

Trabalho prático 1 de Ambientes de Execução Seguros

Autores

- João Almeida (118340)
- Simão Andrade (118345)

Objetivos

Implementar um cofre digital à **prova de adulteração de arquivos** (TPDV), utilizando **Intel SGX enclaves**. O cofre pode ser destruído, mas nunca vai ser possível mudar o conteúdo dos arquivos sem que o cofre perceba. O foco desta implementação é a **integridade** dos arquivos, **não a confidencialidade**.

O programa deve ser capaz de:

- ☒ Criar um ficheiro TPDV.
- ☒ Adicionar um arquivo ao TPDV.
- ☒ Listar os arquivos no TPDV.
- ☒ Extrair um arquivo (ou todos) do TPDV.
- ☒ Calcular o hash de um arquivo no TPDV.
- ☒ Alterar a password do TPDV.
- ☒ Clonar o TPDV para outro SGX enclave.

Implementação

Os dados do TPDV são selados (*sealed*) e armazenados num ficheiro. Toda a informação é armazenada num **unsigned int array**.

O cabeçalho do ficheiro é composto por:

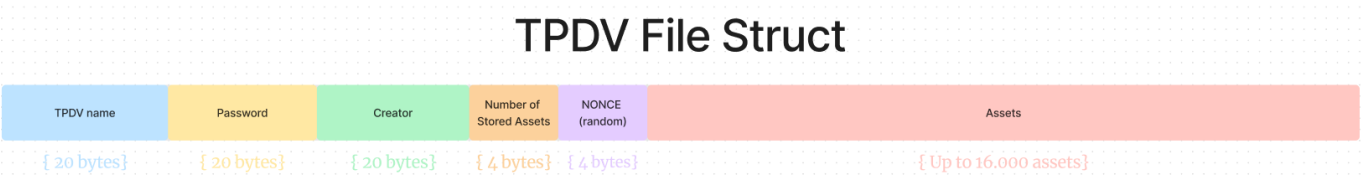


Fig. 1 - Header do TPDV

Nota: O campo **NONCE** representa os últimos 4 bytes do hash de todos os assets. Este é usado para verificar a integridade do TPDV, adicionando uma nova camada de segurança.

Cada ficheiro adicionado ao TPDV é composto por:



Fig. 2 - Estrutura de um arquivo no TPDV

Arquitetura do Programa

O programa é dividido em **dois tipos** de funções:

- **Seguras:** são executadas dentro do enclave e têm acesso a memória selada.
- **Não seguras:** são executadas fora do enclave e têm acesso a memória não selada.

Funções seguras	Funções não seguras
unsealed	create_tpdv
sealed	add_asset
get_sealed_size	list_assets
e1_check_password	change_password
e1_add_asset	retrieve_asset
e1_list_all_assets	check_asset_integrity
e1_get_asset_size	clone_tpdv
e1_retrieve_asset	
e1_change_password	
e1_get_asset_hash_from_vault	
e1_unseal_and_cipher	
e1_decipher_and_seal	

Funcionalidades

Criação do TPDV

Esta funcionalidade é responsável por criar um ficheiro TPDV. O ficheiro é criado com o cabeçalho do TPDV conforme o descrito anteriormente e sem arquivos.

A função no `App.cpp` tem o seguinte cabeçalho:

```
int create_tpdv(const uint8_t *filename, const uint32_t filename_size, const
uint8_t *password, const uint32_t password_size, const uint8_t *creator, const
uint32_t creator_size);
```

E devolve `0` em caso de sucesso e `1` em caso de erro.

Nesta implementação, a criação do TPDV é toda feita fora do enclave.

Adicionar um arquivo ao TPDV

Esta funcionalidade é responsável por adicionar um ficheiro ao TPDV. O ficheiro é adicionado ao array TDPV, sendo novamente selado e guardado no ficheiro.

A função no `App.cpp` tem o seguinte cabeçalho:

```
int add_asset(const uint8_t *filename, const uint32_t filename_size, const
uint8_t *password, const uint32_t password_size, const uint8_t *asset, const
uint32_t asset_size);
```

E devolve `0` em caso de sucesso e `1` em caso de erro.

Para manipular o ficheiro TPDV é feita uma ECALL (chamada para dentro do enclave) com a função `e1_add_asset`. Esta função é responsável por adicionar o arquivo ao TDPV dando *unseal* ao conteúdo do TPDV, adicionando o ficheiro e *seal* o conteúdo do TPDV.

Este fluxo é ilustrado usando a seguinte figura:

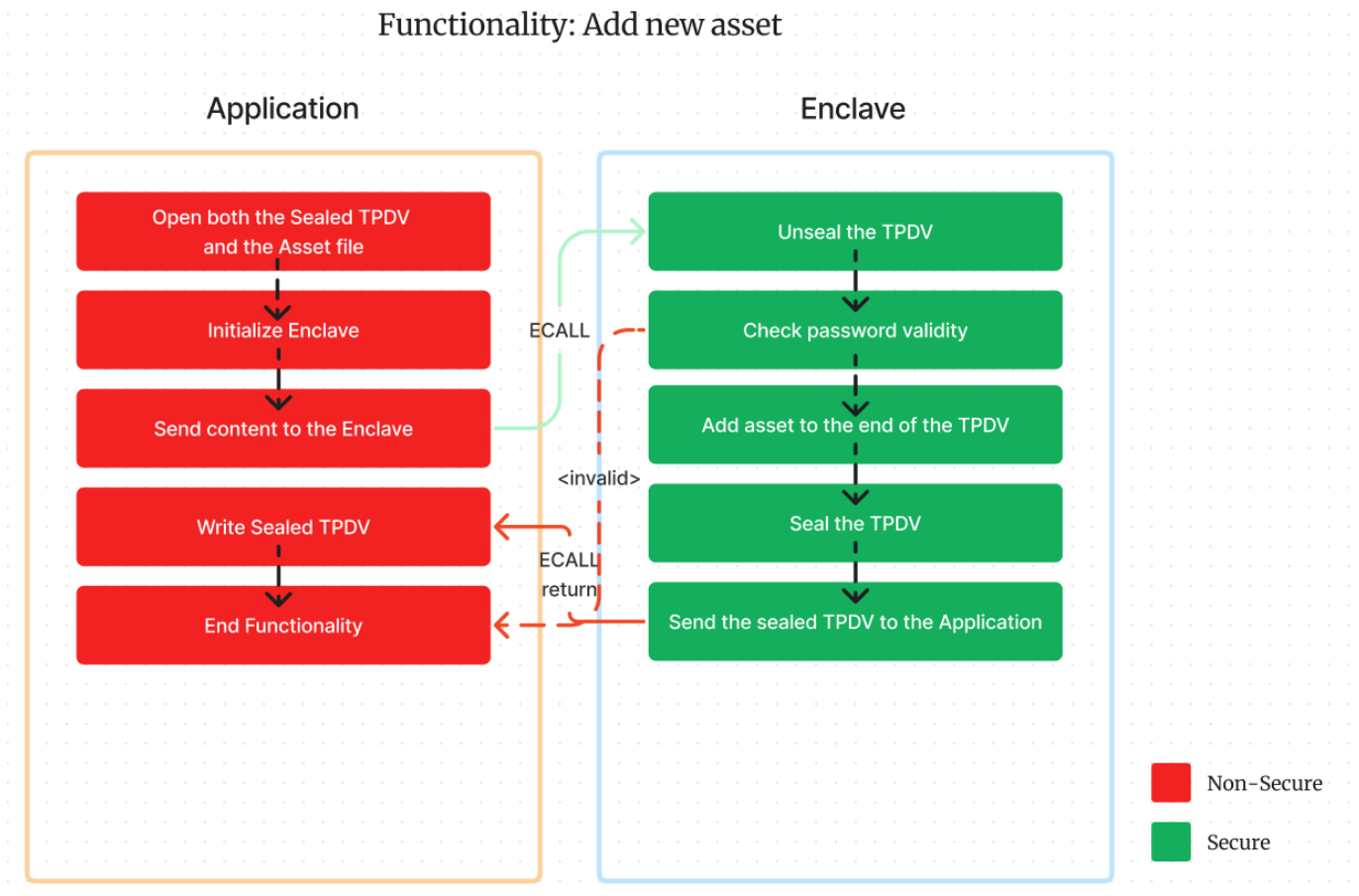


Fig. 3 - Adicionar um ficheiro ao TPDV

O processo de manipulação do ficheiro TPDV é feito **dentro do enclave** de modo a garantir que a informação é manipulada apenas pela informação fornecida pela aplicação cliente. Deste modo a integridade da informação é garantida, uma vez que a informação é selada e não pode ser alterada fora do enclave sem que seja detetado.

Listar os arquivos no TPDV

Esta funcionalidade é responsável por enumerar todos os ficheiros no TPDV. A função dá *unseal* ao TPDV e lê o conteúdo do array, enviando o resultado para o *stdout* através de um OCALL para a aplicação cliente.

A função no *App.cpp* tem o seguinte cabeçalho:

```
int list_assets(const uint8_t *filename, const uint8_t *password);
```

E devolve *0* em caso de sucesso e *1* em caso de erro.

Para dar *unseal* ao TPDV de forma segura, é feita uma ECALL para a função *e1_list_all_assets*. Esta função é responsável por dar *unseal* ao conteúdo do TPDV e devolver o conteúdo.

Este fluxo é ilustrado usando a seguinte figura:

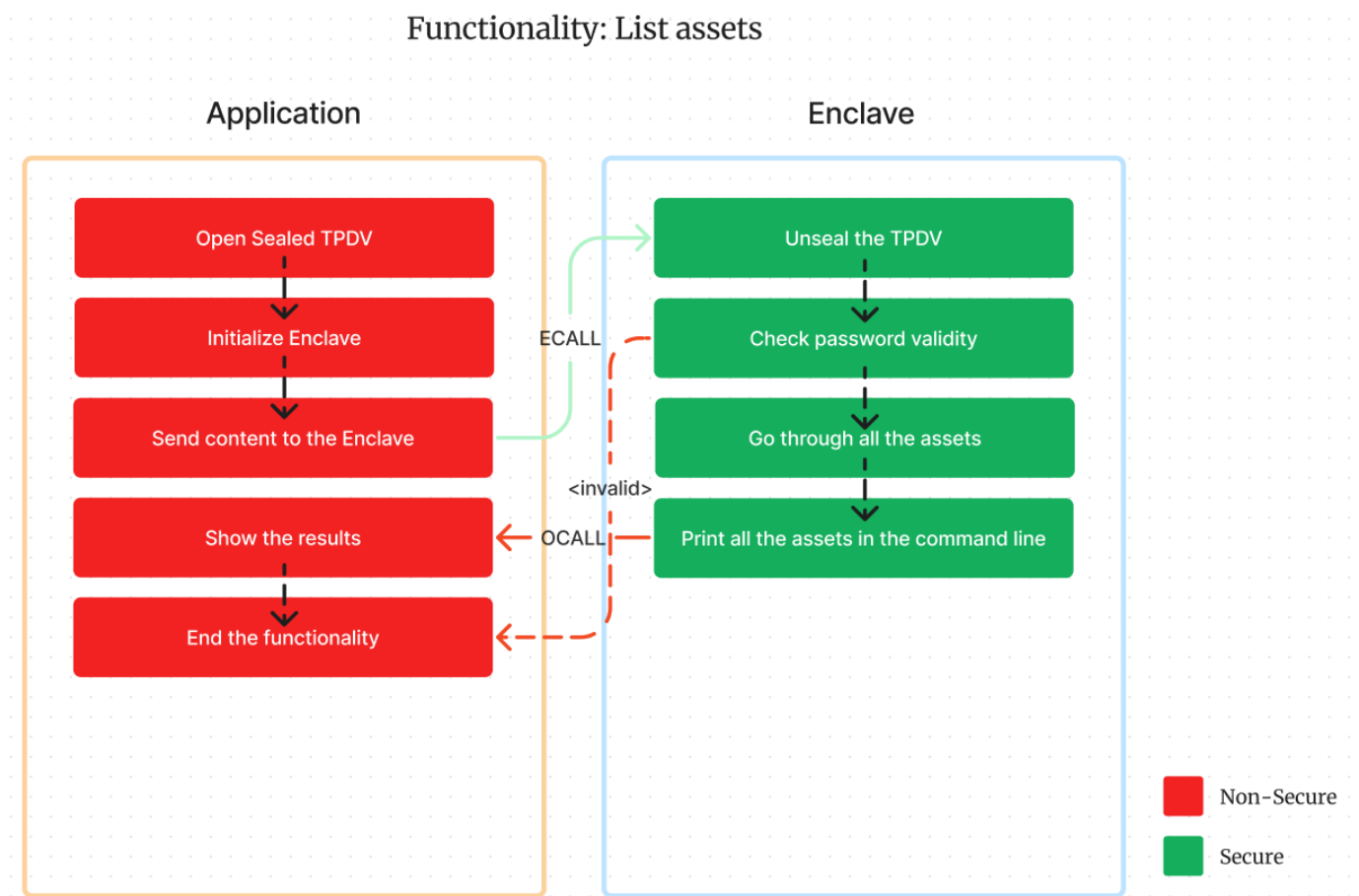


Fig. 4 - Listar os ficheiros no TPDV

Alterar as credenciais do TPDV

Esta funcionalidade é responsável por alterar a password do TPDV. A função dá *unseal* ao TPDV, verifica a password antiga, altera a password, dá *seal* ao TPDV e guarda o conteúdo no ficheiro.

A função no *App.cpp* tem o seguinte cabeçalho:

```
int change_password(const uint8_t *filename, const uint8_t *old_password,
const uint8_t *new_password);
```

E devolve **0** em caso de sucesso e **1** em caso de erro.

Para alterar a password de forma segura, é feita uma ECALL para a função `e1_change_password`. Esta função é responsável por dar *unseal* ao conteúdo do TPDV, verificar a password antiga, alterar a password, *seal* o conteúdo do TPDV e guardar o conteúdo no ficheiro.

Este fluxo é ilustrado usando a seguinte figura:

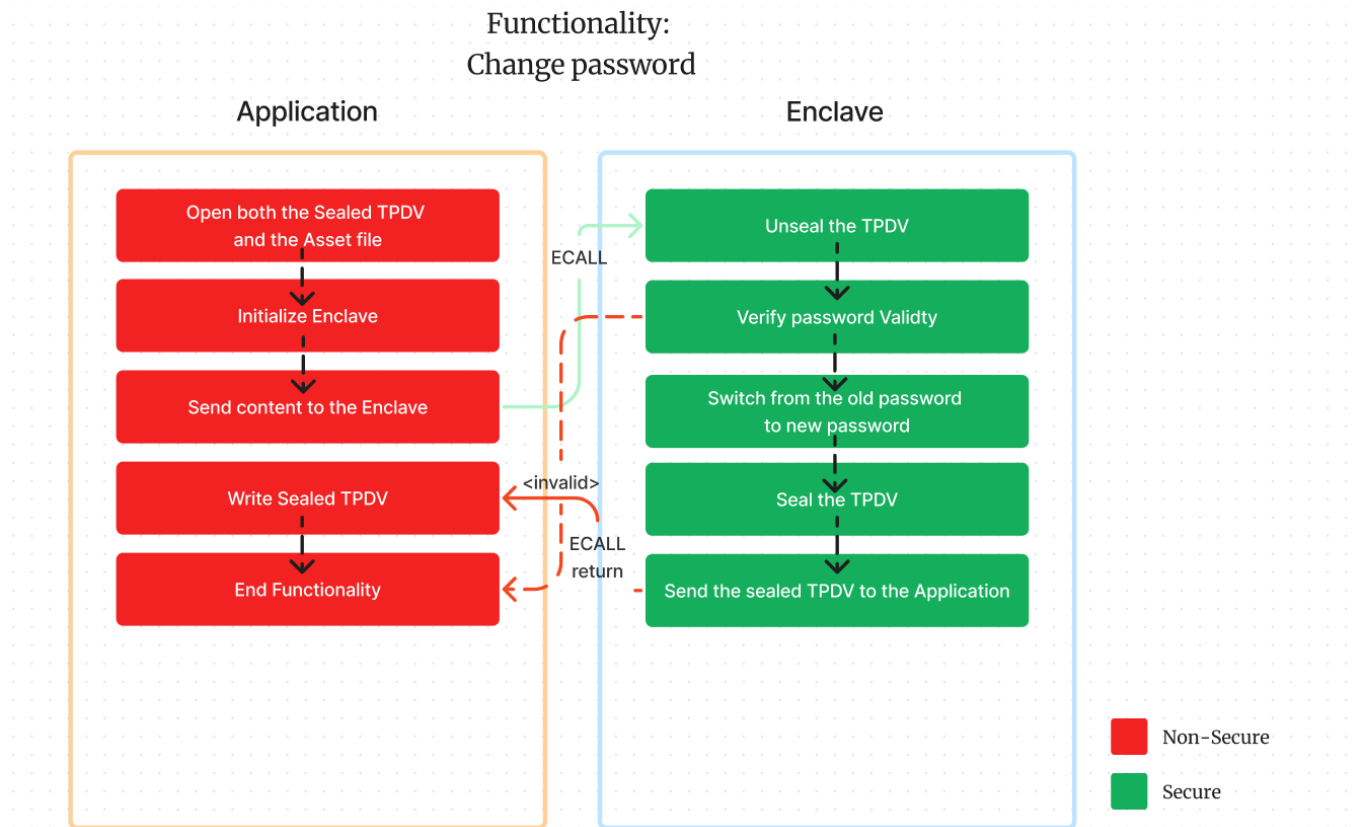


Fig. 5 - Alterar a password do TPDV

Verificar a integridade de um ficheiro no TPDV

Esta funcionalidade é responsável por verificar a integridade de um ficheiro no TPDV. A função dá lê o conteúdo do ficheiro, dá *unseal* ao TPDV e verifica se o hash do ficheiro corresponde ao hash do conteúdo guardado no TPDV.

A função tem o seguinte cabeçalho:

```
int check_asset_integrity(const uint8_t *filename, const uint8_t *password,
const uint8_t *asset_filename)
```

E devolve **0** em caso de sucesso e **1** em caso de erro.

Para verificar a integridade de um ficheiro de forma segura, é feita uma ECALL para a função `e1_get_asset_hash_from_vault`. Esta função é responsável por dar *unseal* ao conteúdo do TPDV, procurar o ficheiro e devolver o hash do ficheiro.

Este fluxo é ilustrado usando a seguinte figura:

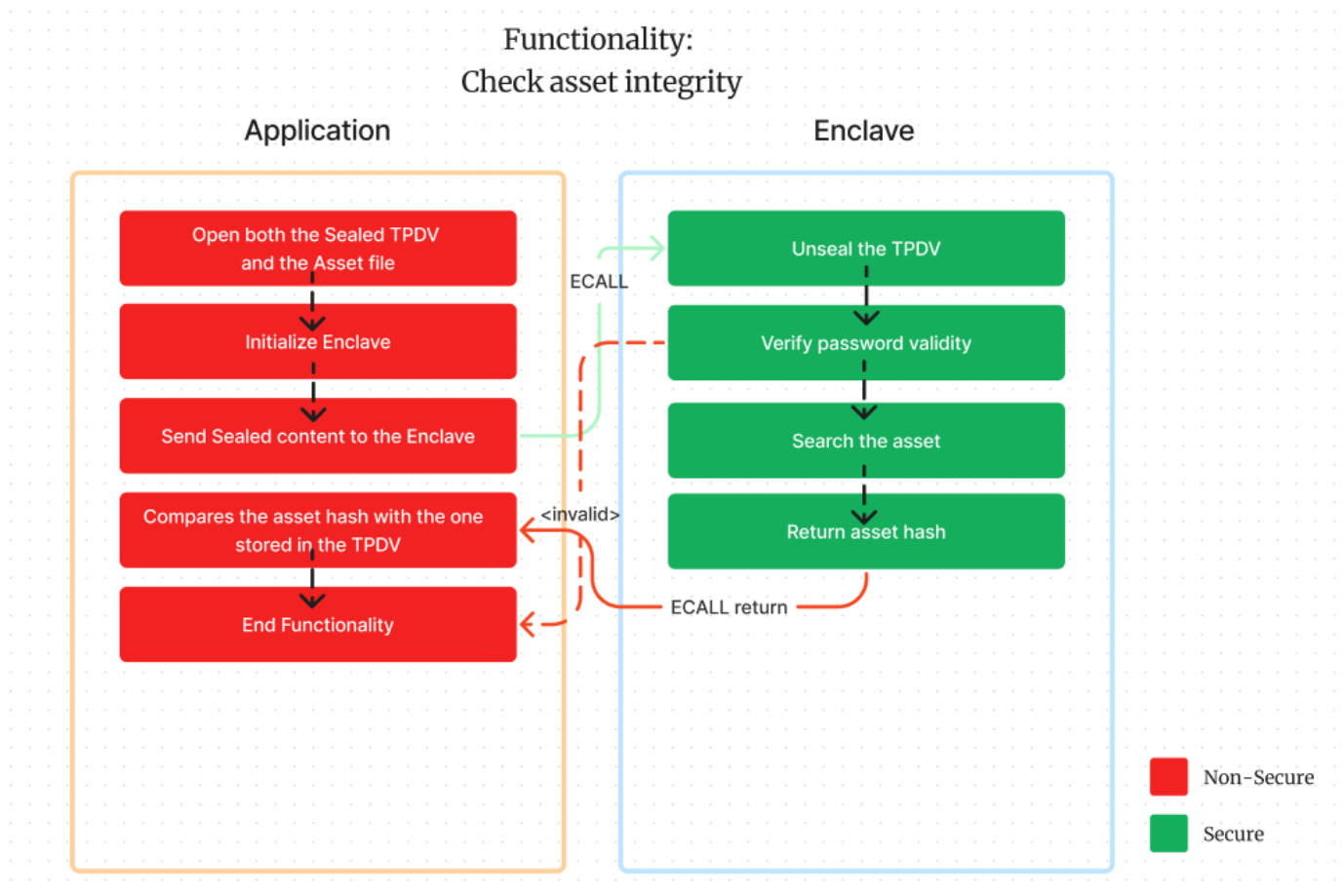


Fig. 5 - Verificar a integridade de um ficheiro no TPDV

Extrair um ficheiro do TPDV

Esta funcionalidade é responsável por extrair um ficheiro do TPDV. A função dá *unseal* ao TPDV, procura o ficheiro e devolve o conteúdo do ficheiro.

A função tem o seguinte cabeçalho:

```
int retrieve_asset(const uint8_t *filename, const uint8_t *password, const
uint8_t *asset_filename)
```

E devolve **0** em caso de sucesso e **1** em caso de erro.

Para extrair um ficheiro de forma segura, é feita uma ECALL para a função `e1_retrieve_asset`. Esta função é responsável por dar *unseal* ao conteúdo do TPDV, procurar o ficheiro e devolver o conteúdo.

Este fluxo é ilustrado usando a seguinte figura:

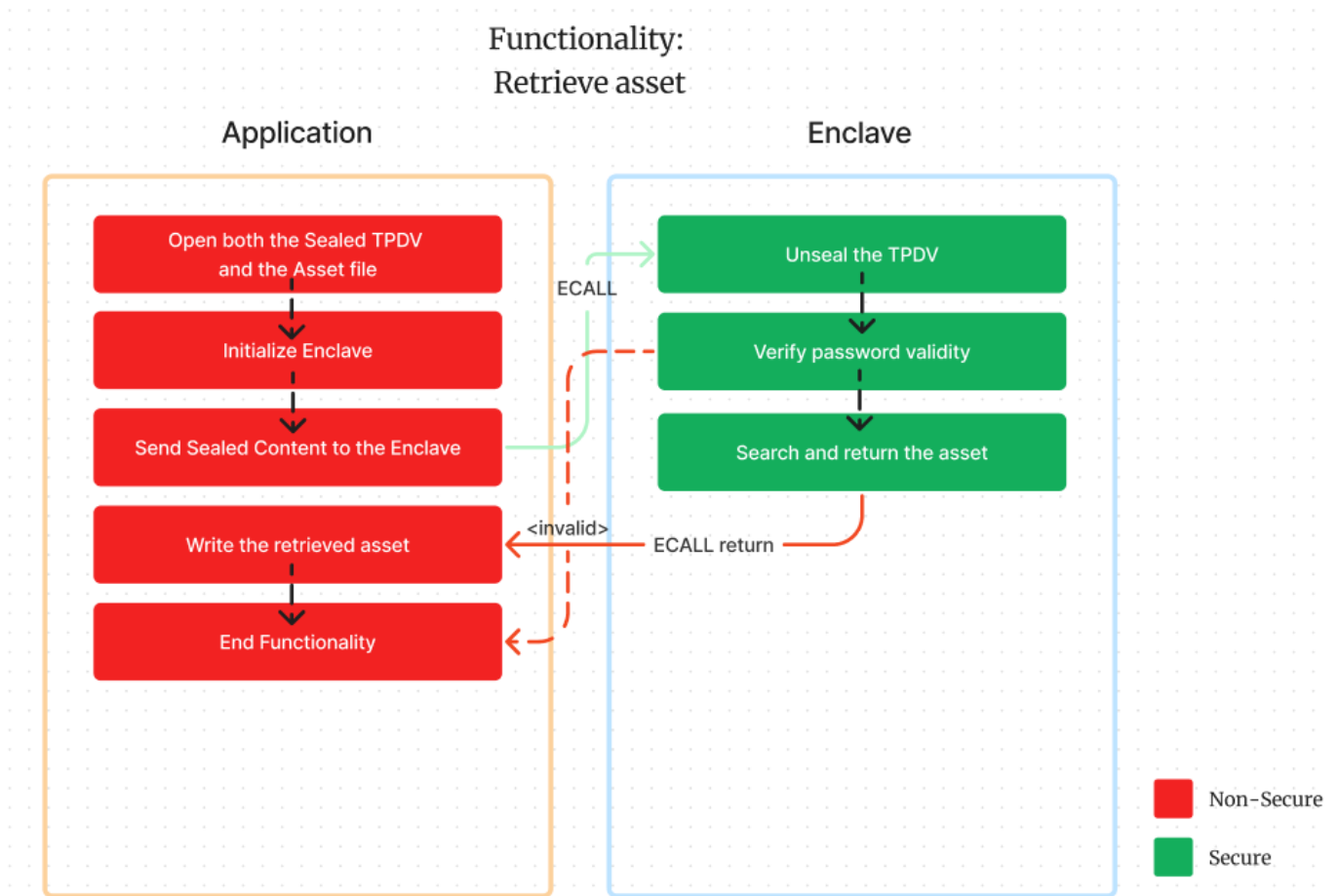


Fig. 5 - Extrair um ficheiro do TPDV

Copiar o TPDV para outro enclave

Esta funcionalidade é responsável por clonar o TPDV para outro enclave.

A função tem o seguinte cabeçalho:

```
int clone_tpdv(const uint8_t *original_tpdv, const uint8_t
*original_password, const uint8_t *cloned_tpdv, const uint8_t
*cloned_password)
```

Para clonar o TPDV de forma segura, foi feita uma troca de mensagens entre os dois enclaves, para estabelecer uma chave secreta partilhada. Para isso, foi utilizada a biblioteca `sgx_dh.h` da Intel SGX, que fornece funções para a troca de chaves Diffie-Hellman.

Key exchange in SGX Enclaves: Using Diffie-Hellman

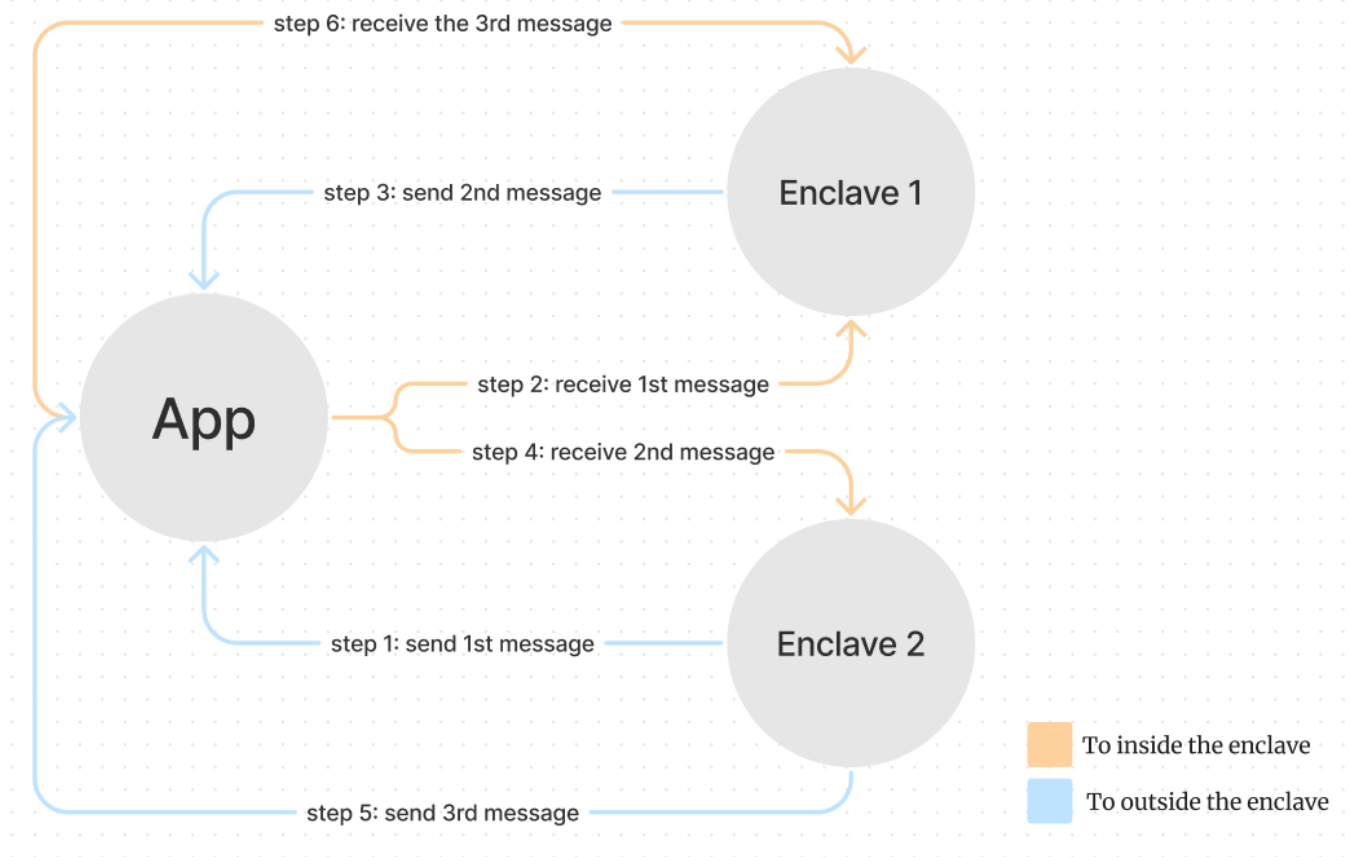


Fig. 6 - Troca de chaves c/SGX enclaves

Na prática:

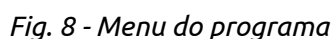
1. `e1_init_session` (Enclave 1): Enclave 1 começa o processo de troca de chave, inicializando uma sessão Diffie-Hellman.
2. `e2_init_session` (Enclave 2): Enclave 2 também inicia sua sessão Diffie-Hellman.
3. `e2_create_message1` (Enclave 2): Enclave 2 gera a primeira mensagem que contém informações para a troca de chave.
4. Enviar Mensagem 1: Enclave 2 envia esta mensagem para Enclave 1.
5. `e1_process_message1` (Enclave 1): Enclave 1 recebe a mensagem e processa-a para gerar sua própria mensagem baseada nela.
6. Enviar Mensagem 2: Enclave 1 envia sua mensagem gerada para Enclave 2.
7. `e2_process_message2` (Enclave 2): Enclave 2 recebe a mensagem de Enclave 1 e processa-a para completar a troca de chave.
8. Enviar Mensagem 3: Enclave 2 envia sua última mensagem para Enclave 1.
9. `e1_process_message3` (Enclave 1): Enclave 1 recebe a mensagem final e finaliza o processo, concordando com a chave secreta compartilhada.

Esta chave é usada para cifrar o conteúdo do TPDV dentro do enclave de origem e decifrar o conteúdo no enclave de destino, onde de seguida o mesmo irá ser selado usando o segundo enclave e guardado num ficheiro.

Este fluxo é ilustrado usando a seguinte figura:



Aqui é apresentadas as verificações que foram feitas ao funcionamento do programa, bem como os resultados obtidos.



Dentro do enclave, o TPDV é manipulado de forma segura, garantindo a integridade dos ficheiros. Caso o TPDV seja manipulado fora do enclave, o programa deteta a adulteração, pois é feita a verificação do hash dos arquivos.

Neste teste foi alterado apenas um byte do ficheiro TPDV.

```
^L<96>"<85>^YôSôÄË; <9b>^_iM<9e>îâBÉ¹%tEv!
7_ "<88>-<87>b }<9d>^H<9b>GŸQÔ^R<94>UV<9c>; <96>=wS^[@é×<89><9f><91>^?<9e>e^M<9e><9e>é^T!î<84> { ±ZÈñèÙÙîQdyY^A^Mly^ [0+â^CÔ<86>_ :ÿ«<84>·m^QÁ^X; ^B``<9
2><93>Yç0^0*«% i) ^Wi""Êsê0je30<9a>»/ÊÁ»±±'c]5^ [Çuú<è0B<9b>û | ^@^VÜ#^@; }8· ^G<8f>; k^A@i^D^X^A^i^A^Lqç^ \r<91>ô+H<84>300Äâ<9c>¿<9f>^Á^T^A, <92><87>_ °0î<9
5><99>| ^R^A"snMB?¶) Ç^S<89>¹~VÜ}ÿ<89>ôDÜ<8f>>jTDî^K*ëîî^Y8Yp%ax!â«; ^Z»îl 8#¹<89>ûû@=^]crµ<82>ý»î0 , <95>¹Ysôp%AF<92>A , äÄó^U<82>^Cu<96>é×±, Ä^F1<83>^B
K<98>d^Dp<9d>^L^J^Sd#T"DE·Y^KkJæ2î<85>N«<8c>^W#E""<86>8<87>Vâ<84>Ä!î!0»ôËpî^W ^Y^ [ó' , <95>J0±<97>^A^DN^FNôÊ90^Noq<8a> , XältÉ^?<80>ùBT<8e> ¹<8a>~² »N^c
^ [^AÿKä<95>³^]6ôg aô%<8b>@i0Äâ^CÔÄÜ^FI¹g¹-<87>wpÿ[<9b>^E+h^ \; îmwE<94>"<81>^Gî' <8d>.òY1}ipÆrÉ+^MÄ5á^PiWiî [îÜ<89>àîi/× _DÇi×@ i@cÿrK<98>^HP<95>e^M<
83>N<80>|@; " i ; <8a>Yfz%<9d>azeî="δ&c+^ \<8c><98>^?Y^Q<92>"^Y^A^2 Ä+ç^KY<8c>D; b^Z^] ^_ , <94><96>b^Qÿ<92>-vó)W ¹Qµ<89>gR^Fi^F@µ^F<99>djHiî0^Y (]iô1îUm<8
e>V#^V<87>$<95>^T|^GyÄvâ8^Ü^ \; ^LîSFNVÜ0ÆL*δ2%Ê^E<96>µX^T<8c>Vé" <99>ñîK<87>z<96>9Ü<95>φ<84>f<91><83>XV^?ôôÄ î"ô00<80>^DKÊ±ko<8f>^G[§+8hZtú^Y^@ [E< , á
0<83>ñçZ0Y^B^K^2^AY^W0«y . <94>P^A]s<97>· " % (7±t-0ôx~î"¥Á^]µgo<8e>^ ^_<96> tr7<8f>B^Â±â<9d>R . T<82>Ç [áY^K<85>â%<9b><93>î^B^Jð0gµÊ<8e>ôrd ^M i=5±e<9
2>^D^XîY^WPE&§gBçÆ^[²b?ô6_J<8e>^Eoûb%N&δñ-ñôLY^V^QHî^0^Q<97>î-ä+îhss2° <9e>2i±Qmôsi>d<89>ÄVki ; 'nçV^E<99>UôdÆnN. % ^Káq| \^F90T , ÚJ<91><83>Aa
±0Êt<9f>XpY^D^Wjs<8b>B^S§Gâ]î#^0· i!Y& <81>êm_æJ(÷p [ ôhY^B<80>AHi<8f>4ü^[îÆ±HVE<9e>c<8f><8b>V^CL^DÄ (^K^_N^L^N^Cbô^N; <9f>ôîî^S<99>t^_Ê3Tdôu$ {<8b>· _
^0^W6n5Sm^MMÜ^Yê
% " ^A(KL \ ^ \ -ÜB?BP%ÿ; Üäb÷rf""jxÊ<8b> ^ac^Lf<9b>I] %sbîY.
vé(%<82>^A·ô~<87>!'@ÜîHê <88>ü <96> , h<96>^ \ ^@E10ô¥Ä<9a>7{D<91>^P±^A^Z<8d>î>^E«^D^N@<87>ô0MPúQ<91>; ê?Ä^X^R ^AXn
2îô^K^_<9b>ó <@4^@^YD00{aÜ<8e><80>îÜcqhu9ñ-h<89>÷CK^Z^@A^] ^A^] üM^À^FmøU^É , 2<88>î , Q^ \<8b>Ä ^KÜ Ük%<99>%â" <83>^AV<8e>XP^ZtNh ^P' ~s^W<92>Kg<99>7<89
>^F<96>To<8a> , î^Fbzñ~D0% ^S^?<9c>±^@<96>î0a<82>z}UÜ^Dó<81><9f>D<99> [I^Aîâ0<98>â-^0x^P^b^AQ±^WJo<89>D<82> , ôK j á^N] ^? · <9f>^D<90>ù0ë0<8a>-ÜQ^Z0 ñ^0<
81>B0Hj î0q° .
~
~
~
"vaults/vaultdosimao" [converted] 7L, 2719B
```

4,1

Bot

Fig. 9 - Resultado da manipulação do TPDV usando o VIM

Devolvendo a seguinte mensagem de erro:

```
Failed to unseal the data
The integrity of the vault has been compromised
Error: Failed to list all assets in the vault.
Press ENTER to continue... [Enter]
```

Fig. 10 - Resultado da manipulação do TPDV

Esta verificação provém do `check_nonce_integrity`.

Listagem de ficheiros

Todos os ficheiros presentes no TPDV podem ser listados, referindo o nome e o tamanho dos ficheiros.

Esta funcionalidade requer autenticação, sendo necessário fornecer a credencial do TPDV.

```
|-----|
| 1. Create a new vault |
| 2. Add asset to vault |
| 3. List all assets in vault |
| 4. Retrieve asset from vault |
| 5. Check integrity of an asset |
| 6. Change password |
| 7. Clone vault |
| 8. Exit |
|-----|

Enter your choice: 3
Enter the vault filename: vaultdosimao
Enter the vault password: simao1 [Enter]
```

Fig. 11 - Listagem de ficheiros

Obtendo o seguinte resultado:

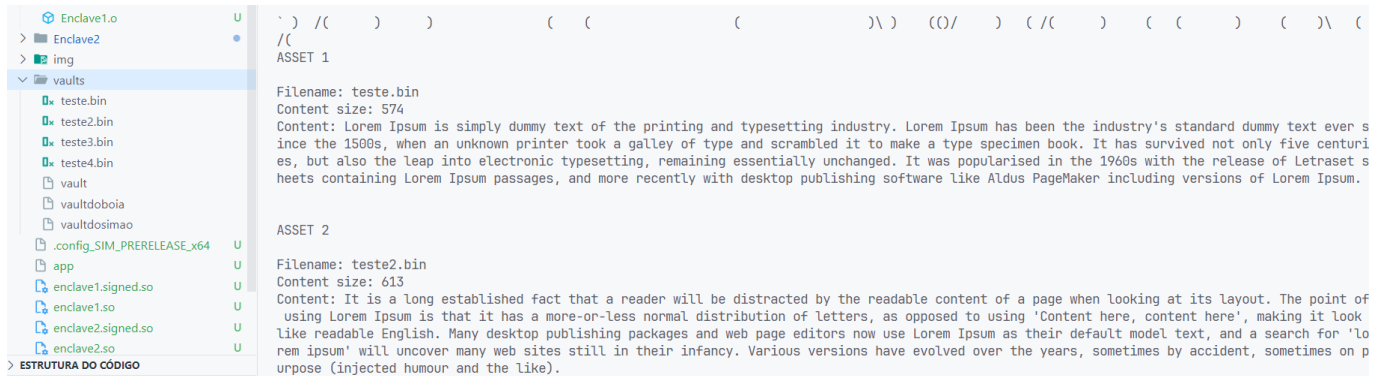


Fig. 12 - Listagem de ficheiros

Alteração das credênciais do TPDV

As credênciais do TPDV podem ser alteradas, sendo necessário fornecer a credencial atual e a nova credencial.

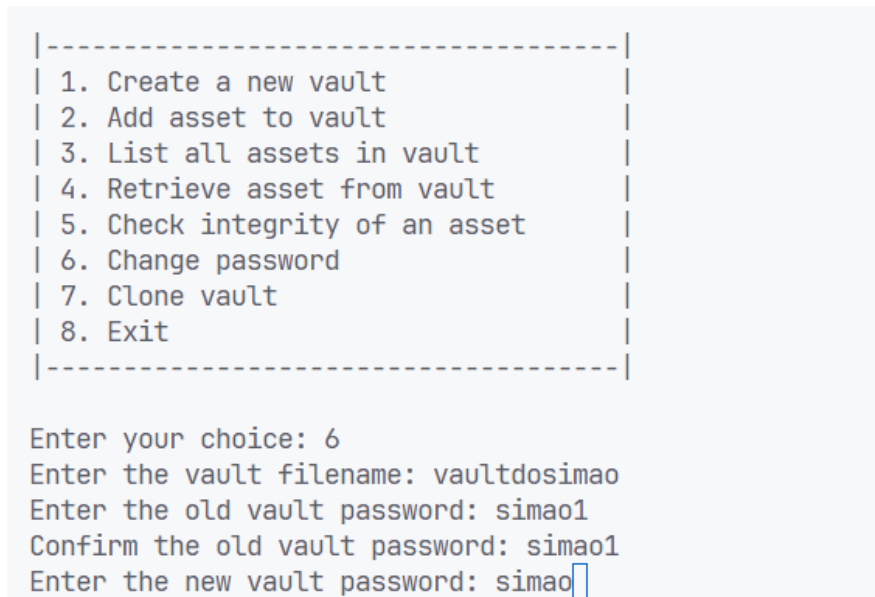


Fig. 13 - Alteração das credênciais do TPDV

Dando a seguinte mensagem de sucesso:

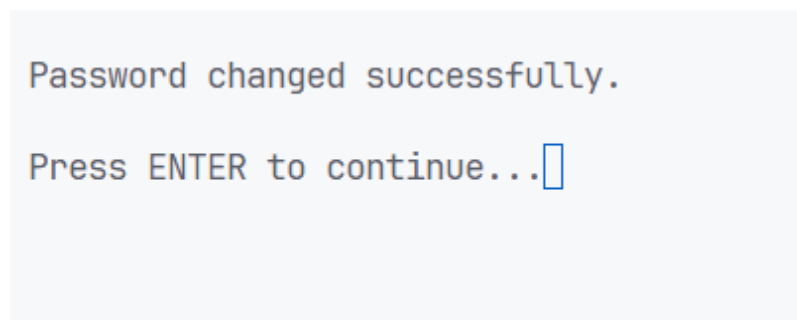


Fig. 14 - Mensagem de sucesso na alteração das credênciais do TPDV

Caso o utilizador insira a credencial antiga, o programa devolve a seguinte mensagem de erro:

```
Error: Failed to list all assets in the vault.  
Press ENTER to continue...
```

Fig. 15 - Mensagem de erro na alteração das credenciais do TPDV

Validação da integridade dos ficheiros

Qualquer arquivo presente no TPDV pode ser validado, verificando se o hash do arquivo corresponde ao hash guardado no TPDV.



Fig. 16 - Validação da integridade dos ficheiros

Caso seja adulterado parte do ficheiro, o programa irá detetar a adulteração.

```
1 | Lorm Ipsum is simply dummy t
```

Fig. 17 - Adulteração de um arquivo no TPDV

E será devolvida a seguinte mensagem de erro:



Fig. 18 - Mensagem de erro na validação da integridade dos ficheiros

Clonagem do TPDV

Para clonar o TPDV, é necessário fornecer as credenciais do TPDV de origem e de destino.

```
Enter the cloned vault filename: clonedosimao  
Enter the cloned vault password: clone
```

Fig. 19 - Autenticação do TPDV a Clonar

```
Enter the vault filename: vaultdosimao
Enter the vault password: simao
```

Fig. 20 - Clonagem do TPDV

Dando a seguinte mensagem de sucesso:

```
Vault cloned successfully.

Press ENTER to continue...
```

Fig. 21 - Mensagem de sucesso na clonagem do TPDV

Criando um novo ficheiro TPDV com o mesmo conteúdo do ficheiro original. No momento, o utilizador pode apenas listar os ficheiros presentes no TPDV clonado.

```
clonedosimao  U
├─ teste.bin
├─ teste2.bin
├─ teste3.bin
├─ teste4.bin
├─ vault
├─ vault_teste  U
├─ vaultdoboia
├─ vaultdosimao
└─ .config_SIM_PRERELEASE_x64  U

2. Add asset to vault
3. List all assets in vault
4. Retrieve asset from vault
5. Check integrity of an asset
6. Change password
7. Clone vault
8. Exit
-----
Enter your choice: 3
Enter the vault filename: clonedosimao
Enter the vault password: clone
```

Fig. 22 - Autenticação do Clone do TPDV

```
ASSET 1
Filename: teste.bin
Content size: 574
Content: Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

ASSET 2
Filename: teste2.bin
Content size: 613
Content: It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

ASSET 3
Filename: teste3.bin
Content size: 41
Content: vinho verde com azeitonas e pão de milho

ASSET 4
Filename: teste4.bin
Content size: 34
Content: este é mesmo de verdade juro joca

Press ENTER to continue...
```

Fig. 23 - Listagem de ficheiros do TPDV clonado

Documentação

A documentação do código foi feita com o Doxygen. Para gerar a documentação, basta executar o seguinte comando:

```
$ doxygen Doxyfile
```

Onde a documentação será gerada na pasta `docs/`.

[!NOTE] Não foi adicionada, pois o Doxygen não se encontra presente na máquina de desenvolvimento.

Execução

Para compilar o programa, basta executar os seguintes comandos:

```
$ make  
$ ./app
```

[!IMPORTANT] Deve ser gerado um novo par de chaves RSA para o enclave, para isso basta executar o script `new_private_key.bash`.

Conclusão

Em conclusão, conseguimos implementar com sucesso o TPDV utilizando Intel SGX enclaves, garantindo assim um ambiente seguro para armazenamento e manipulação de arquivos. Ao longo do desenvolvimento, enfrentamos desafios significativos, especialmente na definição das funcionalidades críticas a serem executadas dentro dos enclaves. No entanto, superamos esses obstáculos e conseguimos finalizar todos os objetivos estabelecidos.

Durante os testes realizados, o TPDV demonstrou a sua eficácia ao detectar falhas de integridade, listar e extrair arquivos, realizar alterações de credenciais e validar a integridade dos arquivos armazenados, tudo sem comprometer a segurança dos dados. Além disso, foram feitos esforços para melhorar o desempenho da aplicação.

Esta experiência foi extremamente enriquecedora, proporcionando conhecimentos valiosos sobre o desenvolvimento de *software* utilizando segurança a nível de *hardware* e sobre o desenvolvimento de aplicações com foco na integridade.

O êxito na implementação do TPDV não apenas ressalta a importância da integridade dos dados em ambientes digitais, mas também destaca o potencial das tecnologias como Intel SGX enclaves na proteção desses dados.

Referências

[Overview of Intel SGX Enclave - by Intel](#)

[Repositório Github do Intel SGX](#)

[Repositório Exemplo: HelloEnclave](#)