

Trabalho de Criptografia Pós Quântica: Troca de chaves com TLS.

Rafael Begnini de Castilhos (20205642)

28 de julho de 2022

Resumo

O presente trabalho possui como objetivo entender e experimentar algoritmos, protocolos e criptografia pós quântica na troca de chaves, utilizando TLS.

Sumário

1	Introdução	2
2	Teoria	2
2.1	Encapsulamento	3
2.2	Construção da troca	3
3	Experimento	4
3.1	Instalação e construção	5
3.2	Execução	5
4	Análise e conclusão	8

1 Introdução

Desde que se descobriu o potencial de criptoanálise do computador quântico, especialmente com o algoritmo de Shor, criou-se a idéia de que a criptografia moderna se tornaria inútil assim que esse tipo de computador começasse a ser desenvolvido em larga escala. Com isso, todas as mensagens trocadas com o uso de algoritmos criptográficos atuais seriam facilmente interceptáveis.

Desse modo, a troca de chaves que é um método de criptografia para trocas de chaves de maneira segura em canal público. Entretanto, deixaria de ser pois estaria utilizando um protocolo que poderia ser descriptografado por um computador quântico.

Em resposta a isso, surgiu a criptografia pós-quântica, ramo da criptografia que estuda classes de algoritmos criptográficos resistentes à criptoanálise quântica. Como exemplos de classes de algoritmos, podemos mencionar os baseados em hash, os baseados em látice, os baseados em códigos, entre outros. Estes algoritmos demorariam um tempo exponencial para serem quebrados, mesmo em computadores quânticos.

O TLS (Transport Layer Security) assim como o seu antecessor SSL (Secure Sockets Layer) é um protocolo de segurança projetado para fornecer segurança nas comunicações sobre uma rede de computadores. O protocolo TLS, que atualmente está na versão 1.2, visa principalmente fornecer privacidade e integridade de dados entre dois ou mais aplicativos de computador que se comunicam. Entretanto, com o surgimento da computação quântica, uma nova ameaça aos problemas de segurança surge.

Contextualizado os itens acima, em sequência será apresentado e experimentado a troca de chaves utilizando o protocolo TLS, será utilizado o projeto de código aberto Open Quantum Safe (OQS) que fornece algoritmos finalistas no NIST empacotados na biblioteca liboqs.

2 Teoria

A troca de chave híbrida refere-se ao uso de vários algoritmos de troca de chave simultaneamente e à combinação do resultado com o objetivo de fornecer segurança, mesmo que todos os algoritmos do componente, exceto um, estejam quebrados. É motivado pela transição para a criptografia pós-quântica. Denominado como Transport Layer Security (TLS) versão 1.3.

Troca de chave "híbrida", significa o uso de dois, ou mais algoritmos de

troca de chave com base em diferentes suposições criptográficas, por exemplo, um algoritmo tradicional e um algoritmo de última geração. O principal objetivo é facilitar o estabelecimento de um segredo compartilhado que permaneça seguro enquanto um dos mecanismos de troca de chave permanecer intacto.

2.1 Encapsulamento

Os algoritmos de troca de chaves são formulados como mecanismos de encapsulamento de chaves, sendo eles:

- KeyGen: Geração probabilística, chave pública pk e chave secreta sk .
- Encaps: Encapsulamento probabilístico, recebe como entrada uma chave pk , e gera um texto cifrado ct e segredo compartilhado ss .
- Decaps: Desencapsulamento que recebe como entrada uma chave sk e texto cifrado ct , gerando um texto compartilhado ss .

2.2 Construção da troca

- Negociação: Para cada combinação particular de algoritmos em um troca de será representada por um grupo nomeado. Cada valor que representa uma troca de chave corresponderá a um par ordenado de dois algoritmos.
- Transmissão: As mensagens serão concatenadas e transmitidas como um único valor.
- Cálculo de segredo compartilhado: Os dois segredos compartilhados são concatenados. Inserindo no cronograma de chave no local do segredo compartilhado (EC)DHE.

A biblioteca liboqs fornece os principais mecanismos para encapsulamento de chaves, conforme citado no item 2.1, sendo denotadas como:

- *OQS_KEM_keypar*: Gera um par de chaves pública/privada para troca de chaves.
- *OQS_KEM_encaps*: Gera uma chave pública para a outra parte e um segredo compartilhado para o requisitante.

- *OQS_KEM_decaps*: Gera um valor secreto compartilhado usando a chave pública da outra parte e sua própria chave privada.

Aplicando essas funcionalidades da biblioteca, é possível encontrar o seguinte fluxo:

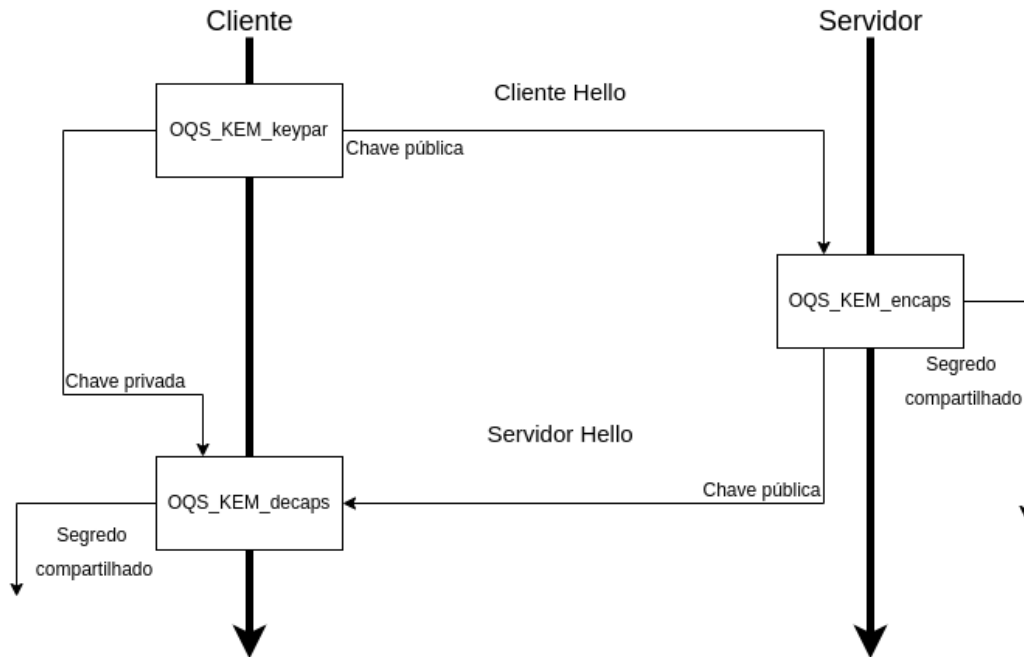


Figura 1: Fluxo de troca de chaves com TLS 1.3.

3 Experimento

O experimento será executado em um notebook com sistema operacional Linux Ubuntu, todos os procedimentos citados a seguir também poderão ser executados em outros sistemas operacionais, porém haverá diferença em alguns comandos que são especiais do Linux. Preambularmente, iremos precisar realizar o download do programa Wireshark que é um programa que analisa o tráfego da rede e organiza por protocolos. Também iremos precisar baixar a biblioteca liboqs e a biblioteca wolfSSL.

3.1 Instalação e construção

Para baixar o Wireshark é necessário executar:

```
sudo apt-get install wireshark
```

Após instalado, deve ser iniciado, executando o comando:

```
sudo wireshark
```

Para realizar *download* e construção do liboqs, execute:

```
wget
  https://github.com/open-quantum-safe/liboqs/archive/refs/tags/0.7.0.tar.gz
tar xf 0.7.0.tar.gz
cd liboqs-0.7.0
mkdir build
cd build
cmake -DOQS_USE_OPENSSL=0 ..
make all
sudo make install
```

Para realizar *download* e construção do wolfSSL, execute:

```
git clone https://github.com/wolfssl/wolfssl
cd wolfssl
./autogen.sh
./configure --with-liboqs
make all
sudo make install
```

3.2 Execução

Em seguida, vamos estabelecer uma comunicação TLS entre um exemplo de servidor e um exemplo de cliente. Para isso, é necessário abrir duas janelas no terminal, uma para o servidor, outra para o cliente. Nesse momento, certifique-se que o Wireshark está aberto e clique no botão para iniciar captura de pacotes.

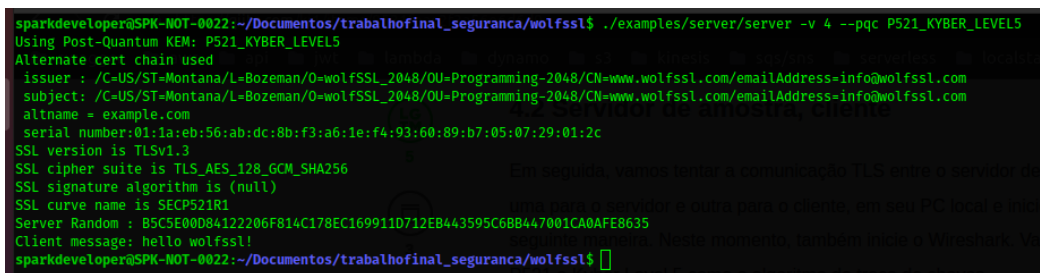
Agora é necessário especificar o início do servidor para troca de chaves híbridas ECC-P521 e Kyber Level 5, em uma janela do terminal:

```
./examples/server/server -v 4 --pqc P521_KYBER_LEVEL5
```

Na outra janela do terminal, execute a conexão do cliente:

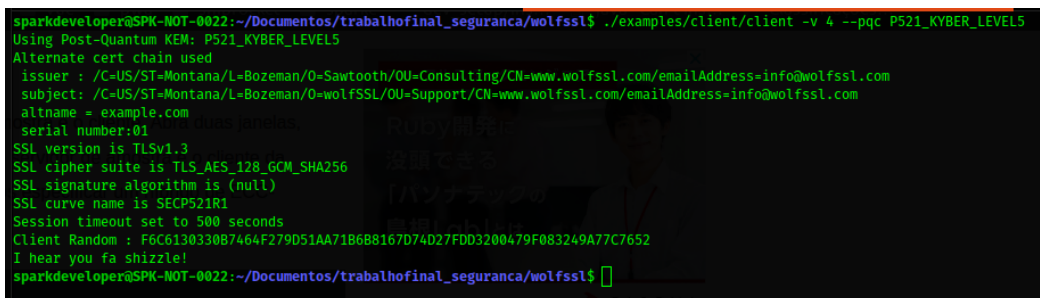
```
./examples/client/client -v 4 --pqc P521_KYBER_LEVEL5
```

O servidor e o cliente realizam uma conexão TLS e uma comunicação com mensagem de ida e volta. Cada janela exibe informações de conexão TLS 1.3 com as seguintes mensagens:



```
sparkdeveloper@SPK-NOT-0022:~/Documentos/trabalhofinal_seguranca/wolfssl$ ./examples/server/server -v 4 --pqc P521_KYBER_LEVEL5
Using Post-Quantum KEM: P521_KYBER_LEVEL5
Alternate cert chain used
issuer : /C=US/ST=Montana/L=Bozeman/O=wolfSSL_2048/OU=Programming-2048/CN=www.wolfssl.com/emailAddress=info@wolfssl.com
subject: /C=US/ST=Montana/L=Bozeman/O=wolfSSL_2048/OU=Programming-2048/CN=www.wolfssl.com/emailAddress=info@wolfssl.com
altname = example.com
serial number:01:1a:eb:56:ab:dc:8b:f3:a6:1e:f4:93:60:89:b7:05:07:29:01:2c
SSL version is TLSv1.3
SSL cipher suite is TLS_AES_128_GCM_SHA256
SSL signature algorithm is (null)
SSL curve name is SECP521R1
Server Random : B5C5E00D84122206F814C178EC169911D712EB443595C6BB447001CA0AFE8635
Client message: hello wolfssl!
sparkdeveloper@SPK-NOT-0022:~/Documentos/trabalhofinal_seguranca/wolfssl$
```

Figura 2: Mensagem do servidor.



```
sparkdeveloper@SPK-NOT-0022:~/Documentos/trabalhofinal_seguranca/wolfssl$ ./examples/client/client -v 4 --pqc P521_KYBER_LEVEL5
Using Post-Quantum KEM: P521_KYBER_LEVEL5
Alternate cert chain used
issuer : /C=US/ST=Montana/L=Bozeman/O=Sawtooth/OU=Consulting/CN=www.wolfssl.com/emailAddress=info@wolfssl.com
subject: /C=US/ST=Montana/L=Bozeman/O=wolfSSL/OU=Support/CN=www.wolfssl.com/emailAddress=info@wolfssl.com
altname = example.com
serial number:01:1a:eb:56:ab:dc:8b:f3:a6:1e:f4:93:60:89:b7:05:07:29:01:2c
SSL version is TLSv1.3
SSL cipher suite is TLS_AES_128_GCM_SHA256
SSL signature algorithm is (null)
SSL curve name is SECP521R1
Session timeout set to 500 seconds
Client Random : F6C6130330B7464F279D51AA71B6B8167D74D27FDD3200479F083249A77C7652
I hear you fa shizzle!
sparkdeveloper@SPK-NOT-0022:~/Documentos/trabalhofinal_seguranca/wolfssl$
```

Figura 3: Mensagem do cliente.

Após realizar a conexão, é possível verificar os pacotes no Wireshark, filtrando apenas os pacotes que utilizam TLS, de acordo com a figura abaixo:

58	21.966405187	192.168.0.14	18.230.171.141	TLSv1.3	612 Client Hello
60	21.994910417	18.230.171.141	192.168.0.14	TLSv1.3	278 Server Hello, Change Cipher Spec, Application Data
62	21.995733797	192.168.0.14	18.230.171.141	TLSv1.3	130 Change Cipher Spec, Application Data
63	21.996087691	192.168.0.14	18.230.171.141	TLSv1.3	158 Application Data
64	21.996551800	192.168.0.14	18.230.171.141	TLSv1.3	744 Application Data
68	22.025580637	18.230.171.141	192.168.0.14	TLSv1.3	593 Application Data, Application Data
70	22.025988460	192.168.0.14	18.230.171.141	TLSv1.3	97 Application Data
72	22.221276218	18.230.171.141	192.168.0.14	TLSv1.3	634 Application Data

Figura 4: Wireshark com filtro para pacotes TLSv1.3.

E também constatar o acontecimento do *handshake*, Client Hello e Server Hello, na imagem abaixo é demonstrado com detalhes as propriedades do TLS no pacote Server Hello:

▼ Transport Layer Security		
▼ TLSv1.3 Record Layer: Handshake Protocol: Server Hello		
Content Type: Handshake (22)		
Version: TLS 1.2 (0x0303)		
Length: 128		
▼ Handshake Protocol: Server Hello		
Handshake Type: Server Hello (2)		
Length: 124		
Version: TLS 1.2 (0x0303)		
Random: 205fd2cf485b4232431d392ae29522a8b821cc573b832f60...		
Session ID Length: 32		
Session ID: f4de1f08a310808ede6c37909df34679923e76f3bdc733b6...		
Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)		
Compression Method: null (0)		
Extensions Length: 52		
▶ Extension: pre_shared_key (len=2)		
▶ Extension: key_share (len=36)		
▶ Extension: supported_versions (len=2)		
▶ TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec		
▶ TLSv1.3 Record Layer: Application Data Protocol: http-over-tls		

0000	ac 67 5d 4c 2a b1 5c 76	95 96 68 4b 08 00 45 00	·g]L*·\v ··hK·E·
0010	01 08 16 55 40 00 31 06	b3 71 12 e6 ab 8d c0 a8	···U@·1· ·q·····
0020	00 0e 01 bb c8 b6 43 8d	c4 e2 e8 ac 1a 55 80 18	·····C· ····U··
0030	00 08 94 67 00 00 01 01	08 0a af fb af 8f 46 aa	···g···· ·····F·
0040	56 51 16 03 03 00 80 02	00 00 7c 03 03 20 5f d2	VQ····· ·· ·· ·
0050	cf 48 5b 42 32 43 1d 39	2a e2 95 22 a8 b8 21 cc	·H[B2C·9 *··"··!·
0060	57 3b 83 2f 60 1e 3a 59	b9 3c 44 4f 6d 20 f4 de	W;·/·`·:Y ·<D0m ··
0070	1f 08 a3 10 80 8e de 6c	37 90 9d f3 46 79 92 3e	·····1 7···Fy·>
0080	76 f3 bd c7 33 b6 70 d1	e6 23 38 8b 89 75 13 01	v···3·p· ·#8··u·
0090	00 00 34 00 29 00 02 00	00 00 33 00 24 00 1d 00	··4·)···· ··3·\$··
00a0	20 37 0b 1c 32 27 a2 a2	43 0d 4b e9 b9 0c b6 1a	7··2'··· C·K·····
00b0	f9 4f 07 59 c5 be f0 79	d0 74 af f2 df 26 6b ef	·0·Y···y ·t···&k·
00c0	2e 00 2b 00 02 03 04 14	03 03 00 01 01 17 03 03	··+····· ······
00d0	00 44 74 0b 72 ed 39 83	ce d0 21 5a f4 ae 87 d8	·Dt·r·9· ··!Z····
00e0	35 82 11 f3 35 60 72 79	68 9e f9 50 10 44 65 96	5···5`ry h··P·De·
00f0	24 90 b6 b9 af e0 d8 3a	f5 11 c0 56 69 49 03 a3	\$·····: ···ViI··
0100	17 7d b5 ec 7d b3 c9 5a	7c 4c 48 d8 58 d5 43 3b	·}··}··Z LH·X·C;
0110	f3 aa a4 30 39 8b		···09·

Figura 5: Detalhes do pacote de Server Hello.

4 Análise e conclusão

Portanto, após realizar testes e executar o programa de *benchmark* disponibilizado pela wolfSSL por meio do comando:

```
./wolfcrypt/benchmark/benchmark
```

Se torna possível constatar que os tempos de processamento desses algoritmos estão entrando em um intervalo comparável aos algoritmos atuais, mas ainda não são iguais ou inferiores.

É possível concluir que os segredos compartilhados calculados na troca de chave híbrida devem ser calculados de uma maneira que atinja a propriedade "híbrida": o segredo resultante é seguro desde que pelo menos um dos algoritmos de troca de chave componente esteja intacto.

Realizando esse trabalho foi possível analisar na prática os tópicos apresentado em sala de aula sobre números criptografia pós-quântica, fazendo referência as possíveis abordagens na área da segurança da computação. Em suma, o trabalho cumpriu seus requisitos pedagógicos e didáticos para a formação de um profissional da ciência da computação.

Referências

- [1] D. Stebila. Troca de chave híbrida em tls 1.3. <https://tools.ietf.org/id/draft-stebila-tls-hybrid-design-03.html>", year=2020, note =.
- [2] K. Wolf. Experimente a criptografia pós-quântica com tls (parte 1: troca de chaves). <https://qiita.com/kj1/items/8531feac2f3a56e6d6d9>", year=2022, note =.