Tecnologia de Segurança (1º Ano MEI)

Trabalho Prático 2

Relatório

Duarte Oliveira, António Guerra, Melânia Pereira e Paulo R. Pereira
{pg47157, pg47032, pg47520, pg47554}@alunos.uminho.pt
Grupo 16

4 de maio de 2022

Resumo

O presente documento tem como objetivo descrever a componente de segurança de informação e estrutura de suporte associada ao desenvolvimento de um serviço de identificação digital . Serve este documento para a análise e identificação de potenciais aspetos que representem algum tipo de risco para o sistema quando em produção. Será feita uma abordagem detalhada de qualquer aspeto que deva ser revisto, apresentando uma avaliação sobre os atributos mais críticos identificados, definindo também uma ordem de prioridade. Opcionalmente poderá existir a exposição de alguma solução alternativa para algum tipo de problema que deva ser corrigido.

Conteúdo

1 Introdução **2**

2 Análise e Especificação **3**

2.1 Especificação de Requisitos 3

3 Vulnerabilidades e *exploits* **4**

3.1 Aplicação mID e Aplicação leitora 4

3.2 Entidade emissora 6

4 Atacantes **14**

5 Outras vulnerabilidades e possíveis soluções **15**

6 Conclusão **16**

Capítulo 1

Introdução

Neste documento começa-se por mostrar o uso de abreviaturas ou macros com parâmetros acima definidas.

- **UC**: Unidade Curricular
- **TS**: Tecnologia de Segurança
- **BLE** Bluetooth Low Energy

Capítulo 2

Análise e Especificação

O processo de estudo e investigação aplicado a este relatório está em correspondência com o material que tem sido apresentado e lecionado ao longo das aulas práticas de TS.

2.1 Especificação de Requisitos

De modo a desenvolver-se uma análise sucinta foi proposta a utilização de algumas das técnicas aqui apresentadas:

- Catalogação de vulnerabilidades e *exploits*
- Catalogação de fraquezas típicas
- Modelação de ameaças:
  - Orientado aos ativos
  - Orientado aos atacantes
  - Orientado ao software
- Análise de risco

Tentando ter um raciocínio linear e organizado, utilizamos estas formas de análise para elaborarmos este relatório.

Capítulo 3

Vulnerabilidades e *exploits*

3.1 Aplicação mID e Aplicação leitora

De forma a proceder à autenticação e download de dados de algum documento ou à atualização de dados relativos ao documento em questão, é utilizada comunicação **TCP/IP** entre os dispositivos que contém ambas as aplicações.
Debilidades associadas ao **TCP/IP**:

- Não existe a garantia de que as mensagens enviadas ou recebidas (segmentos de dados) na comunicação entre a **mID** e a infraestrutura da entidade emissora não possam ser interceptadas por terceiros. Isto cria a possibilidade de diferentes tipos de ataques:
  - Ataque *man in the middle*.
  - *Denial of Service*.
- Estes ataques implicam que não existe proteção contra acessos não autorizados, permitindo que haja *hijacking* de sessões, ou pela previsão do valor de **seq** ou fazendo **sniffing** da rede, usando ferramentas como o *whireshark*.
- Os dados não são sujeitos a qualquer processo de cifragem, contribuindo para que a extração de informação posteriormente à sua obtenção seja facilitada.

O formato de dados que compõe a troca de dados - **JSON** - pode também ser alvo de algum escrutínio na sua análise de segurança.

- Tem um formato *standard*, o que permite que se possam fazer ataques de injeção, substituindo ou eliminando dados, comprometendo a comunicação
- Por norma não é sujeito a algum tipo de cifragem

Pode-se constatar que usar o protocolo **TCP/IP** como forma de comunicação, associado ao uso de ficheiros do formato *JSON* para troca de informação pode facilmente pôr em causa o sistema ao nível da **confidencialidade**, visto que terceiros mal intencionados, e com conhecimento suficiente, podem ter acesso a informações que não devem, de todo, poder ter acesso.

Neste sentido, também é importante referir que estas vulnerabilidades põem também em causa a **disponibilidade** e a **integridade**, visto que a comunicação - ou os dados que a compõem - podem ficar indisponíveis ou corrompidos.

Como possível solução, seria importante usar algum tipo de autenticação para manter a sessão segura ou o uso de um protocolo, como por exemplo o **TLS** na sua versão mais recente - **1.3** - de modo a garantir a segurança dos dados.

Conforme enunciado, a conexão entre o dispositivo do portador e leitor pode usar diferentes tecnologias.

- ***Bluetooth Low Energy***
  - Existe um problema associado ao **tamanho da chave temporária** usada para gerar a chave de cifragem associada à comunicação entre nodos - *o tamanho é demasiado pequeno* - o que pode ser um problema fatal na segurança dos dados.
  - As ferramentas de software e hardware usadas para ataques a plataformas que comunicam com *BLE* são **facilmente obtidas** por indivíduos com um baixo nível de especialização na àrea, definindo os requisitos mínimos necessários para se fazer um ataque destes como muito baixa.
  - Dados trocados em comunicações em que se usa *BLE* podem ser facilmente "***sniffed***"por terceiros e posteriormente examinados.
  - É possível acontecer ***BLE spoofing***, onde um atacante copia o endereço *MAC* de um equipamento *BLE* fazendo-se passar por ele.
  - Equipamentos com *BLE* estão sujeitos a ataques de *DoS* (*Denial of Service*). Aplicado ao nosso caso em concreto, o bombardeamento de pedidos de conexão à aplicação do portador em resposta a pacotes de *advertising* pode impedir a conexão entre o *user* e a aplicação leitora.
  - As comunicações *BLE* estão susceptiveis a ataques ***man in the middle***, podendo haver a alteração de mensagens entre partes por parte de uma entidade maliciosa.
  - Algumas vulnerabilidades, associadas ao funcionamento do *BLE*, foram reportadas em **diferentes equipamentos**, sendo que foi fruto da nossa investigação:
    * a existência de pelo menos **8 relatórios** de vulnerabilidades de segurança ou problemas técnicos associados a *crashes* do serviço do *BLE* em diferentes equipamentos.
    * a existência de pelo menos **7 relatórios** de vulnerabilidades de segurança associados a incidentes de *deadlocks* no serviço do *BLE* em diferentes equipamentos.
    * a existência de pelo menos **2 relatórios** de vulnerabilidades de segurança associados a um *security bypass* do serviço do *BLE* em diferentes equipamentos.

  De forma a evitar problemas técnicos recomenda-se o uso da versão mais atualizada do *Bluetooth Low Energy* visto que quanto mais atualizado o serviço esteja, menos provável ocorram problemas que comprometam o funcionamento do mesmo e mais seguro será o seu funcionamento, visto que os *updates* de versões muitas vezes contém atualizações de segurança que

procuram resolver problemas que tenham sido identificados em versões anteriores.
Para além disso, focando a nível da segurança contra terceiros, seria importante o uso de métodos de pareamento - principalmente o método ***OutOfBand***, visto que envolve autenticação fora do canal de comunicação do *BLE* - garante a participação exclusiva das partes a quem a comunicação se reserva.
Também seria importante reforçar, mais uma vez, que os dados devem ser cifrados *à priori* de serem transmitidos de forma a não comprometer a confidencialidade dos mesmos. O uso de um esquema de cifragem fim a fim, de forma a que só as duas partes envolvidas neste processo tenham acesso a estes dados pode ser uma solução segura para tal.

- ***NFC - Near Field Communication***

  Apesar de aparentemente difícil intromissão dado que a comunicação entre dispositivos está reservada a poucos centímetros, existem alguns casos reportados de ataques que são possíveis.

  - Com equipamento apropriado para a escuta e armazenamento da informação transmitida entre entre dispositivos em comunicação por *NFC*, existe a possibilidade de que um atacante tenha capacidade não só de extrair informação mas também de interferir na comunicação, com a corrupção de diferentes dados que a compõem. Apesar de improvável acesso a estes dados, é também recomendada a cifragem dos mesmos para ter uma comunicação o mais segura possível.

- ***WiFi-Aware***

  - Apesar de se ter feito uma investigação que a nosso ver foi bastante incisiva na procura de vulnerabilidades ou falhas associadas a este componente, não conseguimos encontrar nenhuma que parecesse relevante o suficiente para ser documentada. Neste sentido, este parece-nos ser o equipamento **mais recomendado** para as diferentes interações entre aplicações.

É importante referir que, apesar da necessidade de estarem documentados, estes riscos de ataque previamente registados necessitam de um nível de conhecimento razoável e algum equipamento especializado, pelo que a sua resolução ou troca por outra alternativa não é crítica, mas recomendada.

3.2 Entidade emissora

Infere-se para este entidade todos os riscos associados ao uso de uma comunicação baseada no protocolo *TCP/IP* na comunicação entre esta entidade e a aplicação mID ou com a aplicação de leitura reiterando todas as fraquezas, vulnerabilidades e alternativas apresentadas no respetivo tópico em 3.1.
Relativamente à infra-estrutura que compõe o *backend* do sistema há algumas considerações a se ter.

- *CentOS 7.8.2003*

  - A nossa investigação não obteve qualquer resultado aparentemente relevante de vulnerabilidades ou fraquezas que merecessem ser referidas para esta análise.

De qualquer das formas, é aqui verificável que não existe qualquer tipo de certificação, verificação ou garantia de segurança relativa a software associado ao projeto *CentOS*. Assim, ***se é esperada a utilização do CentOS com garantias de segurança, então esta provavelmente não será a melhor opção para tal***.

- *Django V3.0*
  - ***SQL Injection***
    Tendo sido documentada a 11 de Abril de 2022, verificou-se que os métodos *QuerySet.annotate()*, *aggregate()* e *extra()* são pontos vulneráveis a partir do qual se pode fazer um ataque de SQL Injection a este programa. Sendo uma vulnerabilidade avaliada com um potencial de risco de 8.2, tendo não só uma *baixa complexidade* mas também uma *baixa necessidade de especialização* e um *alto impacto* na **integridade** do sistema em si, esta é sem dúvida uma vulnerabilidade cuja resolução **deve ser priorizada**.
    De forma a solucionar este problema, é recomendada a atualização do *Django* para a versão 3.2.13 ou maior ou igual à 4.0.4, considerando que se pretende uma versão que seja superior à 3.0. referência

  - ***Insecure Permissions***
    A versão do *Django* que está a ser utilizada dispõe de permissões inseguras, comprometendo desde logo três dos principais campos ao nível da segurança de informação - **confidencialidade**, **integridade** e **disponibilidade**. Com um risco avaliado em 8.4 e uma baixa complexidade para se conseguir explorar esta vulnerabilidade, tem que existir uma **alta prioridade** para que haja a resolução da mesma.
    A melhor alternativa seria atualizar o *Django* para uma versão não inferior à 3.1.1. referência

  - ***HTTP Header Injection***
    Introduzida a 6 de Maio de 2021, esta vulnerabilidade tem um potencial de risco de 7.3. A versão do *Django* a ser tida em conta é vulnerável a um ataque de HTTP Header Injection, visto que versões do maiores ou iguais ao *Python3.9.5* removem mudanças de linha e indentações automaticamente ao invocarem o método *urllib.parse()*. Visto que a ferramenta *URLValidator* utilizava os métodos *urllib.urlsplit()* e *urllib.urlunsplit()* para criar uma variante de URL que já não contém indentação nem mudança de linhas, fazendo com que a expressão regular fosse válida para um valor de origem com caracteres inseguros.
    Tendo em conta a avaliação de risco e a baixa complexidade de ataque, este também deve ser uma vulnerabilidade a tratar, neste caso, atualizando também o *Django* para uma versão superior. referência

  - ***Web Cache Poisoning***
    Documentada a 19 de Fevereiro de 2021, esta vulnerabilidade encontra-se presente na versão considerada do *Django*, tornando a mesma desprotegida contra ataques de *Web Cache Poisoning*. **Consertada nas versões acima da 3.1.7 ou superiores**, existe a necessidade de que o *user* seja levado a fazer alguma ação em específico antes realmente a

vulnerabilidade poder ser explorada, sendo que após isso, há uma **total perda de disponibilidade** do serviço, que se manterá de forma persistente ou sustentada. É importante frizar que para se proceder a este ataque, é necessário se ter um alto nível de conhecimento, pelo que não é um tipo de ataque tão acessível, tendo por isso, uma classificação de risco de 5.9. referência

- *UWSGI*
  - ***Directory Traversal***
    Antes da versão 2.0.17, existe uma má gestão da verificação da *DOCUMENT_ROOT* durante o uso da opção *–php-docroot*. Com um ataque deste tipo, o atacante consegue ter acesso a ficheiros e pastas fora da diretoria na qual o mesmo se encontra, mesmo não tendo permissão para o fazer. Com a manipulação de ficheiros com a terminologia "../"ou variações desta, pode ser possível proceder-se ao ataque conforme descrito.
    Dentro desta vulnerabilidade afunilam-se outras. Em primeiro lugar, existe libertação de informação, dado que o atacante ganha acesso à estrutura de ficheiros e ao seu conteúdo. Para além disso também é possível ao atacante escrever ficheiros específicos ou substituir o conteúdo de ficheiros já existentes - **Zip-Slip**.
    Com um *score* de 5.0 e um impacto parcial na confidencialidade do sistema, esta vulnerabilidade não será problema se se tiver a versão mais recente do *UWSGI*. referência
  - ***Overflow***
    Com uma avaliação de 7.5 e um impacto parcial nas diferentes áreas associadas à segurança de informação - **Confidencialidade**, **integridade** e **disponibilidade** - esta vulnerabilidade resume-se ao facto da função *uwsgi_expand_path* - em versões iguais ou inferiores ao 2.0.15 deste serviço - poder sofrer um *buffer overflow* graças a uma diretoria de grande tamanho.
    O facto de ser necessária pouca especialização e não haver qualquer tipo de autenticação requerida, torna imperativo que se tenha a versão mais atualizada desta aplicação, de forma a que o servidor web desta aplicação tenha um funcionamento coeso e seguro. Desta forma, e apesar de considerarmos que o uso deste serviço implica o uso da versão mais recente, achamos por bem mencionar esta vulnerabilidade de forma a precaver problemas. referência

- PostgreSQL 12.4
  - ***Information Disclosure***
    O uso do comando *INSERT ... ON CONFLICT ... DO UPDATE* numa tabela permite que um *user* autenticado da base de dados pudesse ler bytes da memória, podendo assim retirar informações da mesma, violando a confidencialidade do sistema. Apesar disso não há intrusão ao nível da integridade e da disponibilidade do sistema. Por esses motivos o score desta vulnerabilidade é de 4.0. referência
  - ***Man-in-the-middle***
    Nesta versão do *PostgreSQL*, se uma aplicação criar uma conexão à base de dados esta apenas reutiliza os parâmetros básicos de conexão, deixando de parte parâmetros de segurança relevantes. Isto permite que se crie uma oportunidade para um ataque ***man-in-***

the-middle ou a capacidade para se observarem transmissões não cifradas.
Esta é uma vulnerabilidade com um impacto parcial a nível da **confidencialidade**, **integridade** e **disponibilidade** do serviço. Apesar disso, é necessário um nível razoável de especialização para se conseguir explorar esta vulnerabilidade, avaliada em 6.8. referência

- ***Elevation of Privilege***
  Um atacante com permissões para criar objetos permanentes em pelo menos um *schema* pode executar funções *SQL* sob a identidade de um *superuser*. Isto compromete a confidencialidade, integridade e disponibilidade do sistema, e ainda que o seja de forma parcial, há um grau considerável de informação que é libertada, um impacto moderado na modificação de ficheiros ou informação e possíveis interrupções na forma que o serviço é prestado. Com um *score* de 6.5 na escala *CVSS* está é outra vulnerabilidade que merece atenção, sendo que como solução seria viável garantir que uma versão deste serviço que não contemple esta vulnerabilidade, contando que a essa atualização não comprometa o correto funcionamento da base de dados principal deste sistema. referência
- ***Spoofing***
  No terminal interativo *psql*, se se usar o comando *\gset* ao fazer-se uma *query* de um servidor comprometido, o atacante pode executar código como a conta sistema operativo que está a executar o *psql*. Esta vulnerabilidade é de alto risco, sendo avaliada em 7.6, com total e completo impacto nos diferentes campos que garantem a segurança da informação do sistema - confidencialidade, integridade disponibilidade. A única atenuante é que são necessárias bastantes pré condições e algum conhecimento especializado para se conseguir proceder a este *exploit*. Ainda assim não existe nenhum mecanismo de autenticação que seja preciso ultrapassar. referência
- ***Man-in-the-Middle/SQL injection*** Quando o servidor está configurado para confiar ou utilizar um determinado tipo de autenticação, um atacante pode injetar *SQL queries* quando uma conexão é estabelecidade, mesmo existindo verificação, cifragem e certificação *SSL*. O impacto parcial na **confidencialidade**, **integridade** e **disponibilidade** do sistema conferem um *score* de 5.1 na escala CVSS. referência
- ***Overflow***
  Foi encontrada uma falha aquando a alteração de determinados valores em *arrays SQL*, onde campos por preencher permitem que utilizadores da *database* possam escrever *bytes* livremente numa grande escala da memória do servidor. Verificam-se falhas parciais ao nível da confidencialidade, com informação a ser libertada sem autorização. Um nível equivalente de impacto regista-se na integridade do sistema, onde se evidencia modificações ao nível de ficheiros e informação, não havendo total controlo sobre o que se pode alterar. Finalmente há também incidentes registados ao nível da disponibilidade do sistema, com ligeiras falhas de *performance* ou interrupções no serviço. referência

- *Ubuntu 20.04*
  - ***Samba***
    - O **Samba** file server permite que haja partilha de ficheiros entre diferentes sistemas operativos na mesma rede. Permite que se acedam e partilhem ficheiros com utilizadores de *Windows* e *macOS*.

O *Samba AD DC* pode ficar confuso com o *user* que um *ticket* representa. Isto pode levar um comprometimento total do domínio. Desta forma, esta é uma vulnerabilidade de alto risco, com um *score* de 9.0 na escala *CVSS*, tendo um **completo** impacto ao nível da **confidencialidade**, **integridade** e **disponibilidade**.
Finalmente, é preciso um baixo nível de especialização para explorar esta vulnerabilidade, o que a torna ainda mais perigosa. referência cve

* O **Samba** com versões anteriores à 4.13.17, 4.14.12, 4.15.5, com o *vfs_fruit* configurado vai permitir leitura e escrita na *heap*, especialmente a partir de ficheiros propositadamente criados para o efeito. Um atacante com acesso remoto, contendo permissões a ficheiros deste tipo pode executar código arbitrário com *root privileges*.
É uma vulnerabilidade de alto risco, tendo um score de 9.0 e um impacto completo em todos os parâmetros que avaliam a segurança de informação do sistema em si. Com a adjuvante de ser facilmente executável por qualquer indivíduo, esta é sem dúvida uma vulnerabilidade com elevada importância de resolução.referência cve
* Foi encontrada uma falha na forma que o **Samba** mapeia *users* do domínio para *users* locais. Um atacante autenticado pode usar esta falha para causar problemas a nível de permissões.
É uma falha avaliada em 8.5, com um impacto completo na **confidencialidade** e **integridade** do sistema, mas sem efeito ao nível da **disponibilidade** do mesmo. Apesar disto, este é mais uma vez, um ataque sem grande nível de especialidade necessária, sendo necessária a sua monotorização. referência cve

– ***Code execution***
Foi encontrado um escalonamento de privilégios na *utility pkexec* pertencente ao *polkit*. Esta aplicação é uma ferramenta desenhada para permitir *users* sem privilégios a correrem comandos como se fossem *users* com privilégios de acordo com políticas pré-definidas.
O problema começa quando a versão atual do *pkexec* (ultima revisão a 20/04/2022) não consegue gerir os parâmetros chamados de forma correta e acaba por tentar executar variáveis de ambiente como comandos. Um atacante pode tomar partido disto, definindo variáveis de ambiente de forma a que induza o *pkexec* a executar código à sua vontade.
Quando feito com sucesso, este ataque pode causar um escalonamento de privilégios dando privilégios de *admin* a *users* sem essa devida hierarquização na máquina alvo.
Esta é uma vulnerabilidade com um total impacto nas variáveis de **confidencialidade**, **integridade** e **disponibilidade**, com o mau indicador de que se trata de um falha fácil de se explorar, não sendo sequer necessário algum tipo autenticação para o fazer. Por esses motivos, encontra-se avaliada dentro dos parâmetros **CVE**, em 7.2. referência cve

– ***SEV-ES exploit***
Com um total impacto ao nível da **confidencialidade**, **integridade** e **disponibilidade** do serviço, esta vulnerabilidade caracteriza-se na falha do código *AMD* relativo ao *KVM* - um *hypervisor* disponível no *kernel module* do Linux. Este *hypervisor* utilizando o *SEV-ES* (Secure Encrypted Virtualization-Encrypted State) pode levar a leituras e escritas no *kernel host* por meio de um *VMGEXIT* malicioso para uma instrução de leitura ou escrita, usando o código de saída *SVM_EXIT_IOIO*.
Este problema dá origem numa **falha total do sistema** ou num **potencial ataque** de

Ghuest-Host (onde um sistema operativo "convidado"manipula o sistema operativo *host*)
Por este motivo esta é uma **falha crítica**, de valor 7.2, que não precisa de grande especialização ou autenticação. referência cve

- ***Overflow***
  O **AIDE** - *advanced intrusion detection environment*, que verifica e deteta alterações a ficheiros de sistema - em versões anterior à 0.17.4 permite que *users* locais obtenham *root privileges* a partir de metadados de ficheiros, devido a um *overflow* do *buffer* ao nível da *heap*.
  Sendo esta uma vulnerabilidade com um *score* de 7.2 na escala *CVSS* e tendo um impacto total na **confidencialidade** do sistema - visto que há revelação total de ficheiros de sistema - para além de uma classificação igualmente gravosa para os restantes parâmetros avaliativos, esta é, sem dúvida alguma, uma vulnerabilidade de alto risco, que deve ser tratada, procurando atualizar a componente *AIDE* para uma das versões superiores à 0.17.4, caso tal ainda não se verificasse. referência cve

É de notar que o *Ubuntu 20.04* **contém um vasto leque de vulnerabilidades**, muitas delas bastante recentes e atuais. Compreendemos a importância da enumeração das mesmas mas tendo em conta o interesse deste relatório tomamos a liberdade de nos cingirmos às mais preocupantes e com maior potencial de *exploit*.

- *Flask 1.0*
  - ***Remote Code execution and Gain Privileges***
    O *Flask* depende da serialização *pickle* que pode levar à execução de código de forma remota, ou alterações nos privilégios de forma local. Se um atacante tiver acesso ao armazenamento em *cache* existe a possibilidade de construir um *payload* fictício, envenenar a cache e executar código *Python*. Desta forma, e com um score *CVSS* de 7.5, existe um impacto na **confidencialidade**, **integridade** e **disponibilidade** que é parcial. Em contrapartida há um baixo nível de especialização necessário para explorar esta vulnerabilidade não havendo também verificação de algum tipo de ganho de acesso. referência

- *Gunicorn*
  - ***HTTP Request Smuggling***
    Existe uma vulnerabilidade associada ao facto deste serviço falhar ao processar os headers de *Transfer-Encoding* e *Content-Length*, quando ambos estão presentes nalgum pedido. Isto permite que informação conflituosa seja enviada relativamente ao tamanho do *package* levando a que no seu processamento se ganhasse acesso a dados sensíveis, comprometendo outros *users*, tendo sido dado *bypass* às medidas de segurança deste serviço. referência
    Apesar de tudo este *exploit* é avaliado com um nível médio - 5.6 - dado que necessita de uma grande complexidade de ataque para se conseguir ter sucesso na exploração desta vulnerabilidade.
    Como solução é aconselhado a atualização do *gunicorn* para uma versão maior ou igual à 20.0.1. referência

- ***HTTP Response Splitting***
  Esta vulnerabilidade compromete totalmente a **integridade** do serviço em si. Existe a possibilidade da alteração parcial ou total os ficheiros abrangidos pelo *gunicorn*. Basicamente o sistema fica facilmente desprotegido contra ataques HTTP Response Splitting graças à função de *process_headers*. Com um valor de risco avaliado em 7.5, esta vulnerabilidade é, sem dúvida, uma prioridade, pelo que é aconselhada a atualização do serviço para uma versão igual ou superior à 19.5.0. referência

Nota: Dada a não especificação de uma versão deste serviço, considera-se que ambas as vulnerabilidades apresentadas são para efeitos de **documentação**, visto que a versão mais atual é a ***20.0.1***, a qual já engloba *patches* referentes à resolução destas duas fraquezas.
Ainda assim, sentimos que seria boa prática apresentá-las.

- PostgreSql 12.1
  - ***Tampering***
    Existe uma falha no *ALTER... DEPENDS ON EXTENSION*, visto que subcomandos não fazem verificações das autorizações. Um atacante autenticado pode usar esta falha em certas configurações executar comandos de forma a comprometer a integridade da base de dados. Neste sentido apenas a integridade sofre algum tipo de impacto, sendo que não há qualquer falha na confidencialidade ou disponibilidade do serviço. A complexidade para explorar esta falha é média, o que se finda com um *score* de 3.5. referência
  - ***Sql Injection***
    Verificou-se que não há um tratamento correto do *search_path* quando se faz replicação lógica. Um atacante **autenticado** pode usar esta falha para executar um comando *sql* no mesmo contexto que o utilizador utilizou para a replicação. Vulnerabilidade avaliada em 4.6, com impacto confidencial nos diferentes campos de **confidencialidade**, **integridade** e **disponibilidade**.referência
  - ***Extension flaws***
    Algumas extensões do PostgreSQL não usam o *search_path* de forma segura no seu *script* de instalação. Um atacante com privilégios suficientes pode usar esta falha para induzir um administrador do sistema a executar um *script* feito com intenções malignas, durante a instalação de uma extensão. O facto de ter um impacto parcial na **confidencialidade**, **integridade** e **disponibilidade** deste sistema, conferem assim uma avaliação de 4.4 na escala *CVSS*. referência.

  Mais uma vez se refere que a melhor solução para contradizer estas vulnerabilidades é a atualização do serviço para a versão mais recente, desde que isso não comprometa o normal funcionamento desta base de dados de gestão.

- Docker 19.03.6
  - ***Daemon Crashes***
    Utilizar uma *Docker image* vai levar a falhas no *Doeckerd daemon*, pondo em causa o funcionamento da aplicação. Esta é uma vulnerabilidade com um score *CVSS* de 4.3, sem impacto direto na confidencialidade e na integridade da aplicação, mas trazendo alguns

problemas na disponibilidade do sistema.
Apesar disso, é necessário conhecimento especializado para conseguir fazer um ataque desta complexidade.
Independentemente de tudo, este problema está tratado no *Docker* 19.03.15, portanto essa seria a solução mais viável, se não houverem contra-argumentos para esta *atualização*.

– ***Extended privileges***
Existe uma vulnerabilidade ao utilizar a opção *–userns-remap* que permite que haja um escalonamento imprevisto dando acesso a permissões *root* que conseguem ter acesso e privilégios de leitura e escrita dos ficheiros contidos na pasta */var/lib/docker/(...)*, trazendo algumas preocupações ao nível da **integridade** do programa.
Com um *score* de 2.7 esta é uma vulnerabilidade de baixo risco, sem impacto aparente ao nível da **confidencialidade** e **disponibilidade**, podendo ser facilmente resolvida com a atualização para a versão do *Docker* 20.10.3.

Capítulo 4

Atacantes

De forma a complementar toda esta análise dos riscos, potencialidades e respetiva proposta de resolução das mesmas, quando indicado, segue-se uma discriminação de potenciais atacantes que possam ter algum tipo de interesse na exploração das vulnerabilidades anteriormente descritas, de forma a prever o comportamento do mesmo e haver uma preparação para potenciais ataques.
Desta forma:

- **Organizações criminosas ou hostis**
  - Sendo o mID um serviço com um grande volume de troca de dados e informações sensíveis, é também bastante atrativo para grupos com motivações económicas.
  - Noutra perspetiva, é facilmente percetível que um ataque informático com vista a debilitar ou comprometer a disponibilidade do serviço pode ser bastante importante não só para uma organização dita *convencional* como também para outras organizações associadas a governos ou regimes hostis que por algum motivo em específico vêm com bons olhos a interrupção desta aplicação.
- **Indivíduos com interesse pessoal**
  - Existirá sempre alguém que por algum motivo (prejudicial ou benéfico) precise de adulterar ou manipular o serviço, quer seja utilizador da aplicação do portador ou utilizador da aplicação leitora, de forma a sair beneficiado de alguma forma - por exemplo não poder ser convenientemente identificado, ou os seus dados não serem registados.
- **Indivíduos associados ao desenvolvimento da aplicação**
  - Existe o risco de alguma *backdoor* ser criada por um ou mais dos desenvolvedores deste projeto, de maneira a poder beneficiar de alguma vulnerabilidade, futuramente. É importante procurar fazer auditorias por parte de entidades independentes e verificar constantemente o código que é desenvolvido de forma a se obterem opiniões imparciais.

Capítulo 5

Outras vulnerabilidades e possíveis soluções

Tendo em conta as interações apresentadas na arquitetura deste sistema, apercebemo-nos que seria conveniente garantir que as aplicações são sempre acedidas por quem as deve usar. Desta forma seria importante ser requerido algum tipo de autenticação na inicialização da aplicação, quer na leitora, quer na do portador, ou de cada vez que houvesse a interação da mesma com outra entidade. Esta autenticação poderia ser com uma *password* ou código. Isto impediria que terceiros não legítimos com acesso direto aos equipamentos com esta aplicação tivessem pelo menos uma barreira a ultrapassar. Seria pertinente também adicionar também *two factor authentication*, idealmente em todas as operações mas especialmente naquelas que possam ser consideradas as mais sensíveis, tanto para o *user* como para o *leitor*.
Relativamente às diferentes vulnerabilidades apresentadas em capítulos anteriores, foi sempre feita uma abordagem de forma a apresentar a possível solução recomendada, procurando sempre enquadrar o tipo de vulnerabilidades com o modelo **STRIDE**. É de notar que só é viável optar pela solução sugerida desde que isso não comprometa o total funcionamento do sistema em si, e que independentemente da atualização para novas versões, isso não invalida a existência de outras vulnerabilidades a si inerentes. Em contrapartida, caso haja pelo menos a mitigação de um problema então deve-se ter em conta essa solução.
Desta forma seria sensato priorizar aquelas que possuem pelo menos, uma avaliação *CVSS* acima de 6, com especial foco para as vulnerabilidades críticas (com um *score* acima de 9).

Capítulo 6

Conclusão

Em conclusão, podemos referir que este trabalho prático se revelou de extrema importância, foi importante aprender e também rever conceitos e formas de investigação já lecionadas e utilizadas em anteriores trabalhos. Isto permitiu-nos a consolidação de conhecimentos e deu-nos uma boa familiarização com diferentes técnicas de levantamento de vulnerabilidades e riscos.
A temática também foi bastante interessante, visto que nos permitiu criar algum sentido de responsabilidade e de inserção no projeto em si. O facto de termos que organizar ideias e fazer algum *brainstorming* e investigação para produzir resultados foi também bastante importante e, convenientemente, faz parte do processo de análise de riscos.
Em retrospectiva sentimos que foi um trabalho bem feito, onde consideramos que conseguimos prever diferentes ideias, e também nos pormos em diferentes perspetivas de maneira a podermos avaliar diferentes lacunas que compõem este projeto.