IPG Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

RELATÓRIO DE ESTÁGIO

Curso Técnico Superior Profissional
em Cibersegurança

Bruno Rodrigues Gil

novembro | 2020

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE ESTÁGIO

Bruno Rodrigues Gil

RELATÓRIO PARA A OBTENÇÃO DO DIPLOMA DE TÉCNICO SUPERIOR PROFISSIONAL EM CIBERSEGURANÇA

Novembro/2020

Elementos Identificativos

Aluno
Nome: Bruno Rodrigues Gil
Número de aluno: 1012363
Curso: TeSP Cibersegurança
Ano Letivo: 2019/2020
E-mail: 1012363@sal.ipg.pt

Estabelecimento de Ensino
Estabelecimento de ensino: Escola Superior de Tecnologia e Gestão - Instituto Politécnico da Guarda
Morada: Av. Dr. Francisco Sá Carneiro, 50 – 6300-559 Guarda
Telefone: +351 271 220 100
E-mail: ipg@ipg.pt
Orientador: Prof. Doutor José Carlos Coelho Martins da Fonseca

Local de Estágio
Empresa de acolhimento: XCUDO
Morada: R. Marcos de Assunção 4, 2800-663 Pragal
Telefone: +351 927 484 445
E-mail: info@xcudo.pt
Website: xcudo.pt
Supervisor na empresa de acolhimento: Gonçalo Lourenço

Período do Estágio: 26 de Fevereiro a 10 de Julho

Duração do Estágio: 750 horas

Agradecimentos

Um primeiro agradecimento ao Instituto Politécnico da Guarda pela iniciativa de criar este curso especializado na área de Cibersegurança, algo ainda não muito desenvolvido pelas instituições de ensino pelo país, assim como a todos os professores que me acompanharam durante a duração do curso assim como toda a ajuda que me foi oferecida tendo entrado sem quaisquer bases de informática.

Também agradecer à empresa XCUDO e ao meu supervisor Gonçalo Lourenço por me aceitarem para a realização deste estágio, pela oportunidade de trabalhar diretamente na área da Cibersegurança, pela maneira como fui recebido e integrado na empresa, pela positividade e bom espírito de equipa, por toda a ajuda que me foi oferecida e pela forma como foram tratadas as dificuldades que fui tendo durante todo o percurso.

Resumo

Este estágio foi realizado na empresa XCUDO para obtenção do grau de técnico superior profissional em Cibersegurança no período de 26 de fevereiro a 10 de julho de 2020.

Durante este espaço de tempo tive a oportunidade de pôr em prática os conhecimentos que obtive durante o curso assim como aprender a expandir a partir dos mesmos de forma a integrar novas tecnologias.

Numa fase inicial foi-me oferecida formação nas tecnologias que iria precisar durante a duração do estágio, assim como alguns conceitos de segurança de informação. Após esta fase, o meu primeiro projeto foi colaborar no desenvolvimento da aplicação Caravela, onde programei em Python uma forma de gerar uma lista de credenciais como combinação email/senha e uma forma de a distribuir ao público de forma a obter informação sobre que tipo de atacantes iria atrair.

Na segunda fase deste desenvolvimento, foi-me requisitado a instalação e configuração de um *honeypot*, assim como um IDS, que formariam um sistema que permitisse centralizar a informação a ser transmitida pelas várias máquinas para facilidade de obtenção de *logs*, de forma a detetar e analisar qualquer atividade suspeita.

Como última parte do estágio fui desafiado ajudar num caso de procura de informação, onde o objetivo seria procurar quaisquer informações de um cliente que pudessem ter sido publicadas tanto em fóruns conhecidos como outros mais obscuros na *dark web*, ou mesmo não intencionalmente, como pela má configuração de servidores, através da técnica de Google Dorking.

Palavras-Chave: Threat Intelligence, Honeypots, Desenvolvimento de aplicações, Engenharia social, Virtualização.

Índice

Índice de Figuras

Lista de Acrónimos

API - Application Programming Interface

FTP - File Transfer Protocol

IDE – Integrated Development Environment

IDS - Intrusion Detection System

JSON - JavaScript Object Notation

RAM – Random Access Memory

SSH – Secure Shell

1. Introdução

Nos dias de hoje, a segurança de informação é tão importante como a segurança física, e estando cada vez mais serviços informatizados, mais problemas de segurança são abertos a possíveis atacantes. No entanto, não é possível a uma empresa ou pessoa singular ter os recursos necessários para se proteger de qualquer tipo de ataques, sendo necessário ter uma visão realista do perigo que se enfrenta ao ter informação sensível na internet. Desta forma, é então necessário um estudo das vulnerabilidades a que se está exposto, uma melhor compreensão dos danos que seriam causados por um ataque informático bem-sucedido, qual o tipo de informação que é mais cobiçada pelos atacantes, e qual o seu método de operação.

É nesta área da segurança de informação que o presente relatório se foca, na procura de informação sobre atacantes para que melhor se possam definir estratégias de defesa consoante o tipo de perigo em que se encontra, assim como a instalação de medidas de segurança que permitam ter uma visão dos ataques que seriam utilizados contra a empresa no caso da descoberta de uma vulnerabilidade tornada pública.

1.1 Objetivos

O objetivo geral deste estágio é colaborar no desenvolvimento da aplicação Caravela que já está em curso por parte da empresa, sendo o estágio focado no módulo Deception. Este módulo tem como objetivo a criação de ferramentas de contrainteligência para integração na aplicação. A última fase do estágio é colaborar num caso de procura de *leaks* de informação.
O estágio encontra-se dividido em quatro fases:

1. Fase inicial de formação em Python, Docker, MongoDB e RabbitMQ.
2. Desenvolvimento de *script* para gerar credenciais.
3. Configuração de *honeypot* e dos serviços de deteção de intrusões Security Onion e Graylog.
4. Procura de possíveis *leaks* de informação através de Google Dorking.

1.2 Plano Trabalho

Após ser aceite para estágio pela empresa, o meu supervisor apresentou um plano de estágio (Figura 1) que consiste na ajuda ao desenvolvimento da aplicação Caravela, através de uma fase inicial de formação nas tecnologias que ela usa, assim como formação em *threat intelligence*, uma fase no desenvolvimento do módulo de *deception* da aplicação, uma fase de configuração de sistemas e uma última fase de *counter intelligence*.

Módulo	Tarefa	Horas
Formação	Curso Python	38
Formação	Curso RabbitMQ	24
Formação	Curso MongoDB	24
Formação	Curso Docker	24
Formação	Cybersec	40
Desenvolvimento	Script Cybersec Módulo 1 Dev 1	80
Desenvolvimento	Script Cybersec Módulo 1 Dev 2	80
Desenvolvimento	Script Cybersec Módulo 1 Dev 3	80
Desenvolvimento	Script Cybersec Módulo 1 Dev 4	80
Configuração	Ambiente Cybersec 1	24
Configuração	Ambiente Cybersec 2	24
Configuração	Ambiente Cybersec 3	24
Configuração	Ambiente Cybersec 4	24
Configuração	Ambiente Cybersec 5	24
Intel	Intel Case 1	80
Intel	Intel Case 2	80
		750

Figura 1 - Plano de trabalho

1.3 Apresentação Empresa

A XCUDO (pronunciado 'ESCUDO', logotipo Figura 2), é uma empresa *startup* fundada em 2018 por uma equipa portuguesa de três profissionais na área da Cibersegurança com uma vasta experiência no setor bancário e de telecomunicações, desde o nível regulatório até às habilidades mais técnicas.

Figura 2 - Logotipo XCUDO

A empresa oferece uma vasta gama de serviços de Cibersegurança, como análise de superfície, monitorização de *leaks* de informação, políticas de segurança, análise forense, gestão de incidentes de segurança, entre outros.

1.4 Aplicação Caravela

Este projeto inclui-se no desenvolvimento da aplicação Caravela (Figura 3), em desenvolvimento pela XCUDO, com o objetivo de facilitar a relação da Cibersegurança a empresas, disponibilizando várias ferramentas que permitem rápida e automaticamente criar e implantar novas defesas contra possíveis atacantes na rede da própria empresa. Dentro das várias funcionalidades da Caravela, o trabalho desenvolvido foca-se no módulo de Counter-Intelligence, responsável por detetar e analisar o risco que o cliente tem em sofrer um ciberataque, assim como uma simulação dos danos causados após uma fuga de informação.

Figura 3 - *Mockup* Aplicação Caravela

1.5 Estrutura do relatório

O Capítulo 1 dá a introdução ao projeto assim como à empresa em que fui inserido, assim como enquadramento do trabalho realizado e objetivos para o estágio.

O Capítulo 2 demonstra as ferramentas e tecnologias mais importantes utilizadas durante a realização do estágio.

O Capítulo 3 dá a conhecer as tarefas realizadas durante toda a duração do estágio, expressa as dificuldades e ideias apresentadas durante o decorrer do desenvolvimento da aplicação assim como trabalho para clientes.

O Capítulo 4 finaliza o relatório com a conclusão, dando uma reflexão sobre a experiência obtida estando a trabalhar para uma empresa de Cibersegurança.

2. Apresentação de tecnologias utilizadas

Neste capítulo são explicadas as tecnologias mais relevantes utilizadas durante a realização do estágio e as suas funcionalidades, como o IDE que foi utilizado, *software* de virtualização e distribuições de IDS.

2.1 Visual Studio Code

Figura 4 - Logotipo Visual Studio Code

O Microsoft Visual Studio Code (Figura 4) foi o IDE utilizado durante a fase de formação (figura 5) e toda a fase de desenvolvimento, com Python e JSON. Esta é uma ferramenta de Ambiente de Desenvolvimento Integrado *open-source* desenvolvida pela Microsoft com data de lançamento a 29 de Abril de 2015, fruto do projeto VSCode da Microsoft. Ela tem suporte para centenas de linguagens e com várias funcionalidades como suporte para *debugging*, *Refactoring*, destacamento de sintaxe, capacidade de *auto-complete* de código, criação de *snippets* e integração com Git assim como instalação de um grande leque de plugins para facilitar o desenvolvimento[1].

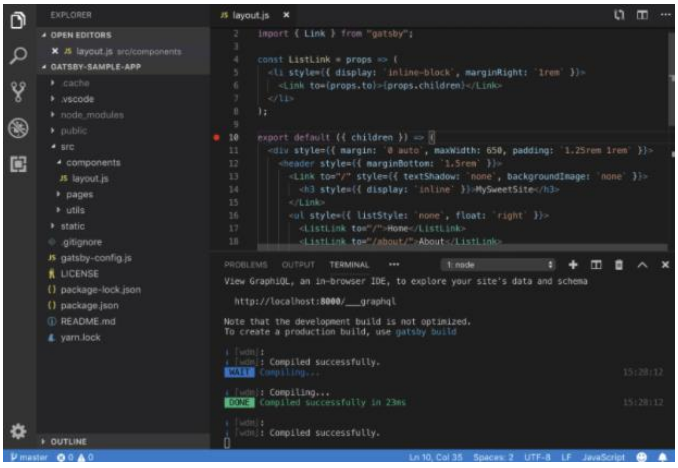

Figura 5 - Interface Visual Studio Code

2.2 Git

O Git é um sistema de controlo de versões utilizado para projetos de qualquer tamanho, permitindo guardar o histórico de edições de um ou mais ficheiros, com capacidade de revisão de código e de *branching*, que funciona através de repositórios, contendo o histórico completo dos ficheiros lá inseridos, facilmente revertendo a uma versão anterior caso seja necessário.

Este *software* foi utilizado durante toda a fase de desenvolvimento para o *upload* dos *scripts* escritos por mim e sincronização com o módulo da aplicação em desenvolvimento, dando assim possibilidade de facilmente implementar os *scripts* na aplicação e de ter o código revisto pela equipa.

2.3 VirtualBox

Figura 6 - Logotipo VirtualBox

O VirtualBox (Figura 6) é uma aplicação de virtualização *open-source* desenvolvida pela Oracle que pode ser instalada em Windows, macOS, Solaris e Linux.

Este programa permite ao utilizador carregar vários sistemas operativos dentro de um único sistema operativo *host* (Figura 7). Esta tecnologia foi utilizada durante o estágio para a criação do servidor de *honeypot* em Ubuntu Server e dos servidores de *intrusion detection* (IDS) Security Onion e Graylog.

Este programa também permite liberdade na configuração do hardware a ser utilizado pelas máquinas, se o sistema operativo de origem assim o permitir, como a modificação da placa de rede a ser reconhecida, possibilidade de deteção de periféricos, modificação da quantidade e qualidade de armazenamento assim como resolução e vários elementos gráficos (Figura 7)[2].

De notar a capacidade de uma máquina virtual ser isolada do sistema operativo em que está a correr, não permitindo que qualquer *malware* que se encontre instalado na máquina se transfira para o sistema principal. É, portanto, uma ótima ferramenta para deteção de atacantes, sendo possível a rápida reconfiguração de uma nova máquina caso algum sistema seja comprometido.

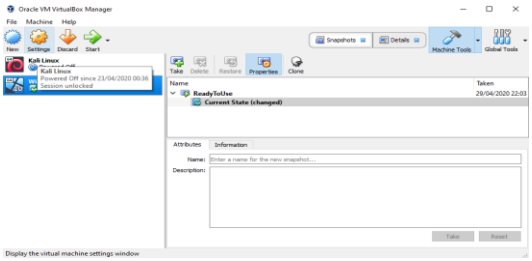

Figura 7 - Interface VirtualBox

2.4 Rsyslog

O Rsyslog é um software *open-source* para envio de mensagens contendo *logs* do sistema através de uma rede IP, com capacidades como envio através de TCP, configurações flexíveis, criação de *logs* diretamente para base de dados e possibilidade de obter o caminho percorrido por uma determinada mensagem[3].

Este software foi utilizado durante o estágio para o envio de *logs*, desde o servidor *honeypot* que foi criado para o servidor de deteção de intrusões (IDS Security Onion e Graylog).

2.5 Security Onion

Figura 8 - Logotipo Security Onion

Security Onion (Figura 8) é uma distribuição Linux grátis e *open-source* para monitorização de Cibersegurança empresarial, gestão de *logs* e avaliação de ameaças à rede. Inclui um vasto leque de ferramentas, como Kibana, Logstash, Suricata, Snort, Elasticsearch, Sguil, Squert, CyberChef, NetworkMiner, entre outros [4]. O sistema vem com um instalador fácil de utilizar que permite a sua configuração em minutos.

Esta distribuição foi o primeiro IDS que me foi apresentado pela equipa, e o que era previsto utilizar durante a fase de configuração de *honeypots*. No entanto, foi mais tarde trocado pelo IDS graylog.

2.6 Graylog

Figura 9 - Logotipo Graylog

Graylog (Figura 9) é uma distribuição Linux para gestão de *logs* que permite a captura, armazenamento e análise em tempo real de terabytes de informação, com simplicidade na forma de procura e visualização da informação (Figura 10). Permite também a instalação de *plugins* que facilitam a organização e a facilidade de utilização [5].

Este foi o IDS sugerido depois de não ter sido possível trabalhar com o IDS Graylog, e o que acabei por utilizar durante toda a fase de configuração para receber e analisar os *logs* do *honeypot* configurado na fase anterior.

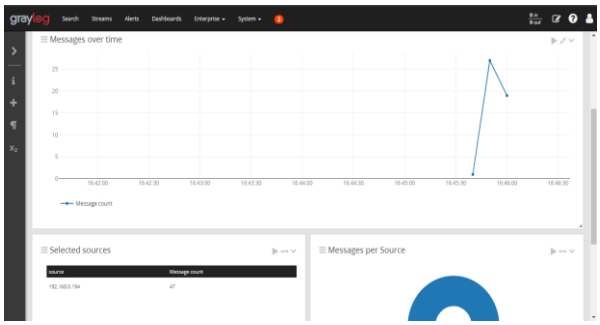

Figura 10 - Interface Graylog

3. Tarefas realizadas

Este capítulo explica as tarefas realizadas durante o curso do estágio, estando divido em quatro subcapítulos consoante o plano de estágio para mais fácil compreensão: fase de formação (150h), desenvolvimento (320h), configuração (120h) e fase final de *inteligence* (160h), em que foi utilizada para procura de informação sensível.

3.1 Formação

A fase inicial do estágio foi de formação nas tecnologias que iriam ser necessárias durante o decorrer do estágio. Nesta fase foi introduzido o projeto assim como a forma como a aplicação iria funcionar, o que seria esperado da minha parte e algumas tarefas a desenvolver para lá do desenvolvimento da aplicação.

A primeira fase foi formação na linguagem Python, que iria ser utilizada durante uma grande parte da duração do desenvolvimento da aplicação. Tendo já aprendido Java durante o período de aulas, a transição para Python foi muito fácil.

Uma grande vantagem da linguagem Python em relação ao Java é a sua simplicidade, permitindo criar a mesma funcionalidade com muito menos linhas de código e complexidade. Isto faz o Python ser uma linguagem bastante útil para *pequenos, embora poderosos, scripts*.

Após aprender Python, tive alguma formação em MongoDB, programa de gestão de bases de dados, Docker, *software* de virtualização ao nível do sistema operativo e RabbitMQ, *software* de mensagens. A formação nestas tecnologias tinha como objetivo a familiarização com a forma como a informação é trocada dentro de uma aplicação, para que mais facilmente conseguisse compreender o que me seria pedido durante a fase de desenvolvimento no período de estágio.

Na parte final de formação, foi-me introduzido o projeto em que iria trabalhar, as tecnologias que utilizava, de que forma estava desenvolvido e o que seria esperado de mim, assim como formação na área de *threat intelligence*, com um foco na criação de *decoys*.

3.2 Fase de desenvolvimento

Durante a fase de desenvolvimento, o objetivo seria ajudar no desenvolvimento da aplicação Caravela, nomeadamente o seu módulo de *deception*, onde me foi pedido que criasse uma forma de gerar credenciais e as disponibilizar ao público.

Inicialmente foi me apenas pedido o *script* que gerasse credenciais para atrair um atacante. Para tal, o *script* que programei gera informação a partir de dois ficheiros .txt, em que um dos ficheiros contém uma lista de apelidos e o outro contém passwords.

O *script* escolhe aleatoriamente uma letra, e dos ficheiros um apelido e uma senha. Depois cria uma lista de emails em estilo *letra.apelido@example.com* e atribui-lhe uma senha (Figura 11).

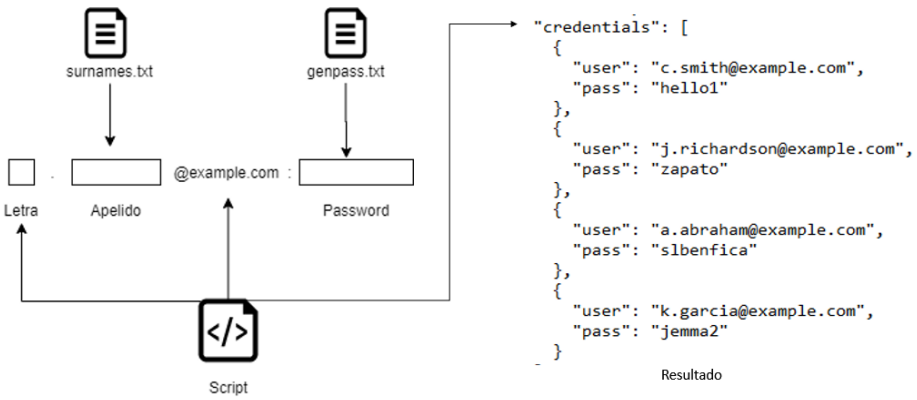

Figura 11- Script para gerar credenciais

Para além destas credenciais, contém indicação do IP da máquina alvo, porta ou serviço ativo. O ficheiro é então gerado em JSON (Figura 12) para facilmente comunicar com a aplicação Caravela. O script encontra-se em **Anexos 6.1.**


```
{
    "orderid": "12ediewi29",
    "count": 1,
    "data": [
      {
        "service": "nginx",
        "ip": "23.213.231.12",
        "action": "post",
        "count": 1,
        "credentials": [
          {
            "user": "c.smith@example.com",
            "pass": "hello1"
          },
          {
            "user": "j.richardson@example.com",
            "pass": "zapato"
          },
          {
            "user": "a.abraham@example.com",
            "pass": "slbenfica"
          },
          {
            "user": "k.garcia@example.com",
            "pass": "jemma2"
          }
        ]
      }
    ]
}
```


Figura 12 - Informação gerada pelo script

Na segunda fase de desenvolvimento, o objetivo seria criar uma forma de disponibilizar a informação gerada, de forma a atrair possíveis ataques informáticos. Para tal, o meu supervisor apresentou-me a forma como a aplicação funcionava e como interagia com Python.

Desenvolvi então um segundo *script* que formata o ficheiro criado anteriormente em JSON (Figura 12) em texto legível (removendo a formatação) e o envia para o fórum Pastebin através da API do site, sob conta anónima (Figura 13). O *script* em questão encontra-se em **Anexos, 6.2**.

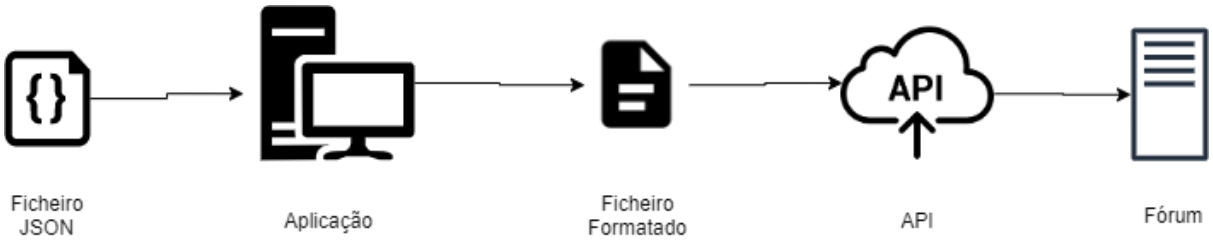

Figura 13 - Script para *post* de credenciais

Na imagem (Figura 14) está representado a forma como se poderia ver no site após o *upload* do ficheiro. Todos os conjuntos email/senha foram gerados aleatoriamente, mas de forma a que pareçam minimamente credíveis.

```
service:nginx
ip:127.0.0.1
action:post
count:20
credentials:
m.stevens@example.com:mexico
f.williams@example.com:lorena
l.evans@example.com:999999
m.anderson@example.com:whatever
z.thomas@example.com:friends
d.anderson@example.com:pass123
```

Figura 14 - Exemplo credenciais genéricas postadas

Esta fase seria não só para atração de possíveis atacantes, mas também para a propagação de informação falsa para ofuscação de qualquer perda legítima. Isto obriga a uma maior cautela por parte de alguém que consiga obter informação sensível.

Durante toda a fase de desenvolvimento, à medida que iam sendo criados os *scripts* era feito o seu *upload* para o repositório através da tecnologia Git, e o código era revisto pelo resto da equipa, sendo que quando precisava de ajuda facilmente conseguia cooperar com outro membro da equipa ou reverter uma alteração.

3.3 Fase de Configuração

Depois de terminada a fase de desenvolvimento dos *scripts*, a próxima fase seria a criação de um servidor que servirá de *honeypot*, estando propositadamente vulnerável, assim como um sistema de deteção de intrusões (IDS) que receba informação sobre os *logs*, enviados remotamente (Figura 15).

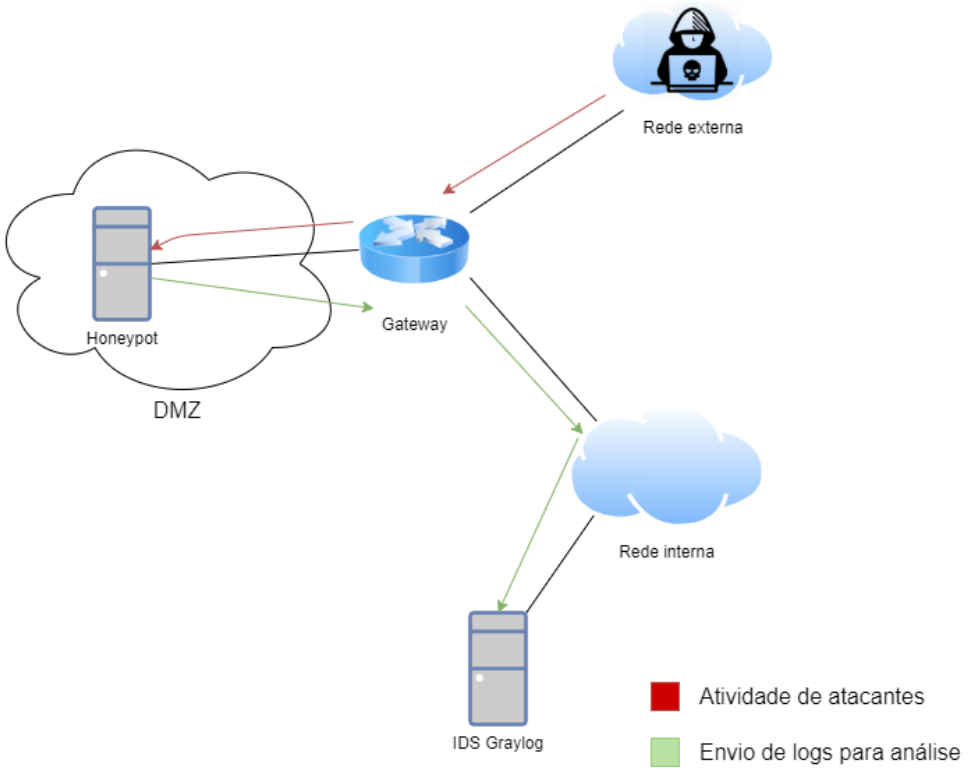

Figura 15 - Arquitetura honeypot e IDS

Para a fase de testes do *honeypot*, foi utilizado como sistema operativo a distribuição Ubuntu Server 18.04. Esta distribuição foi instalada numa máquina virtual com 2048 Megabytes de RAM através do *software* VirtualBox, sendo ativados os serviços de SSH, FTP e rsyslog.

Foi então configurado o *software* rsyslog, que irá enviar os *logs* deste servidor para o IDS Graylog remoto (que também irá ser configurado), permitindo assim uma fácil visualização da informação em tempo real. Para enviar os *logs* pela rede, basta adicionar ao fim do ficheiro de configuração do rsyslog (Figura 16) que se encontra em /etc/rsyslog.conf a linha `" *.* @ip:porta "`, onde ip e porta são o endereço do servidor para onde os *logs* serão enviados e a porta à escuta nesse endereço.

```
$PrivDropToGroup syslog

#
# Where to place spool and state files
#
$WorkDirectory /var/spool/rsyslog

#
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf

$ActionQueueFileName queue
$ActionQueueMaxDiskSpace 1g
$ActionQueueSaveOnShutdown on
$ActionQueueType LinkedList
$ActionResumeRetryCount -1

*.* @192.168.0.77:514
```

Figura 16 - Configuração ficheiro rsyslog.conf

Após o *honeypot* estar configurado de forma a enviar os seus *logs* remotamente, foi instalada a distribuição Linux SecurityOnion (Figura 17). Esta distribuição foi fácil de instalar, no entanto demorava muito a responder, e após a sua instalação estar completa, ficava bloqueado no ambiente de trabalho. O reinício do sistema não resolvia o problema, continuando a bloquear após o *boot*.

Procedeu-se à reinstalação e reconfiguração da distribuição por várias vezes, com diferentes configurações de RAM e disco, mas todas sem sucesso. Concluiu-se que o problema estava ligado à falta de capacidade de processamento por causa da virtualização, tendo sido necessário procurar uma alternativa.

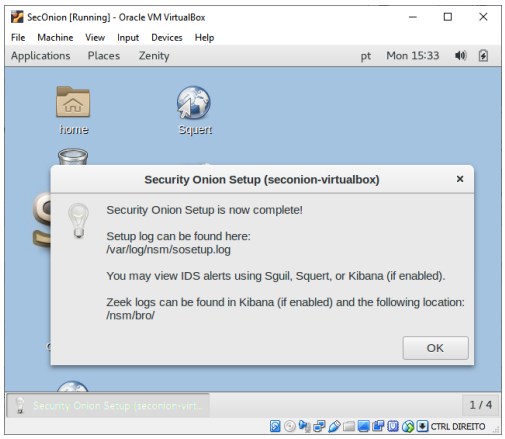

Figura 17 - Instalação Security Onion

Ao explicar o problema ao meu supervisor sobre a performance do Security Onion, foi sugerido alterar a distribuição em uso para o IDS Graylog, também com o objetivo de capturar informação sobre falhas de segurança exploradas no *honeypot* a correr Ubuntu Server.

A instalação do Graylog foi muito fácil, pois o próprio *website* permite o *download* de uma imagem já pré-instalada, sendo apenas necessário importar o ficheiro para o software de virtualização (Figura 18). Depois de importado, basta configurar as definições de rede.

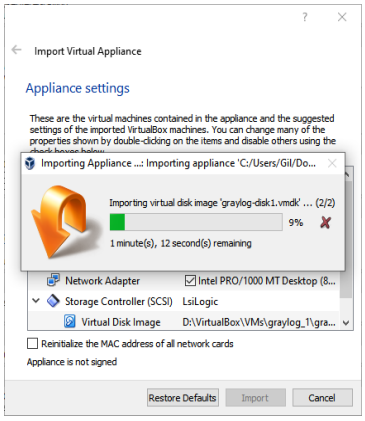

Figura 18 - Importar ficheiro graylog.ova

Depois de configuradas as definições de rede, basta abrir a interface *web* do Graylog e seguir as instruções de configuração que são apresentadas.

Nesta fase são configuradas que portas abrir para receção de informação, definir alertas para atividade suspeita e como serão apresentados ao utilizador (Figura 19).

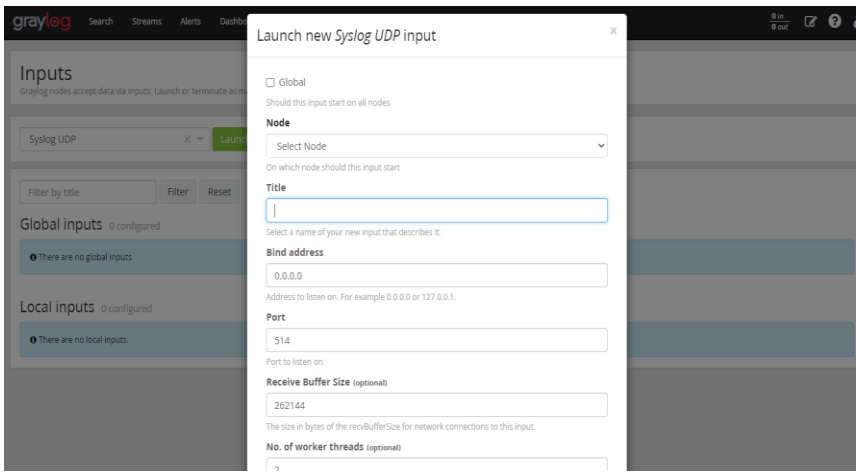

Figura 19 - Configuração Graylog

Se configurado corretamente, os *logs* relativos ao servidor Ubuntu (*honeypot)* configurado previamente deverão aparecer na interface *web* do IDS Graylog (Figura 20). Isto permite facilmente supervisionar uma ou mais máquinas para atividade de atacantes, tendo assim uma visão de um cenário em que a empresa sofra uma fuga de informação.

```
sshd[3176]: PAM service(sshd) ignoring max retries; 6 > 3
sshd[3176]: Disconnecting invalid user iewajro 192.168.0.56 port 62397: Too many authentication failures [preauth]
sshd[3176]: PAM 5 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.0.56
sshd[3176]: error: maximum authentication attempts exceeded for invalid user iewajro from 192.168.0.56 port 62397 ssh2 [preauth]
sshd[3176]: Failed password for invalid user iewajro from 192.168.0.56 port 62397 ssh2
sshd[3176]: pam_unix(sshd:auth): check pass; user unknown
sshd[3176]: Failed password for invalid user iewajro from 192.168.0.56 port 62397 ssh2
sshd[3176]: pam_unix(sshd:auth): check pass; user unknown
sshd[3176]: Failed password for invalid user iewajro from 192.168.0.56 port 62397 ssh2
```

Figura 20 - Output logs do honeypot

Nesta fase, o objetivo principal é a recolha de informação sobre os métodos de ataque e tecnologia à disposição, assim como a qualidade e complexidade dos agressores. Qualquer tentativa de acesso é registada e qualquer *software* que seja inserido por um atacante é guardado na máquina para futura análise, podendo assim ser criado um mapa com origem de ataques, qual das informações geradas pela aplicação foi utilizada pelo atacante, qual o método de ataque e que serviço foi explorado.

A seguinte figura (Figura 21) demonstra o resultado de um *honeypot* de SSH criado pela XCUDO em Novembro 2019, onde é possível diferenciar atacantes por país de origem.

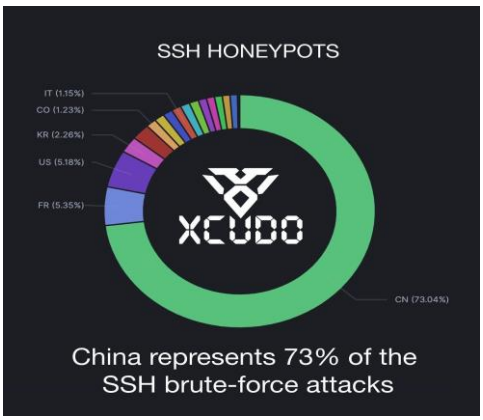

Figura 21 - Estudo de ataques por país [6]

3.4 Intelligence Case (Google Dorking)

Na fase de inteligência, foi introduzida a técnica de *Google Dorking*. Esta técnica utiliza pesquisas no motor de busca Google com parâmetros especiais, que permitem a procura de informação específica por *headers* ou certa informação contida no corpo da mensagem. Com esta técnica de pesquisa consegue-se facilmente encontrar serviços mal configurados dentro de uma organização que estão expostos ao público.

Durante este período de estágio, trabalhei com a equipa na procura de possíveis *leaks* para uma empresa, procurando emails que tivessem sido comprometidos em fóruns que são frequentados por *hackers* (Figura 22), assim como qualquer outra informação pessoal que pudesse ser utilizada contra a empresa.

Figura 22 - Exemplo Fórum de Hacking

Uma ferramenta bastante útil, foi a utilização do *site* hunter.io (Figura 23), que automaticamente procura endereços de email relativos a uma determinada empresa, assim como a fonte onde foi encontrado.

Esta ferramenta faz com que seja facilmente obtida informação sobre possíveis *leaks*, assim como o nome de funcionários e segurança na empresa relativa a dados pessoais.

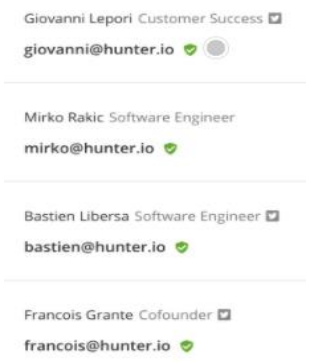

Figura 23 - Exemplo resultados hunter.io

Para além de simples endereços de email, é possível a obtenção de informação pessoal associando nomes de funcionários a perfis nas redes sociais e, consequentemente, obter formas de perceber os horários e procedimentos da empresa. Isto dá, a um atacante, formas de facilmente criar um ataque de engenharia social para obter ainda mais informação.

Sendo que a segurança de informação não depende apenas de programas informáticos, mas também dos recursos humanos da empresa, muitos dos ataques são feitos através de emails que parecem legítimos (Figura 24), vulnerabilidade que muito dificilmente é mitigada por uma equipa informática sem a cooperação de toda a empresa.


AT autoridade tributária e aduaneira

AVISO IMPORTANTE

Exmo(a) Senhor(a)

Está em curso o procedimento de actualização da Lista de Devedores na Internet.

Verifica-se que em seu nome existem processos de execução fiscal para cobrança de dívidas fiscais, que preenchem os requisitos para a inclusão na Lista de Devedores.

Assim, venho por este meio recomendar que proceda, o quanto antes, à regularização da sua situação tributária, procedendo ao pagamento da dívida, desse modo, ponto termo, ao procedimento de inclusão do seu nome na Lista de Devedores na Internet.

Total de débitos a partir de 30 de Janeiro de 2017 - € 2419,88

DOWNLOAD e-fatura

Esta mensagem de correio eletrónico e qualquer dos seus ficheiros anexos, caso existam, são confidenciais e destinados apenas ao(s) pessoa(s) ou entidade(s) acima referida (s). Se não for o destinatário desta mensagem, ou se ela lhe foi enviada por engano, agradecemos que não faça uso ou divulgação da mesma. A distribuição ou utilização da informação nela contida é VEDADA. Se recebeu esta mensagem por engano, por favor avise-nos de imediato, por correio eletrónico, para o endereço acima e apague este e-mail do seu sistema.

Figura 24 - Tentativa de engenharia social

No final de cada semana era feita uma reunião entre a equipa, onde cada elemento dava um relatório da informação recolhida, assim como discutidos eventos relacionados com novos tipos de ataques e novas formas de proteger a empresa que poderiam pôr em risco a segurança já em prática.

Isto permitia que, consoante a informação encontrada, fossem facilmente dirigidos os esforços na procura de informação para o que estaria sob maior risco e, consoante as intenções da empresa, gerir o seu orçamento no setor da sua segurança de informação.

4. Conclusão

Este estágio foi uma ótima forma de ter a minha primeira experiência na área profissional da informática, tendo tido oportunidade de aplicar os conhecimentos que tinha obtido ao longo da realização do curso.

As disciplinas de Programação e Bases de Dados, assim como Administração de Redes e Serviços foram determinantes para a conclusão das tarefas que me foram pedidas na duração do estágio, ainda que as tecnologias não fossem as mesmas, criaram bases que fizeram a transição bastante fácil.

O que mais senti falta foi não estar familiarizado com a tecnologia Git, que considero ser essencial para trabalho de desenvolvimento de aplicações, sendo uma das primeiras coisas que tive de aprender antes do início do estágio por me ser requisitado por parte da empresa, ainda que me tivessem ajudado bastante nessa área mesmo após o início do estágio.

Durante toda a duração do estágio tive vários desafios a superar, no entanto o bom dinamismo da equipa permitiu que conseguisse atingir todos os objetivos que me tinham sido propostos, expondo as minhas dúvidas e partilhando ideias quando possível, facilitando bastante o meu trabalho. Ter estado integrado na XCUDO permitiu me também ter noção do dia a dia de uma empresa focada na área da segurança informática, das necessidades das empresas e das expectativas por parte dos clientes.

5. Webgrafia

[1] Visual Studio Code – Code Editing. Redifined, code.visualstudio.com, consultado a 20/07/2020

[2] Oracle VM VirtualBox Manual, virtualbox.org/manual/ch01.html , consultado a 1/08/2020

[3] Features – rsyslog, https://www.rsyslog.com/features/ consultado em Novembro 2020

[4]Can Security Onion replace your commercial IDS?, por J.M. Porup, https://www.csoonline.com/article/3453199/what-is-security-onion-and-is-it-better-than-a-commercial-ids.html, consultado em 04/06/2020

[5] GrayLog: Como Instalar E Configurar Esta Poderosa Ferramenta De Logs Para Sua Rede, por Pedro Delfino, https://e-tinet.com/linux/graylog/ consultado em Outubro 2020

[6] We turned on our SSH honeypots and instantly got friendly visitors, por XCUDO.pt, https://pt-pt.facebook.com/XCUDO.PT/photos/a.500493853813578/637234626806166/, consultado em 20/07/2020

6. Anexos

6.1 Script para gerar credenciais

```
import random
import json
from string import ascii_lowercase

nmails = int(input("Number of emails to generate:")) #should be parsed as an argument instead

#var init
slines = 0
plines = 0
credlist = []

try:
  with open("surnames.txt","r") as surfile: #save number of lines in files
    for line in surfile:
      slines += 1
except FileNotFoundError:
  print("File surnames.txt not found")

try:
  with open("genpass.txt","r") as passfile:
    for line in passfile:
      plines += 1
except FileNotFoundError:
  print("File genpass.txt not found")

if slines > 0 and plines > 0: #if files not empty
  alphabet = ascii_lowercase
  def genCred(ncreds):
    with open("surnames.txt","r") as surfile, open("genpass.txt","r") as passfile:
      allsur = surfile.readlines()
      allpass = passfile.readlines()
```

```
      for i in range(ncreds):
        letter = random.choice(alphabet)
        surname = random.choice(allsur)
        surname = surname.rstrip('\n')
        password = random.choice(allpass)
        password = password.rstrip('\n')
        cred = "{0}.{1}@example.com".format(letter,surname)
        credlist.append({"user": cred, "pass": password })

  genCred(nmails)

togo =  {
         "service": "nginx", #should come from the app
         "ip": "127.0.0.1",
         "action": "post",
         "count": nmails,
         "credentials": credlist
        }
with open("fakemails.json", "w") as result:
  json.dump(togo, result, indent = 6)
```

6.2 Script para Postar as Credenciais geradas

```
#!/usr/bin/env python
import pika
import json
import os
import time
import socket
import subprocess
import sys
import requests

def main():
    #Connect to RabbitMQ
    connection = pika.BlockingConnection(
        pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    #Queue declaration
    channel.queue_declare(queue='deception_breacrumbs_pastebin_orders')

    #New order received:
    def callback(ch, method, properties, body):
        order = json.loads(body)
        for data in order['data']:
            process_order(order['orderid'],data) #Process the order

    #Waiting for new messages.
    channel.basic_consume(
        queue='deception_breacrumbs_pastebin_orders',            on_message_callback=callback,
auto_ack=True)

    print(' [*] Waiting for messages. To exit press CTRL+C')
```

```
    channel.start_consuming()

def main_local(json_file):
    with open(json_file, 'r') as f:
        order = json.load(f)
        for data in order['data']:
            process_order(order['orderid'],data) #Process the order

#To process the order
def process_order(orderid, data):
    print('Pastebin POST: ' + data['service'])

    url = 'https://pastebin.com/api/api_post.php'


    #Format the output
    output = ('{0}:{1}\n\nCredentials:').format(data['service'],data['ip'])
    for credentials in data['credentials']:
        output = output + ('\n{0}:{1}').format(credentials['user'],credentials['pass'])


    values = {'api_option' : 'paste',
            'api_dev_key' : 'Dev key goes here',
            'api_paste_code' : output,
            'api_paste_private' : '0',
            'api_paste_expire_date' : '1W'
            }

    post = requests.post(url, data=values)
    pasteid = ((post.text).split('/')[-1]) #Get the pastebinID

    #Retrieve the pastebinID from Pastebin:
    if "https://pastebin.com/" in post.text:
        result = {
```

```
            'orderid':orderid,
            'success': True,
            'message': 'Breadcrumb posted successfully',
            'pastebinid': pasteid
            }

    else:
        result = {
            'orderid':orderid,
            'success': False,
            'message': 'Breadcrumb failed to post',
            'pastebinid': 'None'
            }
    print(result)
    return result

if __name__ == '__main__':
    if len(sys.argv) > 1:
        main_local(sys.argv[1])
    else:
        main()
```