Aplicação centralizada para a gestão e fusão de logs

Mestrado em Cibersegurança e Informática Forense

Ana Carolina Silvério Dias

Leiria, julho de 2020

Aplicação centralizada para a gestão e fusão de logs

Mestrado em Cibersegurança e Informática Forense

Ana Carolina Silvério Dias

Trabalho de Projeto realizado sob a orientação do Professor Doutor Mário João Gonçalves Antunes, Coordenador do Mestrado de Cibersegurança e Informática Forense da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria

Leiria, julho de 2020

Originalidade e Direitos de Autor

O presente relatório de projeto é original, elaborado unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para o elaborar.

Dedicatória

À minha avó Maria Rosa, aos meus pais Paula e Adelino e ao meu irmão Eduardo.

Agradecimentos

Em primeiro lugar, um agradecimento especial aos meus pais por todo o apoio, motivação e incentivo demonstrado ao longo deste curso.

Ao professor orientador, Mário Antunes, pelo acompanhamento, apoio, dedicação, disponibilidade e conhecimentos transmitidos durante o decorrer deste projeto, que sem ele não seria possível concretizar-se.

A todos os professores do mestrado de Cibersegurança e Informática Forense, um agradecimento por todos os conhecimentos, experiências e motivações transmitidas.

Por fim, mas não menos importante agradeço aos meus colegas de curso por toda a disponibilidade e entreajuda demonstrada durante o curso, principalmente no decorrer do 1º ano.

Resumo

De uma forma geral, os sistemas computacionais modernos, sejam eles de apoio e gestão da infraestrutura ou aplicacionais, produzem informação relevante sobre a sua atividade operacional, que é mantida em ficheiros específicos designados por registos ou ficheiros de *log*.

Estes ficheiros são normalmente produzidos em formato de texto e têm uma formatação especifica para cada aplicação, guardando apenas a informação relevante da sua atividade. Por exemplo, o ficheiro de log que regista a atividade do sistema Linux, mantido pelo serviço `rsyslogd`, que guarda informação referente ao estado atual ou histórico dos diversos serviços que são realizados no servidor.

Um desafio constante que é colocado aos administradores de sistemas, no âmbito das suas funções, consiste em analisar cuidadosamente estes ficheiros, de forma a identificarem potenciais problemas decorrentes da atividade do sistema operativo e/ou dos servidores aplicacionais. Um bom exemplo de utilização deste tipo de registo, ocorre ao nível de segurança através de tentativas de acesso não permitido a servidores.

Devido à quantidade de informação gerada, esta análise é dificultada pelo volume de informação detalhada que os ficheiros de *log* guardam e pela necessidade constante de análise da informação disponibilizada. Neste caso não há mecanismos nativos que permitam a fusão da informação constante dos vários *logs*, através da sua agregação e sumarização num ficheiro de *log* agregador, ou "meta-log". Tal facto permitiria que o administrador de sistemas se focasse apenas na informação relevante, que consta do "meta-log". Outra dificuldade importante consiste na necessidade de automatizar o tratamento dos *logs*. Só assim será possível lidar, em tempo útil e com objetividade, com a quantidade de informação que é gerada continuamente.

Há várias aplicações que auxiliam a administração de sistemas nestas tarefas, embora nem sempre estejam ao alcance do administrador de sistemas. Algumas aplicações são comerciais e não são flexíveis ao ponto de processar qualquer tipo de *log.* Há ainda aplicações que estão acessíveis em *open source*, mas são difíceis de configurar e, por vezes, apresentam falta de suporte profissional, e não possibilitam a flexibilidade necessária na automatização e sumarização das mensagens de *log.* Por exemplo, a aplicação Nagios é uma solução de monitorização *open source*, que necessita de algumas configurações para obtermos

informação centralizada sobre a infraestrutura, enquanto que a solução comercial *System Center Operation Manager* (SCOM) requere um esforço menor de configuração na monitorização da infraestrutura, redes e aplicações.

Neste projeto desenvolveu-se uma solução de fusão de *logs* alimentada pelos sistemas operativos e aplicações. Desta forma pretende-se contribuir para a melhoria no tratamento automático dos ficheiros de *log*. A solução proposta é composta por cinco blocos principais, designadamente o pré-processamento dos ficheiros de *log*; o processamento de ficheiros de *log* de acordo com um *template* pré-definido; a produção de um meta-log com a informação mais importante que foi selecionada durante a fase de pré-processamento; a correlação das entradas do ficheiro resultante com as entradas dos ficheiros de *log* originais e, por último, a visualização do "meta-log" resultante e dos correspondentes registos originais.

A aplicação foi desenvolvida em PERL, onde se realizaram testes de aceitação da solução e, comprovadas no presente projeto. A prova de conceito foi realizada com ficheiros de *log* provenientes do sistema operativo Windows e aplicações, tendo sido utilizados 430 *logs* com criticidade dos tipos "erro" e "informativo". Com os resultados obtidos conclui-se que a solução proposta permite reduzir o número de campos analisados nos logs, assim como o tamanho do ficheiro utilizado para análise.

Palavras-chave: fusão de *logs*, administração segura de sistemas, monitorização

Abstract

Modern computer systems, in general, produce relevant information, whether infrastructure or application, about their operational activity, which is kept in specific files called records or log files.

This kind of files are usually produced in text format and have a specific format for each application, retaining only the relevant information of the activity. For example, a specific log file that records the activity of an Linux system, maintained by the rsyslogd service, stores information regarding the actual or historical status of the various services that are run on the server.

A constant challenge that system administrators face daily, is to carefully analyze these files, in order to identify potential problems arising from regular activities of the operating system and / or application servers. A good example of using this kind of record occurs at the security level by identifying access attempts prevention ~~access~~ to servers.

Derived from the amount of information generated, this analysis is hampered by the volume of detailed information that log files stores and the continues need to analyze the information available on those files. In this case, there are no native mechanism that allow the merging of the information contained in multiple log files, through their aggregation and summarization in a single aggregating log file, or “meta-log”. This would allow the system administrator to focus only on the relevant information, which appears in the “meta-log”. Another important difficulty is the need to automate the handling of logs. Only then will it be possible to deal, in a timely manner and with objectivity, with the amount of information that is generated continuously.

There are several applications that assist systems administration in these tasks, although they are not always available to the system administrator. Some applications are commercial and are not flexible enough to process all type of log. There are still applications that are accessible in open source, but they are difficult to configure and sometimes lack professional support, and do not provide the necessary flexibility in the automation and summarization of log messages. For example, Nagios is an open source monitoring solution, which requires some customization to obtain centralized information about the infrastructure, while the

System Center Operation Manager (SCOM), a commercial solution, requires less configuration effort for infrastructure monitoring, networks and applications.

This project aims to develop a solution for merging logs powered by operating systems and applications. It intends to contribute to improve the automatic handling of log records. A proposed solution consists of five main blocks, namely ~~of~~ log record pre-processing; log files processing according to a predefined model; meta-log production with the most important information selected during a pre-processing phase; a correlation between the entries in the specified file with the entries in the original log records and, finally, the response from the resulting “meta-log” and the corresponding original records.

An application in PERL was developed to perform acceptance tests of the solution, proven in this dissertation. A proof of concept was performed with system and application log records in Windows, in summary 430 logs were used with error and informative critical. The results obtained allow us to conclude that the proposed solution allows to reduce the number of fields in the logs, such as the size of their text file.

Keywords: log fusion, secure system administration, monitoring

Índice

Índice de Figuras

Índice de Tabelas

Lista de siglas e acrónimos

AD DS	*Active Directory Domain Services*
AJAX	*Asynchronous Javascript and* XML
API	*Application Programming Interfaces*
AWS	*Amazon Web Services*
BI	*Business Intelligence*
CPU	*Central Processing Unit*
CGI	*Common Gateway Interface*
CLM	*Centralized Log Management*
COM	*Component Object Model*
DAX	*Data Analysis Expressions*
DB	*Database*
ESE	*Extensible Storage Engine*
FTP	*File Transfer Protocol*
GB	*Gigabyte*
ID	Identificador
IDS	*Intrusions Detectetion System*
IE	*Internet Explorer*
IP	*Internet Protocol*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
JSON	*JavaScript Object Notation*
KB	*Kilobyte*
LTS	*Long Term Support*
MS-DOS	*Microsoft Disk Operating System*
PCR	*Platform Configuration Register*
PERL	*Practical Extraction and Reporting Language*
PNG	*Portable Network Graphics*
PPI	*Personal Identifiable Information*
NAS	*Network Attached Storage*
NIST	*National Institute of Standards and Technology*

RAM	*Random Access Memory*
RFC	*Request for Comments*
RGPD	Regulamento Geral de Proteção de Dados
RPC	*Remote Procedure Call*
SaaS	*Software as a Service*
SAN	*Storage Area Network*
SCP	*Secure Copy Protocol*
SEM	*Security Event Manager*
SFTP	*Secure File Transfer Protocol*
SGBD	Sistema de Gestão de Bases de Dados
SIEM	*Security Information and Event Management*
SIM	*Security Information Management*
SMS	*Short Message Service*
SNMP	*Simple Network Management Protocol*
SSH	*Secure Shell*
TI	Tecnologias de Informação
TPM	*Trusted Platform Module*
TXT	Ficheiro de texto
USB	*Universal Serial Bus*
WinSCP	*Windows Secure Copy*
W3C	*World Wide Web Consortium*
XML	*eXtensible Markup Language*

1. Introdução

Atualmente, a inovação em TI é um fator-chave para o sucesso das organizações. Com essa inovação, as empresas beneficiam de maior produtividade, flexibilidade e lucros. Todos os equipamentos ligados na infraestrutura de rede de uma organização, incluindo os dispositivos móveis, produzem um vasto conjunto de registos de atividade, o que leva à criação de informação variada sobre a sua atividade. Estes registos são produzidos pelo próprio sistema operativo, bem como pelas aplicações que cada equipamento tem em execução, como aplicações de suporte ao negócio, serviços do sistema operativo, entre outros. O registo de atividade é guardado em ficheiros designados por *logs* [1].

Estes ficheiros de *log*, são uma parte fundamental e critica de qualquer sistema, pois permitem aos técnicos obterem informações sobre as atividades em execução ou já realizadas pelo sistema. É igualmente possível verificar as ocorrências e as suas causas. Devido ao aumento de equipamento com ligação à rede (como servidores, *routers*, *switches*) ou aplicações (como sistemas de bases de dados, *firewalls*, IDS), gerir os *logs* torna-se uma tarefa cada vez mais complicada para os administradores de sistemas. A sua importância é fundamental na identificação do crescente aumento das ameaças a bases de dados, redes sociais, lojas virtuais, entre outros [2]. Este aumento do volume e dimensão dos *logs*, também influência o desempenho dos discos no processo de armazenamento destes registos, que poderão atingir a sua capacidade mais rapidamente do que a desejada ou esperada [3].

Genericamente, os *logs* podem ser classificados em três grupos principais [2]:

- Registo das operações dos utilizadores que acedem e realizam operações nos sistemas (clientes e/ou parceiros das organizações);
- Registos das operações de quem planeia, implementa, testa e gere o sistema (colaboradores das organizações);
- Registos da camada de infraestrutura (*hardware* e *software*).

Uma possível solução para o armazenamento e gestão da diversidade de *logs* passa pela sua centralização, especialmente os relacionados com os sistemas mais críticos das organizações, como *logs* de auditoria dos servidores, *firewalls*, IDS, entre outros [1]. Esta centralização de *logs* permite armazenar dados considerados de crescimento rápido e que

ocupam muito espaço em disco. A centralização dos *logs*, durante o processo de armazenamento, disponibiliza a consulta de informações existentes nos *logs* de forma eficiente, não atrasando a execução de atividades envolvidas durante o processo de armazenamento dos *logs* [2].

O mercado disponibiliza várias soluções, comerciais e *open source*, de centralização e análise automática de *logs* que ajuda os administradores de sistemas a agregar os diversos *logs* num único local, tornando esta análise mais simples e apelativa. Estas soluções também disponibilizam *dashboards* e relatórios com gráficos e métricas estatísticas e em caso de uma ocorrência potencialmente anómala, por vezes é possível programar o envio automático de uma notificação para o e-mail ou por mensagem escrita ou de voz para o telemóvel do administrador de sistemas. No entanto, o custo de aquisição destas soluções não é o mais apelativo para as organizações e em alguns casos, são de difícil configuração e requerem conhecimentos prévios nem sempre detidos pelos administradores de sistemas.

Genericamente, nas mensagens de *log* encontram-se informações relacionadas com a resposta às seguintes questões sobre uma determinada ocorrência: **como** (de que forma aconteceu); **quem** (nome do utilizador); **quando** (dia e hora); **onde** (nome da máquina ou processo) **o quê** (comando ou atividade que originou a ocorrência); e **resultado** (descrição do problema) [4].

Embora esta informação exista nos ficheiros de *log*, algumas mensagens são caracterizadas pelo seu texto em bruto, de difícil leitura, interpretação ou pesquisa, não estruturado ou com uma estrutura pouco intuitiva [5]. Também a falta de priorização das mensagens nalguns ficheiros de *log*, relacionada com a sua "severidade", é um desafio para os administradores de sistemas e constitui um obstáculo a uma análise mais rápida e estruturada da informação aí existente. Por exemplo, esta falta de informação origina a que, caso ocorra um problema critico na infraestrutura, serviço ou aplicação, o administrador de sistemas ou programador não terá a perceção da ocorrência e, só terá conhecimento do sucedido quando ocorrem falhas em algum dos componentes da infraestrutura (por exemplo, serviço ou aplicação).

Por outro lado, os eventos registados nos ficheiros de *log* também são úteis para identificar tendências operacionais e apoiar em investigações internas da organização, incluindo a auditoria e análise forense [6], através de monitorização.

A monitorização é um fator chave na produtividade das organizações, pois em caso de falhas (por exemplo, falha total de rede ou lentidão provocada por *malware* não detetado) no acesso a servidores ou aplicações, os colaboradores podem não desempenhar as suas tarefas de forma adequada, diminuindo a sua produtividade. Também os problemas nos sistemas operativos, aplicações ou comunicações, influenciam o correto desempenho das tarefas dos colaboradores. Estes problemas podem ser diminuídos com a implementação de alertas na monitorização de *logs* recolhidos e, posteriormente de notificações (por exemplo, por email ou SMS) sobre os mesmos [7]. Por exemplo, no caso da falha do serviço de Apache pode ser despoletado um alerta na consola de monitorização e simultaneamente, enviado um email ao administrador de sistema.

Os *logs* também são um dos recursos fundamentais para uma monitorização de segurança eficiente e consciente através de orientações que podem originar conclusões em caso de ataque ou incidente de segurança. A monitorização permite identificar o que aconteceu, qual o impacto, o que podemos fazer de seguida ou o que pode ser alterado para prevenir esta situação. Também permite identificar ocorrências de falhas nos sistemas e detetar um ataque em curso [8]. Por exemplo, detetar a ocorrência de um *login* com sucesso a um servidor de um utilizador externo à organização.

Neste projeto pretende-se contribuir para o desenvolvimento de uma solução que possa melhorar a leitura do conteúdo da informação de ficheiros de *logs* de formatos heterogéneos, através da sua fusão num formato normalizado contribuindo assim para a melhoria na análise e usabilidade destas mensagens pelo administrador de sistemas.

1.1. Objetivos

Neste projeto de mestrado pretendeu-se analisar a estrutura e campos relevantes de diversos *logs*, desenvolver um protótipo para automatizar o tratamento dos campos considerados relevantes nos *logs* e convertê-los posteriormente para uma estrutura que possa ser interpretada através de um *browser*.

Para tal, foram definidos os seguintes objetivos para este projeto:

- Definir uma arquitetura para o tratamento automático das mensagens de *log* recolhidas pelos sistemas operativos e aplicações. De entre os vários *logs* que poderão ser recolhidos, podemos identificar os *logs* dos servidores aplicacionais

(*web*, FTP, SSH, entre outros) e os *logs* do funcionamento do sistema operativo (por exemplo, em Linux, o `syslog` e, em Windows, o *Event Viewer*);

- Definir um *template* para cada um dos tipos de *logs* considerados para processamento, recorrendo a uma linguagem de especificação. No desenvolvimento do protótipo foi utilizado o formato XML. O formato do ficheiro de log resultante, que designaremos neste documento por "meta-log", inclui os campos considerados relevantes pelo administrador de sistemas para a análise global da atividade do sistema;
- Definir um *template* para o formato do meta-log, recorrendo igualmente a uma linguagem de especificação. Também neste caso foi utilizado o formato XML;
- Desenvolver uma aplicação piloto que teste a arquitetura definida, designadamente que implemente o pré-processamento e conversão dos ficheiros de *log* e a consequente transformação para o formato do meta-log;
- Desenvolver uma interface *web* de visualização em Power BI para melhorar a usabilidade no tratamento dos ficheiros de *log*, através de funcionalidades de relacionamento entre as mensagens constantes do meta-log e as correspondentes mensagens originais;
- Avaliar o desempenho da solução, através da análise dos padrões redundantes que são rejeitados, da usabilidade da solução e da sua aplicabilidade em cenários reais.

Os contributos deste projeto são os seguintes:

- Aplicação piloto para o tratamento de *logs* originais e posterior conversão para formato XML, disponível na plataforma GitHub (https://github.com/Silveriodias/logfusion-at-perl);
- Aplicação *web* desenvolvida em Power BI para a visualização dos resultados obtidos e correlacionamento com as mensagens originais, permitindo uma análise rápida e simples dos *logs*;
- Especificação dos formatos dos ficheiros de *log* das aplicações e do meta-log, definidos em XML;
- Testes de validação e análise critica aos resultados obtidos à utilização da arquitetura proposta em cenários reais.

1.2. Organização do documento

O documento está dividido em oito capítulos, sendo o primeiro constituído pela motivação, objetivos e contributos com este projeto e a organização do documento.

O segundo capítulo descreve os conceitos fundamentais para auxiliar a leitura e interpretação deste relatório.

O terceiro capítulo apresenta os trabalhos relacionados com o tratamento automático de *logs* e aplicações existentes no mercado, que tenham tido influência no desenvolvimento deste projeto.

O quarto capítulo ilustra a arquitetura proposta na implementação deste projeto e os campos relevantes nos diversos *logs*.

O quinto capítulo descreve o processo de desenvolvimento deste projeto. Este capítulo inclui os principais algoritmos, os *scripts* utilizados no desenvolvimento e a integração com a tecnologia de Power BI.

O sexto capítulo descreve e justifica os testes realizados e resultados obtidos.

O sétimo capítulo apresenta a análise critica dos resultados obtidos, vantagens e pontos fracos da utilização de uma solução deste género, tendo em conta os resultados obtidos.

Por fim, o oitavo capítulo apresenta as conclusões deste projeto, onde são comentados os objetivos propostos inicialmente e o trabalho futuro.

2. Conceitos fundamentais

Este capítulo descreve as tecnologias de sistemas operativos utilizados e as linguagens de *scripting* em que se baseou o desenvolvimento do projeto. É também enquadrado o conceito de ficheiro de *log*, os formatos mais comuns, em que circunstâncias as mensagens são geradas, quais as funcionalidades e a sua importância.

2.1. Sistemas Operativos

O Ubuntu foi usado no desenvolvimento deste projeto. Trata-se de uma distribuição Linux *open source* e gratuita, baseada em Debian e encontra-se disponível em três edições (*Desktop*, Servidor e *Core*). Atualmente, este sistema operativo encontra-se na versão 20.04 e é lançada nova versão a cada seis meses [9]. Foi usada a versão Ubuntu 18.04.3 LTS, (instalado na versão servidor) da máquina virtual de desenvolvimento do projeto. Trata-se de uma versão estável com atualizações frequentes, assegurando o suporte por cinco anos.

O Windows 10, é um sistema operativo cliente mais atual da Microsoft. Esta versão é precedente ao Windows 8.1 foi lançada em outubro de 2014 e distingue-se pelas funcionalidades de múltiplos ambientes de trabalho [10]. O *Windows Event* é um componente dos sistemas operativos da Microsoft e permite aos administradores de sistemas e utilizadores analisarem os *logs* da máquina de forma local ou remota [11]. Este componente foi usado no projeto para consultar e extrair os *logs* para a realização dos testes.

2.2. Linguagens de scripting e de especificação

2.2.1. PERL

O PERL, *Practical Extraction and Reporting Language*, é uma linguagem de programação que pode ser executada em diversas plataformas. Foi criada por Larry Wall e permite desenvolver programas em sistemas operativos Unix, MS-DOS, Windows, Macintosh, entre outros. Pode ainda ser usada como CGI para a *web*, para o processamento de formulários na *web* e ainda para automatização de tarefas de administração de sistemas operativos Unix [12]. É uma linguagem destinada ao desenvolvimento de *scripts*, que podem ser executados em qualquer sistema operativo.

O PERL foi a linguagem escolhida para a implementação dos *scripts* necessários para a criação do meta-log e do *template*, A escolha foi a simplicidade no desenvolvimento, o facto de haver experiência prévia e a possibilidade de ser executada em vários sistemas operativos. Destaca-se ainda pelas seguintes características principais [13]:

- **Rapidez**, rápido a executar determinadas tarefas e mais poderoso;
- ***One-Linear***, disponibiliza atalhos que permitem escrever *scripts* de forma rápida;
- **Expressões regulares**, suporta nativamente a programação com expressões regulares;
- **Relevância**, permite aos programadores integrar de forma fácil, diversas *interfaces* ou componentes de terceiros que não são compatíveis entre si.

Atualmente, o PERL ainda é utilizado porque possui uma vasta comunidade. No entanto, o Python está a ganhar popularidade a nível mundial devido à sua quantidade de pacotes compatíveis com todos os sistemas operativos [14].

2.2.2. XML

O XML, *eXtensible Markup Language*, é uma estrutura formal para ficheiros, que permite organizar os dados (por exemplo texto, bases de dados) de forma hierárquica e extensível, recorrendo a marcadores (como outras linguagens de *markup*, como o HTML). É uma especificação recomendada pela W3C e, pode ser utilizada para partilhar informações entre vários servidores e aplicações, bem como para reutilização de código [15].

2.2.3. JSON

O JSON, *JavaScript Object Notation*, é um modelo de armazenamento e transmissão de informação no formato de texto. É uma estrutura utilizada por aplicações *web* devido à sua capacidade de estruturar a informação, de forma mais compacta que na estrutura XML, tornado o *parsing* da informação mais rápido [16].

2.2.4. Tomada de Decisão

As linguagens de especificação de XML (secção 2.2.2) e JSON (secção 2.2.3) foram consideradas e analisadas na implementação do *template* do projeto. Têm características próprias, sendo algumas semelhantes em ambas e encontram-se presentes nas Tabela 2.1 [16]. Após a análise e tendo em atenção o objetivo do projeto, decidiu-se utilizar o XML na

linguagem de especificação do *template*, pois a implementação da sua estrutura é mais simples e aos marcadores (*tags*) que utiliza permitem a interpretação do conteúdo de forma simples e intuitiva.

Tabela 2.1 – Semelhanças e diferenças nos modelos XML e JSON.

Semelhanças	Diferenças
Apresentam informações no formato texto.	JSON não é uma linguagem de marcação. Não possui *tag* de início nem de fim.
Natureza auto-descritiva.	JSON apresenta as informações de forma mais compacta.
Capacidade de representar informação complexa, difícil de representar no formato de tabela, em objetos compostos (objetos dentro de objetos), relações de hierarquia, *arrays*, dados ausentes, entre outros.	JSON não permite executar instruções de processamento.
Possibilidade de transportar informações para aplicações AJAX.	Possibilidade de transportar informações para aplicações AJAX.
Utilizam padrões para representação de dados. XML é um padrão W3C e JSON formalizado na RFC4627.	JSON destina-se a troca de informações, enquanto XML possui mais aplicações. Por exemplo: Atualmente, existem bases de dados completas armazenados em XML e estruturados em SGBD's XML nativo.
Independentes de linguagem. Os dados representados em XML e JSON podem ser acedidos por qualquer linguagem de programação, através de API's específicas.	

2.3. Visualização e análise de logs

O Power BI é um serviço de análise de negócios da Microsoft, que utiliza o modelo SaaS, lançado em julho de 2015. Este serviço tem o objetivo fornecer visualizações interativas e está orientado para os recursos de *Business Intelligence* (BI), utilizando uma interface simples para que os utilizadores finais criem os seus próprios relatórios e *dashboards*. O Power BI, é composto por vários componentes (como por exemplo, Power BI *Desktop*, Power BI *Service*, Power BI *Gateway* e Power BI *Report Server*) que permitem a integração

com várias soluções [17] [18]. O Power BI foi utilizado no projeto, mais concretamente no módulo de visualização das ocorrências do "meta-log" e das correspondentes entradas nos ficheiros originais.

2.4.Noção de log

Os *logs* são registos de atividades, normalmente formatadas em texto, que são geradas pelas ações realizadas por uma aplicação, sistemas operativos ou interações dos utilizadores [19]. Estes registos ocorrem nos computadores pessoais, servidores, dipositivos móveis ou equipamentos de rede e podem ser guardados em ficheiros, memória RAM do computador e bases de dados, sendo um fator chave na deteção e resolução de problemas.

Baseado nas informações guardadas é possível detetar o uso indevido de um componente da infraestrutura (como por exemplo, um servidor), ataques, exploração de vulnerabilidades, realizar auditorias (através das ações realizadas pelos utilizadores) e detetar problemas no *hardware*, programas e serviços ou políticas instaladas. Torna-se assim possível aos administradores de sistemas tomarem medidas preventivas a problemas de maior ou minimizar os riscos, caso ocorram [20].

Segundo o NIST, a gestão de *logs* define-se como "*the process for generating, transmitting, storing, analyzing, and disposing of computer security log data*", ou seja, o que é necessário registar e por quanto tempo é necessário reter esses registos [21].

Os *logs* são gerados sempre que existem interações reais com o sistema operativo, aplicação ou serviço, guardando-os em ficheiros, durante um intervalo de tempo e podem ser analisados quando necessário.

No entanto, existe uma grande quantidade de aplicações distintas, tornando-se difícil definir um padrão único para todas as mensagens de *logs*. Apesar de cada aplicação possui um formato diferente para estes registos geralmente, existem informações básicas que estão presentes na maioria dos *logs*, como a data, hora e fuso horário que ocorreu uma determinada atividade ocorreu, endereço IP de origem da atividade, dados do que foi enviado, alterado ou removido e o resultado (sucesso ou insucesso) da atividade. Também registam os diferentes níveis de severidade (critico, erro, alerta, informativo, *debug*, entre outros) consoante a aplicação ou sistema operativo com o intuito de corrigir erros ou verificar o estado de saúde [20].

2.5. Caracterização de ficheiros de *log*

Os testes e resultados obtidos neste projeto, descritos respetivamente nos capitulos 6 e 0, baseados na arquitetura proposta no subcapítulo 4.1, foram realizados com eventos de *log* recolhidos pelo *Event Viewer* do Windows. Estes ficheiros de *log* têm as seguintes características principais:

- **`Microsoft-Windows-Kernel-General`**: regista informações sobre o sistema operativo Windows, como por exemplo quando é que o computador foi iniciado [22];
- **`Service Control Manager`**: consiste num servidor RPC, para que os programas de configuração e controle de serviços possam manipular serviços em máquinas remotas. Este serviço é inicializado com o sistema operativo [23];
- **`Trusted Platform Module`**: ocorre quando é executada uma operação de extensão do PCR do TPM [24];
- **`Microsoft-Windows-UserModePowerService`**: disponibiliza dados à interface de gestão de métricas para serviços e aplicações. Esta interface é designada por Windows Performance Monitor [25];
- **`Microsoft-Windows-Kernel-Power`**: informa que o sistema foi desligado de forma incorreta/inesperada [26];
- **`Microsoft-Windows-WindowsUpdateClient`**: é executado no computador cliente para verificar a existência de atualizações [27];
- **`Microsoft-Windows-DistributedCOM`**: é uma plataforma independente, distribuída e orientada a objetos, que permite implementar componentes de *software* binários que podem interagir em si [28];
- **`Microsoft-Windows-Power-Troubleshooter`**: permite corrigir problemas relacionados com as opções e configurações de energia, planos de energia e configurações de *screen timeout* nos sistemas operativos cliente [29];
- **`Microsoft-Windows-Winlogon`**: fornece suporte interativo para *logon* [30];
- **`Microsoft-Windows-Kernel-Boot`**: informa que o sistema foi desligado de forma inesperada [26];
- **`EventLog`**: regista eventos de várias fontes e armazena-os num único local, possíveis de serem consultados [31].

- **`ESENT`**: refere-se ao tempo de execução do *Extensible Storage Engine* (parte de uma base de dados transacional, semelhante ao SQLite). O ESE é utilizado por vários serviços e aplicações da Microsoft para guardar dados (como exemplo, o Microsoft *Edge*, para guardar o histórico das páginas visitadas pelo utilizador) [32].

2.6. Gestão de *logs*

Com o crescimento da infraestrutura das organizações, a quantidade de *logs* produzidos também cresce, tornando-se necessária a gestão dos mesmos. Existem vários métodos automatizados para gerir os *logs*:

- **Soluções de *scripting* desenvolvidas à medida**, que consistem na implementação de *scripts* que recolhem informação de desempenho ou ações ocorridas em servidores ou aplicações. Este tema será abordado com detalhe na subcapítulo 3.1;
- **Soluções de monitorização**, refere-se à centralização de várias fontes de dados e permite a análise do desempenho de vários servidores, equipamentos de rede ou aplicações, em tempo real;
- **Soluções de SIEM**, também centraliza várias fontes de dados e permite realizar uma análise de ameaças e ataques em vários servidores, equipamentos de rede ou aplicações, em tempo real.

Independentemente da solução de gestão de *logs* escolhida, estão sempre disponíveis as seguintes fases [33]:

1. **Definição da política:** Determina o que a organização pretende auditar (por exemplo, deteção de eventos de segurança, gestão de operações e aplicações, *compliance*);
2. **Configuração**: Converter as políticas de auditoria em informação como, detalhar os eventos de *log* que ajudam a atingir objetivos;
3. **Coleção**: Inclui o envio de eventos de mensagens de *log* dos clientes para o servidor de gestão dos *logs*;
4. **Normalização**: Análise (ou estruturação) dos campos dos dados. Estes, por norma, facilitam a indexação, recuperação e divulgação da informação;
5. **Indexação**: Otimiza a recuperação de dados para consultas/pesquisas, filtros e relatórios. É necessário indexar os dados à medida que são armazenados;

6. **Armazenamento**: Os dados recolhidos precisam de se guardados, a médio ou longo prazo. Podem ser guardados em discos rígidos locais ou externamente (SAN e NAS);
7. **Correlação**: Processo que junta diferentes eventos da mesma fonte ou de várias num evento único. Por exemplo, identificar um ataque de "força bruta" a uma *password*, em vez de identificar sessões de *logons* sem sucesso;
8. ***Baselining***: Processo que define o que é "normal" num determinado ambiente, para que em caso de ocorrência de um evento anómalo, seja despoletado um alerta;
9. **Alarmística**: Caso ocorra um evento crítico de segurança ou falha de uma operação, é importante que a equipa de resposta ao problema seja notificada através de aplicações baseadas nos protocolos SNMP, por SMS ou outra forma adequeada;
10. ***Dashboards*/relatórios**: Os *dashboards* e/ou relatórios permitem que as equipas técnicas identifiquem problemas e façam a gestão de conformidade (*compliance*).

Como referido anteriormente, as soluções de gestão de *logs* são desenvolvidas de diferentes formas e com distintas características e aplicações. Algumas destacam-se na vertente de infraestrutura, outras na vertente aplicacional e outras ainda na vertente de deteção de ameaças na rede. Essas soluções classificam-me genericamente em aplicações centralizadas de gestão de *logs* e *Security Information and Event Management* (SIEM) [21].

As soluções de gestão de *logs* centralizados, *Centralized Log Manangement* - CLM, consolidam as diversas fontes de *logs* num único local acessível ao administrador de sistemas. Este tipo de soluções ajudam o utilizador a analisar e visualizar a informação descrita nos *logs* de forma mais rápida, em segundos, em vez de horas, semanas ou dias, tornado a organização mais dinâmica, lucrativa e segura. As funcionalidades mais comuns nas soluções CLM são as seguintes funcionalidades [34]:

- Políticas de retenção dos *logs* ou seja, torná-los disponíveis durante um determinado período temporal;
- Pesquisa fácil de informação dentro dos *logs*;
- Gerar alertas com base nas métricas definidas nos *logs*;
- Partilhar o próprio *workspace*¸ área de trabalho do utilizador autenticado na solução, e registar informações com outras pessoas de forma simples e rápida;
- Custos reduzidos e maior armazenamento (comparando com as soluções SIEM);
- *Backup* e acesso ao histórico dos *logs*;

- Configurar alertas de segurança e gestão de acessos de utilizadores específicos, sem dar acesso privilegiado (*root*) ao servidor.

As plataformas de SIEM oferecem às equipas de TI uma visão global do que está a acontecer nos vários servidores da rede (interna e externa), em tempo real, de toda a infraestrutura ou parte dela, ajudando os administradores de sistemas a serem mais proativos no que se refere a ameaças de segurança. Ou seja, permite que as equipas de TI recolham dados de várias fontes, aplicações, equipamentos, num único local. Também despoletam alertas em caso de anomalia no tráfego existente, por exemplo, na(s) *firewall*(*s*) da organização, permitindo análises mais rápidas e completas.

As soluções do tipo SIEM são a junção dos conceitos SEM e SIM, gestão de eventos de segurança e gestão de informações de segurança, respetivamente. As SEM analisam os dados dos eventos registados em tempo real, com o objetivo de correlacionar eventos através da monitorização das ameaças e respostas a incidentes. As SIM analisam e recuperam os dados dos eventos, gerando relatórios com a informação necessária à organização em tempo real. As SIEM tem os seguintes benefícios para as organizações [35]:

- Deteção de ameaças de segurança na infraestrutura;
- Melhor análise dos *logs* através da criação de relatórios informativos em tempo real;
- Permitem correlacionar *logs* e a analisá-los a longo prazo;
- Maior eficiência;
- Prevenção de possíveis ameaças à segurança;
- Conformidade na infraestrutura.

2.6.1. Regulamento Geral de Proteção de Dados na gestão de logs

O Regulamento Geral de Proteção de Dados (RGPD) surgiu em maio de 2018 [36], com o intuito de *"estabelecer regras referentes à proteção, tratamento e livre circulação de dados pessoais das pessoas singulares em todos os países membros da União Europeia"*. Este regulamento tem como objetivo reforçar a Proteção de Dados, prevista no art.º 8.º da Carta dos Direitos Fundamentais da União Europeia, e adequar a legislação existente nos Estados-Membros da União Europeia, através de bases para o mercado digital [37].

Segundo o RGPD, pode assumir-se que toda a informação processada por uma organização é identificada como sendo de carácter pessoal (PII). Ou seja, toda a informação que permita

identificar um colaborador, como o primeiro nome, o apelido, o número de cartão de cidadão ou de passaporte, endereços IP, *usernames*, endereço de email, entre outros.

Numa solução de gestão de *logs* centralizados, os *logs* de rede e aplicações processados, podem fazer com que a organização viole a conformidade com o RGPD, pois a probabilidade de existirem estes dados é grande. Tal não significa que o regulamento limite a recolha de *logs*. Pelo contrário, ele impulsiona uma postura de segurança encorajando a adoção de medidas adicionais na recolha dos mesmos [38].

Como sugestão do orientador, analisou-se o capítulo 3 da dissertação "O Regulamento Geral de Proteção de Dados e a Pseudonimização de Logs". Para garantir a conformidade com o RGPD sugere-se que os administradores de sistemas sigam uma abordagem de minimização e anonimização dos dados configurados nos *logs* do servidor. Apresentam-se algumas indicações que contribuirão peara essa anonimização:

- "Os endereços IP completos só devem ser armazenados pelo tempo necessário para fornecer um serviço;
  Os *logs* devem incluir apenas os dois primeiros octetos de endereços IPv4 ou os três primeiros octetos de endereços IPv6;
- Os registos de endereços IP de entrada não devem durar mais do que três dias (suficiente para considerar um fim de semana);
- Não devem ser registados identificadores desnecessários (portos de origem e destino, registos de data e hora, números de portos de protocolos);
- Os *logs* devem ser protegidos contra acesso não autorizado." [39]

Os Art.º 4 - *"Princípios relativos ao tratamento de dados pessoais"*, Art.º 25 - *"Proteção de dados desde a conceção e por defeito"*, Art.º 30 – *"Registos das atividades de tratamento"*, Art.º 32 – *"Segurança do tratamento"* e Art.º 33 - *"Notificação de uma violação de dados pessoais à autoridade de controlo"* presentes no regulamento, refletem como o registo de *logs* de auditoria podem ter um grande impacto na implementação da proteção de dados numa organização.

Assim, este tipo de soluções ajudam as organizações a manter a conformidade com o RGPD, através da monitorização de acessos às aplicações, às bases de dados e inclusivamente, aos *logs* aplicacionais. Para além disso, também ajudam na redução de risco de perda de dados, visto que eles estão centralizados; facilidade em analisá-los; redução de complexidade e de

superfície de ataque; *logs* estruturados e/ou encriptados e anonimato de campos de dados confidenciais [38].

2.6.2. Importância dos *logs*

O registo, armazenamento e análise dos vários *logs*, independentemente do motivo, na maioria das vezes, é a única forma que um administrador de sistemas tem para descobrir as causas de um problema ou um comportamento anormal e é um procedimento de elevada importância na auditoria de segurança para as organizações. Também é um fator importante para efeitos de análise forense.

Uma auditoria de segurança para ter sucesso depende da existência de registo de *logs* confiáveis, aumentando a probabilidade de sucesso quando se correlacionam e identificam padrões de incidentes de segurança ocorridos no sistema. Em caso do sistema operativo, rede ou servidor estar comprometido, a segurança pode estar em causa.

Os dados gerados pelos *logs* fornecem diversas informações que podem ser transformadas em indicadores capacitados na medição dos níveis de segurança e na avaliação das medidas de segurança oferecem os resultados esperados e planeados. Estes dados podem ser reproduzidos em relatórios diários, mensais ou anuais, mostrando *logs* registados, estatísticas de desempenho (como CPU, memória e consumo de espaço em disco) dos servidores ou equipamentos de rede. Também é possível, através desses indicadores, identificar tendências nos servidores, prever falhas de serviços ou controlar acessos.

Os sistemas de centralização de *logs* desempenham a função de análise e correlação de *logs*, evitando esforço desnecessário na consulta de vários ficheiros de *logs* em diferentes máquinas, e aumentando a produtividade dos administradores de sistemas [20].

3. Estado de Arte

Este capítulo aborda os trabalhos relacionados com o tratamento automático de *logs* e aplicações, comerciais e *open source*, que existem no mercado e que influenciaram a tomada de decisão para a implementação deste projeto.

3.1. Soluções relacionadas com o tratamento automático de logs

Algumas organizações quando pretendem recolher os *logs* dos seus sistemas, fazem-no através de *scripts* executados de forma manual ou automatizada [40]. Quando são executados de forma automática, normalmente a sua execução é agendada para recolher os dados (como por exemplo, diariamente, semanalmente ou mensalmente) e guarda-os num ficheiro de texto. Por vezes, os ficheiros com a informação recolhida podem ser enviados por email para o administrador de sistemas.

Encontram-se disponíveis vários *scripts* destinados a este propósito para diversos sistemas operativos, serviços e aplicações, como por exemplo visualizar o espaço utilizado das *mailboxs* do serviço de email da Microsoft (Microsoft Exchange) [41]. Em alguns casos, é necessário adaptar a linguagem do código fonte do *script* ao sistema operativo do servidor, onde será executado ou às informações que o utilizador pretende obter da infraestrutura ou serviços. Por exemplo, um *script* desenvolvido na linguagem PowerShell, para analisar o desempenho de memória nos servidores Windows, terá de ser adaptado de modo a ser executado em servidores Linux.

O exemplo ilustrado na Figura 3.1, mostra o *output* de um *script* executado num sistema operativo Linux que recolhe o número de utilizadores ligados ao sistema num intervalo de tempo definido, número de tentativas de *login* com sucesso e sem sucesso [42].

```
# sh /opt/scripts/user-access-details.sh

Enter the Date, Use Double Space for date from 1 to 9 (Nov  3) and use Single Space for date from 10 to 31 (Nov 30): Nov 30
-----------------------------------------
       User Access Report on: Nov 30
-----------------------------------------
Number of Users logged on System: 20
Successful logins attempt: 14
Failed logins attempt: 6
-----------------------------------------
Success User Details:
      1 daygeek
     1 root
     3 u1
     4 u2
     1 u3
     2 u4
     2 u5
-----------------------------------------
Failed User Details:
      3 u1
     3 u4
-----------------------------------------
```

Figura 3.1 – Exemplo de output de recolha de dados de um utilizador [42].

3.2.*Software* para tratamento manual de *logs*

O Notepad++ é um *software* que permite ler e editar ficheiro em vários formatos. Através dele, os utilizadores podem abrir um ficheiro de *logs* e analisá-lo de forma personalizada, mas manual.

Esta aplicação disponibiliza uma secção de definição do conteúdo do ficheiro, onde é possível definir palavras-chave para *logs* de diferentes níveis de severidades, como por exemplo “Erro” (o *log* é sublinhado a vermelho), “Alerta” (a amarelo) e “Informativo” (a azul) [43]. A Figura 3.2, mostra o exemplo referido.

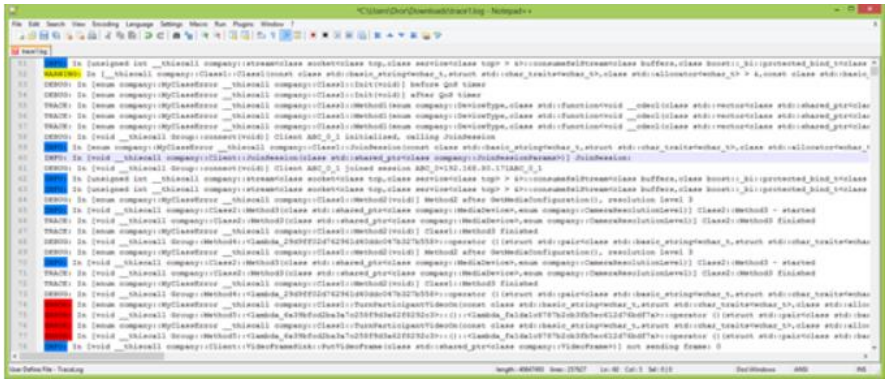


Figura 3.2 – Ficheiro de *logs* personalizado no Notepad++ [43].

Esta solução é simples, prática, sem custos e fácil de executar para um administrador de sistemas. No entanto, tem a desvantagens de ser uma tarefa realizada manualmente e de não ser eficaz quando o ficheiro é muito grande.

3.3. Aplicações comerciais e *open source*

Como referido no subcapítulo 2.6, as soluções de gestão de *logs* diferenciam-se em centralizadas (subcapítulo 3.3.1) e do tipo SIEM (subcapítulo 3.3.2).

3.3.1. Gestão centralizada de *logs*

Atualmente, o mercado disponibiliza inúmeras soluções para gestão de *logs* e gestão centralizada de *logs*, diferenciando-se pelas características, funcionalidades e custos. Por vezes com base na análise das Tabela 3.1 e Tabela 3.2, conclui-se que existem soluções no mercado que disponibilizam funcionalidades de gestão de *logs* e SIEM, como é o caso do SolarWinds e do Splunk.

O *blog* PC&Network[1], considerou como melhores soluções de gestão de *logs* em 2019, o SolarWinds Event Log Consolidator/Manager, LogFusion, Netwrix Event Log Manager, Splunk e WhatsUpGold [44]. Na Tabela 3.1, encontra-se uma breve descrição de cada solução.

Tabela 3.1 - Melhores soluções de gestão de *logs* em 2019

Nome	Características
SolarWinds Event Log Consolidator/Manager	O SolarWinds divide-se em duas partes para gerir *logs* de eventos. O Event Log Consolidator é totalmente gratuito e pode-se considerar uma versão leve do Manager, que é mais robusto. Permite visualizar *logs* de vários sistemas Windows e filtrar os *logs* por ID e padrões nos dados do evento. O Log & Event Manager, também disponibiliza uma versão gratuita que armazena e avalia dados históricos de *log*, envia alertas por email e correlaciona dados de vários dispositivos na rede. A nível de segurança, garante *compliance* através do agendamento de auditorias automatizadas [44].

[1] https://www.pcwdld.com/

Nome	Características
LogFusion	É uma solução considerada básica no que faz, mas fá-lo de maneira limpa e concisa. Ele ingere *logs* baseados em texto, eventos e remotos. A versão gratuita grande parte dos recursos existente na versão com licenciamento. No entanto funcionalidades como pesquisa por *input*, personalização de colunas apenas estão disponíveis na versão com licenciamento [44].
Netwrix Event Log Manager	É um *freeware* e também trabalha com funcionalidades básicas, como consolidação de eventos numa rede inteira através de um único local, despoleta alertas de eventos críticos por email, em tempo real e permite filtragem de alertas e ficheiros, mas de forma limitada. Esta solução não será ideal para infraestruturas grandes [44].
Splunk	Aplicação de gestão de *logs* que se distingue pelo encapsulando de dados de uma gama de dispositivos numa rede. Também é flexível de modo a funcionar *on-premises*, híbrido ou *cloud* [44].
WhatsUpGold	Distingue-se pela capacidade de automação e na gestão de *logs* de rede, mas também possui excelentes recursos de log para eventos do Windows. Também inclui serviços de monitorização em tempo real com capacidades para filtrar e analisar os *logs* recolhidos [44].

Do levantamento realizado, realçam as seguintes considerações:

- A solução Solarwinds foi a mais utilizada em 2019;
- O Solarwinds destaca-se pelas funcionalidades que disponibiliza;
- As soluções centralizam a informação (métricas e *logs*);
- Grande parte das soluções tem a versão gratuita, no entanto limitada por funcionalidades ou tráfego recolhido diariamente;
- As soluções permitem consultar os dados em tempo-real ou em histórico;
- No caso do Solarwinds e Splunk, permitem a integração com vários ambientes (*on-premisses* e *cloud*).
- O Solarwinds e Netwrix Event Log Manager quando detetam anomalias na infraestrutura, informam o responsável da mesma por email.

3.3.2. *Security Information and Event Management*

Da mesma forma que o mercado disponibiliza inúmeras soluções para gestão de *logs*, também disponibiliza outras para SIEM, com funcionalidades e custos distintos. Segundo o site compariTech[2], as plataformas de SIEM, SolarWinds Security Event Manager, ManageEngine EventLog Analyzer, Micro Focus ArcSight ESM, Splunk Enterprise Security, LogRhythm Security Intelligence, IBM QRadar, entre outras, como as melhores em 2019.

Em 2018, a Gartner divulgou o posicionamento dos produtos de SIEM, tendo em conta as caracteristicas de elevada participação no mercado e aumento das receitas [45] e os seguintes quadrades de classificação [46]:

- ***Challengers*:** soluções que possuem grande capacidade de produção e entrega, mas que ainda não atingiram uma porção relevante do mercado
- ***Leaders*:** soluções que possuem uma posição de destaque no mercado, destancando-se entre as concorrentes e encontram-se num nível mais avançado de desenvolvimento tecnológico;
- ***Niche players*:** soluções que com focar em apenas algumas necessidades existentes no mercado;
- ***Visionaries*:** Soluções que possuem boa capacidade de investimento e apresentam novas tecnologias, mas ainda não disponibilizam tudo o que prometem nas suas funcionalidades.

Da análise da Figura 3.3 conclui-se, segundo a avaliação da Gartner, que as empresas lideres no mercado com soluções de SIEM são a `Splunk`, a IBM e a `LogRhythm` [45].

[2] https://www.comparitech.com/

Figura 3.3 - Gartner no quadrante dos líderes para soluções SIEM em 2018 [45].

Ainda com base nos resultados retirados da Figura 3.3 e do site compariTech, compila-se na Tabela 3.2 a comparação de algumas dessas soluções e as suas principais características.

Tabela 3.2 – Melhores soluções de SIEM de 2018 e 2019

Nome	Características
Splunk	- Recolhe dados de diversas máquinas (servidores, servidores de *web*, redes, *exchanges*, *mainframes*, dispositivos de segurança); - *Interface* flexível ao utilizador para pesquisar e analisar os dados em tempo real; - Algoritmo de perfuração para encontrar anomalias e padrões familiares nos ficheiros de *log*; - Sistema de monitorização e alerta para visualizar eventos e ações importantes; - Os relatórios usam um *output* automatizado para os *dashboards* [47]; - Possui de período gratuito de 30 dias; - Integração com a *cloud* [48].

Nome	Características
IBM QRadar	- Visibilidade abrangente dos dados em ambientes *on-premises* e baseados em *cloud* num único painel; - Deteta ameaças conhecidas e desconhecidas, disponibiliza alertas individuais que identificam e priorizam possíveis incidentes. Aplique a Inteligência Artificial para acelerar os processos de investigação em 50%; - Oferece *feedback* em circuito fechado que melhora, de forma continua a deteção e usa segurança automatizada para prosseguir proactivamente ameaças e automatizar processos de controlo [49]; - Possui período de utilização gratuito de 14 dias; - Integração com a *cloud* [48].
LogRhythm	- Tecnologia inteligente que coleta, analisa qualquer tipo de dados (estruturados ou não estruturados); - *Back-end* do Elasticsearch para consultas/pesquisas simples ou elaborados com velocidades extremamente rápidas; - Monitoração de ataques críticos do primeiro ao último segundo da ocorrência; - *Dashboards* avançados que ajudam a interpretar rapidamente se os dados são originários ou se é uma ameaça [47].
SolarWinds Log & Event Manager	- Licenciamento por nó; - Correlação e correção de eventos em tempo real; - Monitoriza a integridade de ficheiros; - Analisa USB; - *Dashboards* configurável; - Agendamentos de consultas; - Grupos definidos pelo utilizador; - Emails personalizáveis; - *Feed* de inteligência sobre ameaças [47]*;* - Integração com a *cloud* [48].
Micro Focus ArcSight ESM	- Recolhe bilhões de eventos por dia, mais de 400 *data sources* de forma centralizada; - Armazena dados de forma mais eficiente através da agregação de eventos do *Logger* e a compacta *logs* até 10:1; - Documentos de *compliance* em *dashboards* e relatórios integrados que aliviam a carga relacionada aos requisitos e auditorias;

Nome	Características
	- Pesquisa de dados através de consultas dinâmicas que facilitam a pesquisa em grandes quantidades de dados [50]; - Possui período gratuito, limitado pelos dados ingeridos; - Integração com a *cloud* (Azure e AWS) [48].

Do levantamento realizado, realçam as seguintes considerações:

- A solução Splunk foi a mais utilizada em 2018 e 2019;
- As soluções recolhem dados de várias fontes de dados;
- As soluções mostram a informação recolhida em *dashboards*;
- A informação visualizada nos *dashboards* é em tempo-real;
- Os dados recolhidos são possíveis de consultar em tempo-real ou em histórico;
- O Solarwinds destaca-se pelas funcionalidades de monitorização da integridade dos ficheiros, análise USB e agendamento de consultas aos dados recolhidos;
- A maioria das soluções apresentam a possibilidade de utilização gratuitamente durante um período temporal definido.

3.4. Aplicações de visualização de dados

As soluções de visualização de dados permitem em representar visualmente (através de gráficos, tabelas, métricas, entre outros) os diferentes dados, divididos em categorias, para facilitar e agilizar a sua análise. Por norma, os dados são recolhidos por aplicações especificas devido ao volume elevado em cada instante [51].

Em 2019, a Gartner divulgou o posicionamento das soluções de análise e visualização de dados, tendo em conta as caracteristicas de elevada participação no mercado e o aumento das receitas, nos quadrantes de classificação *Challengers*, *Leaders*, *Niche players* e *Visionaries* [46]. Cada quadrante de classificação encontra-se descrito na secção 3.3.2.

Da análise da Figura 3.4 conclui-se, que as empresas lideres no mercado com soluções de Análise e Visualização de dados são Microsoft, Tableau e Qlik.

Figura 3.4 - Gartner no quadrante dos líderes para soluções de Análise e Visualização de Dados em 2019 [46].

Ainda com base nos resultados retirados da Figura 3.4 e do site CIO[3], compila-se a Tabela 3.3, onde é possível comparar as principais características de algumas dessas soluções.

Tabela 3.3 – Melhores soluções de Análise e Visualização de Dados em 2018 e 2019

Nome	Características
Microsoft Power BI	- Permite analisar e visualizar dados com origens *on-premises* ou *cloud*; - Disponibilização e partilha de *dashboards*/relatórios interativos na plataforma Power BI; - Integra com diversas soluções plataformas da Microsoft; - Análise centralizada de dados e de forma personalizada; - Simplifica tomadas de decisão baseadas em dados e diferentes fontes; - Visão 360º do negócio; - Configuração de notificações; - Requer licença, no entanto a implementação do *dashbords*/relatórios não requer de licença [52] [53].

[3] https://cio.com.br/

Nome	Características
Qlik	- Recolhe dados em ambientes *on-premises* e em *cloud*; - Disponibiliza acesso dos dados aos colaboradores da empresa com base nas políticas corporativas de governação dos mesmos; - Licença mensal por utilizador, sendo que existe versão gratuita, mas limitada [52].
Board	- Disponibiliza as componentes de BI, análise avançada e gestão do desempenho; - Disponibiliza módulos para diversas áreas, como por exemplo Finanças, Recursos Humanos, *Marketing*, entre outras; - Requer licença para utilização [52] [54].
Domo	- Baseia-se em *cloud*; - Disponibiliza componentes de BI adaptadas a vários setores, como financeiros, assistência médica, educação, entre outros; - Integra-se com as plataformas AWS, Jira, GitHub; - Robusta visualização em dispositivos móveis; - Requer licença para utilização [52].
Dundas BI	- Solução mais usada para implementar *dashboards*, *scorecards* e relatórios *standard;* - Visualização é realizada por *interface* Web que se pode adaptar às exigências dos utilizadores (utilizadores avançados e utilizadores *standard*) através de restrições; - A versão mais recente importa dados do Google *Analytics*, Snowflake e Salesforce Pardot; - Necessita de licença [52].
IBM i2 Analyst's Notebook	- Análise ao registo de chamadas; - Visualização de inteligência; - Correlaciona eventos entre redes; - Possível exportar os dados; - Integração direta com os dados; - Requer licença de utilização, no entanto é possível testar sem custos num intervalo de tempo de 30 dias [55]. Esta solução foi analisada por sugestão do orientador.

Do levantamento realizado, realçam as seguintes considerações:

- A solução Microsoft Power BI foi a mais utilizada 2019 e destaca-se pelas suas funcionalidades;
- As soluções recolhem dados de várias fontes de dados;

- As soluções mostram a informação recolhida em *dashboards*;
- A maioria das soluções apresentam licença para utilização.

Resumindo, o projeto desenvolvido pretende ajudar na análise dos *logs* recolhidos dos servidores com sistema operativos Windows, filtrando os campos dos *logs* mais relevantes.

As soluções descritas anteriormente recolhem dados de qualquer tipo de *logs*, independente da origem ou formato. No entanto não têm geralmente capacidade de filtragem dos dados existentes. Neste projeto a visualização agregada dos vários *logs*, através do "meta-log", será realizada através de um relatório desenvolvido especificamente para o efeito. O relatório assenta-se no formato XML, de modo a facilitar a leitura e interação com o ficheiro dos *logs* originais. A componente de visualização, designadamente o relatório agregador, foi desenvolvido em Power BI, não só por ser a solução que se destacou no mercado em 2019, mas também pela experiência adquirida anteriormente com a aplicação.

Por outro lado, o custo das soluções descritas é elevado e nem todas as organizações têm meios financeiros para as implementar. As organizações que não tenham este tipo de soluções por questões financeiras ou outras, podem utilizar uma aplicação baseada no protótipo desenvolvido, para realizar a análise e filtragem do conteúdo dos *logs*.

4. Arquitetura proposta

Neste capítulo encontra-se o desenho da arquitetura proposta detalhada e a justificação dos campos relevantes nas diversas mensagens de *logs* processadas.

4.1. Desenho da arquitetura

A arquitetura proposta, ilustrada na Figura 4.1, contém os seguintes formatos de ficheiros:

- **Ficheiro de texto**, conjunto de elementos acedidos de forma sequencial, em posições consecutivas. A sua informação encontra-se estruturada em linhas, logo, os elementos são de comprimento variável [56];
- ***Script***, consiste num conjunto de instruções escritas em código PERL que serão executadas por uma determinada ordem pelo servidor, designado por *script* [57].

Esta arquitetura consiste numa aplicação que processa ficheiros de *log* distintos e transforma as suas mensagens num ficheiro com um formato normalizado. Trata-se de um processo de fusão, em que se define um meta-log para o processamento dos ficheiros de log e um outro para o formato de saída, do *template*.

A arquitetura é dividida em três fases:

- **Pré-processamento**, consiste em agrupar ficheiro de texto com os *logs* gerados pelo sistema operativo e aplicações com um ficheiro "confg". O ficheiro "confg" tem o objetivo dar ao utilizador a possibilidade de configurar/selecionar os valores dos campos pré-definidos, consoante a aplicação.

  O processo de "pré-processamento" é realizado através de um *script* que recebe os dois ficheiros referidos anteriormente e gera um meta-log (ficheiro de texto) com um formato genérico, baseado nos campos pré-definidos. Cada ficheiro de *log*, tem o seu respetivo ficheiro "confg".

  Neste processo foi configurado um ficheiro de *logs*, que centraliza os *logs* dos vários meta-logs, que é escrito sempre que o utilizador não insere os parâmetros de forma correta na execução da aplicação;
- **Fusão de *logs***, refere-se à agregação dos meta-logs através da execução de um *script* que recebe os vários meta-logs, resultantes da fase de "pré-processamento" e

converte para uma linguagem de especificação, originando um ficheiro XML. Este ficheiro é designado por *template*;

Neste processo também existe um ficheiro de *logs* da aplicação, que é escrito sempre que o utilizador não insere os parâmetros corretamente.

- **Visualização**, nesta fase é integrado o *template* originado na fase "Fusão de *logs*" numa solução de visualização de dados, tornado a leitura do conteúdo do *template* rápida e simples.

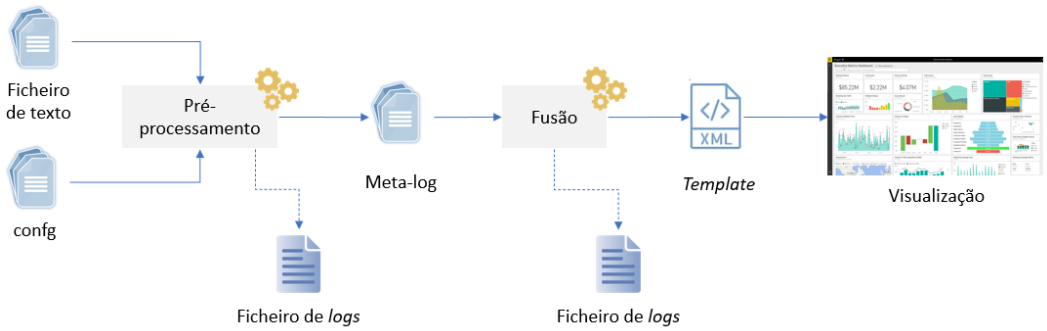


Figura 4.1 – Desenho da arquitetura.

4.2. Tomada de decisão no conteúdo das mensagens de logs

Após a análise de diversos formatos e campos existentes nos *logs* de sistemas operativos Windows e Linux, detetou-se a repetição dos campos "*timestamp*", "nome do serviço/comando/aplicação", "origem do *log*", "nível de severidade" e "descrição". No caso dos *logs* de sistema operativo Windows, também se considerou relevantes os campos que indicam o ID com evento, processo e *thread*.

A Tabela 4.1 descreve os campos considerados relevantes para o resultado do projeto, com a respetiva descrição detalhada.

Tabela 4.1 – Campos de *logs* definidos, descritos e utilizados no *template*.

Nome do campo	Descrição
`timestamp`	Conjunto de caracteres que fornece precisão de uma determinada ocorrência através da data e hora.
`source`	Identifica o servidor que originou a ocorrência.
`levelLogName`	Identifica a categoria do *log* para uma filtragem mais rápida das possíveis causas do problema.

Nome do campo	Descrição
`user`	Identifica o utilizador autenticado no servidor que originou a ocorrência.
`sourceComputer`	Identifica o servidor que sofreu o *log*.
`severityName`	Identifica o nível de prioridade na resolução ou tomada de alguma decisão perante o *log*.
`eventID`	Identifica de forma única um determinado *log*.
`description`	Texto descritivo e resumido com o objetivo de enumerar caraterísticas próprias do problema.
`processID`	Identifica de forma única o processo que se refere ao *log*, quando aplicável.
`threadID`	Identifica de forma única a *thread* que se refere ao *log*, quando aplicável.

Relativamente ao nível de severidade das mensagens de *log*, a Tabela 4.2 descrever os três níveis considerados para o estudo e resultado deste projeto.

Tabela 4.2 – Níveis de severidade de *logs* definidos, descritos e utilizados.

Nível de Severidade	Descrição
`error`	Mensagens de erro existentes que podem originar erros graves ou paragem no funcionamento do equipamento, sistema operativo ou aplicação, se não for tomada uma ação de imediato.
`warning`	Mensagens de alertas gerados pelo equipamento, sistema operativo ou aplicação, que podem originar erros ou erros graves, se não for tomada uma ação a tempo.
`information`	Mensagens do estado das funcionalidades que o equipamento, sistema operativo ou aplicação executam ou estão a ocorrer.

Por fim, na Tabela 4.3, consideraram-se as duas categorias com as respetivas descrições a ter em conta no estudo e resultados deste projeto.

Tabela 4.3 – Categorias de *logs* definidas, descritos e utilizados.

Categoria	Descrição
`Aplication`	Informa que o *log* está relacionado com a(s) aplicação(ões).
`System`	Informa que o *log* está relacionado com o sistema operativo.

5. Desenvolvimento

Este capítulo apresenta a implementação da arquitetura proposta com os principais algoritmos desenvolvidos nos *scripts*. Este capítulo também descreve a integração do *template*, *output* obtido através da fusão dos meta-logs, no Power BI.

Para o desenvolvimento das aplicações configurou-se o sistema operativo Ubuntu 18.04.3 LTS, edição servidor, numa máquina virtual com 2 GB de RAM. Para esta máquina virtual, foram importados todos os ficheiros de *logs* necessários à implementação e testes realizados pelos *scripts* referidos na arquitetura proposta na subcapítulo 4.1.

5.1. Principais algoritmos do pré-processamento de logs

Neste subcapítulo apresentam-se duas abordagens de implementação da aplicação de pré-processamento: a primeira consiste numa implementação não genérica da aplicação e a segunda, que garante uma implementação genérica.

5.1.1. Implementação não genérica da aplicação

O *script* recebe um ficheiro de texto com cerca de 20 campos, de vários *logs* com graus de severidade, formatos e origens diferentes, como ilustra a Figura 5.1.

Nível	Data e hora	Origem	ID do evento	Categoria de Tarefa	Palavras-chave	Utilizador	Código operacional		Registo	Computador	ID do processo	
Informações	09/12/2019 21:13:44	Service Control Manager	7040	Nenhum	ClÃ¡ssica	S-1-5-18	Info	System	DESKTOP-DDD0ABG	796	2480	
Informações	09/12/2019 21:08:55	TPM	17	Nenhum	S-1-5-18	Info	System	DESKTOP-DDD0ABG	4	268		
Informações	09/12/2019 21:08:54	TPM	17	Nenhum	S-1-5-18	Info	System	DESKTOP-DDD0ABG	4	268		
Informações	09/12/2019 21:08:54	TPM	17	Nenhum	S-1-5-18	Info	System	DESKTOP-DDD0ABG	4	268		
Informações	09/12/2019 21:06:32	Microsoft-Windows-UserModePowerService	12	(10)		S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	7352	
Informações	09/12/2019 21:06:22	Microsoft-Windows-Kernel-Power	105	(100)	(1024),(4)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	4	1748	
Informações	09/12/2019 21:06:10	Microsoft-Windows-UserModePowerService	12	(10)		S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	7352	
Informações	09/12/2019 21:06:10	Microsoft-Windows-Kernel-Power	105	(100)	(1024),(4)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	4	13392	

Figura 5.1 – Estrutura exemplo do ficheiro original de *logs*.

Com base na análise realizada aos campos relevantes das mensagens de *log*, implementou-se um *array* que percorre todos os campos do *log* em todas as linhas do ficheiro original, exceto a primeira linha (que é ignorada pelo código) pois corresponde ao nome dos campos. A posição inicial do *array* corresponde ao número inteiro 0.

Após alguns testes, verificou-se que o número de campos nos *logs* varia consoante a origem e severidade do *log*, sendo necessário implementar uma condição, com o operador lógico "*if else*" como ilustra o seguinte excerto de código:

```
foreach $_ (<IN>){
      next if /Relativo Nome da Origem de Evento/;
      chomp $;
      $_ =~ tr/\t/&/s;
      my ($datetime, $source, $levelLogName, $sourceComputer, $user, $severityName, $eventID, $description, $processID, $threadID) = (split /&/, $_)[1, 2, 0, 7, 4, 6, 3, 12, 8, 9];
      my ($sourceComputerSCM, $severityNameSCM, $userSCM, $descriptionSCM, $processIDSCM, $threadIDSCM) = (split /&/, $_)[8, 7, 6, 13, 9, 10];
if($source eq "Service Control Manager"){
      print OUT "$datetime&";
      print OUT "$source&";
      print OUT "$levelLogName&";
      print OUT "$sourceComputerSCM&";
      print OUT "$userSCM&";
      print OUT "$severityNameSCM&";
      print OUT "$eventID&";
      print OUT "$descriptionSCM&";
      print OUT "$processIDSCM&";
      print OUT "$threadIDSCM\n";
```

Desta forma, a cada nova origem na mensagem de *log*, seria acrescentado uma condição no código.

Relativamente ao ficheiro de entrada e de saída (ficheiro meta-log), os nomes dos ficheiros seriam inseridos de forma manual no código, assim como as posições dos campos dos *logs*. O seguinte excerto de código exemplifica os ficheiros de entrada e saída:

```
open(IN, "dataset_34_windows_system_erro.txt");
open(OUT, ">>metalog_error_34.txt");
```

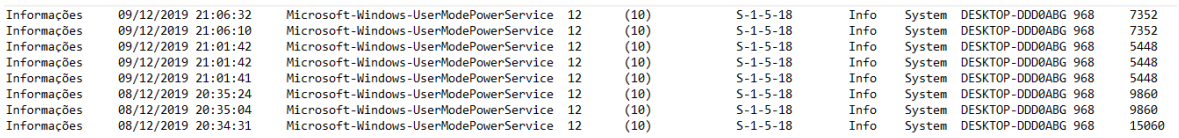
Concluiu-se que esta abordagem seria impraticável, difícil de gerir e não genérica, pois quando existisse um *log* com uma origem diferente das refletidas no código fonte e/ou com os campos com ordem diferente, o utilizador teria de adicionar a condição ao código e/ou as novas posições dos campos. O código fonte completo desta abordagem pode ser consultado no Anexo A – "Versão 1 - Código fonte da aplicação de pré-processamento".

5.1.2. Implementação genérica da aplicação

Após a abordagem descrita na secção 5.1.1 não ter o resultado esperado, implementou-se um novo *script*. Este *script* recebe dois ficheiros de entrada com as seguintes características:

- **Ficheiro com os *logs*:** refere-se a um ficheiro de texto, com vários *logs* com 20 campos recolhidos de um sistema operativo. Este ficheiro tem os seguintes pré-requisitos que devem ser tidos em conta para o correto funcionamento da aplicação:
  - Os *logs* tem de pertencer ao sistema operativo do mesmo tipo (por exemplo, Windows);
  - Apenas podem existir *logs* da mesma origem e severidade (por exemplo, *logs* de erro do serviço Microsoft-Windows-DistributedCOM);
  - Os *logs* tem de ter o mesmo formato;
  - Cada *log* corresponde a uma linha no ficheiro.

  A Figura 5.2 ilustra um exemplo de estrutura deste ficheiro com os pré-requisitos necessários.



Informações	09/12/2019 21:06:32	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	7352
Informações	09/12/2019 21:06:10	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	7352
Informações	09/12/2019 21:01:42	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	5448
Informações	09/12/2019 21:01:42	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	5448
Informações	09/12/2019 21:01:41	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	5448
Informações	08/12/2019 20:35:24	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	9860
Informações	08/12/2019 20:35:04	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	9860
Informações	08/12/2019 20:34:31	Microsoft-Windows-UserModePowerService	12	(10)	S-1-5-18	Info	System	DESKTOP-DDD0ABG	968	15060

Figura 5.2 – Estrutura exemplo de um ficheiro de *logs*.

- **Ficheiro "confg":** consiste num ficheiro de texto com os campos pré-definidos, os respetivos valores (posição do campo no ficheiro de *logs*) e o delimitador, caracter especial que separa os valores entre os campos. O valor definido consiste num número inteiro de 0 até ao número máximo de campos existentes no ficheiro original de *logs*.

  O ficheiro permite a alteração das posições dos campos e delimitador que o utilizador pretende processar e o delimitador utilizado pelo ficheiro de origem dos *logs*, evitando alterar o código fonte (*script*) da aplicação.

  Este ficheiro "confg" é composto pelos seguintes campos: *timestanp*, *source*, *level name*, *source computer*, *user*, *severity name*, *eventID*, *description*, *processID*, *threadID* e o delimitador. A **Figura 5.3**, ilustra a estrutura do ficheiro "confg.txt".

```
#Datetime, Source, Level, Source Computer, User, Serverity, Event ID, Description, Process ID, Thread ID, Delimitador
1,2,0,8,5,7,3,13,9,10,\t
```

Figura 5.3 – Estrutura exemplo de um ficheiro "confg.txt".

Quando o *script* recebe estes três ficheiros de entrada, é verificado o número de parâmetros de entrada e existência dos mesmos. Neste caso, são registados como parâmetro de entrada os ficheiros "confg.txt", "ficheiro_com_os_logs.txt" e "ficheiro_output.txt", como ilustra a Figura 5.4.

```
ubuntu@ubuntu:~/MCIF/Metalog/Teste-Nfiles_V3/Windows_Sistema_Testel_v2$ ./metalog_windows.pl confg.txt
dataset_1_windows_system_info_EventLog.txt metalog_information_EventLog.txt
```

Figura 5.4 – Execução do *script* com parâmetros.

Caso o número de parâmetros inserido for diferente do definido (3 ficheiros) ou se os ficheiros de entrada ("confg.txt" e "ficheiro_com_os_logs.txt") não existam, é mostrada uma mensagem de erro na consola e gravada num ficheiro de *logs* centralizado.

O ficheiro de *logs* centralizado agrega a seguinte informação das validações implementadas aos parâmetros de entrada:

- ***Datetime***, mostra a hora e data (dd-mm-aaaa hh:mm:ss) que ocorreu a mensagem de erro;
- ***Source***, identifica a máquina onde ocorreu o erro;
- ***User***, identifica o utilizador que provocou o erro;
- ***Message***, descreve a mensagem de erro.

De seguida, o *script* lê as posições definidas no ficheiro "confg.txt". Estas posições, podem ser consultadas na Tabela 5.1 e Tabela 5.2 e correspondem à posição no *array* do "ficheiro_com_os_logs.txt". As posições definidas são percorridas por todas as linhas existentes no ficheiro "ficheiro_com_os_logs.txt" e são escritos os campos definidos no "ficheiro_output.txt", designando por meta-log.

Caso o ficheiro de meta-log não exista, quando é inserido no parâmetro, ele é criado automaticamente.

A Tabela 5.1, mostra as posições do *array* tendo em conta a origem do *log* e o grau de criticidade "informativo" no sistema operativo Windows.

Tabela 5.1 – Posições no *array* consoante a origem e criticidade "Informativo" do *log* em Windows.

Origem do *log*	*timestanp*	*source*	*level name*	*source computer*	*user*	*severity name*	*eventID*	*description*	*processID*	*processID*
Microsoft-Windows-Kernel-General	1	2	0	8	5	7	3	13	9	10
Service Control Manager	1	2	0	9	6	8	3	15	10	11
Trusted Platform Module	1	2	0	8	5	7	3	13	9	10
Microsoft-Windows-UserModePowerService	1	2	0	8	5	7	3	13	9	10
Microsoft-Windows-Kernel-Power	1	2	0	9	6	8	3	14	10	11
Microsoft-Windows-WindowsUpdateClient	1	2	0	9	6	8	3	14	10	11
Microsoft-Windows-Power-Troubleshooter	1	2	0	8	5	7	3	13	9	10
Microsoft-Windows-Winlogon	1	2	0	9	6	8	3	14	10	11
Microsoft-Windows-Kernel-Boot	1	2	0	7	4	6	3	12	8	9
EventLog	1	2	0	7	4	6	3	12	8	9

A Tabela 5.2, mostra a posição do array tendo em conta a origem do *log* e o grau de criticidade "erro" no sistema operativo Windows.

Tabela 5.2 – Posições no *array* consoante a origem e criticidade "Erro" do *log* em Windows.

Origem do *log*	*timestanp*	*source*	*level name*	*source computer*	*user*	*severity name*	*eventID*	*description*	*processID*	*processID*
Service Control Manager	1	2	0	8	4	7	3	14	9	10
Trusted Platform Module	1	2	0	7	4	6	3	12	8	9
Microsoft-Windows-DistributedCOM	1	2	0	9	6	8	3	15	10	11

Como referido, o meta-log é constituído pelas entradas dos *logs* presentes no ficheiro original de *logs*, refletindo apenas os campos indicados no ficheiro "confg". Os campos do meta-log estão separados pelo delimitador "&" e o seu conteúdo é adicionado após a última linha já existente, sempre que o *script* é executado. Isto é, acrescenta as novas linhas com o novo conteúdo, ao conteúdo já existente.

O meta-log também reflete o nome do ficheiro e a linha a que o *log* pertence nesse ficheiro, para que no processo de fusão dos *logs* no *template*, subcapítulo 5.2, seja possível identificar a origem do mesmo.

Na Figura 5.4, é possível visualizar um exemplo de *output* deste meta-log, sendo constituído pelos seguintes campos "*filename*", "*line*", "*timestanp*", "*source*", "*level name*", "*source computer*", "*user*", "*severity name*", "*eventID*", "*description*", "*processID*", "*threadID*".

```
dataset_167_windows_system_info_TPM.txt&149&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&150&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&151&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&152&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&153&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&154&08/12/2019 20:36:00&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&155&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&156&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&157&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&158&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&159&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&160&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&161&08/12/2019 20:35:59&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&162&08/12/2019 20:35:58&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&163&08/12/2019 20:35:58&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&164&08/12/2019 20:34:37&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&165&08/12/2019 20:34:31&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
dataset_167_windows_system_info_TPM.txt&166&08/12/2019 20:34:30&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao exec
```

Figura 5.5 – *Output* do ficheiro de meta-log.

O código fonte desenvolvido para originar o meta-log encontra-se documentado no Anexo B – "Código fonte da aplicação de pré-processamento" deste relatório.

5.2. Principais algoritmos para fusão de *logs*

A implementação deste *script* tem como objetivo desenvolver um *template* em formato XML, com os meta-logs referidos no subcapítulo 5.1.

O *script* recebe múltiplos meta-logs, em ficheiro de texto, da aplicação desenvolvida no subcapítulo 5.1, que estão numa diretoria comum juntamente com a aplicação. Assim, sempre que existe um novo meta-log, é necessário adicioná-lo a esta diretoria.

O processamento do *script* inicia-se com a verificação da existência de dois parâmetros de entrada, neste caso o nome da diretoria que tem os ficheiros de meta-log e o ficheiro de *output*, "template.xml". Caso o número de parâmetros inseridos seja diferente do definido, 2, ou o nome da diretoria não exista, mostra uma mensagem de erro na consola e o *script* não é executado, como mostra a Figura 5.6.

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ./metalog_windows_Nfiles.pl
/home/ubuntu/MCIF/Template/final_v3/
Incomplete number of arguments.
```

Figura 5.6 – Validação do número de parâmetros de entrada no *script*.

De seguida, são filtradas as extensões dos ficheiros, retirando da leitura da diretoria os ficheiros com extensão ".pl" e ".xml" e todas as diretorias que comecem por ".", sendo apenas lidos os ficheiros com a extensão ".txt".

Por fim, são percorridos todos os ficheiros de meta-logs existentes na diretoria e respetivas linhas dos mesmos. Os campos dos meta-logs encontram-se separadas pelo caracter especial "&" que corresponde a uma *tag* XML. As *tags* tem o nome dos campos definidos no *output* do meta-log (*timestanp*, *source*, *level name*, *source computer*, *user*, *severity name*, *eventID*, *description*, *processID*, *threadID*, *filename* e *line*) e o seu conteúdo é preenchido automaticamente com o valor do campo correspondente.

Sempre que o *script* é executado, também é acrescentado um conjunto de *tags* com base nos campos existentes no meta-log. Estas *tags* descritas na Tabela 5.3Tabela 2.1 – Semelhanças e diferenças nos modelos XML e JSON., baseiam-se na informação refletida na Tabela 4.1 e é acrescentada ao *template*, após a última linha escrita.

Tabela 5.3 – Descrição das *tags* utilizadas no *template*.

Nome da tag	Descrição
`<timestamp> </timestamp>`	Conjunto de caracteres que fornece precisão de uma determinada ocorrência através da data e hora.
`<source> </source>`	Identifica o servidor que originou a ocorrência.
`<levelLogName> </levelLogName>`	Identifica a categoria do *log* para uma filtragem mais rápida das possíveis causas do problema.
`<user> </user>`	Identifica o utilizador autenticado no servidor que originou a ocorrência.
`<sourceComputer> </sourceComputer>`	Identifica o servidor que registou o *log*.
`<severityName> </severityName>`	Identifica o nível de prioridade na resolução ou tomada de alguma decisão perante o *log*.
`<eventID> </eventID>`	Identifica de forma única um determinado evento.
`<description> </description>`	Texto descritivo e resumido com o objetivo de enumerar caraterísticas próprias do problema.
`<processID> </processID>`	Identifica de forma única o processo que se refere ao *log*, quando aplicável.
`<threadID> </threadID>`	Identifica de forma única a *thread* que se refere ao *log*, quando aplicável.

Nome da tag	Descrição
`<file>` `</file>`	Ficheiro que indica a origem da mensagem de *log* antes do pré-processamento.
`<line>` `</line>`	Linha do ficheiro que indica a origem da mensagem de *log* antes do pré-processamento.

O código desenvolvido para criar o *template* encontra-se no Anexo C – "Código fonte da aplicação de fusão" deste relatório.

5.3. Implementação do relatório em Power BI

Para visualizar as informações no relatório em formato de *desktop layout* do *template*, obtido no subcapítulo 5.2, utilizou-se a tecnologia Power BI. A aplicação de visualização descrita no relatório foi implementada em Power BI *Desktop* e dividiu-se em duas partes: a primeira consiste em integrar o *data source*; a segunda consiste em desenvolver tabelas, gráficos e filtros para uma visualização mais apelativa dos *logs*.

A integração de *data source* consiste em selecionar o ficheiro do tipo XML, na opção "*Get Data*" e adicionar o *template* para que seja elaborado o relatório, como é exemplificado na Figura 5.7. Após selecionar o ficheiro "*template*", é necessário selecionar o nó "*event*" correspondente à *tag* "`<event></event>`", de modo a que exista leitura posterior das restantes *tags* com informação relevante.

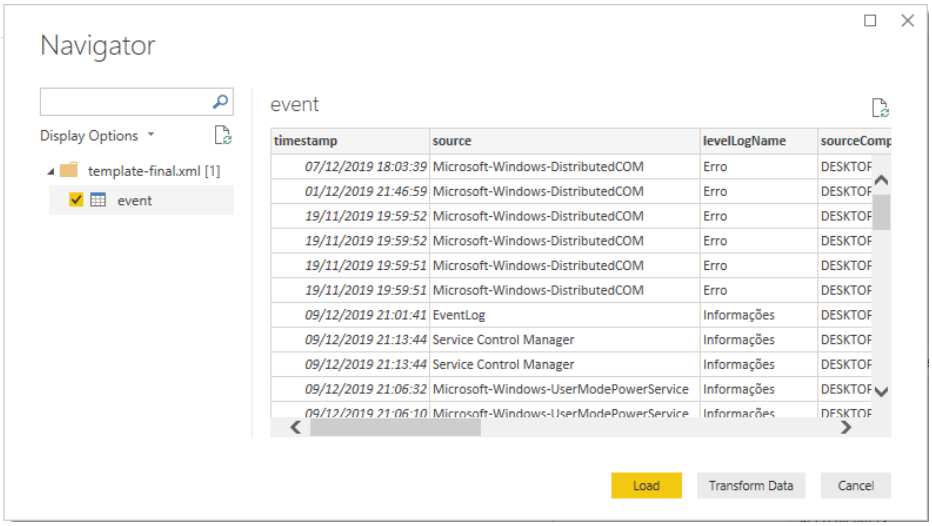


Figura 5.7 – Integração do *template* no Power BI.

A integração deste ficheiro originou alguns problemas, como referido na secção 6.3.1, que foi necessário contornar para tornar possível a implementação deste projeto.

De forma a contornar os problemas que ocorreram, dividiu-se o *template* com todos os meta-logs, nos seguintes ficheiros de menor tamanho:

- Um ficheiro apenas com os *logs* do tipo informativo do sistema operativo;
- Um ficheiro com os *logs* de erros de sistema do sistema operativo;
- Um ficheiro com alguns *logs* informativos e de erro de sistema.

Assim, foi possível implementar um relatório com três páginas, uma fazendo referência aos *logs* informativos, outra aos *logs* de erro e outra com os *logs* informativos e de erro.

5.3.1. Páginas "*Template* Windows-*Information*" e "*Template* Windows-*Error*" do relatório

As três páginas do relatório apresentam uma estrutura, dividindo-se nas seguintes quatro formas para visualizar a informação:

- **Filtros**, permitem visualizar os dados do ficheiro de XML em conjuntos menores, disponibilizando a visualização mais pormenorizada nas páginas do relatório. Neste relatório foram utilizados dois "*Slicer*", um que filtra por ficheiro de meta-logs (DCOM.txt, SCM.txt, TPM.txt, entre outros) e outro com o intervalo temporal (mês-dia-ano) em que ocorreram os *logs*. A Figura 5.8, ilustra os "*Slicer*" implementados.

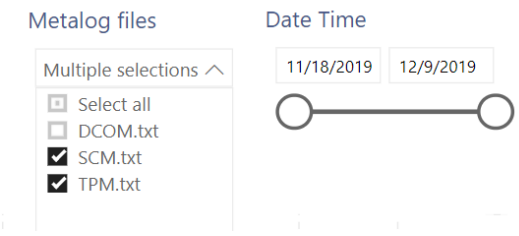


Figura 5.8 – Filtros implementados no relatório em Power BI.

O conteúdo presente na *tag* `<file></file>` é extenso, sendo parte dele redundante com o título do relatório e irrelevante à leitura e análise para o utilizador. Deste modo, foi utilizada a função "split" na coluna "*file*" (coluna que corresponde à *tag*

<*file*></*file*> do ficheiro de XML), separada pelo caracter especial "_". Assim, obteve-se a coluna "*file* - *split*", como ilustra a Figura 5.9.

Esta formatação dos dados com a função "*split*", não é replicada para a origem dos dados e foi realizada através do "Editor do *Power Query*" disponível no Power BI *Desktop* [58].

file	file - split
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_6_windows_system_erro_DCOM.txt	DCOM.txt
dataset_3_windows_system_erro_SCM.txt	SCM.txt
dataset_3_windows_system_erro_SCM.txt	SCM.txt
dataset_3_windows_system_erro_SCM.txt	SCM.txt
dataset_24_windows_system_erro_TPM.txt	TPM.txt
dataset_24_windows_system_erro_TPM.txt	TPM.txt
dataset_24_windows_system_erro_TPM.txt	TPM.txt
dataset_24_windows_system_erro_TPM.txt	TPM.txt

Figura 5.9 – Implementação da função "*split*" no relatório em Power BI.

- **Contagens**, representam a contagem (*count*) de um determinado valor (*tag*). Utilizou-se o "*Card*" para representar a quantidade de eventos, quantidades de ficheiros e a quantidade de *Sources*, consoante os filtros selecionado, como ilustra a Figura 5.10.

Figura 5.10 – Contagens implementadas no relatório em Power BI.

- **Gráficos**, ilustra os dados recebidos em informação numérica mediante uma ou mais linhas que permitem visualizar a relação entre os mesmos. Implementaram-se dois tipos de gráficos, "*Clustered bar chart*" que apresenta a quantidade de *logs* por meta-log e outro que mostra a quantidade de tipos de eventos por meta-log; "*Pie chart*" que agrupa os ID dos eventos e apresenta o valor de ocorrências em percentagem,

consoantes os filtros selecionados. A Figura 5.11, ilustra os tipos de gráficos implementados.

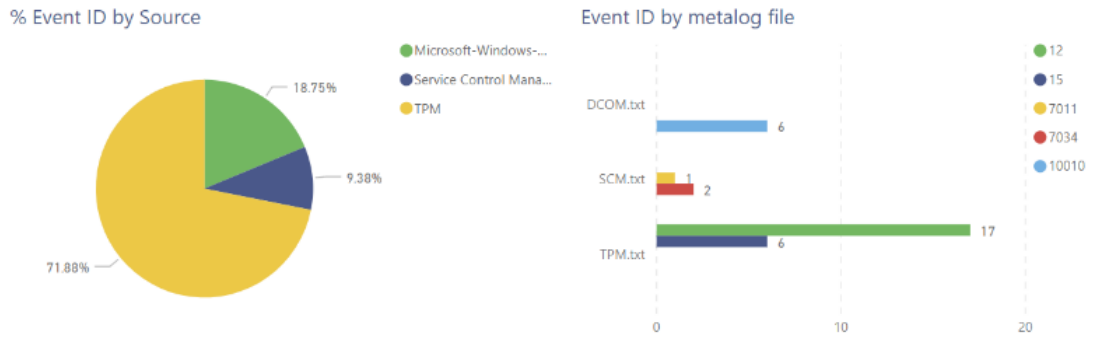
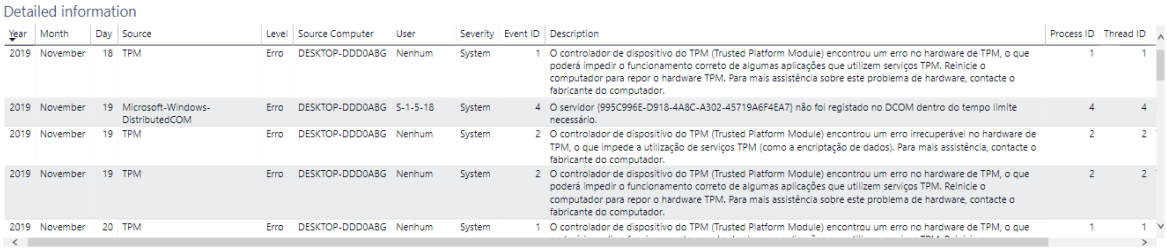


Figura 5.11 – Gráficos implementados no relatório em Power BI.

- **Tabela**, representa a informação presente nas múltiplas *tags* do ficheiro de XML, mas formatada em linhas e colunas. Implementou-se uma tabela em que cada coluna é uma *tag*, consoante os filtros selecionados. A Figura 5.12, ilustra a tabela implementada.



Year	Month	Day	Source	Level	Source Computer	User	Severity	Event ID	Description	Process ID	Thread ID
2019	November	18	TPM	Erro	DESKTOP-DDD0ABG	Nenhum	System	1	O controlador de dispositivo do TPM (Trusted Platform Module) encontrou um erro no hardware de TPM, o que poderá impedir o funcionamento correto de algumas aplicações que utilizem serviços TPM. Reinicie o computador para repor o hardware TPM. Para mais assistência sobre este problema de hardware, contacte o fabricante do computador.	1	1
2019	November	19	Microsoft-Windows-DistributedCOM	Erro	DESKTOP-DDD0ABG	S-1-5-18	System	4	O servidor {995C996E-D918-4A8C-A302-45719A6F4EA7} não foi registado no DCOM dentro do tempo limite necessário.	4	4
2019	November	19	TPM	Erro	DESKTOP-DDD0ABG	Nenhum	System	2	O controlador de dispositivo do TPM (Trusted Platform Module) encontrou um erro irrecuperável no hardware de TPM, o que impede a utilização de serviços TPM (como a encriptação de dados). Para mais assistência, contacte o fabricante do computador.	2	2
2019	November	19	TPM	Erro	DESKTOP-DDD0ABG	Nenhum	System	2	O controlador de dispositivo do TPM (Trusted Platform Module) encontrou um erro no hardware de TPM, o que poderá impedir o funcionamento correto de algumas aplicações que utilizem serviços TPM. Reinicie o computador para repor o hardware TPM. Para mais assistência sobre este problema de hardware, contacte o fabricante do computador.	2	2
2019	November	20	TPM	Erro	DESKTOP-DDD0ABG	Nenhum	System	1	O controlador de dispositivo do TPM (Trusted Platform Module) encontrou um erro no hardware de TPM, o que	1	1

Figura 5.12 – Tabela implementada no relatório em Power BI.

Através da opção "*Horizontal Scroll*", disponível na tabela da Figura 5.12, é possível o utilizador consultar o ficheiro e a linha original no ficheiro de texto, de cada mensagem de log, como ilustra a Figura 5.13, nas colunas "*File Name*" e "*File Line*".

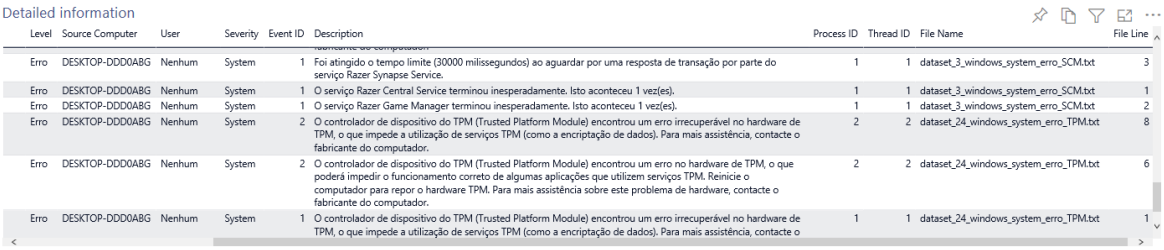


Figura 5.13 – Tabela com ficheiro e linha originária da mensagem de *log* implementada no relatório em Power BI.

A Figura 5.14 ilustra a página "*Template* Windows – *Error*" do relatório publicado na *cloud*, com os filtros disponíveis.

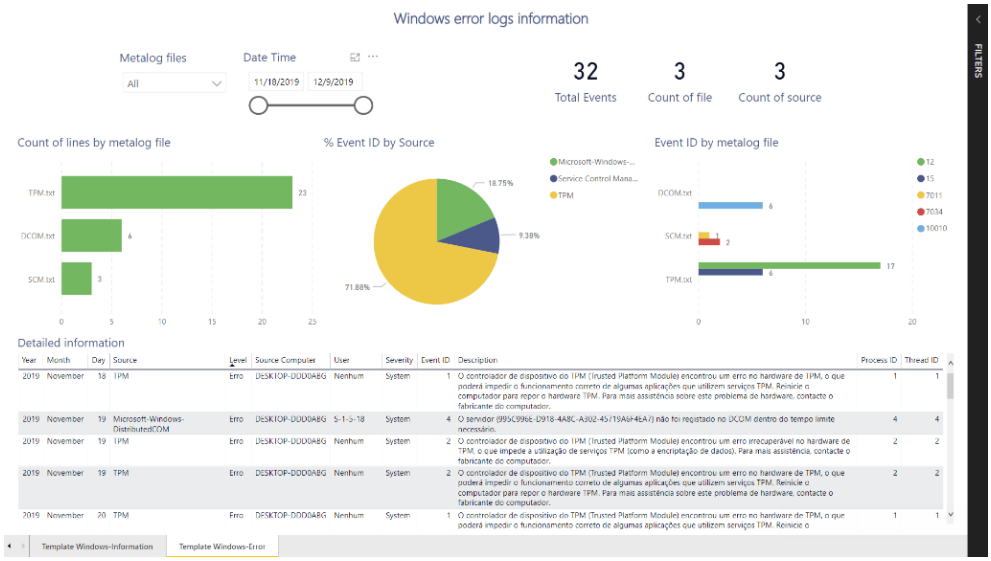


Figura 5.14 – Visualização do relatório implementado em Power BI.

A Figura 5.15 ilustra a página "*Template* Windows – *Error*" do relatório publicado na *cloud*, com informações dos ficheiros "SCM.txt" e "TPM.txt".

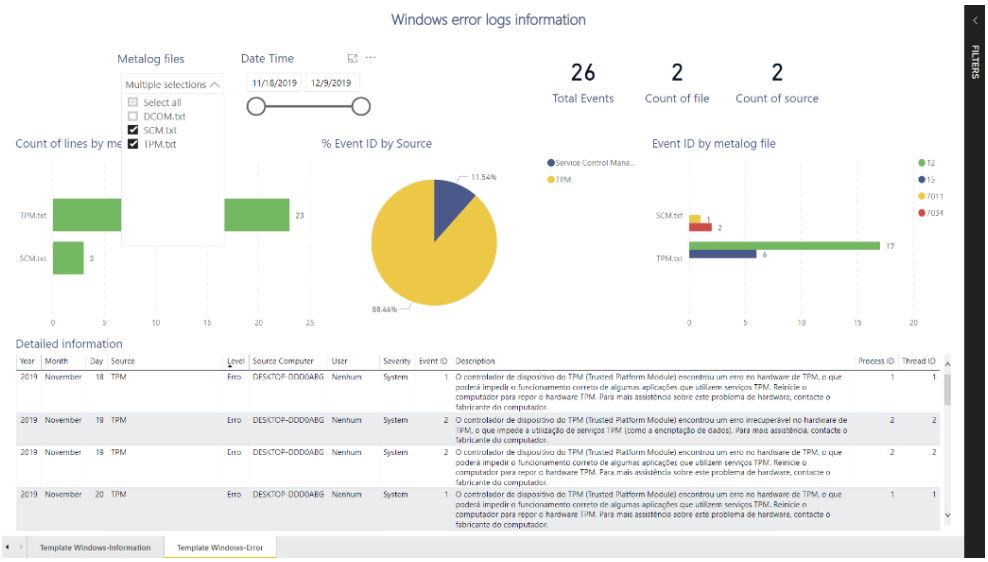


Figura 5.15 – Visualização do relatório implementado, com interação do filtro "Meta-log *files*" em Power BI.

A Figura 5.16 ilustra iteratividade disponível na página "*Template* Windows – *Error*" do relatório publicado na *cloud*, com informações dos ficheiros "SCM.txt" e "TPM.txt".

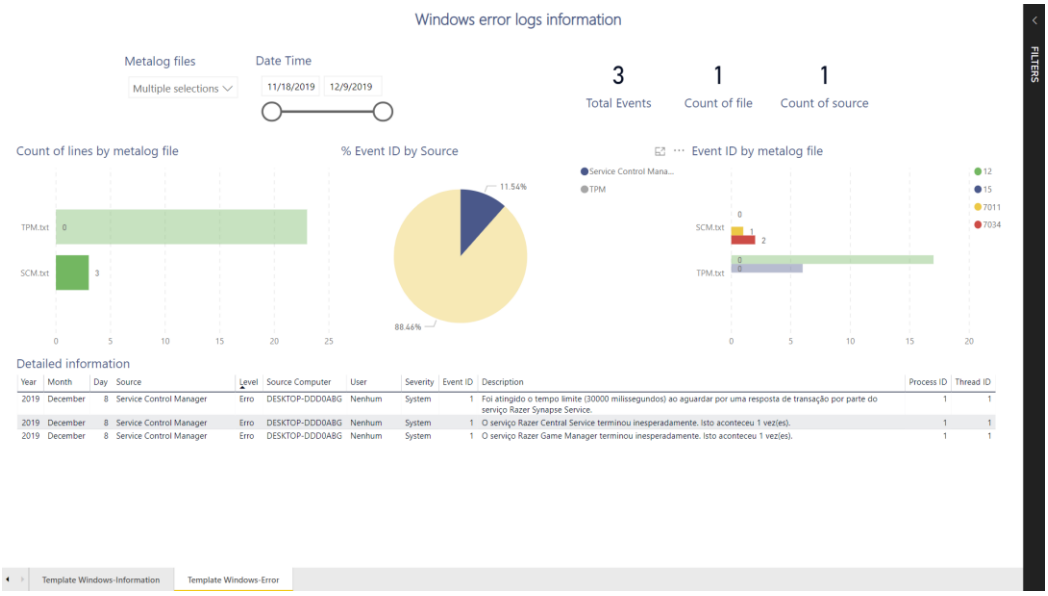


Figura 5.16 – Visualização do relatório implementado, com interação do filtro "Meta-log *files*" e a seleção de um determinado valor no relatório em Power BI.

5.3.2. Página "*Template* Windows" do relatório

A página "*Template* Windows" do relatório têm como base a estrutura e conteúdos presentes nas páginas da secção 5.3.1, no entanto foi implementado a atribuição de cores (vermelho,

amarelo e azul) consoante o nível de severidade do evento ocorrido na tabela ilustrada pela Figura 5.12.

A Figura 5.17, ilustra as cores vermelho, amarelo e azul que correspondem aos níveis de severidade “Erro”, “Alerta” e “Informações”, respetivamente.

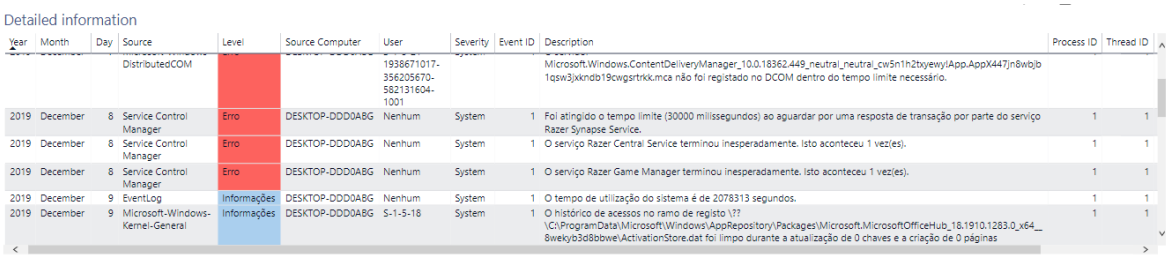


Figura 5.17 – Tabela com cores pelos vários níveis de severidade dos *logs*.

A implementação das cores na tabela é realizada na opção “*Measure* “, existente na secção “Editor do *Power Query*”. Através desta opção implementou-se uma métrica designada por “ColorID”, que utiliza a função “switch” para associar a cada nível de severidade presente na coluna “levelLogName” (Erro, Alerta e Informações) do ficheiro XML “Windows_system_info_error”, os números inteiros 1, 2 e 3. A Figura 5.18, ilustra a função implementada que associa um nível de severidade a um número inteiro.

A linguagem utilizada para implementar esta função designa-se por DAX. O DAX agrega funções, operadores e constantes para serem utilizadas em fórmulas, ou expressões, para calcular e devolver um ou mais valores [59].

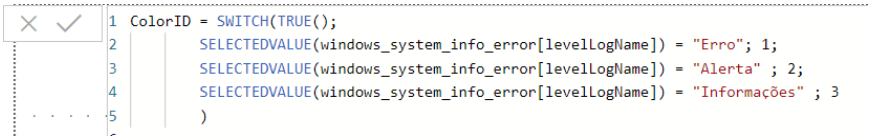


Figura 5.18 – Implementação de função DAX em Power BI.

Após definir o número inteiro correspondente ao nível de severidade existente, é associada a respetiva cor como ilustra a Figura 5.19, na coluna “levelLogName” da tabela implementada.

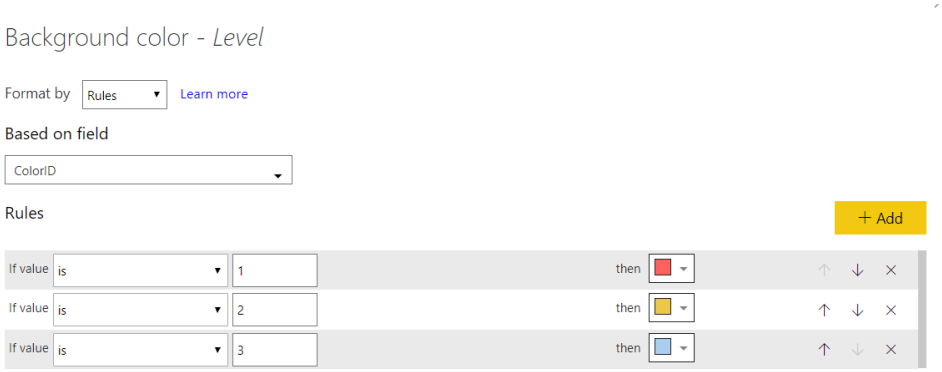


Figura 5.19 – Correspondência das cores com o nível de severidade na tabela em Power BI.

Por fim, o relatório foi publicado na *cloud*, usufruindo das funcionalidades presentes na componente "Power BI *services*", disponibiliza a consulta dos dados em qualquer lugar e momento, através da opção "*Publish*" disponível do Power BI *Desktop*.

Esta publicação requer que a conta de utilizador utilizada no "*Sign In*", seja Office 365 com a respetiva licença de Power BI.

5.3.3. Envio automático do relatório

Com o objetivo de automatizar o processo de leitura dos dados, aumentar a produtividade e aumento de monitorização da infraestrutura e aplicações, implementou-se o envio automático, por email da página "*Template* Windows" do relatório para um utilizador. Esta configuração requer o preenchimento dos seguintes campos:

- ***Subscribe***, email do(s) utilizador(es) que vão receber o relatório;
- ***Report page***, selecionar a página do relatório que será enviada por email;
- ***Frequency***, define a frequência (hora a hora, diariamente, semanalmente, mensalmente e depois de um *refresh* - diariamente) que o email será enviado
- ***Scheduled Time***, a hora a que será envio o email, tendo em conta o fuso horário selecionado. Este campo terá em consideração o conteúdo definido no campo "*Frequency*";
- ***Interval date*** (*start date – end date*), define o intervalo de tempo (mês-dia-ano) que o relatório será enviado por email, tendo em conta o conteúdo definido nos campos "*Scheduled Time*" e "*Frequency*".

Deste modo, na secção "*Subscribe*" disponível na compomente "Power BI services", configurou-se a página "*Template* Windows" do relatório para que seja enviada diariamente, às 7:30 para o email do utilizador Carolina Dias. Este envio começou no dia 19 de abril de 2020 e não foi definida data de término. A Figura 5.20, ilustra os campos referidos corretamente preenchidos.

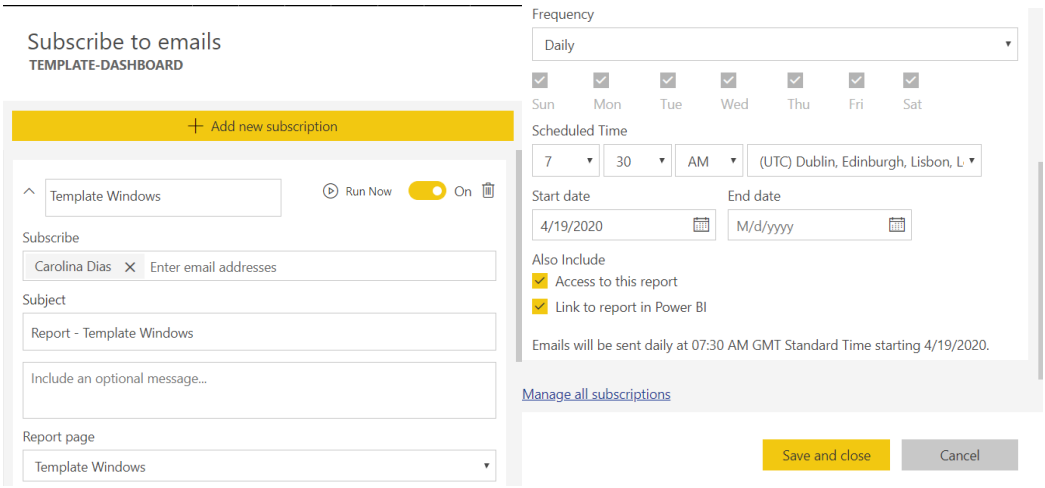


Figura 5.20 – Configuração do envio por email do relatório em Power BI.

Os Anexo D – "Utilização do *template* com os meta-logs informativos de sistema no relatório implementado em Power BI"e Anexo E – "Utilização do *template* com os meta-logs informativos e erros de sistema no relatório implementado em Power BI" descrevem a forma de visualização das páginas completas do relatório "*Template* Windows-*Information*" e "*Template* Windows" que foram implementadas no projeto.

6. Testes e Resultados

Este capítulo justifica a metodologia utilizada para a realização dos testes, com base nos componentes desenvolvidos para esta solução.

Existem inúmeras metodologias de testes que podem ser utilizadas durante o processo de desenvolvimento da aplicação, de modo a que os requisitos da mesma operem com o sucesso esperado em vários ambientes e/ou plataformas.

Os testes podem ser categorizados por funcionais e não funcionais. Os testes funcionais envolvem testes entre a aplicação e os requisitos de negócio, como por exemplo testes unitários, de integração, de sistemas e de aceitação. Enquanto que os testes não funcionais focam-se nos aspetos operacionais da aplicação, como por exemplo testes de desempenho, segurança, usabilidade e compatibilidade.

A realização dos testes para esta solução incidiu na usabilidade. Os testes de usabilidade, consistem em avaliar a facilidade de utilização da aplicação numa perspetiva de utilizador final. Isto é, tem como objetivo avaliar se o desenho e a estética da aplicação correspondem ao fluxo de trabalho do público alvo [60].

Após analisar as várias metodologias de testes, decidiu-se dividi-los em três fases: a aplicação para gerar os meta-logs: a aplicação de fusão dos vários meta-logs; a integração do *template* (ficheiro que origina a fusão dos vários meta-logs) no Power BI.

6.1. Estrutura e organização dos Meta-logs

Neste subcapítulo encontram-se os testes realizados às duas abordagens utilizadas para a implementação da aplicação de meta-log.

6.1.1. Implementação com logs de múltiplas origens num único ficheiro

No primeiro teste recolheram-se dois *logs* informativos dos serviços *Microsoft-Windows-Time-Service* e *Microsoft-Windows-Kernel-General* e um de erro do serviço *Microsoft-Windows-DistributedCOM*, de sistema através do software *Event Viewer* de uma máquina com sistema operativo Windows 10 para um ficheiro de texto.

Após a execução do *script*, detetaram-se alguns problemas no ficheiro de texto de entrada, nomeadamente quebras de linhas entre as linhas com *logs*, quebras de linha na descrição do *log* e quebras de linhas no final do ficheiro. Também se identificou que os *logs* informativos tinham menos campos com informações do que os *logs* de erro. Desta forma, decidiu-se remover todas as quebras de linha e separar os *logs* por *level*, por ficheiro para tornar a análise dos campos dos *logs* mais simples e fidedigna. A Figura 6.1 ilustra o exemplo do *output* alcançado após as alterações realizadas.

```
---------------Log---------------
Date: 19/11/2019 13:30:08
Source: Microsoft-Windows-Time-Service
Level Log Name: System
Source Computer: LPT-DIASCAR.
User: S-1-5-19
Severity Name: Information
Event ID: 158
Description: The time provider 'VMICTimeProvider' has indicated that the current hardware and operating
environment is not supported and has stopped. This behavior is expected for VMICTimeProvider on non-
HyperV-guest environments. This may be the expected behavior for the current provider in the current
operating environment as well.
Process ID: 1488
Thread ID: 19188
---------------Log---------------
Date: 19/11/2019 13:30:10
Source: Microsoft-Windows-Kernel-General
Level Log Name: System
Source Computer: LPT-DIASCAR.
User: S-1-5-18
Severity Name: Information
Event ID: 16
Description: The access history in hive \??\C:\Users\             \AppData\Local\Packages
\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Settings\settings.dat was cleared updating 5 keys and creating 1
modified pages.
Process ID: 20112
Thread ID: 13168
```

Figura 6.1 – *Output* do meta-log com *logs* informativos no primeiro teste.

O segundo teste foi realizado com base nas alterações implementadas no *script* testado anteriormente. Os *data sources* utilizados também foram retirados de uma máquina com sistema operativo Windows 10 e guardados num ficheiro de texto. Estes *logs* são dos serviços *Service Control Manager*, TPM, *Microsoft-Windows-UserModePowerService*, *Microsoft-Windows-Kernel-Power*, *Microsoft-Windows-WindowsUpdateClient*, no caso de *logs* de informação e de TPM, *Service Control Manager*, *Microsoft-Windows-DistributedCOM* para os de erro.

Iniciou-se o teste com um ficheiro de *logs* informativos, composto por 221 *logs*. Após executar o *script* pela primeira vez, detetou-se que alguns campos definidos não estavam

corretos, ou seja, os campos de origem, severidade, utilizador, descrição, ID do processo, ID da *thread* e *level* variavam consoante a origem.

Foi necessário analisar as origens dos *logs* e alterar o *array* implementado conforme a posição do campo. A Figura 6.2, mostra um exemplo de *output* do meta-log.

```
---------------Log---------------
Date: 09/12/2019 21:06:32
Source: Microsoft-Windows-UserModePowerService
Level Log Name: Informações
Source Computer: DESKTOP-DDD0ABG
User: S-1-5-18
Severity Name: System
Event ID: 12
Description: O processo C:\Program Files (x86)\ASUS\ASUS Smart Gesture\AsTPCenter\x64\AsusTPCenter.exe
(ID do processo:5916) repôs o esquema de políticas de {8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c} para
{8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c}
Process ID: 968
Thread ID: 7352
---------------Log---------------
Date: 09/12/2019 21:06:22
Source: Microsoft-Windows-Kernel-Power
Level Log Name: System
Source Computer: DESKTOP-DDD0ABG
User: S-1-5-18
Severity Name: Info
Event ID: 105
Description: Alteração da fonte de energia.
Process ID: 4
Thread ID: 1748
```

Figura 6.2 – *Output* do meta-log com *logs* de erro no segundo teste.

Depois de testar os *logs* de informação, executou-se o mesmo processo para os de erro, alterando-se o ficheiro, composto por 34 *logs* de erro, o nome do ficheiro de saída para guardar os dados processados e os campos de origem, severidade, utilizador, descrição, ID do processo, ID da *thread* e *level*. Após realizadas estas alterações, o resultado do *script* foi o esperado, como ilustra a Figura 6.3.

```
---------------Log---------------
Date: 07/12/2019 18:03:47
Source: TPM
Level Log Name: Erro
Source Computer: DESKTOP-DDD0ABG
User: Nenhum
Severity Name: System
Event ID: 12
Description: O controlador de dispositivo do TPM (Trusted Platform Module) encontrou um erro no hardware
de TPM, o que poderá impedir o funcionamento correto de algumas aplicações que utilizem serviços TPM.
Reinicie o computador para repor o hardware TPM. Para mais assistência sobre este problema de hardware,
contacte o fabricante do computador.
Process ID: 4
Thread ID: 268
---------------Log---------------
Date: 07/12/2019 18:03:39
Source: Microsoft-Windows-DistributedCOM
Level Log Name: Erro
Source Computer: DESKTOP-DDD0ABG
User: S-1-5-21-1938671017-356205670-582131604-1001
Severity Name: System
Event ID: 10010
Description: O servidor
Microsoft.Windows.ContentDeliveryManager_10.0.18362.449_neutral_neutral_cw5n1h2txyewy!
App.AppX447jn8wbjb1qsw3jxkndb19cwgsrtrkk.mca não foi registado no DCOM dentro do tempo limite
necessário.
Process ID: 608
Thread ID: 17296
```

Figura 6.3 – *Output* do meta-log com *logs* informativos no segundo teste.

No terceiro teste, alterou-se o *output* dos ficheiros de meta-log pois verificou-se no teste anterior que o ficheiro tinha muitas linhas e informação desnecessária (ex. Date:) que tornava a implementação do *template* XML mais complexa, conforme se ilustra na Figura 6.4. Contudo, esta informação adicional, que permitiu validar os campos dos *logs* permitiu validar os campos dos *logs*.

```
09/12/2019 21:08:55&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:08:54&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:08:54&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:06:32&Microsoft-Windows-UserModePowerService&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&12&O processo C:\Program Files (x86)\ASUS\ASUS Smart Gesture
\AsTPCenter\x64\AsusTPCenter.exe (ID do processo:5916) repôs o esquema de políticas de {8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c} para {8c5e7fda-e8bf-4a96-9a85-
a6e23a8c635c}&968&7352
09/12/2019 21:06:22&Microsoft-Windows-Kernel-Power&System&DESKTOP-DDD0ABG&S-1-5-18&Info&105&Alteração da fonte de energia.&4&1748
09/12/2019 21:06:10&Microsoft-Windows-UserModePowerService&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&12&O processo C:\Program Files (x86)\ASUS\ASUS Smart Gesture
\AsTPCenter\x64\AsusTPCenter.exe (ID do processo:5916) repôs o esquema de políticas de {8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c} para {8c5e7fda-e8bf-4a96-9a85-
a6e23a8c635c}&968&7352
09/12/2019 21:06:10&Microsoft-Windows-Kernel-Power&System&DESKTOP-DDD0ABG&S-1-5-18&Info&105&Alteração da fonte de energia.&4&13392
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:47&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
09/12/2019 21:04:46&TPM&Informações&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.&4&268
```

Figura 6.4 – Atualização do *output* do meta-log com *logs* informativos no terceiro teste.

6.1.2. Implementação apenas com uma origem de *logs* por ficheiro

Como referido no subcapítulo 5.1, os testes realizados na secção 6.1.1 à abordagem utilizada seria impraticável. Assim, no ficheiro “confg.txt” adicionaram-se as posições pretendidas, consoante o nome do serviço e o nível da ocorrência.

De seguida, executou-se o *script* com os ficheiros de entrada e o de saída, através do seguinte comando:

```
./metalog_windows.pl confg.txt
dataset_164_windows_system_info_TPM.txt
metalog_information_TPM.txt
```

Caso o utilizador não preencha o segundo ou terceiro parâmetro com o nome correto, é apresentada a mensagem "Ficheiro não existe" no ecrã, como ilustra a Figura 6.5. Assim como, se a execução da aplicação não respeitar o número de parâmetros de entrada definido, mostra a mensagem "Número de parâmetros incompleto.", como ilustra a Figura 6.6.

```
ubuntu@ubuntu:~/MCIF/Metalog/Teste-Nfiles_V3/Windows_Sistema_Teste1$ ./metalog_windows.pl confg.txt
dataset_1_windows_system_info_TPM.txt metalog_information_TPM.txt
File doesn't exists.
```

Figura 6.5 – Mensagem de erro "Ficheiro não existe.".

```
ubuntu@ubuntu:~/MCIF/Metalog/Teste-Nfiles_V3/Windows_Sistema_Teste1$ ./metalog_windows.pl confg.txt
dataset_164_windows_system_info_TPM.txt
Incomplete number of files.
```

Figura 6.6 - Mensagem de erro "Número de parâmetro incompleto.".

As mensagens de erro refletidas na Figura 6.5 e Figura 6.6, também são redirecionadas para um ficheiro de texto centralizado, ilustrado na Figura 6.7. Este ficheiro agrega as validações dos parâmetros para executar a aplicação de pré-processamento dos meta-logs, com os campos "*Datetime*", "*Source*", "*User*" e "*Message*".

```
Datetime:07-06-2020 20:54:55
 Source:ubuntu
 User:ubuntu
 Message:File doesn't exists.
Datetime:07-06-2020 20:54:58
 Source:ubuntu
 User:ubuntu
 Message:Incomplete number of files.
```

Figura 6.7 – Ficheiro com as mensagens de validação dos parâmetros dos meta-logs.

O ficheiro de entrada, "confg.txt", reflete as posições dos campos do log e o delimitador que separa esses campos, como ilustra a Figura 6.8.

```
#Datetime, Source, Level, Source Computer, User, Serverity, Event ID, Description, Process ID, Thread ID, Delimitador
1,2,0,8,5,7,3,13,9,10,\t
```

Figura 6.8 – Ficheiro "confg.txt" utilizado para testar o funcionamento do *script*.

O ficheiro de entrada com os *logs* informativos do serviço TPM, ilustrado na Figura 6.9, é composto por 166 linhas.

```
Informações    09/12/2019 21:08:55    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:08:54    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:08:54    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:47    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:46    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
Informações    09/12/2019 21:04:46    TPM    17    Nenhum    S-1-5-18    Info    System  DESKTOP-DDD0ABG 4    268
```

Figura 6.9 – Parte do ficheiro de entrada com os *logs* informativos do serviço TPM.

Após a execução do *script* (Figura 5.4) o resultado do *output* foi o esperado, como ilustra a Figura 6.10. O meta-log era composto por 166 linhas, apenas com os campos definidos no ficheiro "confg.txt", ilustrado na Figura 5.3, com a indicação do nome do ficheiro e número da linha e separados por caracter especial "&".

```
dataset_167_windows_system_info_TPM.txt&1&09/12/2019 21:08:55&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&2&09/12/2019 21:08:54&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&3&09/12/2019 21:08:54&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&4&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&5&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&6&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&7&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&8&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&9&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) ao
dataset_167_windows_system_info_TPM.txt&10&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) a
dataset_167_windows_system_info_TPM.txt&11&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) a
dataset_167_windows_system_info_TPM.txt&12&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) a
dataset_167_windows_system_info_TPM.txt&13&09/12/2019 21:04:47&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) a
dataset 167 windows system info TPM.txt&14&09/12/2019 21:04:46&TPM&DESKTOP-DDD0ABG&S-1-5-18&System&17&Falha do hardware do TPM (Trusted Platform Module) a
```

Figura 6.10 - *Output* do ficheiro de meta-log do serviço TPM.

Este teste realizou-se nos *logs* informativos dos serviços EventLog, SCM, TPM, UMPS, WKB, WKG, WKP, WLogon, WTP, WUC e com os *logs* de erro dos serviços DCOM, SCM e TPM e o resultado foi o esperado (semelhante à figura anterior).

Por fim, para o quarto teste recolheram-se 188 *logs* aplicacionais do serviço ESEN também do sistema operativo Windows. Estes *logs* foram divididos em dois ficheiros de texto, um com os 72 *logs* de erro e o outro com os 116 *logs* informativos.

Este teste é semelhante ao anterior, foram atualizadas as posições dos campos do *log* e o delimitador que separa os campos dos *logs* no ficheiro de entrada, "confg.txt". De seguida,

executou-se o *script* com os ficheiros de entrada e o de saída. O resultado foi também obtido com sucesso, como ilustra a Figura 6.11.

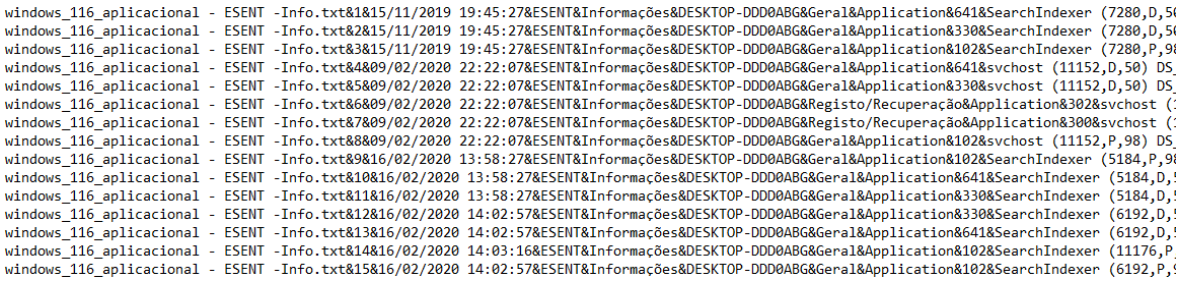


Figura 6.11 - *Output* do ficheiro de meta-log do serviço ESEN.

Realizou-se o mesmo teste para os *logs* informativos do mesmo serviço e o resultado foi também obtido com sucesso (semelhante ao apresentado na figura anterior).

6.2.Estrutura e organização do *Template*

Neste subcapítulo encontram-se os testes realizados com o *template* XML. São descritos os testes à estrutura das *tags* e ao preenchimento automático das mesmas.

6.2.1. Definição das *tags*

Implementou-se o ficheiro de XML manualmente para testar a estrutura e organização das *labels* referida no subcapítulo 4.2. Foram utilizados *logs* de sistema do tipo informação e aplicacional dos tipos informativos e de erro, retirados do *Event Viewer* de um sistema operativo Windows 10, como ilustrado na Figura 6.12.

Figura 6.12 – *Output* no *browser* IE do ficheiro XML criado manualmente.

6.2.2. Ficheiro XML dinâmico

Com base no ficheiro XML estruturado anteriormente, o *template* foi desenvolvido e testado várias vezes até chegar à versão final.

O primeiro teste consistiu na implementação de forma automática das *tags* XML, indentação e preenchimento do respetivo conteúdo nas *tags*, utilizando o meta-log com *logs* de informação, obtido e validado na secção 6.1.2. A Figura 6.13 mostra um excerto o *output* no *browser* IE do teste.

```
<?xml version="1.0" encoding="UTF-8"?>
- <eventLog eventCategory="system">
    <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
    <description>O tipo de inicio do serviço Serviço de Transferência Inteligente em Segundo Plano foi modificado de Iniciar
      automaticamente para Início pedido.</description>
    <eventID>7040</eventID>
    <user>S-1-5-18</user>
    <levelLogName>Informações</levelLogName>
    <severityName>System</severityName>
    <threadID>2480</threadID>
    <source>Service Control Manager</source>
    <timestamp>09/12/2019 21:13:44</timestamp>
    <processID>796</processID>
    <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
    <description>Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.</description>
    <eventID>17</eventID>
    <user>S-1-5-18</user>
    <severityName>System</severityName>
    <levelLogName>Informações</levelLogName>
    <processID>4</processID>
    <threadID>268</threadID>
    <source>TPM</source>
    <timestamp>09/12/2019 21:08:55</timestamp>
    <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
    <description>Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.</description>
    <eventID>17</eventID>
    <user>S-1-5-18</user>
    <severityName>System</severityName>
    <levelLogName>Informações</levelLogName>
    <threadID>268</threadID>
    <timestamp>09/12/2019 21:08:54</timestamp>
    <source>TPM</source>
    <processID>4</processID>
    <processID>4</processID>
    <source>TPM</source>
    <threadID>268</threadID>
    <timestamp>09/12/2019 21:08:54</timestamp>
    <severityName>System</severityName>
    <levelLogName>Informações</levelLogName>
    <user>S-1-5-18</user>
    <eventID>17</eventID>
    <description>Falha do hardware do TPM (Trusted Platform Module) ao executar um comando de TPM.</description>
    <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
    <processID>968</processID>
    <timestamp>09/12/2019 21:06:32</timestamp>
    <threadID>7352</threadID>
    <source>Microsoft-Windows-UserModePowerService</source>
    <levelLogName>Informações</levelLogName>
    <severityName>System</severityName>
    <description>O processo C:\Program Files (x86)\ASUS\ASUS Smart Gesture\AsTPCenter\x64\AsusTPCenter.exe (ID do
      processo:5916) repôs o esquema de políticas de {8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c} para {8c5e7fda-e8bf-4a96-9a85-
```


Figura 6.13 – *Output* no *browser* IE com *logs* informativos no segundo teste.

Foi realizado o mesmo teste com o meta-log de *logs* de erro, também obtido e validado na secção 6.1.2. Para este teste, apenas foi necessário substituir o ficheiro de entrada que contém os *logs* informativos para os de erro, como ilustra a Figura 6.14.

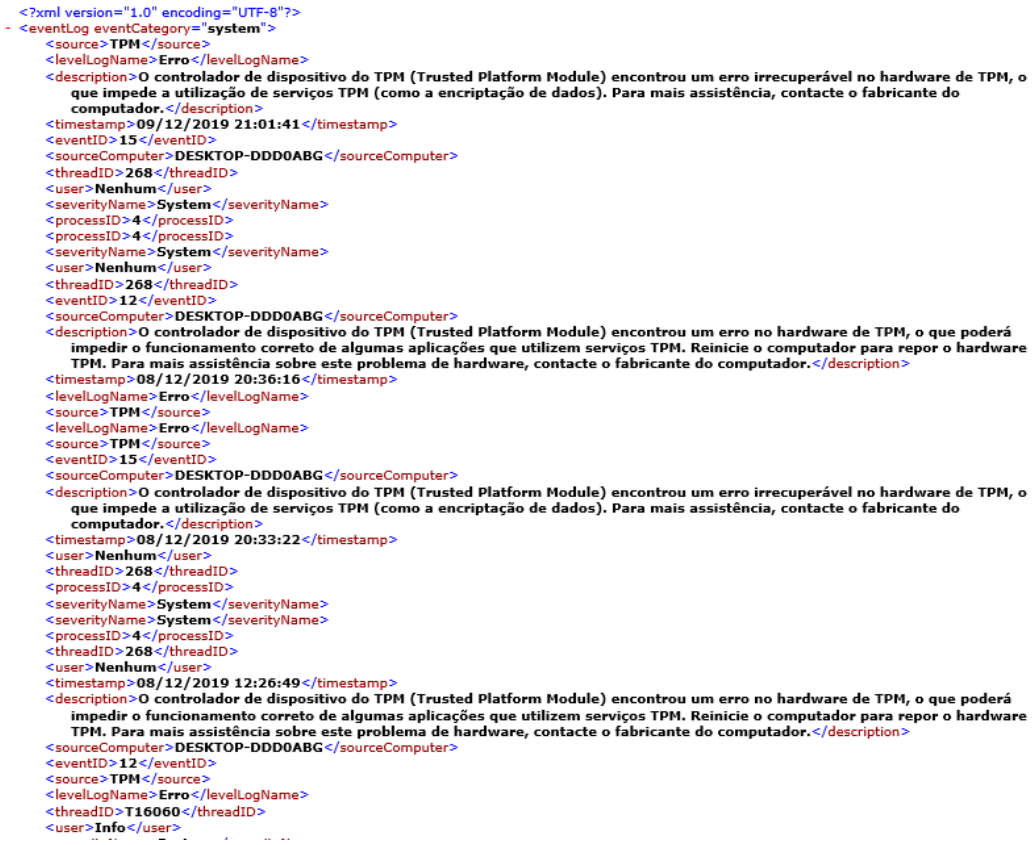


Figura 6.14 – *Output* no *browser* IE com *logs* de erro no segundo teste.

Com este teste, verificou-se em ambos os meta-log, que as *tags* não estavam ordenadas e identificadas como definido na subseção anterior. Tal deve-se ao facto de ter sido utilizado o módulo "XML::LibXML", baseado na biblioteca libxml2, escrita na linguagem C [61].

A solução encontrada foi escrever as *tags* no código PERL estaticamente. Desta forma a organização e indentação estaria sempre correta no *output*. Após ser implementada esta solução, testou-se com os dois meta-logs referidos anteriormente e o resultado foi o esperado. O *output* do *template* encontra-se organizado, com os *logs* dos dois níveis e indentado, como ilustra a Figura 6.15.

Figura 6.15 – *Output* no *browser* IE com *logs* de erro e informação no terceiro teste.

O segundo teste, foi realizado com base nos meta-logs obtidos dos testes realizados na secção 6.1.2. Criou-se uma diretoria na máquina virtual que contém esses meta-logs e o *script*, como ilustra a Figura 6.16.

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ll
total 900
drwxrwxrwx 2 ubuntu ubuntu   4096 May 17 14:05 ./
drwxr-xr-x 5 ubuntu ubuntu   4096 Jun  7 12:57 ../
-rw-rw-r-- 1 ubuntu ubuntu  23780 Apr 11 14:32 metalog_aplicacional_error_esent.txt
-rw-rw-r-- 1 ubuntu ubuntu  42632 Apr 11 14:33 metalog_aplicacional_info_esent.txt
-rw-rw-r-- 1 ubuntu ubuntu   1829 Apr 10 16:51 metalog_DCOM.txt
-rw-rw-r-- 1 ubuntu ubuntu    475 Apr 10 16:51 metalog_information_SCM.txt
-rw-rw-r-- 1 ubuntu ubuntu  33511 Apr 10 16:56 metalog_information_TPM.txt
-rw-rw-r-- 1 ubuntu ubuntu   2962 Apr 10 16:59 metalog_information_UMPS.txt
-rw-rw-r-- 1 ubuntu ubuntu    950 Apr 10 16:59 metalog_information_WKB.txt
-rw-rw-r-- 1 ubuntu ubuntu   4936 Apr 10 17:00 metalog_information_WKG.txt
-rw-rw-r-- 1 ubuntu ubuntu    637 Apr 10 17:00 metalog_information_WKP.txt
-rw-rw-r-- 1 ubuntu ubuntu    256 Apr 10 17:00 metalog_information_WLogon.txt
-rw-rw-r-- 1 ubuntu ubuntu    386 Apr 10 17:01 metalog_information_WTP.txt
-rw-rw-r-- 1 ubuntu ubuntu   3259 Apr 10 17:01 metalog_information_WUC.txt
-rw-rw-r-- 1 ubuntu ubuntu    708 Apr 10 17:02 metalog_SCM.txt
-rw-rw-r-- 1 ubuntu ubuntu   9826 Apr 10 17:03 metalog_TPM.txt
-rw-rw-r-- 1 ubuntu ubuntu    186 Apr 10 17:04 metalog_windows_eventlog.txt
-rwxrwxrwx 1 ubuntu ubuntu   1591 Apr  2 10:39 metalog_windows_Nfiles.pl*
```

Figura 6.16 – Diretoria com os ficheiros necessários para testar o *template*.

Após realizadas as alterações necessárias no *script* referidas no subcapítulo 5.2, executou-se o *script* com os parâmetros de entrada, a diretoria com o nome da diretoria que contém todos os meta-logs e o nome do ficheiro de XML, com o seguinte comando:

```
./metalog_windows_Nfiles.pl     /home/ubuntu/MCIF/Template/final_v3
template.xml
```

Caso o utilizador não preencha o segundo parâmetro com o nome da diretoria correta, é apresentada a mensagem "Diretoria não existe" no ecrã, como ilustra a Figura 6.17. Assim como, se a execução da aplicação não respeitar o número de parâmetros de entrada definido, mostra a mensagem "Número de parâmetros incompleto.", como ilustra a Figura 6.18.

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ./metalog_windows_Nfiles.pl
/home/ubuntu/MFIC/Template/final_v444444 template33.xml
Directory doesn't exists.
```

Figura 6.17 - Mensagem de erro "Diretoria não existe.".

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ./metalog_windows_Nfiles.pl
/home/ubuntu/MCIF/Template/final_v3/
Incomplete number of arguments.
```

Figura 6.18 - Mensagem de erro "Número de parâmetros incompleto.".

As mensagens de erro refletidas nas figuras anteriores, também são redirecionadas para um ficheiro de texto, ilustrado na Figura 6.19. Este ficheiro escreve as novas mensagens na linha abaixo da mensagem já escrita.

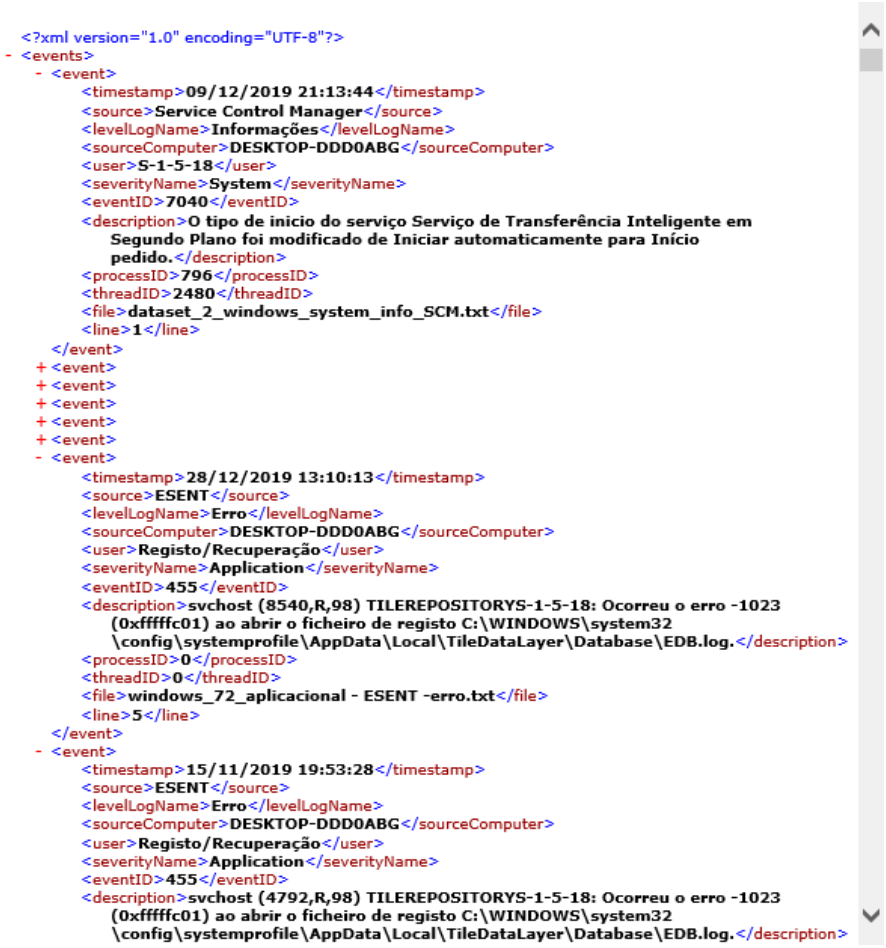
O ficheiro de *logs* agrega todas as validações dos parâmetros para executar a aplicação de fusão dos meta-logs e também é composto pelos campos "*Datetime*", "*Source*", "*User*" e "*Message*".

```
Datetime:08-06-2020 11:21:38
 Source:ubuntu
 User:ubuntu
 Message:Incomplete number of files.
Datetime:08-06-2020 11:21:55
 Source:ubuntu
 User:ubuntu
 Message:Directory doesn't exists.
```

Figura 6.19 - Ficheiro com as mensagens de validação dos parâmetros da fusão dos meta-logs.

O resultado da execução do *script*, era o esperado, como ilustra a Figura 6.20. Como se pode verificar os campos *timestanp*, *source*, *level name*, *source computer*, *user*, *severity name*,

eventID, *description*, *processID*, *threadID*, *filename* e *line* estão corretamente preenchidos, ordenados e contém todos os *logs* existentes nos múltiplos meta-logs.



```
<?xml version="1.0" encoding="UTF-8"?>
- <events>
  - <event>
      <timestamp>09/12/2019 21:13:44</timestamp>
      <source>Service Control Manager</source>
      <levelLogName>Informações</levelLogName>
      <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
      <user>S-1-5-18</user>
      <severityName>System</severityName>
      <eventID>7040</eventID>
      <description>O tipo de início do serviço Serviço de Transferência Inteligente em
          Segundo Plano foi modificado de Iniciar automaticamente para Início
          pedido.</description>
      <processID>796</processID>
      <threadID>2480</threadID>
      <file>dataset_2_windows_system_info_SCM.txt</file>
      <line>1</line>
    </event>
  + <event>
  + <event>
  + <event>
  + <event>
  + <event>
  - <event>
      <timestamp>28/12/2019 13:10:13</timestamp>
      <source>ESENT</source>
      <levelLogName>Erro</levelLogName>
      <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
      <user>Registo/Recuperação</user>
      <severityName>Application</severityName>
      <eventID>455</eventID>
      <description>svchost (8540,R,98) TILEREPOSITORYS-1-5-18: Ocorreu o erro -1023
          (0xfffffc01) ao abrir o ficheiro de registo C:\WINDOWS\system32
          \config\systemprofile\AppData\Local\TileDataLayer\Database\EDB.log.</description>
      <processID>0</processID>
      <threadID>0</threadID>
      <file>windows_72_aplicacional - ESENT -erro.txt</file>
      <line>5</line>
    </event>
  - <event>
      <timestamp>15/11/2019 19:53:28</timestamp>
      <source>ESENT</source>
      <levelLogName>Erro</levelLogName>
      <sourceComputer>DESKTOP-DDD0ABG</sourceComputer>
      <user>Registo/Recuperação</user>
      <severityName>Application</severityName>
      <eventID>455</eventID>
      <description>svchost (4792,R,98) TILEREPOSITORYS-1-5-18: Ocorreu o erro -1023
          (0xfffffc01) ao abrir o ficheiro de registo C:\WINDOWS\system32
          \config\systemprofile\AppData\Local\TileDataLayer\Database\EDB.log.</description>
```


Figura 6.20 – *Output* do *template*.

Por fim, o quarto teste baseou-se no processo anterior, mas utilizando todos os meta-logs criados. Assim, executou-se o *script* com o nome da diretoria como parâmetro e entrada e o nome do ficheiro de XML.

O resultado deste teste foi o esperado, originando um ficheiro XML com tamanho de 240 KB, com todos os *logs* existentes nos meta-logs da diretoria.

6.3.Relatório em Power BI

Neste subcapítulo encontram-se os testes realizados à integração do *template* com o Power BI.

6.3.1. Integração do *template*

Após a realização do último teste descrito anteriormente (subcapítulo 6.2), integrou-se o ficheiro em XML no Power BI *Desktop*. No processo de integração, detetou-se que o ficheiro XML era extenso para integrar no Power BI, como ilustra a Figura 6.21.

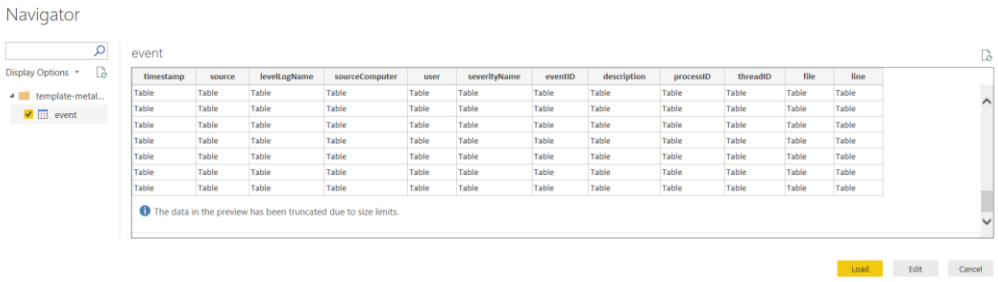


Figura 6.21 – Integração do *template* no Power BI *Desktop*.

De modo a ultrapassar este problema, a solução encontrada foi dividir o conteúdo em dois ficheiros, um com os meta-logs de sistema e outro com os meta-logs aplicacionais. A divisão dos meta-logs realizou-se com a aplicação desenvolvida, adicionando na diretoria da máquina virtual Linux, apenas os meta-logs pretendidos. Deste modo, foi possível integrar corretamente o *template* com os *logs* de sistema no *software*, como ilustra a Figura 6.22.

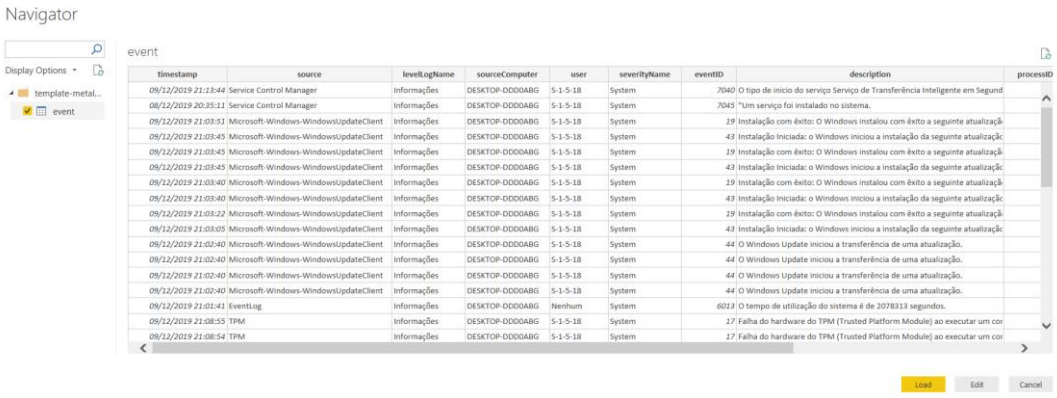


Figura 6.22 – Integração do *template* com os meta-logs de sistema no Power BI *Desktop*.

No entanto, encontrou-se outra limitação ao integrar o template com os meta-logs aplicacionais, o ficheiro era composto por 72 logs (linhas) e apenas foram inseridas 41, mostrando a seguinte mensagem "Os dados na visualização foram truncados devido a limites de tamanho.", como ilustra a Figura 6.23.

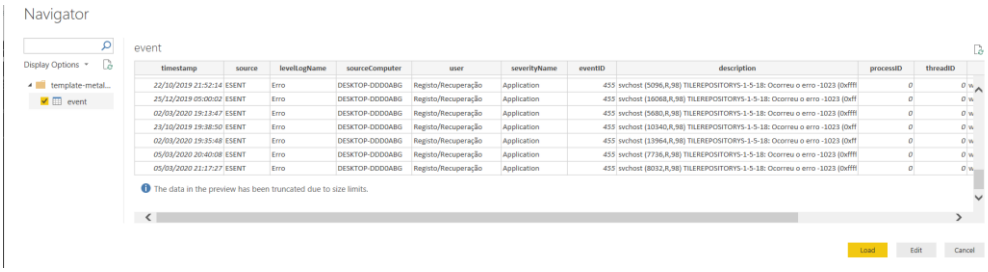


Figura 6.23 - Integração do *template* com os meta-logs aplicacionais no Power BI *Desktop*.

Assim, como referido no subcapítulo 5.3, dividiu-se o *template* com todos os meta-logs, nos seguintes ficheiros de menor tamanho:

- Um ficheiro apenas com os *logs* do tipo informativo do sistema operativo;
- Um ficheiro com os *logs* de erros de sistema do sistema operativo;
- Um ficheiro com alguns *logs* informativos e de erro de sistema.

Esta divisão permitiu integrar os três ficheiros no Power BI sem nenhuma limitação e elaborar um relatório com três páginas.

6.3.2. Envio automático do relatório

Após configurar o envio do relatório por email, como especificado no subcapítulo 5.3, validou-se que a configuração estava ativa através da opção "*On*" ilustrada na Figura 6.24.

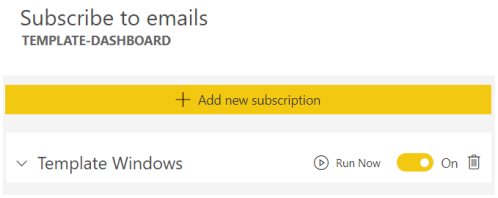


Figura 6.24 – Validação do envio automático do relatório por email.

Para efeitos de testes, o envio do relatório por email foi testado no momento através da opção "*Run Now*", presente na figura anterior. O envio do relatório ocorreu com sucesso, como ilustra a Figura 6.25.

No email encontra-se disponível o relatório em anexo em formato PNG, um link "*Go to report* >" para consultar o relatório na *cloud* e a cópia da página do relatório no corpo do email.

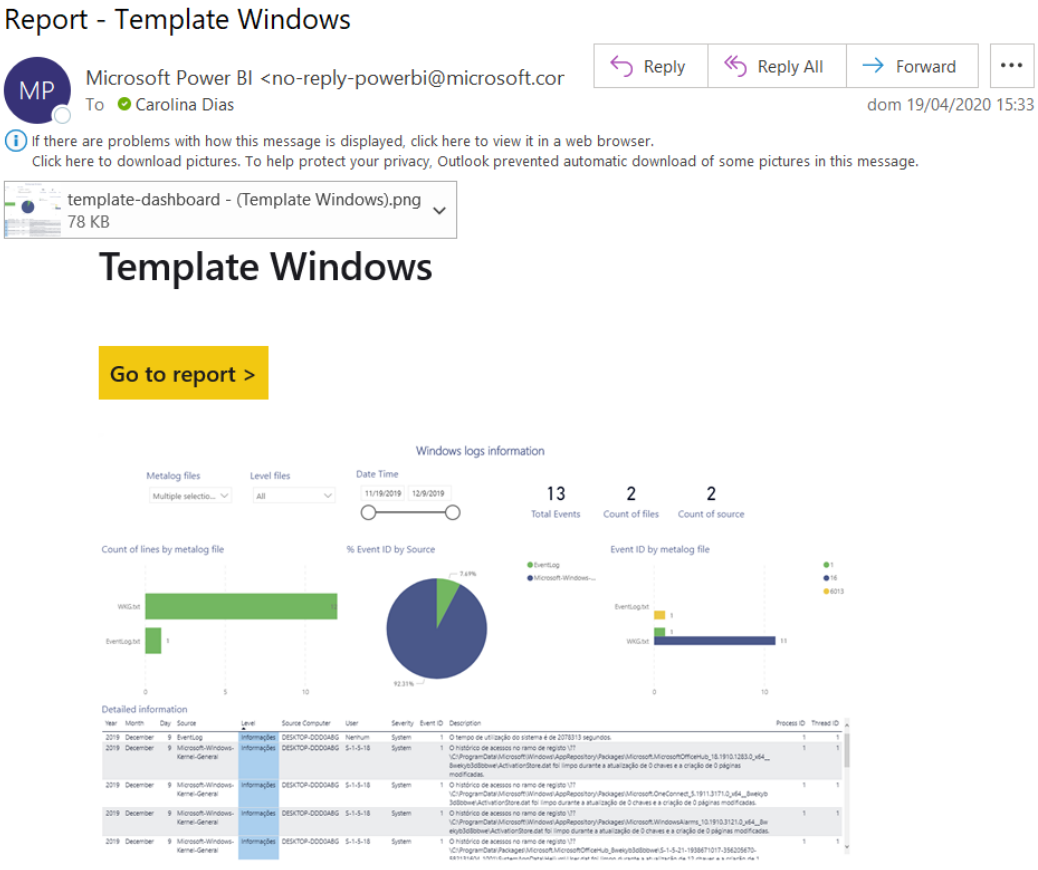


Figura 6.25 – Receção do relatório implementado em Power BI, por email.

7. Análise dos resultados

Neste capítulo são analisados e comentados os resultados obtidos nos testes de usabilidade realizados no capítulo 6. Também serão refletidos os pontos fortes e fracos desta solução.

7.1. Análise crítica dos resultados obtidos

A amostra de *logs* recolhida para realizar os testes deste projeto é restrita e com características comuns, pois são mensagens de sistema e aplicações de um sistema operativo Windows.

Nesta amostra foram recolhidos um total de 430 mensagens de *logs* diferenciados em 2 níveis de criticidade (erro e informativo) de múltiplos serviços Windows. A Tabela 7.1, exemplifica os valores referidos de forma resumida.

Tabela 7.1 – Resumo da quantidade de mensagens de *logs* por criticidade utilizadas nos testes.

	Erro	Informativo	**Total**
Sistema	33	209	242
Aplicacional	72	116	188
Total	105	325	430

Com a análise da tabela anterior, conclui-se que 105 *logs* correspondem a mensagens de erro e 325 a mensagens informativas ocorridas no sistema.

Na Tabela 7.2, pode analisar-se de forma detalhada, o número de *logs* por origem de serviço e criticidade.

Tabela 7.2 – Resumo da quantidade de mensagens de *logs* por origem de serviço utilizada nos testes.

Nível	Origem	Erro	Informativo	**Total**
Aplicacional	ESENT	72	116	188
Sistema	DCOM	6	-	6
	EventLog	-	1	1
	SCM	3	2	5
	TPM	24	164	188
	UMPS	-	8	8
	WinLogon	-	1	1
	WKB	-	5	5
	WKG	-	12	12
	WKP	-	3	3

Nível	Origem	Erro	Informativo	**Total**
	WTP	-	1	1
	WUC	-	12	12
Total		105	325	430

Com a análise da tabela, conclui-se que de um total de 430 *logs* utilizados nos testes, os ficheiros com mais linhas, independentemente da criticidade, são dos serviços ESENT (total de 188 *logs*) e TPM (total de 188 *logs*). O serviço SCM destaca-se pelos seus ficheiros com menor número de *logs* (total de 5 *logs*) nos dois níveis de criticidade.

Os serviços WKG e WUC destacam-se pelo maior número de *logs* utilizados nos testes no nível de criticidade informativa (12 *logs*, em cada serviço). No entanto, os serviços se destacam com menor número de linhas por ficheiros correspondem a EventLog (1 log), WinLogon (1 log) e WTP (1 log).

Na Tabela 7.3, pode realizar-se uma analise-se comparativa entre o tamanho dos ficheiros com os *logs* originais e com os meta-logs¸ ficheiros criados após o pré-processamento.

Tabela 7.3 – Comparação do tamanho do ficheiro original e após o pré-processamento.

		Tamanho do ficheiro original (KB)		**Tamanho do ficheiro pré-processado (KB)**	
Nível	Origem	Erro	Informativo	Erro	Informativo
Aplicacional	ESENT	27	47	24	42
Sistema	DCOM	3	-	2	-
	EventLog	-	1	-	1
	SCM	1	1	1	1
	TPM	11	33	10	41
	UMPS	-	4	-	3
	WinLogon	-	1	-	1
	WKB	-	2	-	2
	WKG	-	6	-	5
	WKP	-	1	-	1
	WTP	-	1	-	1
	WUC	-	5	-	4

Conclui-se, com base na tabela anterior, que após a execução do pré-processamento, transformação do ficheiro original com as mensagens de log em meta-log, os ficheiros com tamanho superior a 2 KB reduziram entre 1 a 8 KB. O ficheiro do serviço TPM com criticidade informativa, é um exemplo de uma redução de 8 KB no tamanho.

O número de linha não sofreu alteração, pois no ficheiro original cada mensagem de log correspondia a uma linha do ficheiro.

Relativamente, aos campos dos *logs* recolhidos, eram compostos por 20 (Data e Hora, Origem, ID do Evento, Categoria de Tarefa, Palavras-chave, Utilizador, Código operacional, Registo, Computador, ID do processo, ID do tópico, ID do processador, ID da sessão, Tempo de *Kernel*, Tempo de utilizador, Tempo do Processador, ID de Correlação, ID de Correlação Relativo e Nome da Origem de Evento), ficando apenas com 10 (*Datetime*, *Source*, *Level*, *Source Computer*, *User*, *Serverity*, *Event* ID, *Description*, *Process* ID e *Thread* ID) após a execução da aplicação na fase de "pré-processamento".

Em suma, da análise realizada destacam-se os seguintes pontos:

- A amostra consiste em 430 *logs* de sistema e aplicacional de sistema operativo Windows;
- Os *logs* utilizados nos testes diferenciam-se entre erro e informativos;
- 105 das mensagens de *logs* da amostra tinha criticidade "Erro";
- 325 dos *logs* recolhidos tinham criticidade "Informativa";
- A solução permitiu que existisse uma redução no tamanho dos ficheiros originais;
- Com base nos testes realizados, os ficheiros de texto e XML não apresentaram nenhuma limitação na escrita dos dados.

7.2. Pontos fortes e fracos da solução

Com base na análise critica dos resultados obtidos caracterizada anteriormente, a solução desenvolvida destaca-se pelos seguintes pontos fortes:

- A solução é escalável, com custos reduzidos e de simples utilização;
- A aplicação para pré-processamento das mensagens, tem um ficheiro específico para o utilizador configurar os campos do *log* que considera relevantes. Isto é, não é necessário selecionar os campos que pretende recolher no código fonte da aplicação;
- Leitura do *template* (ficheiro em XML) na *web*;
- Uniformização e fusão de *logs* num único ficheiro;
- Envio do relatório ao utilizador por email;

- Integração do ficheiro em XML com a plataforma de visualização de dados, com maior relevância no mercado;
- Também permite integrar com aplicações desenvolvidas em linguagens de programação como por exemplo, C#.

Relativamente aos pontos fracos destacam-se os seguintes, que podem ser melhorados no futuro:

- O ficheiro com os *logs* originais tem um conjunto de requisitos para ser utilizado na aplicação;
- O processo de execução entre as duas aplicação é manual;
- A integração do *template* (ficheiro em XML) com o Power BI também é um processo manual;
- A partilha do relatório Power BI na *cloud*, requere uma licença.

8. Conclusões e trabalho futuro

Com o desenvolvimento deste projeto, analisou-se a constituição das mensagens de *logs*, realçando os campos que as mesmas devem ter para uma interpretação eficaz e rápida de problemas que ocorram na infraestrutura.

O mercado atual disponibiliza soluções que centralizam os *logs*, monitoriza-os e correlaciona-os de forma a detetar ocorrência e ameaças na infraestrutura. Em caso de anomalia as soluções existentes notificam o utilizador e disponibilizam a informação em relatórios. No entanto, estas soluções apresentam custos elevados de licenciamento ou, em alguns casos, necessitam de elevado esforço de implementação para atingir a informação pretendida pelo utilizador. Por vezes, o utilizador também não tem os conhecimentos necessários para a utilizadas deste tipo de soluções.

Na arquitetura proposta, pode analisar-se o processo de transferência dos dados entre as aplicações e a plataforma de visualização. Nesta arquitetura destacam-se os ficheiros de *logs* das aplicações na fase "Pré-processamento" e "Fusão", que permite a centralização das mensagens de erro refletidas na consola ao utilizador.

Com base na arquitetura proposta, conseguiu-se implementar uma solução escalável, com *feedback* a erros do utilizador e de simples utilização. O relatório implementado também disponibiliza ao utilizador uma visualização dos dados apelativa e interativa.

Nos testes de usabilidade realizados à solução, conclui-se que a sua utilização é simples, pois o utilizador não necessita de adaptar o código fonte consoante os campos que pretende filtrar ou fundir nas múltiplas mensagens de *log*. A nível de visualização das mensagens de *log*, a informação é representada em contagens, gráficos e tabelas, visível em *cloud* e enviada por email.

Através da análise dos resultados apresentados no capítulo 0, verificou-se que a solução permite que exista a redução no tamanho dos ficheiros originais, uniformização e fusão dos *logs* num único ficheiro.

Conclui-se que os objetivos propostos foram atingidos, tornando possível automatizar o tratamento dos campos considerados relevantes nos *logs*, convertê-los para uma estrutura

que é interpretada através de um *browser* e integrá-la com uma solução de visualização e análise de dados.

Nos Anexos é possível consultar o código fonte das aplicações desenvolvidas na linguagem PERL, *print screen* das páginas, do relatório implementado em Power BI, não refletidas ao longo do subcapítulo 5.3 e o "Manual de utilização da solução".

Como trabalho futuro, poderá melhorar-se a aplicação desenvolvida e acrescentar-lhe funcionalidades de forma a aumentar a diversidade de serviços disponíveis. Como por exemplo, dar a possibilidade ao utilizador de selecionar os campos que prende visualizar no *template* (ficheiro XML). No entanto, o utilizador pode selecionar os campos que pretende analisar através dos filtros implementados no relatório em Power BI.

Relativamente ao processo de execução da aplicação até à visualização dos dados no Power BI, poderá automatizar-se o processo existente, adaptando o código fonte desenvolvido.

Por fim, poderá adaptar-se esta solução a uma aplicação que utilize abordagens baseadas em ontologias, para que seja analisado, extraído conhecimento e aprendizagem das mensagens de *log* de forma automática.

Referências Bibliográficas

[1] R. A. Grimes, “Why you need centralized logging and event log management,” CSO, 12 06 2018. [Online]. Available: https://www.csoonline.com/article/3280123/why-you-need-centralized-logging-and-event-log-management.html. [Acedido em 17 11 2019].

[2] “A Importância da Centralização de Logs,” gerencianet, 04 04 2016. [Online]. Available: https://gerencianet.com.br/blog/importancia-da-centralizacao-de-logs/. [Acedido em 08 03 2020].

[3] J. Wilder, “Centralized Logging,” 03 06 2012. [Online]. Available: http://jasonwilder.com/blog/2012/01/03/centralized-logging/. [Acedido em 18 11 2019].

[4] A. Brown, “Application Logging: What, When, How,” DZone, 01 09 2009. [Online]. Available: https://dzone.com/articles/application-logging-what-when . [Acedido em 09 11 2019].

[5] B. Johnson, “HOW WE SOLVED NODE'S HARD TO USE LOGS,” Timber, [Online]. Available: https://timber.io/blog/how-we-solved-nodes-hard-to-use-logs/ . [Acedido em 09 11 2019].

[6] “Security Audit Logging Guideline,” UC Berkeley, [Online]. Available: https://security.berkeley.edu/security-audit-logging-guideline. [Acedido em 09 11 2019].

[7] R. Martins, “A importancia da monitorização,” ComputerWorld, 09 11 2011. [Online]. Available: https://www.computerworld.com.pt/2011/11/09/a-importancia-da-monitorizacao/. [Acedido em 08 03 2020].

[8] “Introduction to logging for security purposes,” National Cyber Security Center, 08 07 2018. [Online]. Available: https://www.ncsc.gov.uk/guidance/introduction-logging-security-purposes. [Acedido em 16 11 2019].

[9] "The story of Ubuntu," Ubuntu, 2020. [Online]. Available: https://ubuntu.com/about. [Acedido em 19 04 2020].

[10] "Windows 10 (Microsoft Windows 10)," TechTarget, 12 2017. [Online]. Available: https://searchenterprisedesktop.techtarget.com/definition/Windows-10. [Acedido em 19 04 2020].

[11] C. HOFFMAN, "What Is the Windows Event Viewer, and How Can I Use It?," How-To Geek, 12 11 2018. [Online]. Available: https://www.howtogeek.com/123646/htg-explains-what-the-windows-event-viewer-is-and-how-you-can-use-it/. [Acedido em 19 04 2020].

[12] "About Perl," Perl, [Online]. Available: https://www.perl.org/about.html. [Acedido em 19 04 2020].

[13] A. Balakrishnan, "What is Perl? How relevant it is and how to get started!," 9 05 2018. [Online]. Available: https://blog.usejournal.com/what-is-perl-how-relevant-it-is-and-how-to-get-started-d802e7aba2cd. [Acedido em 19 04 2020].

[14] EDUCBA, "Perl vs Python," EDUCBA, [Online]. Available: https://www.educba.com/perl-vs-python/. [Acedido em 05 07 2020].

[15] A. P. Pereira, "O que é o XML?," Tecmundo, 18 03 2009. [Online]. Available: https://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm. [Acedido em 16 11 2019].

[16] "Uma introdução ao JSON," Devmedia, [Online]. Available: https://www.devmedia.com.br/json-tutorial/25275. [Acedido em 16 11 2019].

[17] "What is Power BI?," Microsoft, 2020. [Online]. Available: https://powerbi.microsoft.com/en-us/what-is-power-bi/. [Acedido em 08 03 2020].

[18] "Turn data into opportunity," Microsoft, 2020. [Online]. Available: https://powerbi.microsoft.com/en-us/. [Acedido em 08 03 2020].

[19] "Você sabe o que é log de dados ? Entenda sua importância," Security Strong, 07 11 2017. [Online]. Available: https://www.strongsecurity.com.br/blog/voce-sabe-o-que-e-log-de-dados-entenda-sua-importancia/. [Acedido em 09 11 2019].

[20] "A Importância de Gerar e Manter Logs," Ezequiel Juliano Müller, 27 09 2014. [Online]. Available: http://www.ezequieljuliano.com.br/?p=76. [Acedido em 11 12 2019].

[21] D. Torre, "What is log management and how to choose the right tools," CSO, 18 10 2010. [Online]. Available: https://www.csoonline.com/article/2126060/network-security-what-is-log-management-and-how-to-choose-the-right-tools.html. [Acedido em 14 12 2019].

[22] "Event ID: 12 Source: Microsoft-Windows-Kernel-General," EventID.net, [Online]. Available: http://www.eventid.net/display-eventid-12-source-Microsoft-Windows-Kernel-General-eventno-11542-phase-1.htm. [Acedido em 08 03 2020].

[23] "Service Control Manager," Microsoft, 31 05 2018. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/services/service-control-manager. [Acedido em 08 03 2020].

[24] "TPM event logs," IBM, 01 06 2018. [Online]. Available: https://www.ibm.com/support/knowledgecenter/POWER9/p9ia9/p9ia9_tpm_event_logs.htm. [Acedido em 08 03 2020].

[25] "User-Mode Power Service," Microsoft, 20 04 2017. [Online]. Available: https://docs.microsoft.com/en-us/windows-hardware/drivers/powermeter/user-mode-power-service. [Acedido em 08 03 2020].

[26] "How To Fix Kernel-Power Error?," PCRisk, [Online]. Available: https://blog.pcrisk.com/windows/12891-how-to-fix-kernel-power-error. [Acedido em 08 03 2020].

[27] "Windows Update Agent," Microsoft, 02 02 2010. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc735627(v=ws.10). [Acedido em 08 03 2020].

[28] "COM," Microsft, 02 02 2010. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc774403(v=ws.10). [Acedido em 08 03 2020].

[29] S. Bennuri, "How to Run Power Troubleshooter in Windows 10," Howtoconnect, 02 11 2019. [Online]. Available: https://www.howto-connect.com/run-power-troubleshooter-windows-10/. [Acedido em 08 03 2020].

[30] "W (Security Glossary)," Microsoft, 31 05 2018. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/secgloss/w-gly?redirectedfrom=MSDN. [Acedido em 08 03 2020].

[31] "Event Logging," Microsoft, 31 05 2019. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/eventlog/event-logging. [Acedido em 08 03 2020].

[32] T. Tang, "How do you troubleshoot ESENT entries in the event viewer?," 17 09 2018. [Online]. Available: https://social.technet.microsoft.com/Forums/Windows/en-US/d3fb010d-abc3-4043-bd26-5a8fa06fa918/how-do-you-troubleshoot-esent-entries-in-the-event-viewer?forum=win10itprogeneral. [Acedido em 11 04 2020].

[33] R. A. Grimes, "Living the log management lifecycle," InfoWorld, 04 08 2010. [Online]. Available: https://www.infoworld.com/article/2625977/living-the-log-management-lifecycle.html. [Acedido em 14 12 2019].

[34] J. Morgan, "What is Centralized Log Management (CLM)?," mission, 31 05 2016. [Online]. Available: https://www.missioncloud.com/blog/what-is-centralized-log-management-clm/. [Acedido em 18 11 2019].

[35] "What is SIEM?," Forcepoint, [Online]. Available: https://www.forcepoint.com/cyber-edu/siem. [Acedido em 18 11 2019].

[36] K. Leal, "Regulamento Geral de Proteção de Dados (RGPD): o que vai mudar no digital?," Nomadismo, 19 04 2018. [Online]. Available: https://www.nomadismodigital.pt/regulamento-rgpd/. [Acedido em 22 12 2019].

[37] Primavera, “Regulamento Geral de Proteção de Dados,” Primavera, [Online]. Available: https://pt.primaverabss.com/pt/tudo-que-precisa-saber-sobre-o-rgpd/o-que-e-o-regulamento-geral-de-protecao-de-dados-rgpd/. [Acedido em 22 12 2019].

[38] L. Filipe, “Gestão de Logs Centralizados e o RGPD,” LinkedIn, 19 10 2018. [Online]. Available: https://www.linkedin.com/pulse/gest%C3%A3o-de-logs-centralizados-e-o-rgpd-luis-filipe/. [Acedido em 22 12 2019].

[39] A. E. L. T. Varanda, “O Regulamento Geral de Proteção de Dados e a Pseudonimização de Logs,” 09 2019. [Online]. Available: https://iconline.ipleiria.pt/handle/10400.8/4362. [Acedido em 14 03 2020].

[40] M. M. ·, “Bash script to monitor messages log (Warnings, Errors and Critical) in Linux,” 2DAYGEEK, 05 09 2019. [Online]. Available: https://www.2daygeek.com/linux-bash-script-to-monitor-messages-log-warning-error-critical-send-email/. [Acedido em 23 12 2019].

[41] P. Cunningham, “Generate Mailbox Size and Information Reports using PowerShell,” Script Center, 12 2008. [Online]. Available: https://gallery.technet.microsoft.com/scriptcenter/Generate-Mailbox-Size-and-3f408172. [Acedido em 14 03 2020].

[42] M. MARUTHAMUTHU, “Bash Script to Check Successful and Failed User Login Attempts on Linux,” 2DAYGEEK, 03 12 2019. [Online]. Available: https://www.2daygeek.com/bash-script-to-check-successful-and-failed-user-login-attempts-on-linux/. [Acedido em 23 12 2019].

[43] D. Helper, “Tip: Using Notepad++ to Read Log Files,” DZone, 13 06 2013. [Online]. Available: https://dzone.com/articles/tip-using-notepad-read-log. [Acedido em 15 12 2019].

[44] M. Wilson, “https://www.pcwdld.com/best-event-log-monitor-and-siem-tools,” PC&Network, 24 01 2019. [Online]. Available: https://www.pcwdld.com/best-event-log-monitor-and-siem-tools. [Acedido em 23 12 2019].

[45] John, "Gartner Magic Quadrant for SIEM Products (2018-2010)," Cyber Security Memo, 11 01 2019. [Online]. Available: https://www.51sec.org/2019/01/11/gartner-magic-quadrant-for-siem-products-2016-2015-2014-2013-2012-2011-2010/. [Acedido em 24 11 2019].

[46] A. NASCIMENTO, "Gartner publica versão 2019 de seu Quadrante Mágico – Confira a nossa análise," 15 02 2019. [Online]. Available: http://www.insightdataservices.com.br/gartner-publica-versao-2019-de-seu-quadrante-magico/. [Acedido em 17 05 2020].

[47] "Best Log Management Tools: 51 Useful Tools for Log Management, Monitoring, Analytics, and More," Stackify, 26 05 2017. [Online]. Available: https://stackify.com/best-log-management-tools/. [Acedido em 22 12 2019].

[48] S. T. Help, "Top 11 Best SIEM Tools In 2020 For Real-Time Incident Response And Security," 10 03 2020. [Online]. Available: https://www.softwaretestinghelp.com/siem-tools/. [Acedido em 15 03 2020].

[49] "IBM QRadar," IBM, [Online]. Available: https://www.ibm.com/security/security-intelligence/qradar. [Acedido em 22 12 2019].

[50] "ArcSight Logger," MicroFocus, [Online]. Available: https://www.microfocus.com/pt-br/products/siem-log-management/overview. [Acedido em 22 12 2019].

[51] A. Telecom, "O EFEITO DA VISUALIZAÇÃO DE DADOS NA DEFINIÇÃO DE ESTRATÉGIAS DE NEGÓCIOS," 29 02 2016. [Online]. Available: https://blog.algartelecom.com.br/gestao/o-efeito-da-visualizacao-de-dados-na-definicao-de-estrategias-de-negocios-2/. [Acedido em 23 05 2020].

[52] "12 principais ferramentas de Business Intelligence em 2019," 05 12 2018. [Online]. Available: https://cio.com.br/12-principais-ferramentas-de-business-intelligence-em-2019/. [Acedido em 23 05 2020].

[53] R. Santos, “O que é Power BI e quais as vantagens de usar essa ferramenta,” 12 08 2019. [Online]. Available: https://blog.deskmanager.com.br/o-que-e-power-bi/. [Acedido em 23 05 2020].

[54] “Board: The #1 platform for decision-making,” [Online]. Available: https://www.board.com/en. [Acedido em 23 05 2020].

[55] IBM, “IBM i2 Analyst's Notebook,” [Online]. Available: https://www.ibm.com/products/analysts-notebook. [Acedido em 23 05 2020].

[56] Programação, “9.1 Noção de ficheiro e operações básicas,” [Online]. Available: https://web.fe.up.pt/~jmsa/programacao/Cap-9.htm. [Acedido em 20 03 2020].

[57] Significados, “Significado de Script,” 12 09 2013. [Online]. Available: https://www.significados.com.br/script/. [Acedido em 20 03 2020].

[58] Microsoft, “Get started with Power BI Desktop,” 13 03 2020. [Online]. Available: https://docs.microsoft.com/en-us/power-bi/desktop-getting-started. [Acedido em 14 04 2020].

[59] Microsoft, “Apply DAX basics in Power BI Desktop,” 21 10 2019. [Online]. Available: https://docs.microsoft.com/en-us/power-bi/desktop-quickstart-learn-dax-basics. [Acedido em 17 04 2020].

[60] K. Aebersold, “Software Testing Methodologies,” [Online]. Available: https://smartbear.com/learn/automated-testing/software-testing-methodologies/. [Acedido em 31 05 2020].

[61] “Perl XML::LibXML by Example,” 29 01 2020. [Online]. Available: http://grantm.github.io/perl-libxml-by-example/. [Acedido em 25 04 2020].

[62] Microsoft, “DataSets, DataTables e DataViews,” 30 03 2019. [Online]. Available: https://docs.microsoft.com/pt-br/dotnet/framework/data/adonet/dataset-datatable-dataview/. [Acedido em 20 03 2020].

Anexos

Anexo A – "Versão 1 - Código fonte da aplicação de pré-processamento"

```
#!/usr/bin/perl

use strict;
#use warnings;

open(IN, "dataset_34_windows_system_erro.txt");
open(OUT, ">>metalog_error_34.txt");

foreach $_ (<IN>){
      next if /Relativo     Nome da Origem de Evento/;
      chomp $_;
      $_ =~ tr/\t/&/s;
      my ($datetime, $source, $levelLogName, $sourceComputer,
$user, $severityName, $eventID, $description, $processID,
$threadID) = (split /&/, $_)[1, 2, 0, 7, 4, 6, 3, 12, 8, 9];
      my ($sourceComputerSCM, $severityNameSCM, $userSCM,
$descriptionSCM, $processIDSCM, $threadIDSCM) = (split /&/, $_)[8,
7, 6, 13, 9, 10];
      my ($levelLogNameWDCOM, $severityNameWDCOM,
$sourceComputerWDCOM, $userWDCOM, $descriptionWDCOM,
$processIDWDCOM, $threadIDWDCOM) = (split /&/, $_)[0, 8, 9, 6, 15,
10, 11];
if($source eq "Service Control Manager"){
      print OUT "$datetime&";
      print OUT "$source&";
      print OUT "$levelLogName&";
      print OUT "$sourceComputerSCM&";
      print OUT "$userSCM&";
      print OUT "$severityNameSCM&";
      print OUT "$eventID&";
      print OUT "$descriptionSCM&";
      print OUT "$processIDSCM&";
      print OUT "$threadIDSCM\n";
      } elsif($source eq "Microsoft-Windows-DistributedCOM"){
      print OUT "$datetime&";
      print OUT "$source&";
      print OUT "$levelLogNameWDCOM&";
      print OUT "$sourceComputerWDCOM&";
      print OUT "$userWDCOM&";
      print OUT "$severityNameWDCOM&";
      print OUT "$eventID&";
      print OUT "$descriptionWDCOM&";
      print OUT "$processIDWDCOM&";
      print OUT "$threadIDWDCOM\n";
      } else {
      print OUT "$datetime&";
      print OUT "$source&";
```

```
      print OUT "$levelLogName&";
      print OUT "$sourceComputer&";
      print OUT "$user&";
      print OUT "$severityName&";
      print OUT "$eventID&";
      print OUT "$description&";
      print OUT "$processID&";
      print OUT "$threadID\n";}
}

close(IN);
close(OUT);

exit(0);
```

Anexo B – "Código fonte da aplicação de pré-processamento"

```
#!/usr/bin/perl

use strict;
#use warnings;

open(IN2, "<", $ARGV[0]);
open(IN, "<", $ARGV[1]);
open(OUT, ">>", $ARGV[2]);
open(OUT2, ">>", "logs_metalogs.txt");

my @tmp;
my $datetimelog = `date +'%d-%m-%Y %H:%M:%S'`;
my $userlog= `whoami`;
my $hostnamelog = `hostname`;

if(@ARGV != 3) {
       print "Incomplete number of files.\n";
       print OUT2 "Datetime:$datetimelog Source:$hostnamelog
User:$userlog Message:Incomplete number of files.\n";
}
 elsif ((! -e $ARGV[0]) or (! -e $ARGV[1])) {
       print "File doesn't exists.\n";
       print OUT2 "Datetime:$datetimelog Source:$hostnamelog
User:$userlog Message:File doesn't exists.\n";
}
else {
       foreach $_ (<IN2>){
              next if /#Datetime, Source, Level, Source Computer,
User, Serverity, Event ID, Description, Process ID, Thread ID,
Delimitador/;
              chomp $_;
              $_ =~ tr/=/,/s;
              @tmp = split(/,/,$_);
       }
       my $line = 0;
       foreach $_ (<IN>){
              chomp $_;
              $line++;
```

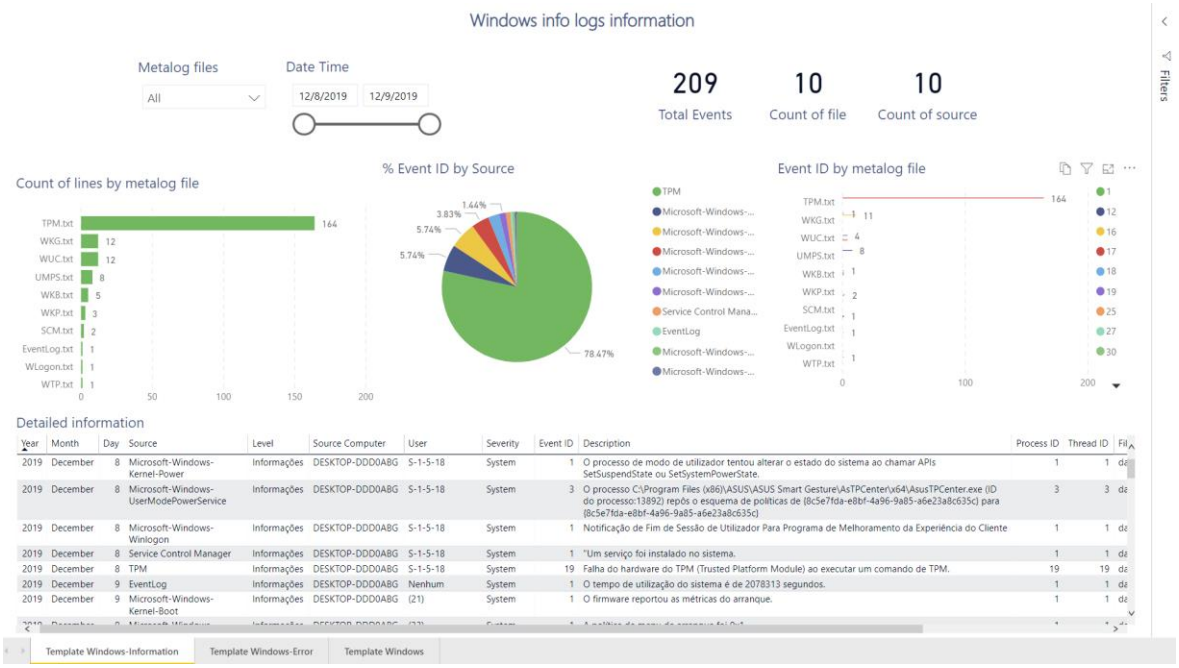
```
                $_ =~ eval "tr/$tmp[10]/&/s";
                my ($datetime, $source, $levelLogName,
$sourceComputer, $user, $severityName, $eventID, $description,
$processID, $threadID) = (split /&/,
$_)[$tmp[0],$tmp[1],$tmp[3],$tmp[4],$tmp[5],$tmp[6],$tmp[7],$tmp[8
],$tmp[9]];
                print OUT
"$ARGV[1]&$line&$datetime&$source&$levelLogName&$sourceComputer&$u
ser&$severityName&$eventID&$description&$processID&$threadID\n";
        }
}

close(IN);
close(IN2);
close(OUT);
close(OUT2);

exit(0);
```

Anexo C – "Código fonte da aplicação de fusão"

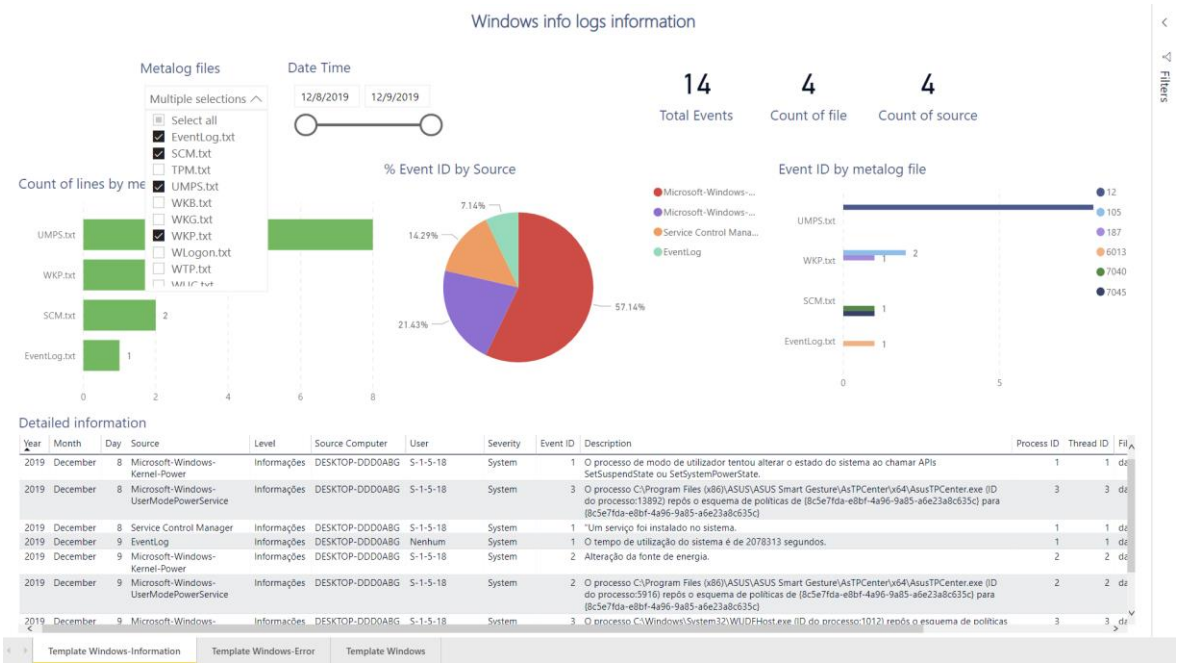
```
#!/usr/bin/perl

use strict;
#use warnings;

my $directory = $ARGV[0];
opendir(DIR, $directory);

my $filename;
my @array;
my $datetimelog = `date +'%d-%m-%Y %H:%M:%S'`;
my $userlog= `whoami`;
my $hostnamelog = `hostname`;

open(OUT, ">>", $ARGV[1]);
open(OUT2, ">>", "logs_template.txt");

if(@ARGV != 2) {
        print "Incomplete number of arguments.\n";
        print OUT2 "Datetime:$datetimelog Source:$hostnamelog
User:$userlog Message:Incomplete number of files.\n";
}
 elsif ((! -e $ARGV[0])) {
        print "Directory doesn't exists.\n";
        print OUT2 "Datetime:$datetimelog Source:$hostnamelog
User:$userlog Message:Directory doesn't exists.\n";
}
else {
        print OUT "<?xml version=\"1.0\" encoding=\"utf-8\"?>\n";
        print OUT "<events>\n";
        while ($filename = readdir(DIR)) {
                next if $filename =~ /^\./;
```

```
                next if $filename =~ /^*\.pl/;
                next if $filename =~ /^*\.xml/;

                open (IN, "<", "$filename");

                @array = <IN>;

                foreach my $i (@array) {
                        chomp $i;
                        my ($file, $line, $datetime, $source,
$levelLogName, $sourceComputer, $user, $severityName, $eventID,
$description, $processID, $threadID) = (split /&/, $i)[0, 1, 2, 3,
4, 5, 6, 7, 8, 9, 10, 11];
                        print OUT "<event>\n";
                        print OUT
"<timestamp>$datetime</timestamp>\n";
                        print OUT "<source>$source</source>\n";
                        print OUT
"<levelLogName>$levelLogName</levelLogName>\n";
                        print OUT
"<sourceComputer>$sourceComputer</sourceComputer>\n";
                        print OUT "<user>$user</user>\n";
                        print OUT
"<severityName>$severityName</severityName>\n";
                        print OUT "<eventID>$eventID</eventID>\n";
                        print OUT
"<description>$description</description>\n";
                        print OUT
"<processID>$processID</processID>\n";
                        print OUT
"<threadID>$threadID</threadID>\n";
                        print OUT "<file>$file</file>\n";
                        print OUT "<line>$line</line>\n";
                        print OUT "</event>\n";
                }
        close(IN);
        }
        print OUT "</events>";
}

closedir(DIR);
close(OUT);
close(OUT2);

exit(0);
```

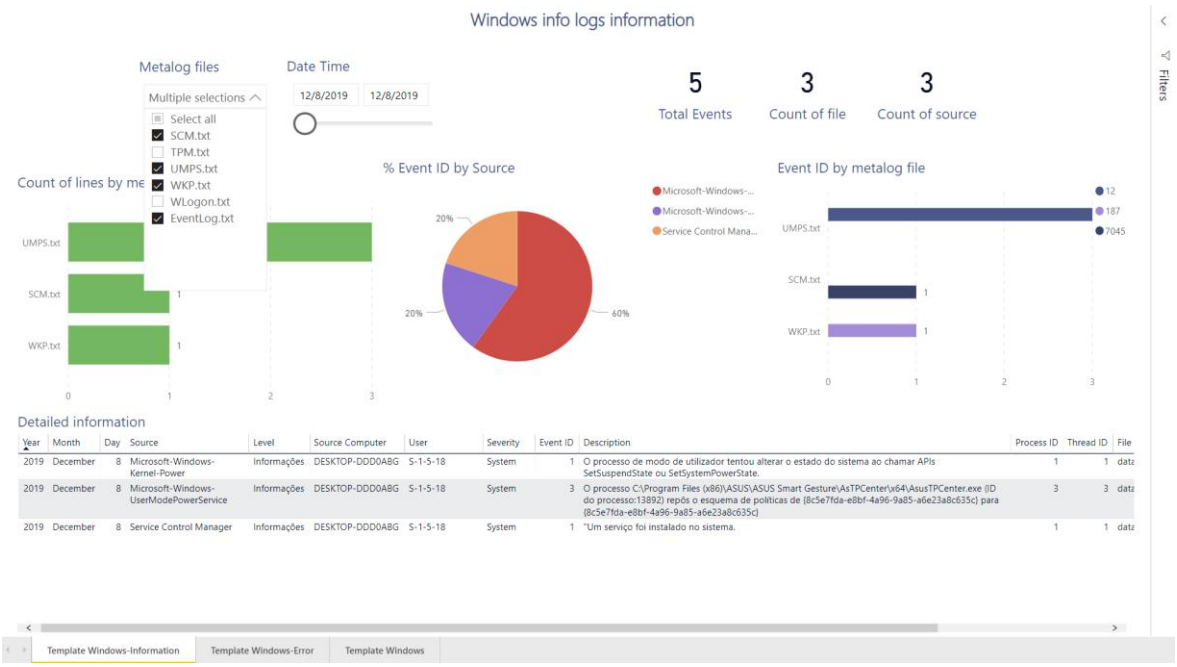
Anexo D – "Utilização do *template* com os meta-logs informativos de sistema no relatório implementado em Power BI"



Visualização do relatório com *logs* informativos implementado em Power BI.

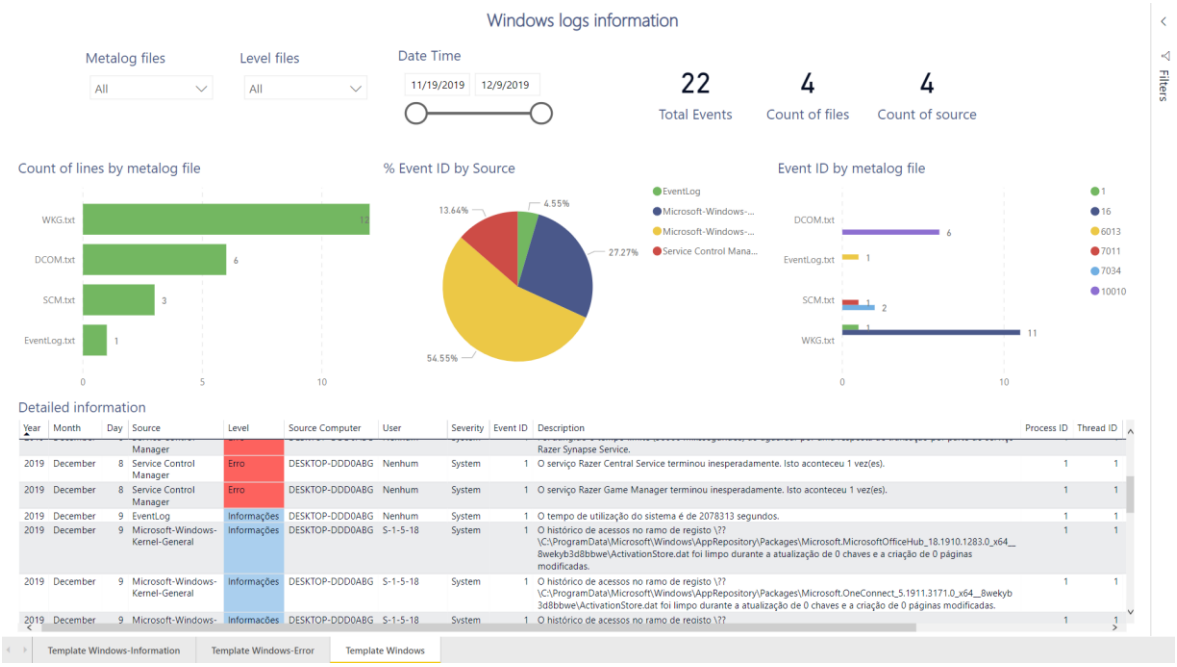


Visualização do relatório implementado, com interação do filtro "Meta-log *files*" em Power BI.

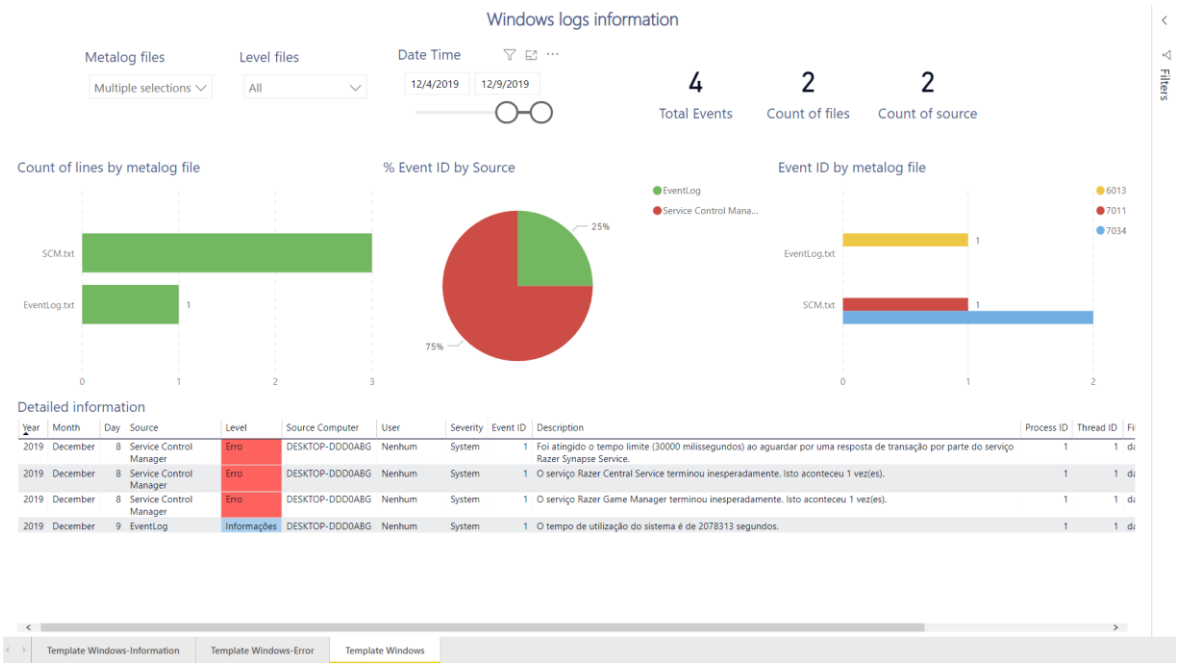


Visualização do relatório implementado, com interação dos filtros "Meta-log *files*" e "*Date Time*" em Power BI.

Anexo E – "Utilização do *template* com os meta-logs informativos e erros de sistema no relatório implementado em Power BI"



Visualização do relatório com *logs* informativos e erros implementado em Power BI



Visualização do relatório implementado, com interação dos filtros "Meta-log *files*", "Level files" e "*Date Time*" implementado em Power BI.

Anexo F – "Manual de utilização da aplicação"

O presente manual destina-se à utilização das aplicações de meta-logs, fusão de *logs* e visualização dos dados recolhidos pelas mesmas.

1. **Aplicação de meta-logs e fusão de *logs***
    a. Requisitos de utilização da solução
        - Máquina virtual com sistema operativo Ubuntu 18.04.3 LTS e 2 GB de memória;
        - Ficheiro de texto com os *logs* com os seguintes requisitos:
            - Todos os *logs* tem de estar numa única linha;
            - Todos os *logs* tem de pertencer a um único serviço e nível.
        - Uma diretoria dedicada ao armazenamento dos meta-logs;
        - A aplicação de agrega os meta-logs, tem de estar dentro da diretoria definida acima;
        - Os ficheiros (logs_metalogs.txt e logs_template.txt) que guardam as mensagens de erro na execução das aplicações, tem de estar nas diretorias das aplicações.

b. Utilização da solução

- Execução de meta-logs

Passo 1: Recolha dos *logs* para um ficheiro de texto, garantindo os requisitos referidos acima.

Passo 2: Garantir que o ficheiro de texto com os *logs* encontra-se na mesma diretoria que o ficheiro de configurações e a aplicação.

Passo 3: Executar o comando na consola da máquina virtual, dentro da diretoria com a sintaxe, ./<nome_aplicação> <ficheiro_configurações> <ficheiro_*logs*_originais> <nome_ficheiro_meta-log.txt>, como ilustra a seguinte figura:

```
ubuntu@ubuntu:~/MCIF/Metalog/Teste-Nfiles_V3/Windows_Sistema_Teste1$ ./metalog_windows.pl
confg.txt dataset_164_windows_system_info_TPM.txt metalog_information_TPM.txt
```

Passo 4: O ficheiro será criado na mesma diretoria com o nome referido em <nome_ficheiro_meta-log.txt> do comando acima.

- Fusão dos meta-logs

Passo 1: Inserir os meta-logs, ficheiro em formato de texto, numa diretoria e garantir que a aplicação existe nessa diretoria.

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ll
total 900
drwxrwxrwx 2 ubuntu ubuntu   4096 May 17 14:05 ./
drwxr-xr-x 5 ubuntu ubuntu   4096 Jun  7 12:57 ../
-rw-rw-r-- 1 ubuntu ubuntu  23780 Apr 11 14:32 metalog_aplicacional_error_esent.txt
-rw-rw-r-- 1 ubuntu ubuntu  42632 Apr 11 14:33 metalog_aplicacional_info_esent.txt
-rw-rw-r-- 1 ubuntu ubuntu   1829 Apr 10 16:51 metalog_DCOM.txt
-rw-rw-r-- 1 ubuntu ubuntu    475 Apr 10 16:51 metalog_information_SCM.txt
-rw-rw-r-- 1 ubuntu ubuntu  33511 Apr 10 16:56 metalog_information_TPM.txt
-rw-rw-r-- 1 ubuntu ubuntu   2962 Apr 10 16:59 metalog_information_UMPS.txt
-rw-rw-r-- 1 ubuntu ubuntu    950 Apr 10 16:59 metalog_information_WKB.txt
-rw-rw-r-- 1 ubuntu ubuntu   4936 Apr 10 17:00 metalog_information_WKG.txt
-rw-rw-r-- 1 ubuntu ubuntu    637 Apr 10 17:00 metalog_information_WKP.txt
-rw-rw-r-- 1 ubuntu ubuntu    256 Apr 10 17:00 metalog_information_WLogon.txt
-rw-rw-r-- 1 ubuntu ubuntu    386 Apr 10 17:01 metalog_information_WTP.txt
-rw-rw-r-- 1 ubuntu ubuntu   3259 Apr 10 17:01 metalog_information_WUC.txt
-rw-rw-r-- 1 ubuntu ubuntu    708 Apr 10 17:02 metalog_SCM.txt
-rw-rw-r-- 1 ubuntu ubuntu   9826 Apr 10 17:03 metalog_TPM.txt
-rw-rw-r-- 1 ubuntu ubuntu    186 Apr 10 17:04 metalog_windows_eventlog.txt
-rwxrwxrwx 1 ubuntu ubuntu   1591 Apr  2 10:39 metalog_windows_Nfiles.pl*
```

Passo 2: Executar o comando na consola da máquina virtual, dentro da diretoria com a sintaxe, ./<nome_aplicação> <diretoria_meta-logs> <nome_ficheiro_*template*.xml>, como ilustra a seguinte figura:

```
ubuntu@ubuntu:~/MCIF/Template/final_v3$ ./metalog_windows_Nfiles.pl /home/ubuntu/MCIF/Template/
final_v3 template.xml
```

Passo 3: O ficheiro será criado na mesma diretoria com o nome referido em <nome_ficheiro_*template*.xml> do comando acima.

2. Aplicação de Visualização

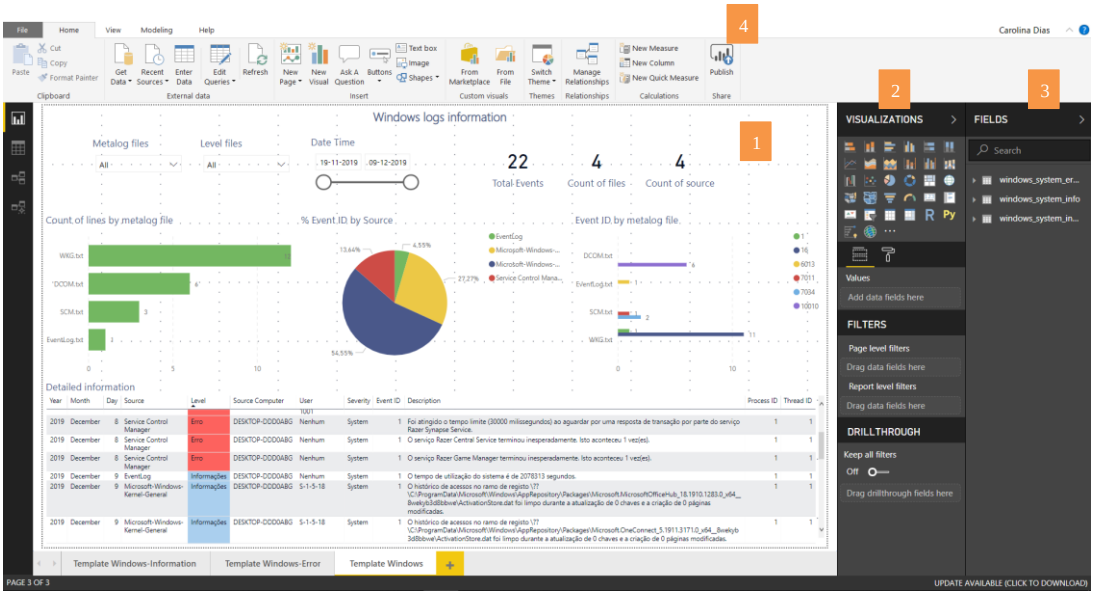
a. Requisitos de utilização da solução

- Máquina virtual com sistema operativo Windows;

  Nota: Atualmente, o Power BI Desktop não é suportado em sistemas operativos Linux (https://community.powerbi.com/t5/Desktop/Power-BI-Desktop-for-Linux/td-p/785915);
- *Software* "Microsoft Power BI Desktop".

b. Utilização da solução

- Adicionar e alterar conteúdo no relatório



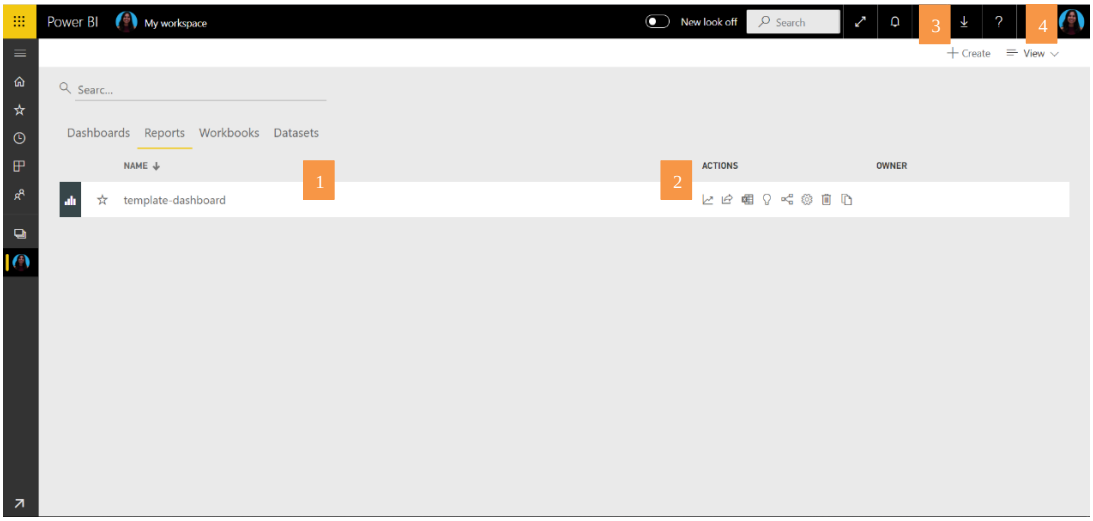
1 Definição e personalização do layout do relatório.

2 Personalização dos tipos de visualização selecionados.

3 Integração do conteúdo existente no ficheiro de XML, com os tipos de visualização selecionados no ponto 2.

4 Publicação do relatório para a *cloud*.

- Consultar os relatórios elaborados

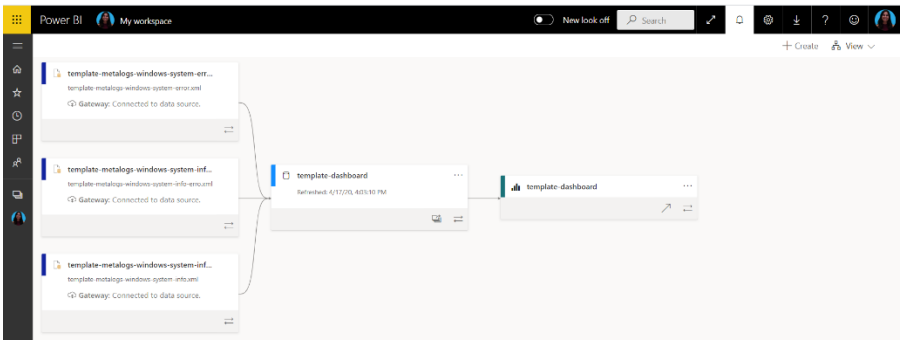


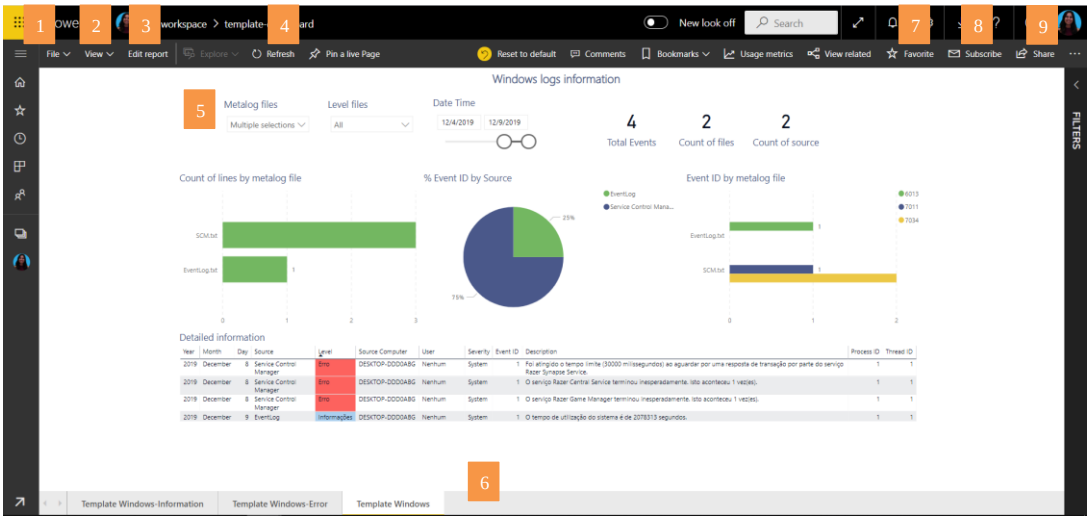
1 Lista de relatórios implementados.

2 Cada relatório implementado, disponibiliza um conjunto de opções personalizadas, como por exemplo, partilhar, eliminar, copiar o relatório.

3 Criar um *dashboard*, relatório, *dataset* ou streaming *dataset*.

4 Permite selecionar o tipo de vista por lista ou por *Lineage* dos relatórios. A vista por *Lineage* permite ver os *data sources* utilizados no relatório selecionado no ponto 1. A seguinte figura, ilustra a vista "*Lineage*".





1 Opções de exportar, guardar e fazer *download* do relatório.

2 Disponibiliza opções de visualização do relatório.

3 Permite editar os conteúdos e *layout* do relatório.

4 Atualiza manualmente os conteúdos do relatório.

5 Filtrar as informações apresentadas no relatório.

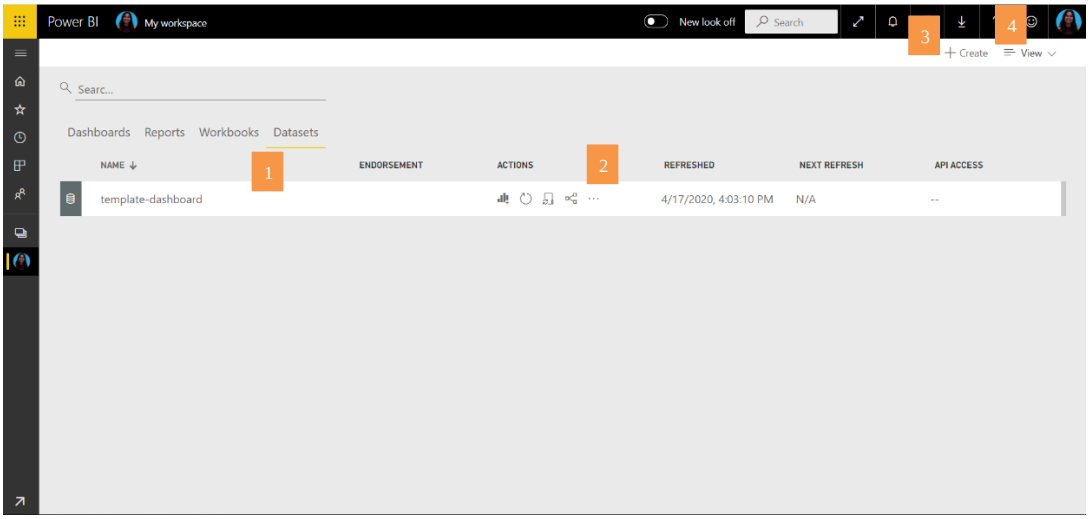
6 Mostra as páginas presentes no relatório.

7 Adicionar o relatório nos “Favoritos”.

8 Enviar o relatório por email.

9 Partilhar o relatório com colaboradores.

- Consultar os *datasets* utilizados nos relatórios



1 Lista de *datasets* utilizados nos relatórios implementados.

2 Cada dataset utilizado, disponibiliza um conjunto de opções personalizadas, como por exemplo, atualização manual dos dados, agendar a atualização.

3 Criar um *dashboard*, relatório, *dataset* ou streaming *dataset*.

4 Permite selecionar o tipo de vista por lista ou por *Lineage* dos relatórios. A vista por *Lineage* permite ver os *data sources* utilizados no relatório selecionado no ponto 1.