Информационная безопасность и компьютерные сети

Практическая работа №1

«*Сертификаты*»

Гетьман Александр Игоревич
Маркин Юрий Витальевич
Обыденков Дмитрий Олегович
Пономаренко Роман Евгеньевич

Практические задания курса

	P1.1	*P1.2*	*P1.3*	*P2*	*P3.1*	*P3.2*	*Бонус*
Максимальный балл	*1*	*1*	*2*	*2*	*1*	*2*	*1*
Дата выдачи	*19.09.2022*	*19.09.2022*	*19.09.2022*	*17.10.2022**	*14.11.2022**	*14.11.2022**	-
Дата окончания приема	*03.10.2022*	*10.10.2022*	*17.10.2022*	*14.11.2022**	*28.11.2022**	05.12.2022*	-
Сложность	☼	☼ ☼	☼ ☼ ☼	☼ ☼ ☼	☼ ☼ ☼ ☼	☼ ☼ ☼ ☼ ☼	-

** не является публичной офертой*

$$Бонус = \ if \ \ P_{1.1} + P_{1.2} + P_{1.3} + P_2 + P_{3.1} + P_{3.2} = 9 \ \ then \ \ 1 \ \ else \ \ 0$$

Соглашение о наименовании

- Меня зовут:
  - *Иванов Петр Сергеевич*
- Мой вуз:
  - *МГУ*
  - *МФТИ*
  - *ВШЭ*
- Моя группа:
  - *999*
  - *М99-999х*
  - *мСП99*

- Подстановки:
  - `<фамилияио> = ivanovps`
  - `<группа> = 999|m99_999x|msp99`
  - `<вуз> = msu|mipt|hse`
- Тема письма с решением:
  - *`msu-999-p1_{1,2,3}`*
  - *`mipt-m99_999x-p1_{1,2,3}`*
  - *`hse-msp99-p1_{1,2,3}`*
- Название архива:
  - `ivanovps-999-p1_{1,2,3}.zip`
  - `ivanovps-m99_999x-p1_{1,2,3}.zip`
  - `ivanovps-msp99-p1_{1,2,3}.zip`
- Посылки с иным форматом будут проигнорированы
  - Дефис является разделителем, используйте исключительно латинские символы, цифры и нижнее подчеркивание

Правила отправки решений

Not before	Not after	Максимальное количество посылок* в день**
	16.10.2022 23:59	1
17.10.2022 00:00	17.10.2022 23:59	2
18.10.2022 00:00		0

1. Посылки с некорректным форматом не будут проверяться (с уведомлением);
2. Если на посылку нет реакции более 2 рабочих дней, то письмо могло попасть в спам — свяжитесь с преподавателем (@dmt_obd);
3. Не объединяйте отправку P1.1/P1.2/P1.3 в один тред;
4. (*) Для P1.1/P1.2/P1.3 раздельные счетчики посылок;
5. (**) Счетчик посылок обнуляется в 00:00.

Рекомендуемое ПО

- Операционная система *Linux*
- Криптографическая библиотека и набор утилит *OpenSSL*

Альтернативные реализации:

- *LibreSSL* (OpenBSD)
- *BoringSSL* (Google)
- *mbedTLS* (ARM)
- *JSSE* (Oracle)

```
$ openssl version -v -b -p

OpenSSL 1.1.1f  31 Mar 2020
built on: Mon Jul  4 11:24:28
2022 UTC
platform: debian-amd64
```

Использование OpenSSL [1/2]

```
$ openssl list -commands
asn1parse         ca                ciphers           cms
crl               crl2pkcs7         dgst              dhparam
dsa               dsaparam          ec                ecparam
enc               engine            errstr            gendsa
genpkey           genrsa            help              list
nseq              ocsp              passwd            pkcs12
pkcs7             pkcs8             pkey              pkeyparam
pkeyutl           prime             rand              rehash
req               rsa               rsautl            s_client
s_server          s_time            sess_id           smime
speed             spkac             srp               storeutl
ts                verify            version           x509
```

Использование OpenSSL [2/2]

```
$ man openssl # не забывайте про man!
$ man openssl-
openssl-asn1parse  openssl-enc        openssl-pkey       openssl-spkac
openssl-ca         openssl-engine     openssl-pkeyparam  openssl-srp
openssl-ciphers    openssl-errstr     openssl-pkeyutl    openssl-s_server
openssl-cms        openssl-gendsa     openssl-prime      openssl-s_time
openssl-c_rehash   openssl-genpkey    openssl-rand       openssl-storeutl
openssl-crl        openssl-genrsa     openssl-rehash     openssl-ts
openssl-crl2pkcs7  openssl-list       openssl-req        openssl-tsget
openssl-dgst       openssl-nseq       openssl-rsa        openssl-verify
openssl-dhparam    openssl-ocsp       openssl-rsautl     openssl-version
openssl-dsa        openssl-passwd     openssl-s_client   openssl-x509
openssl-dsaparam   openssl-pkcs12     openssl-sess_id
openssl-ec         openssl-pkcs7      openssl-smime
openssl-ecparam    openssl-pkcs8      openssl-speed
```

Часть 1
X.509

Сертификат X.509

Структура сертификата X.509:

- Версия
- Серийный номер
- Идентификатор алгоритма подписи
- Имя издателя
- Период действия:
  - Не ранее
  - Не позднее
- Имя субъекта
- Информация об открытом ключе субъекта:
  - Алгоритм открытого ключа
  - Открытый ключ субъекта
- Уникальный идентификатор издателя (обязательно только для v2 и v3)
- Уникальный идентификатор субъекта (обязательно только для v2 и v3)
- Дополнения (для v2 и v3)
  - Возможные дополнительные детали
- Алгоритм подписи сертификата (обязательно только для v3)
- Подпись сертификата (обязательно для всех версий)

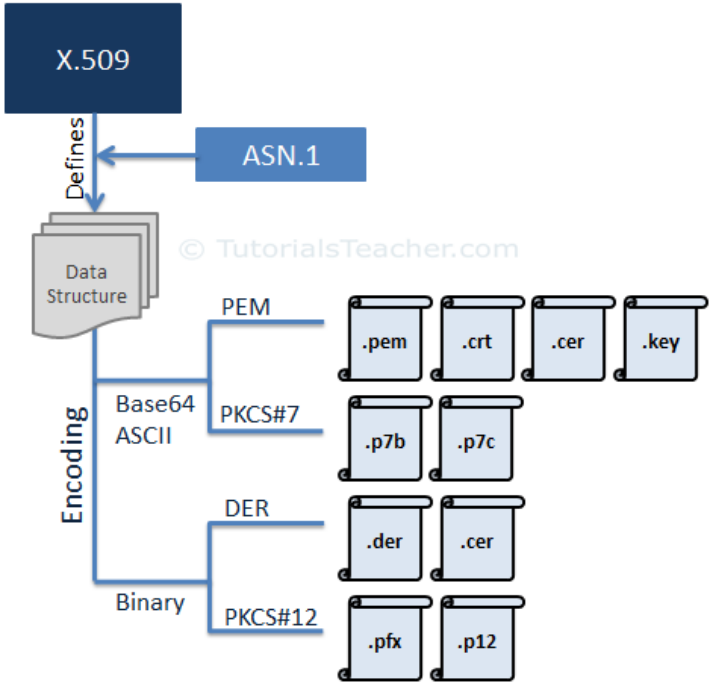
```
$ openssl x509 -text -noout -in test.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4109 (0x100d)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = RU, ST = Moscow, O = Obydenkov Dmitry, OU =
Intermediate, CN = Obydenkov Intermediate CA, emailAddress =
*******@ispras.ru
        Validity
            Not Before: Mar 19 11:57:50 2019 GMT
            Not After : Mar 18 11:57:50 2020 GMT
        Subject: C = RU, ST = Moscow, O = Obydenkov Dmitry, OU =
Obydenkov Client, CN = client.obydenkov.ru, emailAddress =
obydenkov@ispras.ru
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:ca:d6:50:5e:c6:bf:d1:69:51:1c:0b:9b:4d:ea:
                    e7:fa:c3:24:69:95:53:f1:60:d7:52:84:c8:eb:c7:
                    0b:56:89:62:29:e4:35:08:12:ff:5f:06:c6:d9:81:
...
```

Форматы сертификатов

- Текстовые
  - ***PEM*** *(Privacy Enhanced Mail)* — используется преимущественно открытым ПО
  - ***PKCS#7*** *(Public-Key Cryptography Standards)* — используется Java, поддерживается Windows
- Бинарные
  - ***DER*** *(Distinguished Encoding Rules)* — используется преимущественно в Windows
  - ***PKCS#12*** — изначально разрабатывался Microsoft как улучшенный PEM, оформлен как RFC

Конвертация формата:

```
$ openssl x509 -outform der \
   -in test.crt \
   -out test.der
```



Форматы сертификатов — PEM

```
$ cat test.crt
-----BEGIN CERTIFICATE-----
MIIFrjCCA5agAwIBAgICEA0wDQYJKoZIhvcNAQE
MQ8wDQYDVQQIDAZNb3Njb3cxGTAXBgNVBAoMEE9
BgNVBAsMDEludGVybWVkaWF0ZTEiMCAGA1UEAww
aWF0ZSBDQTEiMCAGCSqGSIb3DQEJARYTb2J5ZGV
...
oYFHUFqEZbjJsgFFL+gOiJWDxxhqQyYBKlYXR+B
1unVxeZ8M51C66AuMFb596fKFSynGy8js9MnddE
ZDgFvHHkpo5f6wBDlo5W6bzB
-----END CERTIFICATE-----
```

```
$ cat test.key
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAytZQXsa/0WlRHAubTern+sM
CBL/XwbG2YHMHD2KXltaxXxsVw9i0L0ePZxBkQt
rFnX7TUPdytbFE+uSx2dCtxDP4uYTmLC/Rm7V0R
OTqsDItDdj03c135mV7XfIoB2/siGkDTQ5XETrC
...
RLki/isCgYEArlfvlkp3P5Ip0U9wnMl0sk55JzA
dUGOgJUydshy5y4jdqkzy6w0tYEVJaWRwFVnV+6
9e9rkakVk3w84bRzrMHTDlHp1Z9blB+W4rZ5xoD
-----END RSA PRIVATE KEY-----
```

Форматы сертификатов - DER

```
$ hexdump -C test.der
00000000  30 82 05 ae 30 82 03 96  a0 03 02 01 02 02 02 10  |0...0...........|
00000010  0d 30 0d 06 09 2a 86 48  86 f7 0d 01 01 0b 05 00  |.0...*.H........|
00000020  30 81 98 31 0b 30 09 06  03 55 04 06 13 02 52 55  |0..1.0...U....RU|
00000030  31 0f 30 0d 06 03 55 04  08 0c 06 4d 6f 73 63 6f  |1.0...U....Mosco|
00000040  77 31 19 30 17 06 03 55  04 0a 0c 10 4f 62 79 64  |w1.0...U....Obyd|
00000050  65 6e 6b 6f 76 20 44 6d  69 74 72 79 31 15 30 13  |enkov Dmitry1.0.|
00000060  06 03 55 04 0b 0c 0c 49  6e 74 65 72 6d 65 64 69  |..U....Intermedi|
00000070  61 74 65 31 22 30 20 06  03 55 04 03 0c 19 4f 62  |ate1"0 ..U....Ob|
00000080  79 64 65 6e 6b 6f 76 20  49 6e 74 65 72 6d 65 64  |ydenkov Intermed|
00000090  69 61 74 65 20 43 41 31  22 30 20 06 09 2a 86 48  |iate CA1"0 ..*.H|
…
```

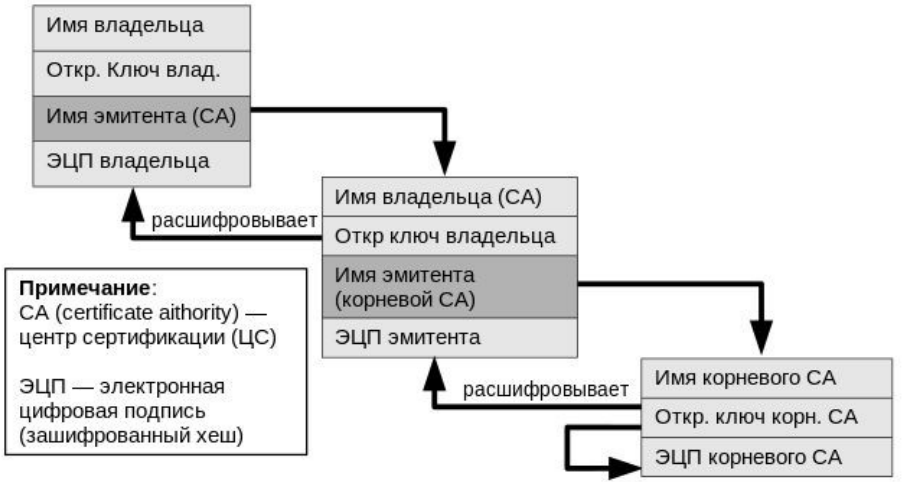
Цепочка сертификатов

Выпуск самоподписанного сертификата (CA):

- Сгенерировать ключевую пару;
- Сгенерировать сертификат с заданными атрибутами, подписанный сгенерированным ранее закрытым ключом.

Выпуск подписанных CA сертификатов:

- Сгенерировать ключевую пару;
- Сгенерировать запрос на сертификат к удостоверяющему центру, содержащего все атрибуты сертификата;
- Удостоверяющий центр выпускает сертификат, используя запрос сертификата как источник данных атрибутов, а также подписывает его при помощи своего закрытого ключа.



Выпуск самоподписного сертификата [1/2]

1.a Генерация ключевой пары RSA 2048 бит:

```
$ openssl genrsa -out testCA.key 2048
Generating RSA private key, 2048 bit long
modulus (2 primes)
.....................................+++++
.................................+++++
e is 65537 (0x010001)
```

1.b Генерация ключевой пары RSA 4096 бит, зашифрованной AES 256 бит:

```
$ openssl genrsa -aes256 -out testCA.key 4096
Generating RSA private key, 4096 bit long
modulus (2 primes)
.................................+++
.....................................+++
e is 65537 (0x010001)
Enter pass phrase for test.key:
Verifying - Enter pass phrase for test.key:
```

2. Выпуск сертификата:

```
$ openssl req -x509 -new -key testCA.key
-days 3650 -out testCA.crt
Enter pass phrase for test.key:
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-
State]:Moscow
Locality Name (eg, city) []:Moscow
Organization Name (eg, company) [Internet
Widgits Pty Ltd]:ISP RAS
Organizational Unit Name (eg, section)
[]:INSECON  CA
Common Name (e.g. server FQDN or YOUR
name) []:insecon.ispras.ru
Email Address []:
```

Выпуск самоподписного сертификата [2/2]

```
$ openssl x509 -text -noout -in testCA.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:

01:8d:0e:5c:0e:3a:ce:2d:3a:44:4c:88:e9:9f:8c:6f:cc:
b5:b1:be
        Signature Algorithm:
sha256WithRSAEncryption
        Issuer: C = RU, ST = Moscow, L = Moscow, O
= ISP RAS, OU = INSECON CA, CN = insecon.ispras.ru
        Validity
            Not Before: Sep 26 09:23:26 2019 GMT
            Not After : Sep 23 09:23:26 2029 GMT
        Subject: C = RU, ST = Moscow, L = Moscow, O
= ISP RAS, OU = INSECON CA, CN = insecon.ispras.ru
        Subject Public Key Info:
...
```

Проверка выпущенного сертификата:

```
$ openssl verify -verbose \
    -CAfile testCA.crt \
    testCA.crt
test.crt: OK
```

Конфигурационный файл

OpenSSL по умолчанию использует конфигурацию:

`/etc/ssl/openssl.cnf`

Утилита `req` поддерживает загрузку конфигурационных файлов:

```
$ openssl req -config openssl.cnf \
    -x509 -new -days 3650 \
    -key testCA.key -out testCA.crt
```

Справка по формату конфигурационного файла:

```
$ man config
$ man x509v3_config
$ cat /etc/ssl/openssl.conf
```

Справка по атрибутам утилиты `req` для конфигурационного файла:

```
$ man openssl-req
```

```
$ cat openssl.conf
...
[req]
days                    = 3650
serial                  = 1
distinguished_name      = req_distinguished_name
x509_extensions         = v3_ca

[req_distinguished_name]
countryName             = RU
stateOrProvinceName     = Moscow
localityName            = Moscow
organizationName        = ISP RAS
organizationalUnitName  = INSECON CA
commonName              = insecon.ispras.ru
#0.emailAddress          = insecon@ispras.ru
...
```

Ключевая пара [1/2]

Генерация ключевой пары:

1. Выбрать два простых числа: p, q
2. Вычислить произведение: $n = p \cdot q$
3. Вычислить функцию Эйлера: $\varphi(n) = (p-1)(q-1)$
4. Выбрать открытую экспоненту: e $(1 < e < \varphi(n))$
5. Вычислить секретную экспоненту: $d \equiv e - 1 \pmod{\varphi(n)}$

Открытый ключ: $\{e , n\}$

Закрытый ключ: $\{d , n\}$

Генерация сертификата:

```
$ openssl genrsa -out testCA.key 2048
$ openssl req -x509 -new -days 3650 \
    -key testCA.key -out testCA.crt
```

Содержимое ***testCA.key***:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAx0Y8m55qnQmIAadq6RWtLp+bQ
oeVjQP9hJIiJ+l5auPPvv1sJ6XBf845hEbjCoTNZ2
hPyXKnS4NBSFTM+Wr7ioLQsTw9rOaWk+/b73uAG5l
MTST+q3DmjZK784Dv4N1Zcijcwd2JM3ENcfbpFTtz
HCLDgXTXarxG53ChY2aqGbRYNBl8AMTNhvdkX8es1
3uSkuzKv0yNgvoJzIdyKmZKeEucEqkkEW8ifhMbKu
lkB4s+gBpuGlYf3tUtTcI9E7/X5XL3mTNQcUhM0mb
TGL9bXw/F9Znidx9k5ph7zOw3zNTEHB/H
...
-----END RSA PRIVATE KEY-----
```

Ключевая пара [2/2]

Генерация ключевой пары:

1. Выбрать два простых числа: p, q
2. Вычислить произведение: $n = p \cdot q$
3. Вычислить функцию Эйлера: $\varphi(n) = (p-1)(q-1)$
4. Выбрать открытую экспоненту: e $(1 < e < \varphi(n))$
5. Вычислить секретную экспоненту: $d \equiv e - 1 \pmod{\varphi(n)}$

Открытый ключ: $\{e , n\}$

Закрытый ключ: $\{d , n\}$

Согласно *RFC 3447* *(A.1.2)*, закрытый ключ содержит структуру *ASN.1* типа *RSAPrivateKey*:

```
RSAPrivateKey ::= SEQUENCE {
          version           Version,
          modulus           INTEGER,  -- n
          publicExponent    INTEGER,  -- e
          privateExponent   INTEGER,  -- d
          prime1            INTEGER,  -- p
          prime2            INTEGER,  -- q
          exponent1         INTEGER,  -- d mod (p-
1)
          exponent2         INTEGER,  -- d mod (q-
1)
          coefficient       INTEGER,
                            -- (inverse of q) mod p
          otherPrimeInfos   OtherPrimeInfos
OPTIONAL
      }
```

Корневые сертификаты [1/3]

- Microsoft Windows
  - Microsoft Root Certificate Program
- *Apple macOS*
  - Apple Root Certificate Program
- *Linux*
  - Отсутствует централизованная программа
  - Широко используется набор криптографических библиотек Mozilla Network Security Services (NSS), включающих сертификаты Mozilla Root Certificate Program
  - Пакет ca-certificates включает сертификаты из Mozilla CA Certificate Store

Корневые сертификаты [2/3]

Ключевые директории и файлы в *Linux*:

- `/usr/share/ca-certificates/`
  - Хранилище сертификатов
- `/usr/local/share/ca-certificates/`
  - Хранилище пользовательских сертификатов
- `/etc/ca-certificates.conf`
  - Конфигурация утилиты `update-ca-certificates`
- `/etc/ssl/certs/ca-certificates.crt`
  - Все сертификаты в одном файле

```
$ ls -R /usr/share/ca-certificates/
 ACCVRAIZ1.crt
 AC_RAIZ_FNMT-RCM.crt
 Actalis_Authentication_Root_CA.crt
 AddTrust_External_Root.crt
 AffirmTrust_Commercial.crt
 AffirmTrust_Networking.crt
 AffirmTrust_Premium.crt
 AffirmTrust_Premium_ECC.crt
 Amazon_Root_CA_1.crt
 Amazon_Root_CA_2.crt
 Amazon_Root_CA_3.crt
 Amazon_Root_CA_4.crt
...
```

Корневые сертификаты [3/3]

Браузер *Mozilla Firefox* использует собственное хранилище корневых доверенных сертификатов:

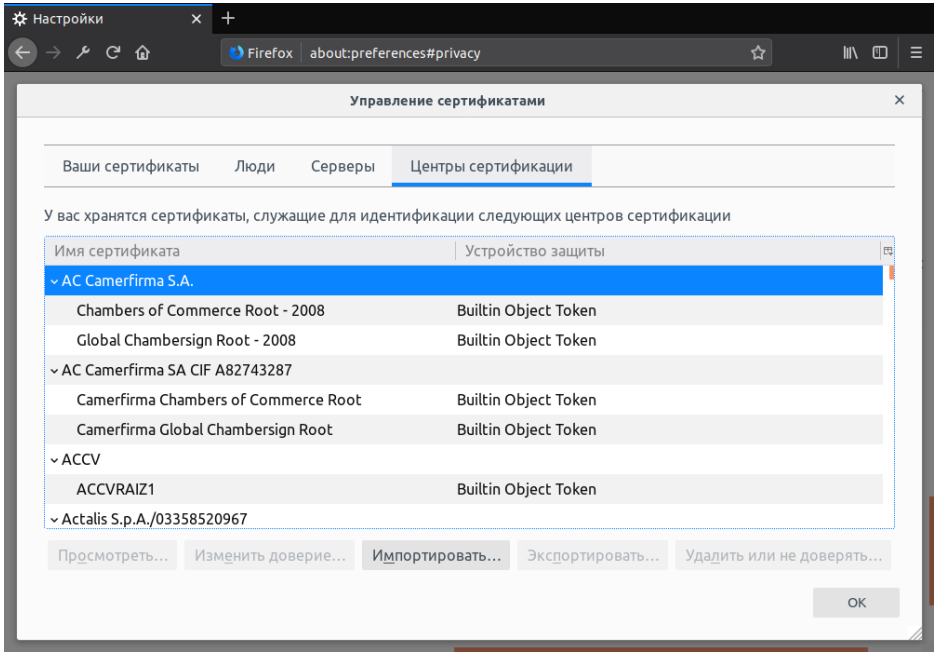
`/usr/lib/firefox/libnssckbi.so`

Браузер *Chromium/Google Chrome* используют системное хранилище корневых сертификатов:

`/usr/lib/x86_64-linux-gnu/nss/libnssckbi.so`

Добавить доверенный сертификат (*Firefox*):

- *Приватность и защита → Сертификаты → Просмотр сертификатов ...*



Выпуск сертификата [1/2]

```
$ openssl genrsa \
      -out test.key 2048
$ openssl req -new \
      -key test.key \
      -out test.csr
$ openssl x509 -req -days 365 \
      -CA testCA.crt \
      -CAkey testCA.key \
      -Cacreateserial \
      -CAserial serial \
      -in test.csr \
      -out test.crt
```

1. Сгенерировать ключевую пару;
2. Сгенерировать **запрос сертификата** к удостоверяющему центру, содержащего все атрибуты сертификата;
3. Удостоверяющий центр выпускает сертификат, используя запрос сертификата как источник данных атрибутов, а также подписывает его при помощи своего закрытого ключа.

Выпуск сертификата [2/2]

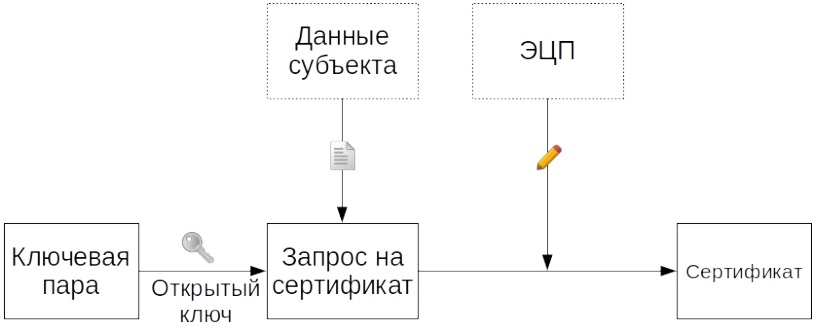
```
$ openssl x509 -text -noout -in test.crt
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            20:4e:cf:ac:2c:6a:a1:49:a8:c1:f4:57:63:bd:d1:0a:ee:24:d3
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = RU, ST = Moscow, L = Moscow, O = ISP RAS, OU =
INSECON CA, CN = insecon.ispras.ru, emailAddress = insecon@ispras.ru
        Validity
            Not Before: Sep 26 14:22:35 2019 GMT
            Not After : Sep 25 14:22:35 2020 GMT
        Subject: C = RU, ST = Moscow, L = Moscow, O = Student, OU =
Student, CN = student.ru, emailAddress = student@ispras.ru
```

Запрос на сертификат

Запрос на сертификат включает:

- Уникальный идентификатор
- Публичный ключ
- Набор атрибутов

Запрос на сертификат отправляется удостоверяющему центру, который преобразует CSR-запрос в *X.509* сертификат.




```
$ cat test.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIC0DCCAbgCAQAwgYoxCzAJBgNVBAYTAlJVMQ8wD
QYDVQQIDAZNb3Njb3cxDzANBgNVBAcMBk1vc2Nvdz
EQMA4GA1UECgwHU3R1ZGVudDEQMA4GA1UECwwHU3R
1ZGVudDETMBEGA1UEAwwKc3R1ZGVudC5tZTEgMB4G
CSqGSIb3DQEJARYRc3R1ZGVudEBpc3ByYXMucnUwg
gEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQ
C3JToDHVRCkTxbYWzdP26TNtBPcAaLbtr8KRgV2nm
TIxfxjL1kNPZAiRuA7h2Y+mt2f7hO1pT
...
LCAwOp8tSC2s5w5I1RCV0xv+hqChsFjXLsYsxzKiO
h6bF89mBYWsB8cmHi3dHtrReGRLe1JTNy2o9bI1yd
8TIfnVpoder8ucbmf2DFm6YO9LMvgDTjyPfck8WkP
GqCNarjmKA==
-----END CERTIFICATE REQUEST-----
```

Практическое задание №1.1

Сгенерировать цепочку сертификатов X.509, включающую следующие элементы:

- Корневой самоподписной сертификат:
  - Сертификат: `<фамилияио>-<группа>-ca.crt`;
  - Ключ: `<фамилияио>-<группа>-ca.key`;
- Промежуточный сертификат:
  - Сертификат: `<фамилияио>-<группа>-intr.crt`;
  - Ключ: `<фамилияио>-<группа>-intr.key`;
- Сертификат базовый:
  - Сертификат: `<фамилияио>-<группа>-basic.crt`;
  - Ключ: `<фамилияио>-<группа>-basic.key.`

Что?	Шесть PEM-файлов в архиве в названием `<фамилияио>-<группа>-p1_1.zip`
Куда?	insecon@ispras.ru (тема: `<вуз>-<группа>-p1_1`)
Когда?	Крайний срок 03.10.2022

Практическое задание №1.1 - Корневой

- Ключевая пара:
  - *RSA 4096* бит;
  - Зашифрован *AES 256*, пароль <фамилияио>;
- Сертификат
  - Самоподписной;
  - Срок действия 3 лет;
  - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_1, CN=<фамилияио> CA, email=<адрес вашей почты>`;
  - *X.509 v3* расширения:
    - *Basic Constrains*:
      - `Critical`
      - `CA=True`
    - *Key Usage:*
      - `Critical`
      - `Digital Signature`
      - `Certificate Sign`
      - `CRL sign.`

Практическое задание №1.1 - Промежуточный

- Ключевая пара:
  - *RSA 4096* бит;
  - Зашифрован *AES 256*, пароль <фамилияио>;
- Сертификат
  - Подписан корневым сертификатом;
  - Срок действия 1 год;
  - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, CN=<фамилияио> Intermediate CA, OU=<фамилияио> P1_1, email=<адрес вашей почты>`;
  - *X.509 v3* расширения:
    - *Basic Constrains*:
      - `Critical`
      - `PathLen=0`
      - `CA=True`
    - *Key Usage:*
      - `Critical`
      - `Digital Signature`
      - `Certificate Sign`
      - `CRL sign.`

Практическое задание №1.1 - Basic

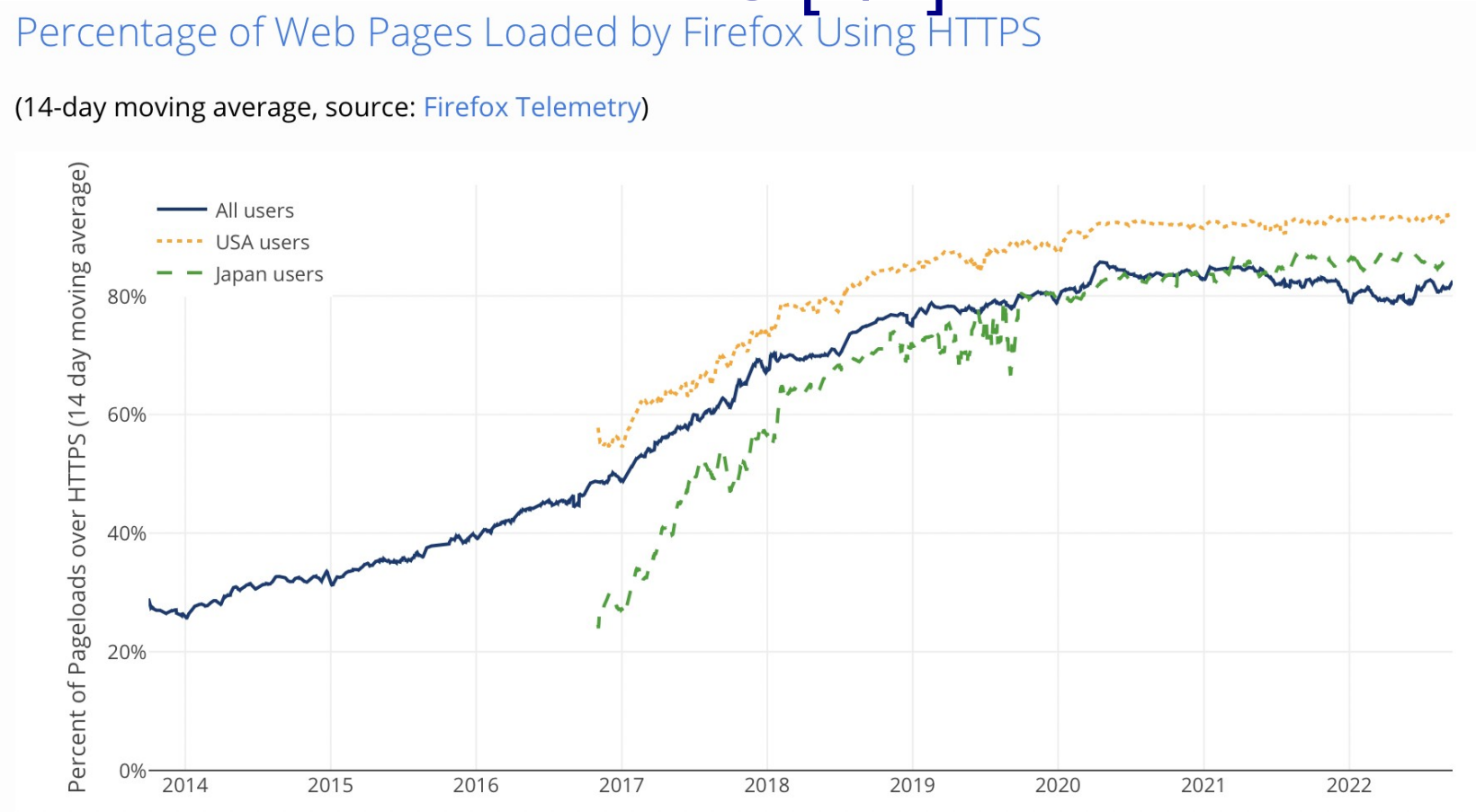
- Ключевая пара:
  - *RSA 2048* бит;
- Сертификат
  - Подписан промежуточным сертификатом;
  - Срок действия 90 дней;
  - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_1,  CN=<фамилияио> Basic, email=<адрес вашей почты>`;
  - *X.509 v3* расширения:
    - *Basic Constrains*:
      - `CA=False`
    - *Key Usage (`Critical`):*
      - `Digital Signature`
    - *Extended Key Usage (`Critical`)*:
      - `TLS Web Server Authentication`
      - `TLS Web Client Authentication`
    - `Subject Alternative Name:`
      - `basic.<фамилияио>.ru`
      - `basic.<фамилияио>.com`

Часть 2
CRL

HTTPS [1/2]

- Протокол *HTTPS* использует *X.509* сертификаты;
- Процесс получения сертификата у *Let's Encrypt* прост;
  - Протокол ACME (Automatic Certificate Management Environment) для автоматизации управления жизненным циклом сертификата:
    - Клиент-бот взаимодействует с веб-сервером (Nginx/Apache/IIS/...)
  - Подтверждение владения доменом (этап I);
    - *DNS-01 challenge;* `_acme-challenge.<YOUR_DOMAIN>`
    - *HTTP-01 challenge;* `http://<YOUR_DOMAIN>/.well-known/acme-challenge/<TOKEN>`
  - Выпуск сертификата (этап II);
    - Отправка CSR центру сертификации (имеющийся или сгенерированный ботом);
  - Бесплатно.
- Поисковики помечают сайты без *HTTPS* версии как «небезопасные».

HTTPS [2/2]


Percentage of Web Pages Loaded by Firefox Using HTTPS
(14-day moving average, source: Firefox Telemetry)
Percent of Pageloads over HTTPS (14 day moving average)
All users
USA users
Japan users
80%
60%
40%
20%
0%
2014
2015
2016
2017
2018
2019
2020
2021
2022

Статус сертификата

Сертификат имеет атрибуты периода действия:

- Не раньше
  `Not Before: Mar 12 15:30:45 2019 GMT`
- Не позднее
  `Not After : Mar 10 15:30:45 2024 GMT`

Механизмы проверки статуса сертификата:

- Список отозванных сертификатов (первый RFC 2459 — январь 1999)
  CRL *- Certificate Revocation List*
- Протокол состояния сертификата (первый RFC 2560 — июнь 1999)
  OCSP *- Online Certificate Status Protocol*

Список отозванных сертификатов

Принцип работы:

- Удостоверяющий центр публикует список, в котором перечислены все отозванные сертификаты;
- Список подписан закрытым ключом CA;
- Список содержит поле *nextUpdate* — время когда будет выпущен новый список;
- Список содержит только отозванные сертификаты, без истекших сертификатов;
- Механизм дельта-обновления списка.

Недостатки:

- Списки публикуются недостаточно часто;
- Списки имеют слишком большой размер;
  - ~229МБ на 1М записей;
  - Десятки миллионов записей;
- Уязвимость перед атакой «отказ в обслуживании»;
  - Частично решается созданием инфраструктуры «зеркал»;
- Let’s Encrypt (2012) создавался изначально без поддержки CRL;
- Механизм проверки удалён из Firefox 24.0+.

Вторая жизнь CRL

- Разработаны технологии *CRLite* (**Mozilla**, 2017) и *CRLSets* (**Google**), механизмы работы схожи:
  - Краулеры периодически загружают CRL **всех** CA (каждые несколько часов);
  - Списки проверяются на корректность и согласованность;
  - Списки отозванных сертификатов сжимаются структуру данных фильтр Блума (все отозванные сертификаты помещаются в 10МБ — менее 1 байта на домен);
  - Поставщик браузера распространяет ежедневные обновления (около 0.5МБ) — информация об отозванных сертификатах распространяется проактивно;
- С *1 октября 2022* в программы распространения корневых сертификатов Apple и Mozilla внесено требование об обязательном включении поля CRL Distribution Point.

Выпуск CRL-совместимого сертификата

- Добавить в конфигурационный файл атрибут:

```
[ basic_cert ]
...
crlDistributionPoints =
URI:http://crl.ivanovii.ru:8080/
ivanovii-123.crl
...
```

- Подробнее в документации:

```
$ man x509v3_config
$ man openssl-ca
```

Сертификат:

```
$ openssl x509 -text -noout \
     -in ivanovii-123-crl-valid.crt
...
  X509v3 CRL Distribution Points:
    Full Name:
      URI:
http://crl.ivanovii.ru:8080/
ivanovii-123.crl
...
```

Выпуск списка отозванных сертификатов

- Выпуск CRL:
```
$ openssl ca
      -config openssl.cnf \
      -gencrl \
      -out ivanovii-123.crl
```
- Отзыв сертификата (добавить в CRL):
```
$ openssl ca \
     -config openssl.cnf \
     -revoke ivanovii-123-crl-
revoked.crt
```
- Просмотр содержимого CRL:
```
$ openssl crl \
     -noout -text \
     -in ivanovii-123.crl
```

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer:
/C=RU/ST=Moscow/O=ivanovii/OU=ivanovii
P1/CN=ivanovii Intermediate CA
CA/emailAddress=ivanovii@ispras.ru
        Last Update: Mar 14 13:55:59 2019 GMT
        Next Update: Apr 13 13:55:59 2019 GMT
        CRL extensions:
            X509v3 Authority Key Identifier:
                keyid:CC:8C:F9:...:F8:91

            X509v3 CRL Number:
                4099
Revoked Certificates:
    Serial Number: 1007
        Revocation Date: Mar 14 11:27:18 2019 GMT
    Signature Algorithm: sha256WithRSAEncryption
         13:ba:d1:e2:d6:22:8c:a7:..:bd:d1:41:ed:
```

Проверка статуса сертификата - CRL

```
$ openssl verify -crl_check \
      -CRLfile ivanovii-123.crl \
      -CAfile ca-chain.crt \
      ivanovii-123-crl-valid.crt
ivanovii-123-crl-valid.crt: OK

$ openssl verify -crl_check
      -CRLfile ivanovii-123.crl \
      -CAfile ca-chain.crt \
      ivanovii-123-crl-revoked.crt
C = RU, ST = Moscow, O = ivanovii, OU = ivanovii CRL Revoked, CN = crl.revoked.ivanovii.ru,
emailAddress = ivanovii@ispras.ru
error 23 at 0 depth lookup: certificate revoked
error ivanovii-123-crl-revoked.crt: verification failed
```

Практическое задание №1.2

Сгенерировать CRL-файл и два сертификата:

- Валидный:
  - Сертификат: `<фамилияио>-<группа>-crl-valid.crt`;
  - Ключ: `<фамилияио>-<группа>-crl-valid.key`;
- Отозванный:
  - Сертификат: `<фамилияио>-<группа>-crl-revoked.crt`;
  - Ключ: `<фамилияио>-<группа>-crl-revoked.key`;
- Список отозванных сертификатов:
  - CRL: `<фамилияио>-<группа>.crl;`
  - Цепочка сертификатов: `<фамилияио>-<группа>-chain.crt.`

Что?	Шесть PEM-файлов в архиве в названием `<фамилияио>-<группа>-p1_2.zip`
Куда?	insecon@ispras.ru (тема: `<вуз>-<группа>-p1_2`)
Когда?	Крайний срок 10.10.2022

Практическое задание №1.2 - Подробнее

- Список отозванных сертификатов:
  - Подписан промежуточным сертификатом;
  - Расширения:
    - Authority Key Identifier
- Валидный сертификат:
  - Свойства соответствуют сертификату Basic из задания №1, кроме:
    - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_2, CN=<фамилияио> CRL Valid, email=<адрес вашей почты>`;
    - Присутствует атрибут `X509v3 Subject Alternative Name: crl.valid.<фамилияио>.ru` (только один домен)
    - Присутствует атрибут `X509v3 CRL Distribution Points`, URL сервера распространения *CRL*: `http://crl.<фамилияио>.ru`
  - **Отсутствовать** в списке отозванных сертификатов;
- Отозванный сертификат:
  - Свойства соответствуют сертификату Basic из задания №1, кроме:
    - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_2, CN=<фамилияио> CRL Revoked, email=<адрес вашей почты>`;
    - Присутствует атрибут `X509v3 Subject Alternative Name: crl.revoked.<фамилияио>.ru` (только один домен)
    - Присутствует атрибут `X509v3 CRL Distribution Points`, URL сервера распространения *CRL*: `http://crl.<фамилияио>.ru`;
  - **Присутствовать** в списке отозванных сертификатов.

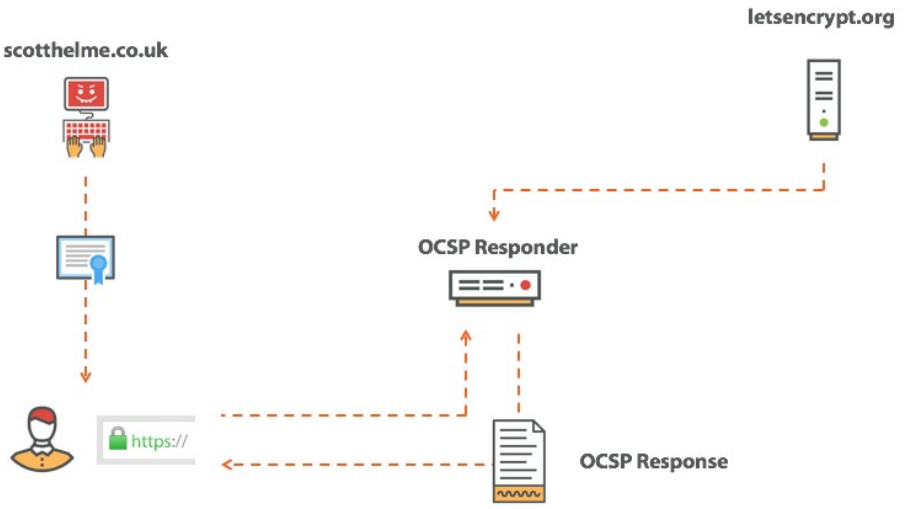
Часть 3
OCSP

Протокол состояния сертификата

Клиент запрашивает статус сертификата, OCSP-сервер отвечает статусом сертификата:

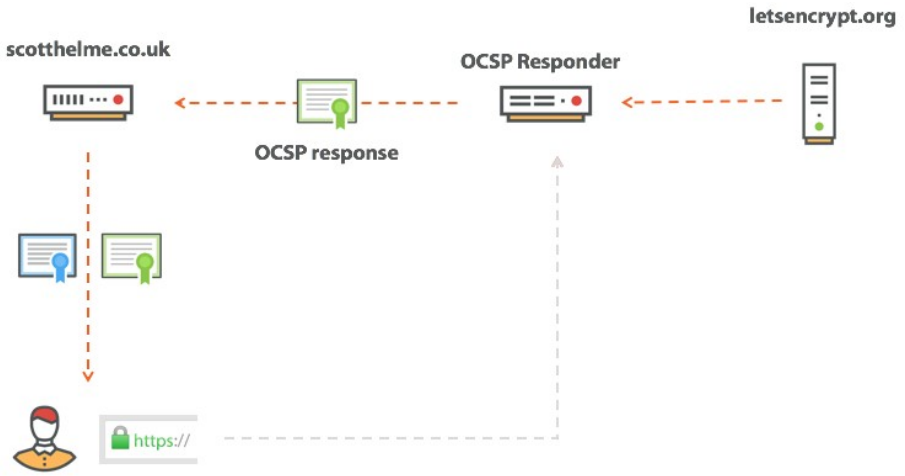
- Годный;
- Отозван;
- Статус неизвестен;

- OCSP-ответ подписывается закрытым ключом удостоверяющего центра;
  - Функция обслуживания OCSP-запросов может быть делегирована другому субъету удостоверяющим центром;
- Уязвим к *replay*-атакам;
  - Существует расширение, добавляющее *nounce* в тело запроса, но данное расширение снижает эффективность кэширования OCSP-ответов;
- OCSP-сервер получает информацию о посещаемых сетевых ресурсах.



OCSP Staple

- Сетевой ресурс, чей сертификат требует проверки, сам запрашивает *OCSP Responder* и добавляет ответ *OCSP Response* к сертификату (*OCSP Staple*);
  - Повышается анонимность клиента;
  - Снижается нагрузка на OCSP-сервер;
- Сертификат может включать атрибут *Must-Staple* — отклонять соединение, если отсутствует *OCSP Response*;
  - Злоумышленник может отключить *OCSP Stapling* на скомпрометированном ресурсе, *Must-Staple* сертификат не допустит этого;
- *HTTP* заголовок Expect-Staple — обратная связь на *OCSP Staple*.



Выпуск OCSP-совместимого сертификата

- Добавить в конфигурационный файл атрибут:

```
[ basic_cert ]
...
authorityInfoAccess =
OCSP;URI:http://ocsp.ivanovii.ru
:2560
...
```

- Подробнее в документации:

```
$ man x509v3_config
$ man openssl-ca
```

Сертификат:

```
$ openssl x509 -text -noout \
     -in ivanovii-123-ocsp-valid.crt
...
  Authority Information Access:
    OCSP -
URI:http://ocsp.ivanovii.ru:2560
...
```

OCSP Responser

Для функционирования *OCSP* серверу нужен отдельный сертификат подписи *OCSP* ответов с атрибутом *Key Usage*:

```
$ openssl x509 -text -noout \
  -in ivanovii-123-ocsp-resp.crt
...
  X509v3 Extended Key Usage:
critical
    OCSP Signing
...
```

Запуск OpenSSL OCSP Responder:

```
$ openssl ocsp \
  -port 2560 \
  -index index.txt \
  -CA ca-chain.cert.pem \
  -rkey ivanovii-123-ocsp-resp.key \
  -rsigner ivanovii-123-ocsp-resp.crt
```

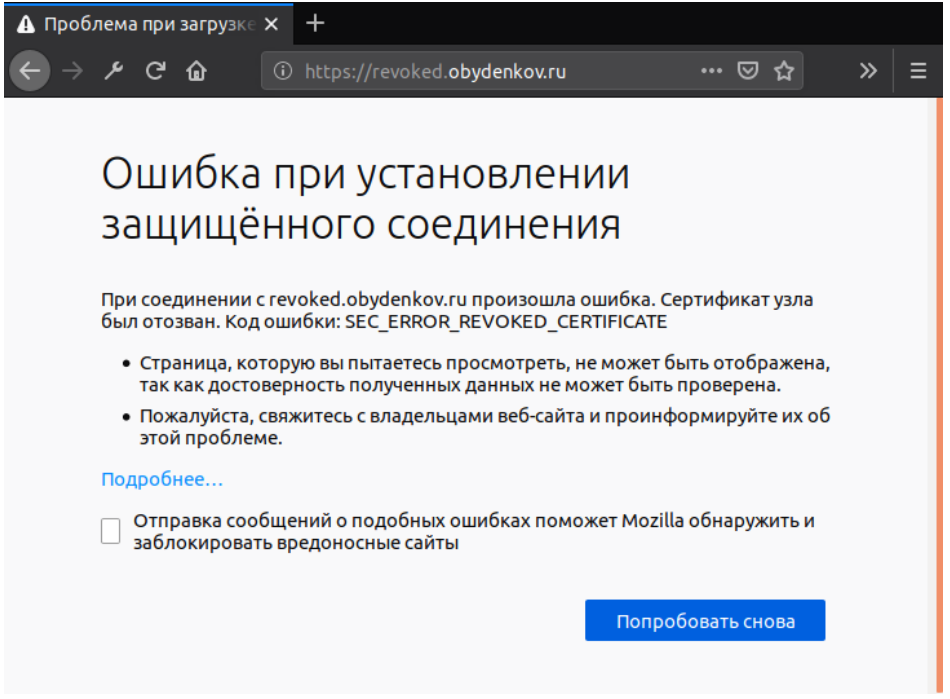
Отзыв сертификата:

```
$ openssl ca \
    -config openssl.cnf \
    -revoke ivanovii-123-ocsp-
revoked.crt
```

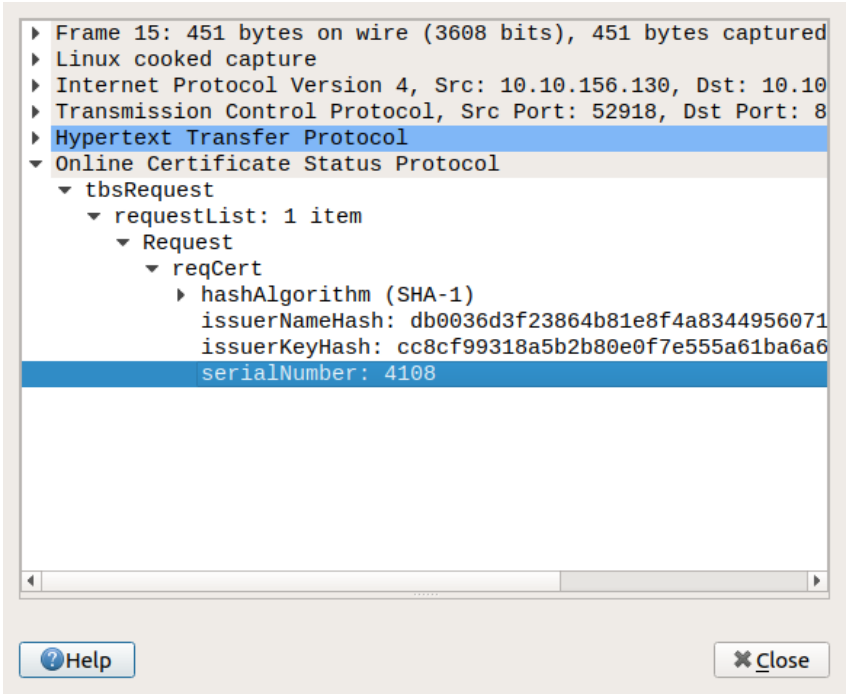
Проверка статуса сертификата - OCSP

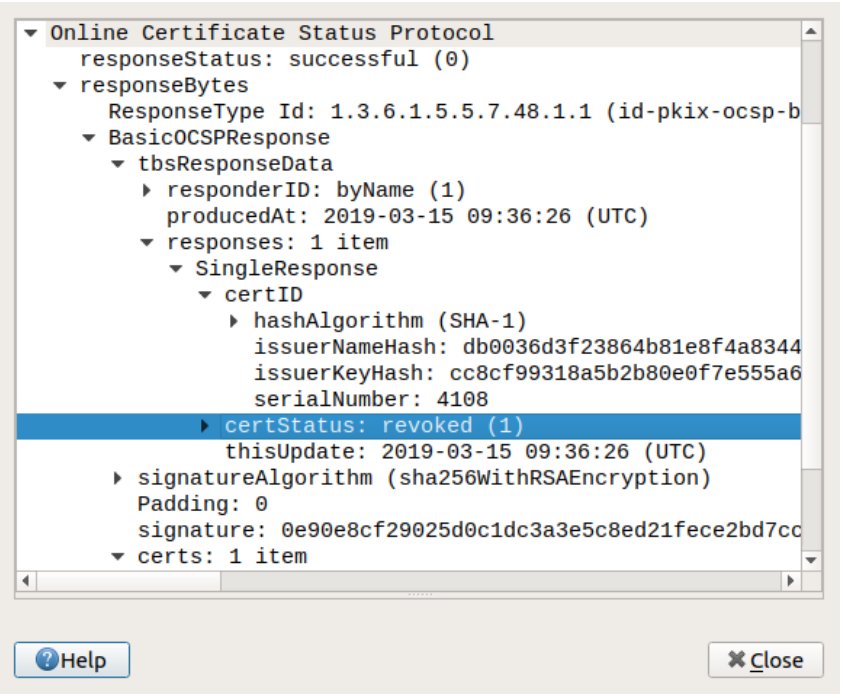
```
$ openssl ocsp \
      -url http://ocsp.ivanovii.ru:2560 \
      -CAfile ca-chain.crt \
      -issuer ivanovii-123-intr.crt \
      -cert ivanovii-123-ocsp-revoked.crt
Response verify OK
ivanovii-123-ocsp-revoked.crt: revoked
This Update: Mar 26 15:01:23 2019 GMT
Revocation Time: Mar 15 09:07:31 2019 GMT

$ openssl ocsp \
      -url http://ocsp.ivanovii.ru:2560 \
      -CAfile ca-chain.crt \
      -issuer ivanovii-123-intr.crt \
      -cert ivanovii-123-ocsp-valid.crt
Response verify OK
intermediate/certs/ocsp.valid.obydenkov.ru.cert.pem: good
This Update: Mar 26 15:01:27 2019 GMT
```

Проверка статуса сертификата - Firefox



Проверка статуса сертификата - Firefox





Практическое задание №1.3

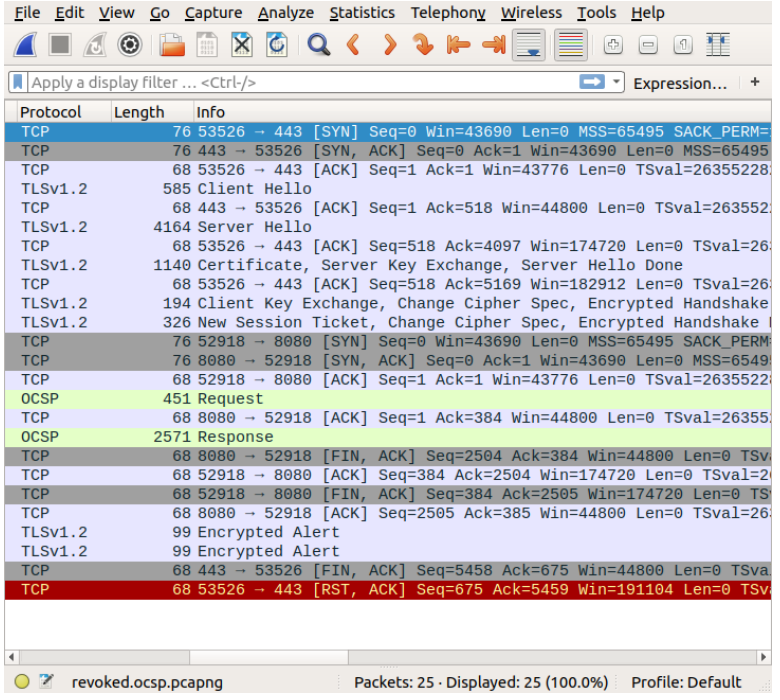
Развернуть тестовую среду с OCSP-сервером и создать два сертификата: отозванный и валидный:

- Валидный:
  - Сертификат: `<фамилияио>-<группа>-ocsp-valid.crt`;
  - Ключ: `<фамилияио>-<группа>-ocsp-valid.key`;
- Отозванный:
  - Сертификат: `<фамилияио>-<группа>-ocsp-revoked.crt`;
  - Ключ: `<фамилияио>-<группа>-ocsp-revoked.key`;
- Сертификат OCSP сервера:
  - Сертификат: `<фамилияио>-<группа>-ocsp-resp.crt`;
  - Ключ: `<фамилияио>-<группа>-ocsp-resp.key;`
  - Цепочка сертификатов: `<фамилияио>-<группа>-chain.crt`
- Сетевые трассы с OCSP запросами и TLS handshake и валидные `SSLKEYLOG`:
  - Валидный сертификат сервера:
    `<фамилияио>-<группа>-ocsp-valid.pcapng`;
    `<фамилияио>-<группа>-ocsp-valid.log`;
  - Отозванный сертификат сервера:
    `<фамилияио>-<группа>-ocsp-revoked.pcapng`;
    `<фамилияио>-<группа>-ocsp-revoked.log`.

Что?	Семь PEM-файлов и два PCAP+LOG в архиве в названием `<фамилияио>-<группа>-p1_3.zip`
Куда?	insecon@ispras.ru (тема: `<вуз>-<группа>-p1_3`)
Когда?	Крайний срок 17.10.2022

Практическое задание №1.3 — Подробнее [1/3]

- Необходимо развернуть окружение, состоящее из веб-сервера (Apache, NGINX, Lighttpd и прочие) и *OCSP Responder*;
- Веб-сервер должен быть сконфигурирован для использования сертификатов:
  - `<фамилияио>-<группа>-ocsp-revoked.crt;`
  - `<фамилияио>-<группа>-ocsp-valid.crt;`
- Веб-сервер должен должен отдавать страницу с произвольным содержимым по адресам:
  - `ocsp.valid.<фамилияио>.ru;`
  - `ocsp.revoked.<фамилияио>.ru;`
- Браузер клиента должен устанавливать TLS соединение с веб-сервером и запрашивать статус сертификата у *OCSP Responder (*без *Stapling)*;
  - Браузер должен показывать зелёный замочек или сообщение об ошибке;
- Сетевая трасса соединение может быть записана утилитами Wireshark, tshark, tcpdump;
  - Материалы по Wireshark [1], [2], [3]



Пример сетевой трассы `ocsp-revoked.pcapng`

Практическое задание №1.3 — Подробнее [2/3]

- Ключевая пара:
  - *RSA 4096* бит;
  - Зашифрован *AES 256*, пароль <фамилияио>;
- Сертификат
  - Подписан промежуточным сертификатом;
  - Срок действия 1 год;
  - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_3, CN=<фамилияио> OCSP Responder, email=<адрес вашей почты>`;
  - *X.509 v3* расширения:
    - *Basic Constrains*:
      - `CA=False`
    - *Key Usage:*
      - `Critical`
      - `Digital Signature`
    - *Extended Key Usage:*
      - `OCSP Signing`

Практическое задание №1.3 — Подробнее [3/3]

- Валидный сертификат:
  - Свойства соответствуют сертификату Basic из задания №1, кроме:
    - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_3, CN=<фамилияио> OCSP Valid, email=<адрес вашей почты>`;
    - Присутствует атрибут `X509v3 Subject Alternative Name: ocsp.valid.<фамилияио>.ru` (только один домен)
    - Присутствует атрибут `X509v3 Authority Information Access`, URL OSCP Responder: `http://ocsp.<фамилияио>.ru`
  - **Отсутствовать** в списке отозванных сертификатов;
- Отозванный сертификат:
  - Свойства соответствуют сертификату Basic из задания №1, кроме:
    - `C=RU, ST=Moscow, L=Moscow, O=<фамилияио>, OU=<фамилияио> P1_3, CN=<фамилияио> OCSP Revoked, email=<адрес вашей почты>`;
    - Присутствует атрибут `X509v3 Subject Alternative Name: ocsp.revoked.<фамилияио>.ru` (только один домен)
    - Присутствует атрибут `X509v3 Authority Information Access`, URL OSCP Responder: `http://ocsp.<фамилияио>.ru`
  - **Присутствовать** в списке отозванных сертификатов.

Certificate Transparency

Проблема:

- CA может выпустить «валидный» сертификат для любого домена
  - Hongkong Post теоретически может выпустить сертификат для yandex.ru
- CA может быть скомпрометирован
  - Нидерландский CA DigiNotar взломан в 2011 году, злоумышленники выпустили 247 сертификатов (в том числе для Google)

Решение:

- Публиковать списки всех выпущенных сертификатов
  - Только добавление новых логов в список
  - На основе дерева Меркла, корень подписан
- Экосистема CT:
  - *Операторы логов*
    Поддерживают список выпущенных сертификатов
  - *Аудиторы*
    Проверяют корректность списка выпущенных сертификатов
  - *Мониторы*
    Сверяют выпущенные сертификаты и задеплоенные на данном домене

Certificate Transparency

- Подходы доставки Signed Certificate Timestamp (SCT):
  - *Расширение X.509v3*
    CA выпускает пресертификат, отправляет операторам логов, получает подписанный SCT, выпускает сертификат
  - *OCSP Stapling*
  - *Расширение TLS*
- Возможны различные политики CT:
  - Chrome и Safari выполняет проверки SCT
  - Firefox не проверяет SCT

```
$ openssl x509 -text -noout -in ispras-ru.pem
...
CT Precertificate SCTs:
    Signed Certificate Timestamp:
        Version   : v1 (0x0)
        Log ID    : 46:A5:55:EB:75:FA:91:20:30:B5:A2:89:69:F4:F3:7D:
                    11:2C:41:74:BE:FD:49:B8:85:AB:F2:FC:70:FE:6D:47
        Timestamp : Jan 21 13:37:45.458 2021 GMT
        Extensions: none
        Signature : ecdsa-with-SHA256
                    30:45:02:20:2A:F3:A0:A2:24:7B:FA:74:AB:C1:C8:76:
                    59:EF:3C:1C:6C:FF:E6:66:36:5A:C1:AE:DF:92:A6:9E:
                    3A:51:18:00:02:21:00:D5:1E:67:F3:A6:94:DE:D4:7E:
                    D4:25:81:6B:6A:81:2B:39:85:31:9D:D8:C4:21:C4:36:
                    73:18:94:DD:00:3C:5B
    Signed Certificate Timestamp:
        Version   : v1 (0x0)
        Log ID    : DF:A5:5E:AB:68:82:4F:1F:6C:AD:EE:B8:5F:4E:3E:5A:
                    EA:CD:A2:12:A4:6A:5E:8E:3B:12:C0:20:44:5C:2A:73
        Timestamp : Jan 21 13:37:45.497 2021 GMT
        Extensions: none
        Signature : ecdsa-with-SHA256
                    30:45:02:21:00:A0:0F:79:BC:7C:5C:CB:51:AF:F7:E3:
                    D9:17:E5:A6:49:2C:C6:4A:49:E4:2D:DE:C7:33:4C:FB:
                    19:0C:C7:E6:95:02:20:1D:9F:C1:54:EC:07:25:07:11:
                    ED:23:13:8E:29:41:E7:8E:19:1F:B9:8A:0C:46:AB:BF:
                    86:A5:39:69:25:9F:7F
...
```

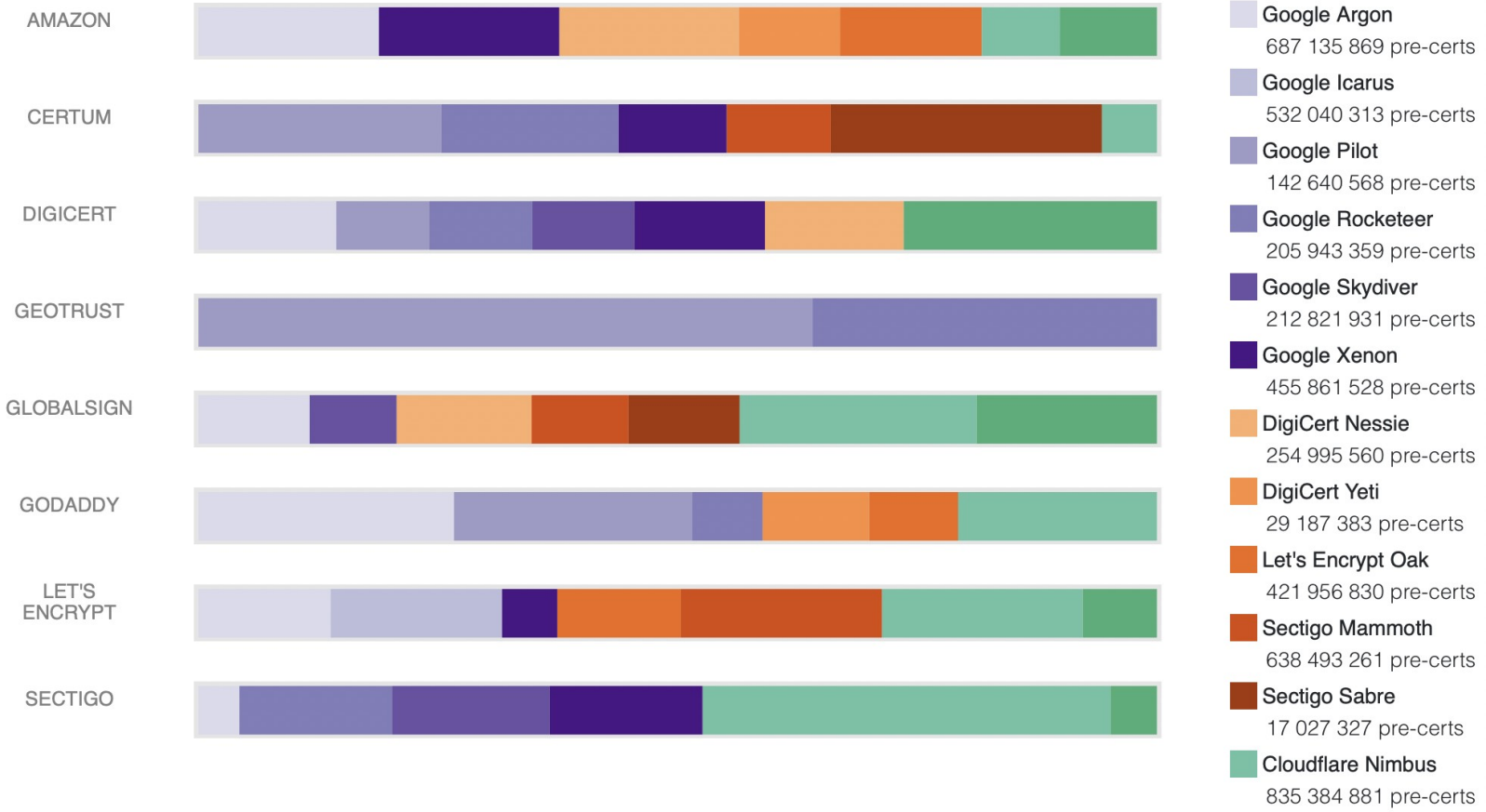
Google Xenon2022

<Log ID> = sha256(pubkey)

<Signature> = E_{PK}(sct_version + signature_type + timestamp + precert + extensions)

Lets Encrypt Oak 2022

CT — Операторы логов



Best Practices

- RSA 2048 достаточно;
  - ECDSA лучше, есть проблемы с совместимостью;
- Не используйте SHA-1 и MD5 для подписи сертификата;
- Используете пароль для ключей если бэкапите ключи вне сервера;
  - 600 (-rw-------) для ключей – минимум, HSM (Hardware Security Module) – максимум;
  - При компрометации сервера пароль не защитит;
- Выпускайте сертификаты на год или меньший срок;
- Проверьте что SAN покрывает все поддомены;
  - Старайтесь не использовать wildcard;
  - Последние версии браузеров не валидируют CN;
- Используйте TLS v1.2 или v1.3;
- Используйте актуальный cipher suites;
  - Предпочтение AEAD и PFS шифрам;
- Деплойте на сервере полную цепочку сертификатов;
  - На клиенте может не оказаться промежуточного сертификата;
- TLS для всего — статический контент;
- Используйте HSTS (HTTP Strict Transport Security) — исключаем HTTP;
- Используйте CSP (Content Security Policy) — защищаемся от XSS и HTTP third-party;
- …

Рекомендации и советы к выполнению практической работы

Описанное далее не является обязательным

OpenSSL

Выпуск сертификата:

- Копируете `/etc/ssl/openssl.conf` в рабочую директорию, удаляете лишнее, модифицируете
- **Приватный ключ**;
  - `$ openssl genrsa … -out *.key 4096`
- **Запрос на сертификат**;
  - `$ openssl req -config openssl.conf -new … -out *.csr`
  - *Request extensions* (например, *SAN*):
    - Конфигурация `[req] req_extensions = ...`
    - Опция `-addext`
- **Сертификат**;
  - `$ openssl ca -config openssl.conf -days 365 ... -out *.crt -infiles *.csr`
  - Сделать самоподписным — опция `-selfsign`
  - Перегрузить `[ ca ] default_ca` — опция `-name`
    - Выбор издателя *CA* или *Intermediate CA*
  - Выбрать набор *x509v3* расширений — опция `-extensions`
    - Разные наборы расширений *CA*, *Intermediate CA*, *Basic*, …
    - Важный параметр `copy_extensions` — обработка *Request extensions*

Настройка тестового стенда с NGINX

- Установите `nginx`
  - Для *Linux* достаточно установки из пакета
- Создайте стартовую страницу `index.html`, поместите в директорию `/var/www/my-site`
  - Для тестового стенда достаточно простейшего сайта из одной стартовой страницы
- Добавьте доменное имя тестового сайта в `/etc/hosts`
- Добавьте конфигурацию сайта в директорию `/etc/nginx/sites-available`
  - Конфигурацию должна включать секцию `server`, в которой указывается путь к ресурсам сайта и сертификатам (ключ и сертификат для сайта, цепочка *CA*-сертификатов)
  - Используйте документацию сервера
- Замените символьную ссылку `/etc/nginx/sites-enabled/default` на созданную конфигурацию сайта
- Перезагрузите конфигурацию сервера
  - `# systemctl reload nginx.service`

Запись сетевой трассы с расшифровкой TLS

- Запустите *Wireshark* и активируйте сетевой трассы с интерфейса lo
- Запустите браузер Firefox с опцией логирования NSS ключей
  - `$ SSLKEYLOGFILE=sslkey.log ./firefox`
- Добавьте корневой сертификат в браузер
- Перейдите на тестовый сайт с использованием браузера
  - Убедитесь, что сайт корректно отображается
  - Не забывайте про кэш
- Закройте браузер
- Остановите запись сетевой трассы
- Укажите в настройках *Wireshark* путь к лог-файлу
  - `Edit→Preferences→Protocols→SSL→(Pre)-Master-Secret log filename`

Запись сетевой трассы с расшифровкой TLS