs": {
        "variables": [
            {
                "name": "type",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventType']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_vendorName": {
    "inputs": {
        "variables": [
            {
                "name": "vendorName",
                "type": "string",
                "value": "@{outputs('Get_VendorName')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_VendorName": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Response": {
    "inputs": {
        "body": "@{variables('response')}\n",
        "statusCode": 200
    },
    "kind": "Http",
    "runAfter": {
        "Switch": [
            "Succeeded"
        ]
    },
    "type": "Response"
```

```
            },
            "Switch": {
                "cases": {
                    "Case_\"Azure_Active_Directory_Identity_Protection\"": {
                        "actions": {
                            "Compose": {
                                "inputs": "@{body('ParserIPC')}\nProvider Name:
@{variables('providerName')}\nVendor Name:
@{variables('vendorName')}\n\n@{variables('eventString')}",
                                "runAfter": {
                                    "ParserIPC": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "Compose"
                            },
                            "ParserIPC": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
                                            "EventSeverity": "@variables('severity')",
                                            "EventString": "@variables('eventString')",
                                            "EventType": "@variables('type')"
                                        }
                                    },
                                    "host": {
                                        "triggerName": "manual",
                                        "workflow": {
                                            "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC"
                                        }
                                    }
                                },
                                "runAfter": {},
                                "type": "Workflow"
                            },
                            "Set_response_\"IPC\"": {
                                "inputs": {
                                    "name": "response",
                                    "value": "@{outputs('Compose')}"
                                },
                                "runAfter": {
                                    "Compose": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "SetVariable"
                            }
                        },
                        "case": "Azure Active Directory Identity Protection"
                    },
                    "Case_\"Office_365_Advanced_Threat_Protection\"": {
                        "actions": {
                            "Compose_\"OATP\"": {
                                "inputs": "@{body('ParserOATP')}\nProvider Name:
@{variables('providerName')}\nVendor Name:
@{variables('vendorName')}\n\n@{variables('eventString')}",
                                "runAfter": {
                                    "ParserOATP": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "Compose"
                            },
                            "ParserOATP": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
```

```
                                        "EventSeverity": "@variables('severity')",
                                        "EventString": "@variables('eventString')",
                                        "EventType": "@variables('type')"
                                    }
                                },
                                "host": {
                                    "triggerName": "manual",
                                    "workflow": {
                                        "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserOATP"
                                    }
                                }
                            },
                            "runAfter": {},
                            "type": "Workflow"
                        },
                        "Set_response_\"OATP\"": {
                            "inputs": {
                                "name": "response",
                                "value": "@{outputs('Compose_\"OATP\"')}"
                            },
                            "runAfter": {
                                "Compose_\"OATP\"": [
                                    "Succeeded"
                                ]
                            },
                            "type": "SetVariable"
                        }
                    },
                    "case": "Office 365 Advanced Threat Protection"
                }
            },
            "default": {
                "actions": {
                    "Compose_\"Sentinel\"": {
                        "inputs":
"@{body('ParserSentinel')}\n\n@{variables('eventString')}",
                        "runAfter": {
                            "ParserSentinel": [
                                "Succeeded"
                            ]
                        },
                        "type": "Compose"
                    },
                    "ParserSentinel": {
                        "inputs": {
                            "body": {
                                "Request": {
                                    "EventDescription": "@variables('description')",
                                    "EventDisplayName": "@variables('displayName')",
                                    "EventID": "@variables('eventId')",
                                    "EventSeverity": "@variables('severity')",
                                    "EventString": "@variables('eventString')",
                                    "EventType": "@variables('type')"
                                }
                            },
                            "host": {
                                "triggerName": "manual",
                                "workflow": {
                                    "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserSentinel"
                                }
                            }
                        },
                        "runAfter": {},
                        "type": "Workflow"
                    },
                    "Set_response_\"Sentinel\"": {
                        "inputs": {
                            "name": "response",
                            "value": "@{outputs('Compose_\"Sentinel\"')}"
                        },
                        "runAfter": {
                            "Compose_\"Sentinel\"": [
```

```
                                "Succeeded"
                            ]
                        },
                        "type": "SetVariable"
                    }
                }
            },
            "expression": "@variables('providerName')",
            "runAfter": {
                "Initialize_response": [
                    "Succeeded"
                ]
            },
            "type": "Switch"
        }
    },
    "contentVersion": "1.0.0.0",
    "outputs": {},
    "parameters": {},
    "triggers": {
        "manual": {
            "inputs": {
                "schema": {
                    "properties": {
                        "Request": {
                            "properties": {
                                "EventDescription": {
                                    "type": "string"
                                },
                                "EventDisplayName": {
                                    "type": "string"
                                },
                                "EventSeverity": {
                                    "type": "string"
                                },
                                "EventString": {
                                    "type": "string"
                                },
                                "EventSystemId": {
                                    "type": "string"
                                },
                                "EventTimeGenerated": {
                                    "type": "string"
                                },
                                "EventType": {
                                    "type": "string"
                                }
                            },
                            "type": "object"
                        }
                    },
                    "type": "object"
                }
            },
            "kind": "Http",
            "type": "Request"
        }
    }
},
"parameters": {}
}
```

Anexo K – **Exemplo do código de uma aplicação de segunda camada de *parsing* – *ParserIPC***

```
{
    "definition": {
        "$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-
06-01/workflowdefinition.json#",
        "actions": {
            "Compose_event_string": {
                "inputs": "@variables('eventString')",
                "runAfter": {
                    "Initialize_description": [
                        "Succeeded"
                    ],
                    "Initialize_displayName": [
                        "Succeeded"
                    ],
                    "Initialize_eventId": [
                        "Succeeded"
                    ],
                    "Initialize_eventString": [
                        "Succeeded"
                    ],
                    "Initialize_severity": [
                        "Succeeded"
                    ],
                    "Initialize_type": [
                        "Succeeded"
                    ]
                },
                "type": "Compose"
            },
            "Get_tenant_Id": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   result = null;\r\n    if
(events.includes(\"TenantId\")){\r\n        var tenantIdAux =
events.split(\"TenantId\\\":\\\"\");\r\n        var tenantId =
tenantIdAux[1].split(\"\\\"\");\r\n        result = tenantId[0];\r\n    }\r\n    return result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_time_generated": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   result = null;\r\n    if
(events.includes(\"TimeGenerated\")){\r\n        var timeGeneratedAux =
events.split(\"TimeGenerated\\\":\\\"\");\r\n        var timeGenerated =
timeGeneratedAux[1].split(\"\\\"\");\r\n        result = timeGenerated[0].split(\"T\")[0] + \"
\" + timeGenerated[0].split(\"T\")[1].split(\".\")[0];\r\n    }\r\n    return result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Initialize_description": {
                "inputs": {
                    "variables": [
                        {
                            "name": "description",
```

```
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventDescription']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_displayName": {
    "inputs": {
        "variables": [
            {
                "name": "displayName",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventDisplayName']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_eventId": {
    "inputs": {
        "variables": [
            {
                "name": "eventId",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventID']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_eventString": {
    "inputs": {
        "variables": [
            {
                "name": "eventString",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventString']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_response": {
    "inputs": {
        "variables": [
            {
                "name": "response",
                "type": "string"
            }
        ]
    },
    "runAfter": {
        "Initialize_tenantId": [
            "Succeeded"
        ],
        "Initialize_timeGenerated": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_severity": {
    "inputs": {
        "variables": [
            {
                "name": "severity",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventSeverity']"
            }
```

```
            ]
        },
        "runAfter": {},
        "type": "InitializeVariable"
    },
    "Initialize_tenantId": {
        "inputs": {
            "variables": [
                {
                    "name": "tenantId",
                    "type": "string",
                    "value": "@{outputs('Get_tenant_Id')?['body']}"
                }
            ]
        },
        "runAfter": {
            "Get_tenant_Id": [
                "Succeeded"
            ]
        },
        "type": "InitializeVariable"
    },
    "Initialize_timeGenerated": {
        "inputs": {
            "variables": [
                {
                    "name": "timeGenerated",
                    "type": "string",
                    "value": "@{outputs('Get_time_generated')?['body']}"
                }
            ]
        },
        "runAfter": {
            "Get_time_generated": [
                "Succeeded"
            ]
        },
        "type": "InitializeVariable"
    },
    "Initialize_type": {
        "inputs": {
            "variables": [
                {
                    "name": "eventType",
                    "type": "string",
                    "value": "@triggerBody()?['Request']?['EventType']"
                }
            ]
        },
        "runAfter": {},
        "type": "InitializeVariable"
    },
    "Response": {
        "inputs": {
            "body": "@variables('response')",
            "statusCode": 200
        },
        "kind": "Http",
        "runAfter": {
            "Switch": [
                "Succeeded"
            ]
        },
        "type": "Response"
    },
    "Switch": {
        "cases": {
            "Case_\"Anomalous_Token\"": {
                "actions": {
                    "Parser_-_IPC_-_AnomalousToken": {
                        "inputs": {
                            "body": {
                                "Request": {
                                    "EventDescription": "@variables('description')",
```

```
                                        "EventDisplayName": "@variables('displayName')",
                                        "EventID": "@variables('eventId')",
                                        "EventSeverity": "@variables('severity')",
                                        "EventString": "@variables('eventString')",
                                        "EventTenantId": "@variables('tenantId')",
                                        "EventTimeGenerated": "@variables('timeGenerated')",
                                        "EventType": "@variables('eventType')"
                                    }
                                },
                                "host": {
                                    "triggerName": "manual",
                                    "workflow": {
                                        "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_AnomalousToken"
                                    }
                                }
                            },
                            "runAfter": {},
                            "type": "Workflow"
                        },
                        "Set_response_\"Anomalous_Token\"": {
                            "inputs": {
                                "name": "response",
                                "value": "@{body('Parser_-_IPC_-_AnomalousToken')}"
                            },
                            "runAfter": {
                                "Parser_-_IPC_-_AnomalousToken": [
                                    "Succeeded"
                                ]
                            },
                            "type": "SetVariable"
                        }
                    },
                    "case": "Anomalous Token"
                },
                "Case_\"Anonymous_IP_address\"": {
                    "actions": {
                        "Parser_-_IPC_-_AnonymousIP": {
                            "inputs": {
                                "body": {
                                    "Request": {
                                        "EventDescription": "@variables('description')",
                                        "EventDisplayName": "@variables('displayName')",
                                        "EventID": "@variables('eventId')",
                                        "EventSeverity": "@variables('severity')",
                                        "EventString": "@variables('eventString')",
                                        "EventTenantId": "@variables('tenantId')",
                                        "EventTimeGenerated": "@variables('timeGenerated')",
                                        "EventType": "@variables('eventType')"
                                    }
                                },
                                "host": {
                                    "triggerName": "manual",
                                    "workflow": {
                                        "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_AnonymousIP"
                                    }
                                }
                            },
                            "runAfter": {},
                            "type": "Workflow"
                        },
                        "Set_response_\"AnonymousIP\"": {
                            "inputs": {
                                "name": "response",
                                "value": "@{body('Parser_-_IPC_-_AnonymousIP')}"
                            },
                            "runAfter": {
                                "Parser_-_IPC_-_AnonymousIP": [
                                    "Succeeded"
                                ]
                            },
                            "type": "SetVariable"
                        }
```

```
                        },
                        "case": "Anonymous IP address"
                    },
                    "Case_\"Atypical_Travel\"": {
                        "actions": {
                            "Parser_-_IPC_-_AtypicalTravel": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
                                            "EventSeverity": "@variables('severity')",
                                            "EventString": "@variables('eventString')",
                                            "EventTenantId": "@variables('tenantId')",
                                            "EventTimeGenerated": "@variables('timeGenerated')",
                                            "EventType": "@variables('eventType')"
                                        }
                                    },
                                    "host": {
                                        "triggerName": "manual",
                                        "workflow": {
                                            "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_AtypicalTravel"
                                        }
                                    }
                                },
                                "runAfter": {},
                                "type": "Workflow"
                            },
                            "Set_response_\"AtypicalTravel\"": {
                                "inputs": {
                                    "name": "response",
                                    "value": "@{body('Parser_-_IPC_-_AtypicalTravel')}"
                                },
                                "runAfter": {
                                    "Parser_-_IPC_-_AtypicalTravel": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "SetVariable"
                            }
                        },
                        "case": "Atypical Travel"
                    },
                    "Case_\"Malicious_IP_address\"": {
                        "actions": {
                            "Parser_-IPC_-_MaliciousIP": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
                                            "EventSeverity": "@variables('severity')",
                                            "EventString": "@variables('eventString')",
                                            "EventTenantId": "@variables('tenantId')",
                                            "EventTimeGenerated": "@variables('timeGenerated')",
                                            "EventType": "@variables('eventType')"
                                        }
                                    },
                                    "host": {
                                        "triggerName": "manual",
                                        "workflow": {
                                            "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_MaliciousIP"
                                        }
                                    }
                                },
                                "runAfter": {},
                                "type": "Workflow"
                            },
                            "Set_response_\"MaliciousIP\"": {
                                "inputs": {
```

```
                                    "name": "response",
                                    "value": "@{body('Parser_-IPC_-_MaliciousIP')}"
                                },
                                "runAfter": {
                                    "Parser_-IPC_-_MaliciousIP": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "SetVariable"
                            }
                        },
                        "case": "Malicious IP"
                    },
                    "Case_\"Malware_linked_IP_address\"": {
                        "actions": {
                            "Parser_-_IPC_-_MalwareLinkedIP": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
                                            "EventSeverity": "@variables('severity')",
                                            "EventString": "@variables('eventString')",
                                            "EventTenantId": "@variables('tenantId')",
                                            "EventTimeGenerated": "@variables('timeGenerated')",
                                            "EventType": "@variables('eventType')"
                                        }
                                    },
                                    "host": {
                                        "triggerName": "manual",
                                        "workflow": {
                                            "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_MalwareLinkedIP"
                                        }
                                    }
                                },
                                "runAfter": {},
                                "type": "Workflow"
                            },
                            "Set_\"MalwareLinkedIP\"": {
                                "inputs": {
                                    "name": "response",
                                    "value": "@{body('Parser_-_IPC_-_MalwareLinkedIP')}"
                                },
                                "runAfter": {
                                    "Parser_-_IPC_-_MalwareLinkedIP": [
                                        "Succeeded"
                                    ]
                                },
                                "type": "SetVariable"
                            }
                        },
                        "case": "Malware linked IP address"
                    },
                    "Case_\"Password_Spray\"": {
                        "actions": {
                            "Parser_-_IPC_-_PasswordSpray": {
                                "inputs": {
                                    "body": {
                                        "Request": {
                                            "EventDescription": "@variables('description')",
                                            "EventDisplayName": "@variables('displayName')",
                                            "EventID": "@variables('eventId')",
                                            "EventSeverity": "@variables('severity')",
                                            "EventString": "@variables('eventString')",
                                            "EventTenantId": "@variables('tenantId')",
                                            "EventTimeGenerated": "@variables('timeGenerated')",
                                            "EventType": "@variables('eventType')"
                                        }
                                    },
                                    "host": {
                                        "triggerName": "manual",
                                        "workflow": {
```

```
                                        "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_PasswordSpray"
                                    }
                                }
                            },
                            "runAfter": {},
                            "type": "Workflow"
                        },
                        "Set_variable_\"PasswordSpray\"": {
                            "inputs": {
                                "name": "response",
                                "value": "@{body('Parser_-_IPC_-_PasswordSpray')}"
                            },
                            "runAfter": {
                                "Parser_-_IPC_-_PasswordSpray": [
                                    "Succeeded"
                                ]
                            },
                            "type": "SetVariable"
                        }
                    },
                    "case": "Password Spray"
                },
                "Case_\"Unfamiliar_sign-in_properties\"": {
                    "actions": {
                        "Parser_-_IPC_-_UnfamiliarSignIn": {
                            "inputs": {
                                "body": {
                                    "Request": {
                                        "EventDescription": "@variables('description')",
                                        "EventDisplayName": "@variables('displayName')",
                                        "EventID": "@variables('eventId')",
                                        "EventSeverity": "@variables('severity')",
                                        "EventString": "@variables('eventString')",
                                        "EventTenantId": "@variables('tenantId')",
                                        "EventTimeGenerated": "@variables('timeGenerated')",
                                        "EventType": "@variables('eventType')"
                                    }
                                },
                                "host": {
                                    "triggerName": "manual",
                                    "workflow": {
                                        "id":
"/subscriptions/censored/Microsoft.Logic/workflows/ParserIPC_UnfamiliarSignIn"
                                    }
                                }
                            },
                            "runAfter": {},
                            "type": "Workflow"
                        },
                        "Set_response_\"UnfamiliarSignInProperties\"": {
                            "inputs": {
                                "name": "response",
                                "value": "@{body('Parser_-_IPC_-_UnfamiliarSignIn')}"
                            },
                            "runAfter": {
                                "Parser_-_IPC_-_UnfamiliarSignIn": [
                                    "Succeeded"
                                ]
                            },
                            "type": "SetVariable"
                        }
                    },
                    "case": "Unfamiliar sign-in properties"
                }
            },
            "default": {
                "actions": {}
            },
            "expression": "@variables('displayName')",
            "runAfter": {
                "Initialize_response": [
                    "Succeeded"
                ]
```

```
                },
                "type": "Switch"
            }
        },
        "contentVersion": "1.0.0.0",
        "outputs": {},
        "parameters": {},
        "triggers": {
            "manual": {
                "inputs": {
                    "schema": {
                        "properties": {
                            "Request": {
                                "properties": {
                                    "EventDescription": {
                                        "type": "string"
                                    },
                                    "EventDisplayName": {
                                        "type": "string"
                                    },
                                    "EventID": {
                                        "type": "string"
                                    },
                                    "EventSeverity": {
                                        "type": "string"
                                    },
                                    "EventString": {
                                        "type": "string"
                                    },
                                    "EventType": {
                                        "type": "string"
                                    }
                                },
                                "type": "object"
                            }
                        },
                        "type": "object"
                    }
                },
                "kind": "Http",
                "type": "Request"
            }
        }
    },
    "parameters": {}
}
```

Anexo L – **Exemplo do código de uma aplicação de terceira camada de *parsing* – *ParserIPC_AtypicalTravel***

```
{
    "definition": {
        "$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-
06-01/workflowdefinition.json#",
        "actions": {
            "Compose": {
                "inputs": "Tenant Id: @{variables('tenantId')}\nTime Generated:
@{variables('timeGenerated')}\nEvent Name: @{variables('displayName')}\nEvent Type:
@{variables('type')}\nSeverity: @{variables('severity')}\nDescription:
@{variables('description')}\nDetailed Description: @{variables('detailDescription')}\n----------
-----------------------------------------------------\nUser Name:
@{variables('userName')}\nUser Account: @{variables('userAccount')}\nAlarm Sign-In Date-Time:
@{variables('timeGenerated')}\nCurrent IP Address: @{variables('currentIp')}\nCurrent Location:
@{variables('currentLocation')}\nPrevious IP Address: @{variables('previousIp')}\nPrevious
Location: @{variables('previousLocation')}\nPrevious Sign-In Date-Time:
@{variables('previousSigninDateTime')}\n-----------------------------------------------------------
--------\n",
                "runAfter": {
                    "Initialize_detailDescription": [
                        "Succeeded"
                    ],
                    "Initialize_previousIp": [
                        "Succeeded"
                    ],
                    "Initialize_previousLocation": [
                        "Succeeded"
                    ],
                    "Initialize_userAccount": [
                        "Succeeded"
                    ]
                },
                "type": "Compose"
            },
            "Compose_event_string": {
                "inputs": "@variables('eventString')",
                "runAfter": {
                    "Initialize_description": [
                        "Succeeded"
                    ],
                    "Initialize_displayName": [
                        "Succeeded"
                    ],
                    "Initialize_eventString": [
                        "Succeeded"
                    ],
                    "Initialize_severity": [
                        "Succeeded"
                    ],
                    "Initialize_tenantId": [
                        "Succeeded"
                    ],
                    "Initialize_timeGenerated": [
                        "Succeeded"
                    ],
                    "Initialize_type": [
                        "Succeeded"
                    ]
                },
                "type": "Compose"
            },
            "Get_current_IP_address": {
                "inputs": {
```

```
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Current IP Address\")){\r\n        var currentIpAddressAux =
events.split(\"Current IP Address\\\\\\\": \\\\\\\"\");\r\n        var currentIpAddress =
currentIpAddressAux[1].split(\"\\\\\");\r\n        result = currentIpAddress[0];\r\n    }\r\n
return result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_current_location": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Current Location\")){\r\n        var currentLocationAux =
events.split(\"Current Location\\\\\\\": \\\\\\\"\");\r\n        var currentLocation =
currentLocationAux[1].split(\"\\\\\");\r\n        result = currentLocation[0];\r\n    }\r\n
return result;"
                },
                "runAfter": {
                    "Compose_event_string": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_detailed_description": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Detail Description\")){\r\n        var detailDescriptionAux =
events.split(\"Detail Description\\\\\\\": \\\\\\\"\");\r\n        var detailDescription =
detailDescriptionAux[1].split(\"\\\"\");\r\n        result = detailDescription[0] +
detailDescription[1] + detailDescription[2];\r\n    }\r\n    return result;"
                },
                "runAfter": {
                    "Initialize_previousSigninDateTime": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_previous_IP_address": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Previous IP Address\")){\r\n        var previousIpAddressAux =
events.split(\"Previous IP Address\\\\\\\": \\\\\\\"\");\r\n        var previousIpAddress =
previousIpAddressAux[1].split(\"\\\\\");\r\n        result = previousIpAddress[0];\r\n    }\r\n
return result;"
                },
                "runAfter": {
                    "Initialize_currentIp": [
                        "Succeeded"
                    ]
                },
                "type": "JavaScriptCode"
            },
            "Get_previous_location": {
                "inputs": {
                    "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Previous Location\")){\r\n        var previousLocationAux =
events.split(\"Previous Location\\\\\\\": \\\\\\\"\");\r\n        var previousLocation =
previousLocationAux[1].split(\"\\\\\");\r\n        result = previousLocation[0];\r\n    }\r\n
return result;"
                },
                "runAfter": {
                    "Initialize_currentLocation": [
                        "Succeeded"
```

```
                ]
            },
            "type": "JavaScriptCode"
        },
        "Get_previous_signin_date_time": {
            "inputs": {
                "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"Previous Signin DateTime\")){\r\n        var previousSigninDateTimeAux =
events.split(\"Previous Signin DateTime\\\\\\\": \\\\\\\"\");\r\n        var
previousSigninDateTime = previousSigninDateTimeAux[1].split(\"\\\\\");\r\n        var horas =
previousSigninDateTime[0].split(\" \")[1];\r\n        var mes =
previousSigninDateTime[0].split(\" \")[0].split(\"/\")[0];\r\n        var dia =
previousSigninDateTime[0].split(\" \")[0].split(\"/\")[1];\r\n        var ano =
previousSigninDateTime[0].split(\" \")[0].split(\"/\")[2];\r\n        result = ano + \"-\" + mes
+ \"-\" + dia + \" \" + horas;\r\n    }\r\n    return result;"
            },
            "runAfter": {
                "Compose_event_string": [
                    "Succeeded"
                ]
            },
            "type": "JavaScriptCode"
        },
        "Get_time_generated": {
            "inputs": {
                "code": "var text = \"Hello world from \" +
workflowContext.workflow.name;\r\n\r\nreturn text;"
            },
            "runAfter": {
                "Compose_event_string": [
                    "Succeeded"
                ]
            },
            "type": "JavaScriptCode"
        },
        "Get_user_account": {
            "inputs": {
                "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"User Account\")){\r\n        var userAccountAux = events.split(\"User
Account\\\\\\\": \\\\\\\"\");\r\n        var userAccount =
userAccountAux[1].split(\"\\\\\");\r\n        result = userAccount[0];\r\n    }\r\n    return
result;"
            },
            "runAfter": {
                "Initialize_userName": [
                    "Succeeded"
                ]
            },
            "type": "JavaScriptCode"
        },
        "Get_user_name": {
            "inputs": {
                "code": "   events =
workflowContext.actions.Compose_event_string.outputs;\r\n   var result = null;\r\n    if
(events.includes(\"User Name\")){\r\n        var userNameAux = events.split(\"User Name\\\\\\\":
\\\\\\\"\");\r\n        var userName = userNameAux[1].split(\"\\\\\");\r\n        result =
userName[0];\r\n    }\r\n    return result;"
            },
            "runAfter": {
                "Compose_event_string": [
                    "Succeeded"
                ]
            },
            "type": "JavaScriptCode"
        },
        "Initialize_currentIp": {
            "inputs": {
                "variables": [
                    {
                        "name": "currentIp",
                        "type": "string",
                        "value": "@{outputs('Get_current_IP_address')?['body']}"
```

```
            }
        ]
    },
    "runAfter": {
        "Get_current_IP_address": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_currentLocation": {
    "inputs": {
        "variables": [
            {
                "name": "currentLocation",
                "type": "string",
                "value": "@{outputs('Get_current_location')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_current_location": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_description": {
    "inputs": {
        "variables": [
            {
                "name": "description",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventDescription']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_detailDescription": {
    "inputs": {
        "variables": [
            {
                "name": "detailDescription",
                "type": "string",
                "value": "@{outputs('Get_detailed_description')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_detailed_description": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_displayName": {
    "inputs": {
        "variables": [
            {
                "name": "displayName",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventDisplayName']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_eventString": {
    "inputs": {
        "variables": [
            {
```

```
                "name": "eventString",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventString']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_previousIp": {
    "inputs": {
        "variables": [
            {
                "name": "previousIp",
                "type": "string",
                "value": "@{outputs('Get_previous_IP_address')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_previous_IP_address": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_previousLocation": {
    "inputs": {
        "variables": [
            {
                "name": "previousLocation",
                "type": "string",
                "value": "@{outputs('Get_previous_location')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_previous_location": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_previousSigninDateTime": {
    "inputs": {
        "variables": [
            {
                "name": "previousSigninDateTime",
                "type": "string",
                "value": "@{outputs('Get_previous_signin_date_time')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_previous_signin_date_time": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_response": {
    "inputs": {
        "variables": [
            {
                "name": "response",
                "type": "string",
                "value": "@{outputs('Compose')}"
            }
        ]
    },
    "runAfter": {
        "Compose": [
            "Succeeded"
        ]
```

```
    },
    "type": "InitializeVariable"
},
"Initialize_severity": {
    "inputs": {
        "variables": [
            {
                "name": "severity",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventSeverity']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_tenantId": {
    "inputs": {
        "variables": [
            {
                "name": "tenantId",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventTenantId']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_timeGenerated": {
    "inputs": {
        "variables": [
            {
                "name": "timeGenerated",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventTimeGenerated']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_type": {
    "inputs": {
        "variables": [
            {
                "name": "type",
                "type": "string",
                "value": "@triggerBody()?['Request']?['EventType']"
            }
        ]
    },
    "runAfter": {},
    "type": "InitializeVariable"
},
"Initialize_userAccount": {
    "inputs": {
        "variables": [
            {
                "name": "userAccount",
                "type": "string",
                "value": "@{outputs('Get_user_account')?['body']}"
            }
        ]
    },
    "runAfter": {
        "Get_user_account": [
            "Succeeded"
        ]
    },
    "type": "InitializeVariable"
},
"Initialize_userName": {
    "inputs": {
```

```
            "variables": [
                {
                    "name": "userName",
                    "type": "string",
                    "value": "@{outputs('Get_user_name')?['body']}"
                }
            ]
        },
        "runAfter": {
            "Get_user_name": [
                "Succeeded"
            ]
        },
        "type": "InitializeVariable"
    },
    "Response": {
        "inputs": {
            "body": "@variables('response')",
            "statusCode": 200
        },
        "kind": "Http",
        "runAfter": {
            "Initialize_response": [
                "Succeeded"
            ]
        },
        "type": "Response"
    }
},
"contentVersion": "1.0.0.0",
"outputs": {},
"parameters": {},
"triggers": {
    "manual": {
        "inputs": {
            "schema": {
                "properties": {
                    "Request": {
                        "properties": {
                            "EventDescription": {
                                "type": "string"
                            },
                            "EventDisplayName": {
                                "type": "string"
                            },
                            "EventID": {
                                "type": "string"
                            },
                            "EventSeverity": {
                                "type": "string"
                            },
                            "EventString": {
                                "type": "string"
                            },
                            "EventTenantId": {
                                "type": "string"
                            },
                            "EventTimeGenerated": {
                                "type": "string"
                            },
                            "EventType": {
                                "type": "string"
                            }
                        },
                        "type": "object"
                    }
                },
                "type": "object"
            }
        },
        "kind": "Http",
        "type": "Request"
    }
}
```

```
    },
    "parameters": {}
}
```

Anexo L – **Código da *Azure Function eventForwarding***

```
module.exports = function (context, myEventHubMessage) {
    context.log(`Alerta: ${myEventHubMessage}`);
    try{
        var syslog = require("syslog-client");
    } catch (e) {
        throw e;
    }
    var SYSLOG_SERVER = GetEnvironmentVariable("SYSLOG_SERVER");
    var SYSLOG_PROTOCOL;
    if (GetEnvironmentVariable("SYSLOG_PROTOCOL")=="TCP") {
        SYSLOG_PROTOCOL = syslog.Transport.Tcp;
    }
    else {
        SYSLOG_PROTOCOL = syslog.Transport.Udp;
    }
    var SYSLOG_HOSTNAME;
    if (GetEnvironmentVariable("SYSLOG_HOSTNAME")==null) {
        SYSLOG_HOSTNAME = "AzureSentinel"
    }
    else {
        SYSLOG_HOSTNAME = GetEnvironmentVariable("SYSLOG_HOSTNAME");
    }
    var SYSLOG_PORT = GetEnvironmentVariable("SYSLOG_PORT");
    var SYSLOG_FACILITY;
    if (GetEnvironmentVariable("SYSLOG_FACILITY") == null) {
        SYSLOG_FACILITY = syslog.Facility.Local0;
    }
    else {
        SYSLOG_FACILITY = syslog.Facility.Local0;
    }
    var options = {
        syslogHostname: SYSLOG_HOSTNAME,
        transport: SYSLOG_PROTOCOL,
        port: SYSLOG_PORT,
        facility: SYSLOG_FACILITY
    };
    var client = syslog.createClient(SYSLOG_SERVER, options);
    myEventHubMessage.forEach((message, index)=>{
        if(typeof message === 'string'){
```

```
            msg = JSON.stringify(JSON.parse(JSON.stringify(message)));
            client.log(msg, options, function(error) {
                if (error) {
                    context.log("Erro ao enviar alerta.");
                    context.bindings.eventForwardingOutput = "Erro ao enviar alerta.\n" + "Alerta:\n" + msg;
                    context.log(error);
                }
            });
        }
    });
    context.log("Alerta(s) enviado com sucesso.");
    context.bindings.eventForwardingOutput = "Alerta(s) enviado com sucesso.\n\n" + "Alerta:\n" +
myEventHubMessage;
    context.done();
};
```

Anexo M – **Exemplo da vista detalhada de um alerta *Azure* no *IBM QRadar – Atypical Travel***

Informações de Evento

Nome do Evento	Atypical Travel						
Categoria de Baixo Nível	Endereço Suspeito						
Descrição do evento	Sign-in from an atypical location based on the user's recent sign-in. This risk event type identifies two sign-ins originating from geographically distant locations, where at least one of the locations may also be atypical for the user, given past behavior. Among several other factors, this machine learning algorithm takes into account the time between the two sign-ins and the time it would have taken for the user to travel from the first location to the second, indicating that a different user is using the same credentials. The algorithm ignores obvious \\\false positives\\\ contributing to the impossible travel conditions, such as VPNs and locations regularly used by other users in the organization. The system has an initial learning period of the earliest of 14 days or 10 logins, during which it learns a new user's sign-in behavior. For more information - https://go.microsoft.com/fwlink/?linkid=2016517\						
Magnitude	(5)	**Relevância**	3	**Gravidade**	6	**Credibilidade**	5
Nome de Usuário	@telecom.pt						
Horário de Início	25/09/2021, 06:26:28	**Horário de Armazenamento**	25/09/2021, 06:26:28	**Horário da Origem de log**	25/09/2021, 06:26:25		
AccountName (customizado)	Luis						
Previous Sign-In Date-Time (customizado)	25/09/2021, 00:57:07						
Previous User IP (customizado)	176.						
Previous User Location (customizado)	, Lisboa, PT						
Product (customizado)	Azure Active Directory Identity Protection						
Time Generated (customizado)	25/09/2021, 06:24:55						
User Agent (customizado)							
User Location (customizado)	Val-D'oise, FR						

Informações de Origem e Destino

IP de origem	89.	**IP de destino**	0.0.0.1
Nome do Ativo-fonte	N/D	**Nome do Ativo de Destino**	N/D
Porta de Origem	0	**Porta de Destino**	0
IP de origem pré-NAT		**IP de destino pré-NAT**	
Porta de Origem pré-NAT	0	**Porta de Destino pré-NAT**	0
IP de origem pós-NAT		**IP de destino pós-NAT**	
Porta de Origem pós-NAT	0	**Porta de Destino pós-NAT**	0
IPv6 de Origem	0:0:0:0:0:0:0:0	**IPv6 de Destino**	0:0:0:0:0:0:0:0
MAC de Origem	00:00:00:00:00:00	**MAC de Destino**	00:00:00:00:00:00