МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

ОТЧЁТ

По лабораторной работе № \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил:

Студент группы ИИ-22

Копанчук Евгений Романович

Проверил\_\_:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Брест – 2023

Ход работы

Генерирование самоподписанного сертификата

Для начала сгенерируем пару ключей. На этот раз выберем тип ключа RSA

openssl genpkey -algorithm RSA -out root\_keypair.pem

..+....+..+....+..+.......+.....+...+....+...+++++++++++++++++++++++++++++++++++++++\*......+...+...+.....+...+.........+.+..+.......+............+...+........+..........+...+............+.........+...........+...+.+.....+.............+.....+++++++++++++++++++++++++++++++++++++++\*...+....+...+........+....+.........+.....+.........+...+....+...+...........+....+...........+.+...+...+.........+.....+...+...+.......+...............++++++

....+...+............+...+......+.....+.......+...+..+.........+...+...+++++++++++++++++++++++++++++++++++++++\*..+..........+......+.....+......+....+++++++++++++++++++++++++++++++++++++++\*.......+...............+.....+......+....+..+...+.+.....+.+.....+..................................+.....+................+...+.....+.+....................+...+......+....+...+..+...+.........+......+.......+..+...+...+.......+..+......+...+....+........+....+.....+.+..+......+.+......+.....+...+......+.+............+...........+....+..+......+............+......+.+........+...+....+......+..............+.+..+.......+...+..+..........+.........+........................+........+...+.+...+.....+...+....+...+...+.................+.+..+.......+..................+........+..........+..+....+........+...+....+........+...+.......+...+..+.+......+..............+.+...+.....+......+......+.+........+....+............+.........+.........+...+..+..........+..+...+.+.....+.........+.......+...+........+...+.+...+..+......+...+..........+...........+......+...+....+...+..+.......+...+..+....+.....+......+.+.....+....+...+...+.....+.......+..+....+.....+..........+..............+.+..+.......+...++++++

Проверим созданный ключ:

openssl pkey -in root\_keypair.pem -noout -text

Private-Key: (2048 bit, 2 primes)

modulus:

00:a7:09:65:cf:08:21:da:b3:3c:38:c9:a5:14:ab:

83:9e:28:94:84:d0:7e:31:54:e7:3f:30:d5:29:2f:

92:d9:3d:41:51:b8:2f:68:dc:fa:ef:1a:a2:a9:54:

fb:a4:04:d9:c0:60:7a:b4:25:33:f9:9e:d8:f9:34:

d5:5c:37:7c:0d:99:6c:ac:1e:81:19:a7:89:52:b4:

0b:4d:24:6e:0d:46:9c:c1:9f:1b:bb:29:f8:1a:59:

91:16:72:2a:2f:5f:72:e4:17:40:64:e4:de:62:5a:

6f:b6:f7:72:56:05:b9:5b:a5:6f:49:65:ea:07:29:

77:27:0c:8e:76:62:67:55:f1:aa:92:ce:de:14:9b:

2b:36:0b:06:eb:4b:db:ba:49:f4:db:1c:46:c5:b1:

78:a0:61:e8:82:9a:ec:08:a3:7a:37:14:ac:73:6a:

2c:72:e6:22:29:fb:a8:cb:4e:fe:ae:b1:45:19:00:

60:17:56:99:85:ad:ef:4c:3f:0d:8e:7a:85:15:a4:

22:a4:ac:34:63:d5:8a:1c:50:73:b5:65:b3:ca:13:

31:31:0b:b6:c2:33:cd:54:14:25:3f:0e:77:7c:29:

6b:4a:a4:c0:8a:35:ed:73:5f:be:7e:8a:e7:df:1d:

06:88:17:9f:96:a8:9c:21:23:70:bb:d1:4b:a5:6b:

9c:75

publicExponent: 65537 (0x10001)

privateExponent:

45:72:5f:4d:31:bc:dd:90:1a:70:23:ca:0a:2b:7b:

07:d5:28:56:85:d6:89:8d:1a:ad:c7:90:40:54:e2:

b1:86:49:b3:7f:a7:da:69:d1:bc:7a:6e:f9:86:f7:

2e:8c:c3:b9:bb:83:18:85:c5:33:b9:ca:cc:e5:9d:

f5:7e:ae:09:7b:69:d6:af:c2:0e:17:fd:3c:90:d5:

88:07:a6:e7:c7:1d:62:54:d3:f3:3f:72:43:06:4d:

0f:af:7c:81:80:a6:24:55:d4:74:7e:41:45:54:03:

51:67:54:b6:96:1f:a8:0e:42:94:1a:16:b0:4e:99:

96:f7:1d:fd:f3:50:b2:46:a0:73:14:ec:d5:4e:60:

33:6f:e0:fd:68:6a:b2:7d:08:1d:0f:db:24:d3:25:

a4:d5:bb:a4:6a:0d:ca:6a:5a:34:34:99:07:7a:53:

22:76:f1:16:4c:39:1b:60:22:b0:17:59:df:df:d2:

19:46:b3:39:9c:7f:e6:c1:84:b9:42:08:f6:6a:07:

e1:94:bd:7f:59:fb:8b:87:94:bb:27:aa:52:f7:35:

a5:32:ea:92:31:4d:17:02:fe:af:40:66:f5:3a:18:

85:61:47:75:0c:f8:ac:56:41:90:68:1b:2c:27:e3:

59:bc:70:9a:37:7e:eb:46:cf:41:c0:a0:52:d8:e8:

8b

prime1:

00:bc:d6:62:8f:95:8c:43:f3:fc:7e:ec:39:4f:19:

0b:44:1d:fd:99:da:54:8a:93:86:af:22:c9:3f:31:

d2:b5:68:5c:f6:a0:09:bc:98:d7:84:d6:ab:2a:96:

b5:b1:25:eb:73:f2:ac:52:d0:81:f6:85:ce:3f:1b:

05:03:4d:7d:cf:33:07:74:6c:77:8c:21:fe:1a:be:

b2:1f:15:b2:dd:90:f8:69:b7:a7:b0:28:a6:83:9e:

2b:4d:9d:99:12:a9:9a:11:d7:ca:02:56:08:94:2e:

1b:89:5d:b8:af:64:90:8f:1c:d6:c0:77:cb:31:b3:

64:91:61:85:b1:53:e5:cc:6b

prime2:

00:e2:72:0f:5b:f1:8f:ea:67:0c:dc:0b:67:9d:29:

71:49:c2:92:f9:55:b7:39:69:19:e2:fc:bf:73:70:

3f:ee:fb:92:aa:b7:08:43:df:c4:cc:a8:31:a1:50:

a3:12:cd:40:1b:77:02:56:03:be:b5:3b:a3:e6:99:

31:19:be:84:1e:77:24:af:59:c3:c7:db:8c:be:2e:

e8:84:bf:d3:b4:e7:e8:dc:51:92:97:61:99:5b:ba:

5a:4d:b1:f4:bc:8e:1d:41:2a:a4:c4:a7:c3:a2:7e:

be:42:44:1e:30:d4:13:94:c0:06:65:79:03:2f:0d:

5b:42:04:79:7a:c8:7e:72:9f

exponent1:

76:82:30:9e:9d:a2:c6:6e:93:72:6f:fb:53:49:8b:

49:77:6c:d5:26:30:d2:3a:dd:56:82:05:3c:54:9f:

40:db:bb:23:c7:b1:f4:9b:12:fe:39:de:77:be:d7:

4d:d2:57:5e:8f:d8:57:ac:a5:33:3a:2c:02:6c:89:

8b:d6:10:f3:7c:b9:db:f5:ca:5c:b6:64:8b:12:4c:

24:b7:f0:65:9e:7a:57:96:3b:75:91:3d:56:f2:67:

88:b1:34:04:83:4d:be:8e:e7:26:ea:29:3e:1d:65:

88:a2:b9:a9:d0:f9:eb:e6:94:5b:7b:92:98:a1:55:

a6:eb:46:04:d0:7b:31:fb

exponent2:

0f:13:3e:d8:85:25:6a:9a:7f:ad:aa:88:b4:5a:cc:

f2:f7:24:27:bb:40:1e:6f:53:cc:db:bc:b7:27:b2:

e9:eb:0f:a1:93:db:d6:00:f0:ae:1b:48:6a:8f:de:

bb:62:b4:b0:df:64:65:6b:8c:67:ae:57:c4:03:50:

d4:80:ed:74:c5:c3:c8:74:b6:66:f8:c3:ce:e9:7a:

8b:53:e5:f9:36:33:37:6f:aa:cd:af:2e:b1:7f:9f:

81:6a:a6:7b:63:0e:a9:86:f8:3f:2b:58:07:da:5e:

3a:35:24:b9:65:75:04:43:20:69:e2:fb:86:47:62:

5a:eb:c7:56:98:40:6a:cb

coefficient:

00:ab:10:5e:f8:6d:8e:4d:e8:66:de:ac:c7:5a:a4:

38:d0:72:53:a6:45:ee:71:d6:5d:b2:bf:af:fd:0c:

2e:49:53:c4:7c:ed:95:c8:50:6a:3a:3c:9b:d8:20:

5f:2b:ad:19:63:e9:09:25:c7:24:30:dc:a2:92:3d:

d3:5f:6f:f5:16:85:81:78:c4:05:d4:1c:27:86:5b:

93:ed:4c:b7:77:cd:88:bb:37:69:17:d3:6b:8f:67:

64:51:b7:d1:d2:df:35:ee:40:a8:1b:7b:93:03:0c:

ae:6f:31:e7:0e:88:9a:18:e9:66:59:57:ca:85:da:

da:2d:f5:49:20:ac:63:2f:5e

Сгенерируем CSR-запрос.

openssl req -new -subj “/CN=Root CA” -addext “basicConstraints=critical,CA:TRUE” -key root\_keypair.pem -out root\_csr.pem

Заметим, что мы добавили расширение CA:TRUE. Это необходимо для сертификатов УЦ.

Ничего не напечатано, и создан файл, root\_csr.pem, содержащий наш CSR-запрос. Посмотрим, что в нем:

openssl req -in root\_csr.pem -noout -text

Certificate Request:

Data:

Version: 1 (0x0)

Subject: CN = Root CA

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (2048 bit)

Modulus:

00:a7:09:65:cf:08:21:da:b3:3c:38:c9:a5:14:ab:

83:9e:28:94:84:d0:7e:31:54:e7:3f:30:d5:29:2f:

92:d9:3d:41:51:b8:2f:68:dc:fa:ef:1a:a2:a9:54:

fb:a4:04:d9:c0:60:7a:b4:25:33:f9:9e:d8:f9:34:

d5:5c:37:7c:0d:99:6c:ac:1e:81:19:a7:89:52:b4:

0b:4d:24:6e:0d:46:9c:c1:9f:1b:bb:29:f8:1a:59:

91:16:72:2a:2f:5f:72:e4:17:40:64:e4:de:62:5a:

6f:b6:f7:72:56:05:b9:5b:a5:6f:49:65:ea:07:29:

77:27:0c:8e:76:62:67:55:f1:aa:92:ce:de:14:9b:

2b:36:0b:06:eb:4b:db:ba:49:f4:db:1c:46:c5:b1:

78:a0:61:e8:82:9a:ec:08:a3:7a:37:14:ac:73:6a:

2c:72:e6:22:29:fb:a8:cb:4e:fe:ae:b1:45:19:00:

60:17:56:99:85:ad:ef:4c:3f:0d:8e:7a:85:15:a4:

22:a4:ac:34:63:d5:8a:1c:50:73:b5:65:b3:ca:13:

31:31:0b:b6:c2:33:cd:54:14:25:3f:0e:77:7c:29:

6b:4a:a4:c0:8a:35:ed:73:5f:be:7e:8a:e7:df:1d:

06:88:17:9f:96:a8:9c:21:23:70:bb:d1:4b:a5:6b:

9c:75

Exponent: 65537 (0x10001)

Attributes:

Requested Extensions:

X509v3 Basic Constraints: critical

CA:TRUE

Signature Algorithm: sha256WithRSAEncryption

Signature Value:

02:24:41:80:81:63:86:41:1b:e5:47:22:ee:b8:d1:25:be:4e:

c9:a4:2f:ea:b2:cf:bb:2e:5a:63:7b:28:d4:32:86:2d:18:76:

79:a2:7f:3c:27:04:ff:2a:72:1d:70:68:e1:91:66:f6:1a:98:

53:6f:98:92:49:24:3a:1c:79:6b:f6:8d:00:d8:16:16:55:76:

cc:17:6e:30:3a:58:3a:8a:67:13:3b:4c:5c:f0:57:8b:7d:ef:

82:56:b1:d0:cd:77:32:e4:2c:7c:25:4c:cc:68:5f:80:27:12:

37:76:c8:19:d7:65:c4:09:fe:80:48:47:c1:77:95:fa:53:6e:

b1:c4:09:1a:6b:6f:b4:dc:ad:9f:60:f9:05:53:9b:1d:b3:9c:

af:b7:a0:43:78:e3:0f:a7:19:aa:f9:97:bd:45:90:ab:5d:20:

71:f1:01:3f:12:7d:26:26:de:23:82:72:b0:ab:ca:55:26:f5:

26:6e:23:b7:9f:33:21:31:ec:23:23:e9:7d:5d:1f:1f:d8:b8:

1e:39:03:22:43:54:58:7b:9c:ef:0d:fa:ea:27:17:b5:23:ab:

28:f2:6a:78:ef:9e:24:32:7d:4d:02:4c:9a:9e:71:fc:5d:fa:

f5:75:40:dc:d4:27:5d:f5:0a:07:07:c6:48:92:c4:d1:23:54:

9b:3a:3b:6b

Теперь сгенерируем самоподписанный сертификат, используя CSR запрос и пару ключей. Срок действия нового сертификата будет составлять 3650 дней, т. е. около 10 лет:

openssl x509 -req -in root\_csr.pem -copy\_extensions copyall -key root\_keypair.pem -days 3650 -out root\_cert.pem

Certificate request self-signature ok

subject=CN = Root CA

Обратите внимание на параметр -copy\_extensions copyall. Он нужен, потому что по умолчанию команда openssl x509 не копирует расширения X509v3 из CSR-запроса в сертификат.

Ничего не напечатано, и создан файл, root\_cert.pem, содержащий самоподписанный сертификат. Посмотрим, что в нем:

openssl x509 -in root\_cert.pem -noout –text

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

59:53:43:6f:05:ea:13:1c:1c:aa:48:39:03:92:9c:82:b4:74:4b:86

Signature Algorithm: sha256WithRSAEncryption

Issuer: CN = Root CA

Validity

Not Before: Nov 23 09:07:43 2023 GMT

Not After : Nov 20 09:07:43 2033 GMT

Subject: CN = Root CA

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (2048 bit)

Modulus:

00:a7:09:65:cf:08:21:da:b3:3c:38:c9:a5:14:ab:

83:9e:28:94:84:d0:7e:31:54:e7:3f:30:d5:29:2f:

92:d9:3d:41:51:b8:2f:68:dc:fa:ef:1a:a2:a9:54:

fb:a4:04:d9:c0:60:7a:b4:25:33:f9:9e:d8:f9:34:

d5:5c:37:7c:0d:99:6c:ac:1e:81:19:a7:89:52:b4:

0b:4d:24:6e:0d:46:9c:c1:9f:1b:bb:29:f8:1a:59:

91:16:72:2a:2f:5f:72:e4:17:40:64:e4:de:62:5a:

6f:b6:f7:72:56:05:b9:5b:a5:6f:49:65:ea:07:29:

77:27:0c:8e:76:62:67:55:f1:aa:92:ce:de:14:9b:

2b:36:0b:06:eb:4b:db:ba:49:f4:db:1c:46:c5:b1:

78:a0:61:e8:82:9a:ec:08:a3:7a:37:14:ac:73:6a:

2c:72:e6:22:29:fb:a8:cb:4e:fe:ae:b1:45:19:00:

60:17:56:99:85:ad:ef:4c:3f:0d:8e:7a:85:15:a4:

22:a4:ac:34:63:d5:8a:1c:50:73:b5:65:b3:ca:13:

31:31:0b:b6:c2:33:cd:54:14:25:3f:0e:77:7c:29:

6b:4a:a4:c0:8a:35:ed:73:5f:be:7e:8a:e7:df:1d:

06:88:17:9f:96:a8:9c:21:23:70:bb:d1:4b:a5:6b:

9c:75

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints: critical

CA:TRUE

X509v3 Subject Key Identifier:

98:4F:70:73:50:07:97:23:C4:E1:F1:00:31:A7:43:85:DC:04:65:50

Signature Algorithm: sha256WithRSAEncryption

Signature Value:

57:b6:a2:4a:3a:a9:17:f3:7c:1a:e8:10:9f:bd:45:a8:cf:cb:

9f:39:d2:49:f3:36:6f:21:79:fe:1a:af:d0:f8:a1:8e:d8:f2:

30:6d:b5:87:67:0b:54:75:f2:d8:e8:a2:ff:b0:4a:c6:a8:4e:

91:80:e7:dd:66:82:c7:b5:58:eb:3c:64:8a:e7:8f:f2:d1:6f:

e5:b6:66:e7:eb:4f:90:95:34:5b:dd:7f:ff:09:ac:a8:48:17:

ed:7a:78:ad:d1:2b:5d:b5:10:0e:a4:00:12:b9:9f:86:1d:75:

a2:4f:6d:10:0d:b9:10:f3:ae:66:1f:38:4b:3f:98:e4:f9:1e:

e0:10:94:47:e4:8f:f6:5d:51:46:29:90:3b:a5:6e:9f:90:ee:

e4:8f:9a:5e:4c:4f:f1:d5:0c:ad:75:6c:36:f1:31:1a:52:8d:

71:b5:b5:fe:5e:ca:c8:0a:01:30:d6:c8:50:61:c1:e7:03:06:

42:00:cc:57:8a:03:52:54:42:6a:aa:15:6b:df:42:82:00:3a:

62:a0:55:db:4a:ae:54:7a:a5:d1:ac:e0:d8:58:53:95:4c:27:

af:10:2d:c4:ef:f0:d2:c6:3d:a2:36:74:e2:b4:7d:f6:e7:b3:

d2:ee:c5:e0:59:e7:bf:b0:90:aa:5f:c6:0e:e8:18:2e:4d:a2:

d1:2b:ae:06

Обратите внимание, что поля Issuer и Subject одинаковы. Это признак самоподписанного сертификата. Мы успешно сгенерировали самоподписанный сертификат, который будем использовать в качестве сертификата корневого УЦ. Теперь сгенерируем два несамоподписанных сертификата.

Генерирование несамоподписанного сертификата

Сначала сгенерируем пару ключей для сертификата промежуточного УЦ:

openssl genpkey -algorithm RSA -out intermediate\_keypair.pem

.....+...................+.........+...........+....+...+...+++++++++++++++++++++++++++++++++++++++\*..+......+..........+.....+.........+.+..+.+..+.............+.....+.......+.....+++++++++++++++++++++++++++++++++++++++\*.........+.+......+..+.+......+......+.....+....+...+........+....+......+...+..+...+....+...+...+...............+.....+.+...............+...+...+..+......+......+...+...+..........+..+......+.......+.....+..........+........................+..............+.......+.................+.+......+...........+..........+............+........+.........+....+..+.............+.....+....+.....+.......+........+.+.....+.+...........+...+.+...+...+........+.........++++++

...+++++++++++++++++++++++++++++++++++++++\*...+++++++++++++++++++++++++++++++++++++++\*......++++++

Затем сгенерируем CSR-запрос:

openssl req -new -subj “/CN=Intermediate CA” -addext “basicConstraints=critical,CA:TRUE” -key intermediate\_keypair.pem -out intermediate\_csr.pem

Теперь выпустим сертификат промежуточного УЦ и подпишем его закрытым ключом корневого сертификата:

openssl x509 -req -in intermediate\_csr.pem -copy\_extensions copyall -CA root\_cert.pem -CAkey root\_keypair.pem -days 3650 -out intermediate\_cert.pem

Certificate request self-signature ok

subject=CN = Intermediate CA

Посмотрим на выпущенный сертификат промежуточного УЦ:

openssl x509 -in intermediate\_cert.pem -noout -text

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

43:50:3f:93:5a:09:2f:e5:93:c1:e9:46:31:3d:ab:78:ef:23:11:c8

Signature Algorithm: sha256WithRSAEncryption

Issuer: CN = Root CA

Validity

Not Before: Nov 23 09:19:06 2023 GMT

Not After : Nov 20 09:19:06 2033 GMT

Subject: CN = Intermediate CA

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (2048 bit)

Modulus:

00:dd:ea:e5:50:15:c0:d6:73:03:39:d7:64:af:08:

5c:1c:25:30:bd:97:4b:8f:93:7c:27:f7:46:3a:5b:

88:86:2c:4a:8e:b8:32:74:95:16:48:c8:1a:d4:f3:

bf:44:1e:21:08:ee:6a:4e:b5:7c:01:7d:b2:f4:e1:

44:c5:2e:a0:c8:ad:3a:0d:ec:04:12:cf:5a:a7:07:

f7:c9:8d:56:10:c7:3e:f3:5d:01:64:a0:96:c4:db:

87:a1:7f:59:e8:50:29:ba:70:92:97:ef:e8:3f:e1:

9f:c5:18:c7:16:ad:79:f0:3f:cd:7d:86:56:7c:b2:

49:99:42:24:09:8a:8f:af:d6:50:96:2f:f1:aa:78:

9b:80:94:ba:b9:ad:14:9a:81:c2:0c:e9:5b:dd:6c:

f3:41:7d:48:4c:02:65:6c:47:47:bc:74:b1:ca:8a:

dc:13:f3:00:6a:70:cf:96:2c:b7:f0:f2:36:6a:7d:

ba:b6:41:36:d9:2e:fd:db:e7:17:5d:59:80:42:25:

8f:b9:9a:2e:a2:d7:72:d5:ca:e7:a7:4e:41:43:b9:

a7:fb:7f:89:1b:f4:7a:4b:41:08:b8:47:8d:70:0b:

63:4a:0c:ca:95:94:bf:37:26:2a:f4:3d:d3:64:d0:

28:16:91:0e:48:6f:0f:62:9d:e7:5a:d7:7e:b5:a4:

6a:f7

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints: critical

CA:TRUE

X509v3 Subject Key Identifier:

DE:AD:6C:7C:FB:A9:B7:BD:D6:9B:92:0D:9D:D4:8B:3B:63:53:5A:B4

X509v3 Authority Key Identifier:

98:4F:70:73:50:07:97:23:C4:E1:F1:00:31:A7:43:85:DC:04:65:50

Signature Algorithm: sha256WithRSAEncryption

Signature Value:

a1:be:d0:57:27:45:90:dd:7f:3a:d7:df:db:b3:2e:b5:28:dc:

07:bd:d9:0e:5d:3a:29:ae:bf:1e:6f:95:87:00:15:db:e1:86:

0d:f3:aa:cc:3d:27:6d:ba:d0:6a:a0:a9:87:33:1e:e7:38:d8:

9c:c3:0d:4a:92:13:42:69:c2:94:71:88:e6:ef:1c:ea:53:2a:

41:5a:dc:18:20:51:32:32:0e:13:9b:4e:93:10:9e:0f:cb:ac:

4b:06:3b:ae:13:52:dc:90:90:e7:20:16:18:cb:38:ff:90:d4:

5f:13:6e:ae:64:3f:25:29:6a:70:58:2b:ee:95:01:1e:a1:bf:

13:f2:00:54:2f:56:09:be:fa:b9:49:bb:d9:ca:17:79:55:ad:

60:65:df:48:90:a6:fa:9b:8d:b6:97:3f:64:e9:32:60:8b:76:

02:99:52:2b:40:cb:07:38:c4:4b:54:1e:40:28:ec:bb:ce:1e:

43:ba:e2:25:31:a9:48:40:c6:75:36:30:d9:5d:93:f4:43:7b:

37:e3:ae:fa:cc:32:7e:f2:68:f2:b8:b8:46:9b:14:1c:53:3d:

54:d9:60:5b:9e:6d:7a:a3:9e:e5:72:ae:c2:e7:cf:72:c1:8b:

9e:0f:d5:48:6a:cf:96:c8:1f:5f:ea:8d:7c:e8:dd:76:6d:c4:

27:ae:e7:36  
Обратите внимание, что поля Issuer и Subject различны. Это значит, что сертификат несамоподписанный.

Теперь выпустим листовый сертификат и подпишем его закрытым ключом сертификата промежуточного УЦ. Делается это так же, как при выпуске сертификата промежуточного УЦ

openssl genpkey -algorithm ED448 -out leaf\_keypair.pem

openssl req -new -subj “/CN=Leaf” -addext “basicConstraints=critical,CA:FALSE” -key leaf\_keypair.pem -out leaf\_csr.pem

openssl x509 -req -in leaf\_csr.pem -copy\_extensions copyall -CA intermediate\_cert.pem -CAkey intermediate\_keypair.pem -days 3650 -out leaf\_cert.pem

Certificate request self-signature ok

subject=CN = Leaf

На этот раз мы задали CA:FALSE, а не CA:TRUE, потому что листовые сертификаты не должны использоваться для выпуска других сертификатов.

Посмотрим на сгенерированный листовый сертификат:

openssl x509 -in leaf\_cert.pem -noout –text

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

5a:98:90:e1:09:51:05:bb:c9:3e:96:7e:af:c4:2b:8f:27:98:0a:b8

Signature Algorithm: sha256WithRSAEncryption

Issuer: CN = Intermediate CA

Validity

Not Before: Nov 23 09:24:21 2023 GMT

Not After : Nov 20 09:24:21 2033 GMT

Subject: CN = Leaf

Subject Public Key Info:

Public Key Algorithm: ED448

ED448 Public-Key:

pub:

7a:40:68:27:4c:bd:5d:8c:02:46:25:7b:fe:cf:8f:

92:ed:08:d5:84:c2:41:39:3e:0b:5d:ae:7b:e3:ac:

d9:b9:d2:cc:e8:89:ec:6f:8f:b8:d7:69:2b:7d:8d:

c7:b1:8e:7a:d9:d3:e2:12:fe:b7:03:00

X509v3 extensions:

X509v3 Basic Constraints: critical

CA:FALSE

X509v3 Subject Key Identifier:

6A:D1:8F:89:FC:24:D8:7E:48:C9:D9:46:A8:6F:6E:26:5C:2A:69:2C

X509v3 Authority Key Identifier:

DE:AD:6C:7C:FB:A9:B7:BD:D6:9B:92:0D:9D:D4:8B:3B:63:53:5A:B4

Signature Algorithm: sha256WithRSAEncryption

Signature Value:

66:a2:81:dd:c4:ff:e1:6c:78:c2:2a:92:9f:64:c7:7b:2d:d4:

34:d3:ac:4d:a9:e9:ab:a3:be:87:b1:ad:25:a5:1e:13:c6:b5:

ce:ca:30:98:43:ea:48:9a:38:8c:81:2d:51:e0:c4:3a:a3:68:

3c:c1:81:a1:21:6e:68:34:65:18:a0:24:3e:44:3e:d0:ec:92:

de:44:45:f3:70:c8:69:c0:61:2e:6c:1c:3b:52:71:27:09:06:

ec:f7:ac:5b:2a:ed:e5:06:9e:b0:79:97:72:03:ac:cb:80:81:

44:4c:a6:87:f9:55:f8:08:d7:fe:f0:b2:b7:63:fa:dd:37:c2:

ef:21:4e:7d:8d:d8:2d:73:1f:16:a7:34:08:28:8b:3a:72:de:

2e:4e:42:68:9a:56:1e:5f:93:ac:01:2b:ac:61:af:be:a2:0c:

83:3e:54:42:ae:50:86:62:7c:8d:35:fa:ee:4a:ac:d0:1e:72:

c8:55:30:3f:25:00:bf:40:6d:14:97:e8:e4:b6:71:a3:c4:2c:

31:39:26:71:c1:e2:38:68:da:b5:e1:80:f0:78:47:d9:ba:52:

ad:3b:98:24:7a:b9:c5:39:2e:2b:57:c2:3e:c3:30:94:e0:e1:

50:7f:7a:76:92:29:3f:37:ef:96:b0:a0:3e:a7:13:50:3c:96:

f8:16:88:6c

Теперь проверим листовый сертификат с помощью двух других.

Проверка сертификата в командной строке

openssl verify -verbose -show\_chain -trusted root\_cert.pem -untrusted intermediate\_cert.pem leaf\_cert.pem

leaf\_cert.pem: OK

Chain:

depth=0: CN = Leaf (untrusted)

depth=1: CN = Intermediate CA (untrusted)

depth=2: CN = Root CA

Обратите внимание на флаги -trusted и -untrusted. Флаг -trusted задает файл, содержащий один или несколько надежных сертификатов. Чтобы проверить сертификат, openssl verify должна построить цепочку от проверяемого сертификата до надежного. Флаг -untrusted задает файл, содержащий один или несколько ненадежных сертификатов. Ненадежные сертификаты олезны в качестве промежуточных в цепочке. Оба флага можно использовать несколько раз для задания нескольких файлов.