

VPN over QUIC システム (RQST) 設定マニュアル ver 0.10

1. RQST の動作説明	2
2. RQST のインストール	3
2.1 仮想 I/F の設定 (クライアント・サーバ共通)	3
2.2 仮想 I/F の追加 (サーバのみ)	4
2.3 仮想 I/F のメトリックの設定 (クライアントのみ)	4
2.4 RQST の動作テスト	4
3. 証明書の設定	7
3.1 認証局を作成する	7
3.2 サーバの秘密鍵・証明書の作成	10
3.3 クライアントの秘密鍵・証明書の作成	11
3.4 秘密鍵・証明書のコピー	11
3.4 証明書検証モードでの RQST の起動	12
3.5 サービスとしてのサーバの起動	12
4. バイナリのビルド	13
4.1 Visual Studio Community 2019 のインストール	13
4.2 Rust のインストール	13
4.3 NASM のインストール	13
4.4 ビルドの実行	13

1. RQST の動作説明

RQST クライアントから RQST サーバに対して QUIC 接続を確立することで、クライアント・サーバ間に仮想のネットワークを作成します (図 1)。仮想ネットワークを流れるパケットは、クライアント・サーバ間の QUIC 接続を通じて Datagram Frame でカプセル化されてクライアント・サーバに届けられます。仮想ネットワークは仮想 Ethernet として動作する L2 となっているので、サーバ上で DHCP サーバ等を動かすことで、クライアントの仮想 I/F に自動的に IP アドレスを付与することができます。また、サーバをルータとして動作させた上で NAT を行うようにしたり、サーバの仮想 I/F をルータと接続されている I/F とブリッジ接続を行うようにしたりすることで、クライアントがサーバ側のネットワークを通じて通信を行うようにすることができます。

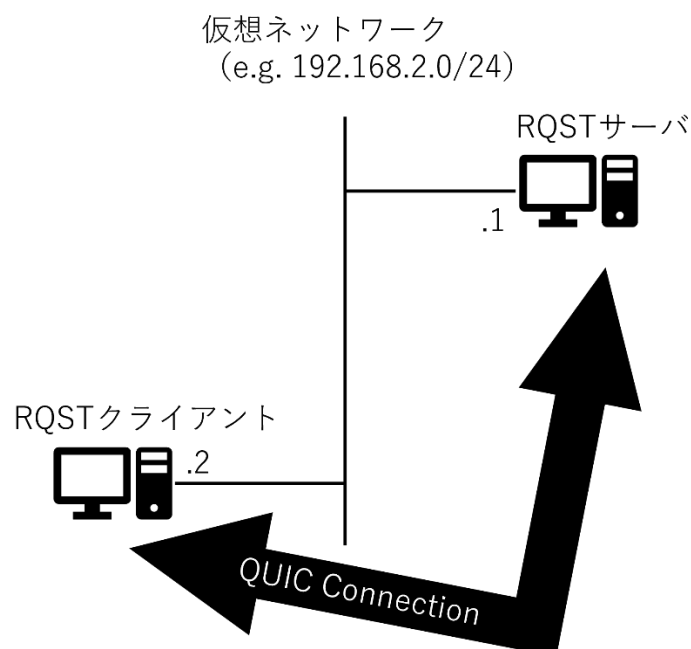


図 1

クライアント・サーバ間の QUIC 接続の確立にあたっては、互いに証明書を検証するようになっており、これにより、クライアントが偽のサーバに接続してしまったり、サーバが無関係の第三者が動作させているクライアントからの接続を許可してしまったりすることが防止されます。もっとも、動作試験のために証明書の検証を省略するモードで動作させることは可能です。

なお、サーバは複数のクライアントの接続を受け付けることができますが、それぞれ独立の仮想 L2 ネットワークが作成されます。そのため、複数のクライアントが接続している状態であってもクライアント同士では直接通信を行うことはできないようになっています。

2. RQST のインストール

2.1 仮想 I/F の設定（クライアント・サーバ共通）

仮想 I/F を作成できるようにするため、OpenVPN Windows 版を[公式サイト](#)からダウンロードしてインストールします。OpenVPN 自体は使用しないため、インストール後にタスクトレイにある”OpenVPN GUI”を右クリックして表示されるメニューから”設定”を選択し、その後表示される設定ウィンドウ（図 2）にある”Windows 起動時に開始”のチェックを外し、”OK”をおしてウィンドウを閉じた後、タスクトレイにある”OpenVPN GUI”を右クリックして表示されるメニューから”終了”を選択して終了させます。

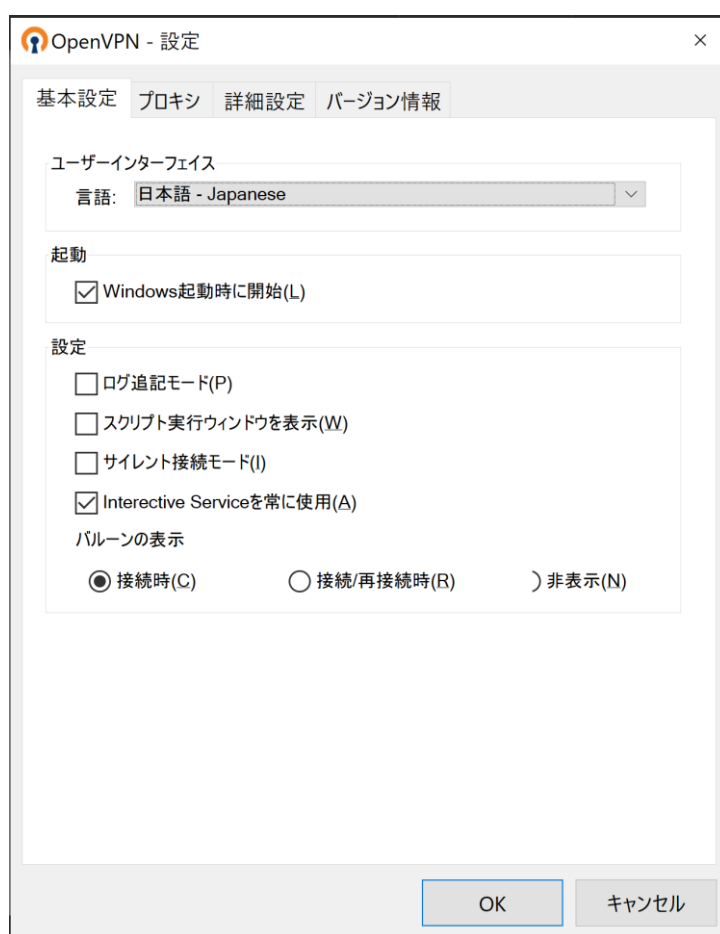


図 2

RQST の最適な動作には仮想 I/F の MTU が 1280 に設定されている必要があります。まず、コマンドプロンプトを管理者権限で実行し、次のコマンドを実行して仮想 I/F の Idx を調べます。対象の仮想 I/F の名前は、デフォルトでは、”OpenVPN Tap-Windows6”です。

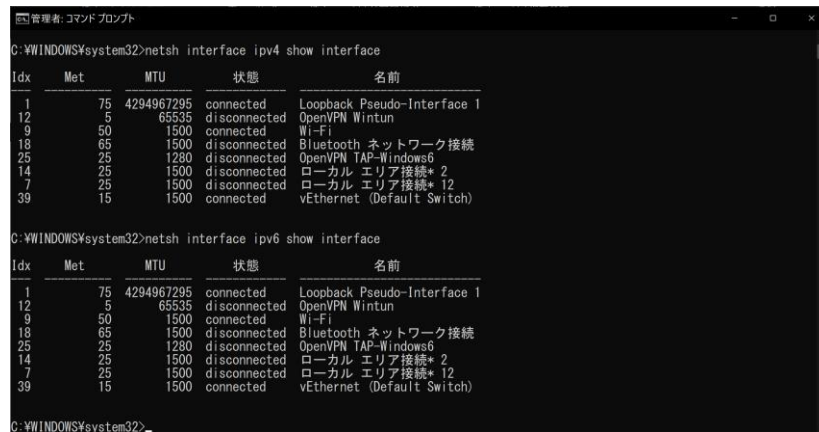
```
C:¥WINDOWS¥system32>netsh interface ipv4 show interface
```

その後、次のコマンドを実行して IPv4/IPv6 のそれぞれで MTU を 1280 に変更します（”25”の部分は先ほど調べた Idx の数値に変更してください）。

```
C:¥WINDOWS¥system32>netsh interface ipv4 set interface 25 mtu=1280
```

```
C:¥WINDOWS¥system32>netsh interface ipv6 set interface 25 mtu=1280
```

最後に確認のため、インターフェース一覧をもう一度表示させます。図 3 のように対象の仮想 I/F の MTU が 1280 と表示されていれば成功です。



Idx	Met	MTU	状態	名前
1	75	4294967295	connected	Loopback Pseudo-Interface 1
12	5	65535	disconnected	OpenVPN Wintun
9	50	1500	connected	Wi-Fi
18	65	1500	disconnected	Bluetooth ネットワーク接続
25	25	1280	disconnected	OpenVPN TAP-Windows6
14	25	1500	disconnected	ローカル エリア接続* 2
7	25	1500	disconnected	ローカル エリア接続* 12
39	15	1500	connected	vEthernet (Default Switch)

Idx	Met	MTU	状態	名前
1	75	4294967295	connected	Loopback Pseudo-Interface 1
12	5	65535	disconnected	OpenVPN Wintun
9	50	1500	connected	Wi-Fi
18	65	1500	disconnected	Bluetooth ネットワーク接続
25	25	1280	disconnected	OpenVPN TAP-Windows6
14	25	1500	disconnected	ローカル エリア接続* 2
7	25	1500	disconnected	ローカル エリア接続* 12
39	15	1500	connected	vEthernet (Default Switch)

図 3

2.2 仮想 I/F の追加（サーバのみ）

サーバが複数のクライアントからの接続を受け付ける場合には仮想 I/F を追加する必要があります。管理者権限のあるコマンドプロンプトで tapctl.exe を次のように実行して必要な数だけ追加します。なお、追加した後は 2.1 の手順に従って MTU を 1280 に設定する必要があります。

```
C:¥WINDOWS¥system32>cd C:¥Program Files¥OpenVPN¥bin
```

```
C:¥Program Files¥OpenVPN¥bin>tapctl create --name "TAP-Windows6 2"
```

2.3 仮想 I/F のメトリックの設定（クライアントのみ）

クライアントが仮想ネットワーク経由でサーバを通じて通信を行うためには、仮想 I/F のメトリックを小さな値に設定しておく必要があります。この設定を行うためには管理者権限のあるコマンドプロンプトで次のようにコマンドを実行します（"25"の部分は 2.1 の場合と同様に対象の仮想 I/F の Idx の数値に変更してください）。

```
C:¥WINDOWS¥system32> netsh interface ipv4 set interface 25 metric=1
```

```
C:¥WINDOWS¥system32> netsh interface ipv6 set interface 25 metric=1
```

2.4 RQST の動作テスト

rqst-bin-v010.zip を適切なフォルダに展開します（以下では C:¥rqst を使用）。クライアントのバイナリとサーバのバイナリはそれぞれ、C:¥rqst¥client¥vpn-client.exe、C:¥rqst¥server¥vpn-server.exe となります。

サーバ上で、ICS の設定を行ってクライアントがサーバを経由して通信できるようにします。サーバの外部との通信に用いるインターフェースのプロパティを開き、共有のページを開きます（図 4）。その後、"ネットワークのほかのユーザーに、このコンピュータのインターネット接続をとおしての接続を許可する（N）"にチェックをいれ、"ホームネットワー

ク接続 (H) ”ではクライアントとの接続に用いる仮想 I/F を選択し、“OK”を押します。

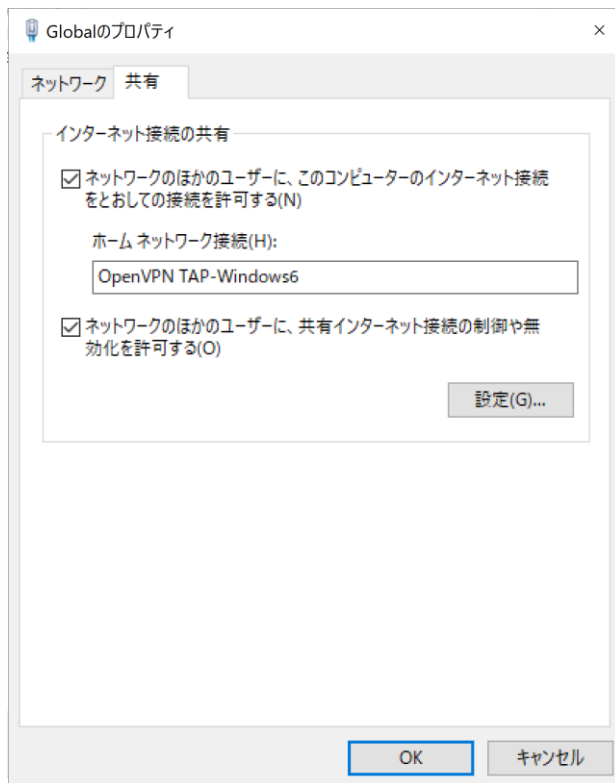


図 4

クライアント上で管理者権限のあるコマンドプロンプトを開き、サーバのアドレス宛の静的経路を追加します。まず、次のコマンドを入力して IPv4 の Nexthop アドレスを調べます。

```
C:\WINDOWS\system32>route print -4 0.0.0.0
```

図 5 のような表示になるので、“IPv4 ルート テーブル”の箇所のゲートウェイのエントリの IP アドレス（ここでは 192.168.11.1）を記録します。

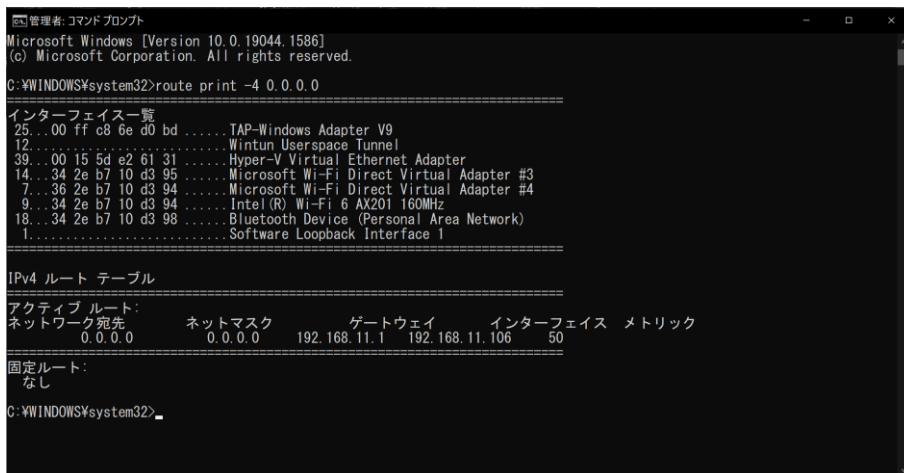


図 5

続いて次のコマンドを入力して IPv6 の Nexthop アドレスを調べます。

```
C:\WINDOWS\system32>route print -6 ::/0
```

図 6 のような表示になるので、“IPv6 ルート テーブル”の箇所のゲートウェイのエントリの IP アドレスおよび IF のエントリの数値（ここでは fe80::271:b9ff:fee2:f776 と 9）を記録します。

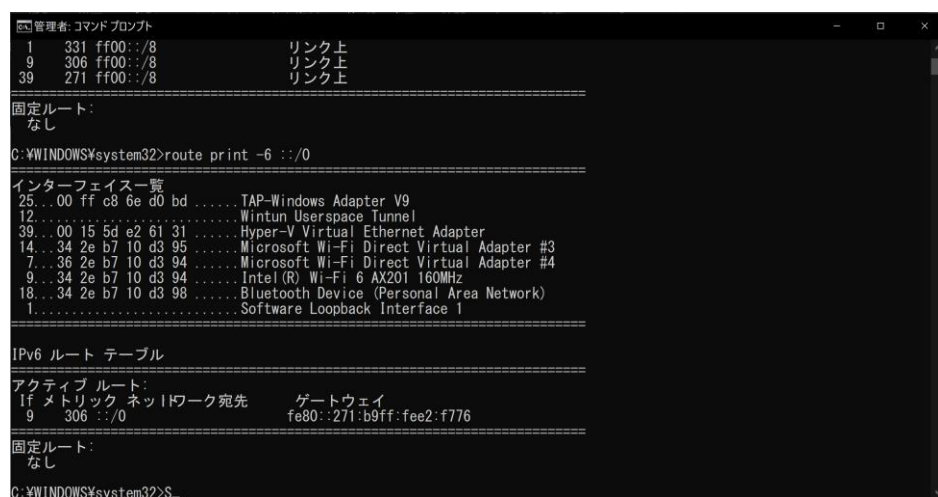


図 6

そして、次のコマンドを入力してサーバのアドレス宛の静的経路を設定します（ここではサーバのアドレスは 203.0.113.1 および 2001:db8::1 とします）。

```
C:\WINDOWS\system32>route add 203.0.113.1 mask 255.255.255.255 192.168.11.106
```

```
C:\WINDOWS\system32>route add 2001:db8::1/128 fe80::271:b9ff:fee2:f776 IF 9
```

サーバおよびクライアントの設定が終わったら、サーバ上で通常のコマンドプロンプトを開き、次のコマンドを入力して vpn-server.exe を証明書の検証を行わないモードで実行します。

```
C:\rqst\server>vpn-server -d
```

続いてクライアント上でもコマンドプロンプトを開き、次のコマンドを入力して vpn-client.exe を証明書の検証を行わないモードで実行します。なお、接続先のサーバ名は適切に変更してください。

```
C:\rqst\client>vpn-client -d vpn://rqst.example.com:3456
```

これで RQST が動作している限り、クライアントの通信はサーバを経由して行われることになります。プログラムを終了するには Ctrl+C を入力してください。

3. 証明書の設定

動作試験が終わったら、認証局を作成し、クライアント・サーバのそれぞれに秘密鍵・証明書を新しく発行します。これらの秘密鍵・証明書を用いると検証を行うモードで RQST を実行できるようになります。

3.1 認証局を作成する

[Download Free OpenSSL for Microsoft Windows \(firedaemon.com\)](https://www.firedaemon.com/openssl/) から Win 用の OpenSSL バイナリを入手し、インストールします。その後、openssl.exe のあるフォルダを環境変数 Path に追加します。

続いて、CA という名前のフォルダを作り（以下では C:\CA とします）、その中で以下の内容で setup.bat ファイルを作ります。

```
@mkdir certs
@mkdir crl
@mkdir newcerts
@mkdir private
@mkdir csr
@type nul > index.txt
@type nul > crlnumber
@echo 1000 > serial
```

そして、通常のコマンドプロンプトを開き次のコマンドを入力して setup.bat を実行します。

```
C:\CA>setup.bat
```

また、以下の内容を C:\CA\openssl.cfg というファイル名で保存します。

```
[ ca ]
default_ca = CA_default

[ CA_default ]
dir           = C:\CA
certs         = C:\CA\certs
crl_dir       = C:\CA\crl
database      = C:\CA\index.txt
new_certs_dir = C:\CA\newcerts
serial        = C:\CA\serial
crlnumber     = C:\CA\crlnumber
crl           = C:\CA\crl.pem
certificate    = C:\CA\certs\ca.crt
private_key   = C:\CA\private\ca.key
```

name_opt	= ca_default
cert_opt	= ca_default
crl_extensions	= crl_ext
default_days	= 365
default_crl_days	= 30
default_md	= sha256
preserve	= no
policy	= policy_match

[policy_match]

countryName	= match
stateOrProvinceName	= optional
organizationName	= optional
organizationalUnitName	= optional
commonName	= supplied
emailAddress	= optional

[policy_anything]

countryName	= optional
stateOrProvinceName	= optional
localityName	= optional
organizationName	= optional
organizationalUnitName	= optional
commonName	= supplied
emailAddress	= optional

[req]

default_bits	= 2048
distinguished_name	= req_distinguished_name
x509_extensions	= v3_ca
string_mask	= utf8only
default_md	= sha256

[req_distinguished_name]

countryName	= Country Name (2 letter code)
countryName_default	= JP


```

countryName_min                = 2
countryName_max                = 2
stateOrProvinceName            = State or Province Name (full name)
stateOrProvinceName_default    = Okayama
localityName                   = Locality Name (eg, city)
localityName_default           = Okayama
0.organizationName              = Organization Name (eg, company)
0.organizationName_default     = SEERA Networks Inc.
organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default = Okayama Office
commonName                     = Common Name (e.g. server FQDN or YOUR
name)
commonName_default             = RQST CA
commonName_max                 = 64
emailAddress                    = Email Address
emailAddress_default           = rqst@example.com
emailAddress_max               = 64

[ server_cert ]
basicConstraints                = CA:FALSE
nsCertType                     = server
nsComment                      = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier           = hash
authorityKeyIdentifier         = keyid,issuer:always
keyUsage                       = critical, digitalSignature, keyEncipherment
extendedKeyUsage                = serverAuth

[ v3_ca ]
subjectKeyIdentifier           = hash
authorityKeyIdentifier         = keyid:always,issuer
basicConstraints                = critical,CA:true
keyUsage                       = critical, digitalSignature, cRLSign, keyCertSign

[ v3_intermediate_ca ]
subjectKeyIdentifier           = hash
authorityKeyIdentifier         = keyid:always,issuer

```

```
basicConstraints      = critical,CA:true, pathlen:0
keyUsage              = critical, digitalSignature, cRLSign, keyCertSign

[ crl_ext ]
authorityKeyIdentifier = keyid:always
```

次に、コマンドプロンプトに次のコマンドを入力し、認証局の秘密鍵を作成します。実行した後は秘密鍵を暗号化するパスワードを入力し、Enter キーを押します。

```
C: ¥CA>openssl genrsa -aes256 -passout stdin -out private¥ca.key 4096
```

そして、コマンドプロンプトに次のコマンドを入力し、認証局の証明書を発行します。実行後はまず秘密鍵のパスワードの入力を待つ状態になっているので、パスワードを入力して Enter キーを押します。その後、認証局の情報を適宜入力します。

```
C: ¥CA>openssl req -config openssl.cfg -key private¥ca.key -passin stdin -new -x509 -days
9999 -sha256 -extensions v3_ca -out certs¥ca.crt
```

3.2 サーバの秘密鍵・証明書の作成

以下の内容を（*_default を適宜変更した上で）C:¥CA¥certreq.cfg に保存します。

```
[ req ]
default_bits          = 2048
distinguished_name = req_distinguished_name

[ req_distinguished_name ]
countryName            = Country Name (2 letter code)
countryName_default    = JP
stateOrProvinceName    = State or Province Name (full name)
stateOrProvinceName_default = Okayama
localityName           = Locality Name (eg, city)
localityName_default   = Okayama
organizationName       = Organization Name (eg, company)
organizationName_default = SEERA Networks Inc.
commonName             = Common Name (e.g. server FQDN or YOUR name)
commonName_default     = rqst.example.com
```

コマンドプロンプトに以下のコマンドを入力し、サーバの秘密鍵を作成します。秘密鍵にはパスワードを設定しないので入力の必要はありません。

```
C: ¥CA>openssl genrsa -out private¥server.key 2048
```

コマンドプロンプトに以下を入力し、サーバの証明書要求を発行します。なお、commonName には RQST のサーバを運用する予定の FQDN のどれかを指定し、subjectAltName には RQST のサーバを運用する予定の FQDN をすべて列挙します（ここでは、rqst.example.com と rqst4.example.com の両方を用いる例とします）。

```
C: ¥CA>openssl req -config certreq.cfg -addext "subjectAltName=DNS:
rqst.example.com,DNS: rqst4.example.com" -key private¥server.key -new -sha256 -out
csr¥server.csr
```

次に subjectAltName に指定した内容に応じて次のような内容で san.txt を作成します。

```
subjectAltName = DNS: rqst.example.com,DNS: rqst4.example.com
```

そしてコマンドプロンプトに以下を入力し、サーバ証明書を発行します。実行後はまず秘密鍵のパスワードの入力を待つ状態になっているので、パスワードを入力して Enter キーを押します。

```
C: ¥CA>openssl ca -config openssl.cfg -passin stdin -days 5000 -notext -md sha256 -in
csr¥server.csr -extfile san.txt -out certs¥server.crt
```

3.3 クライアントの秘密鍵・証明書の作成

コマンドプロンプトに以下のコマンドを入力し、クライアントの秘密鍵を作成します。秘密鍵にはパスワードを設定しないので入力する必要はありません。

```
C: ¥CA>openssl genrsa -out private¥client.key 2048
```

次にコマンドプロンプトに以下を入力し、クライアントの証明書要求を発行します。なお、commonName には何を入力してもかまいません。

```
C:¥CA>openssl req -config certreq.cfg -key private¥client.key -new -sha256 -out
csr¥client.csr
```

そしてコマンドプロンプトに以下を入力し、クライアント証明書を発行します。実行後はまず秘密鍵のパスワードの入力を待つ状態になっているので、パスワードを入力して Enter キーを押します。

```
C: ¥CA>openssl ca -config openssl.cfg -passin stdin -days 5000 -notext -md sha256 -in
csr¥client.csr -out certs¥client.crt
```

3.4 秘密鍵・証明書のコピー

サーバ上で認証局の証明書（ca.crt）、サーバの秘密鍵・証明書（server.key, server.crt）をバイナリと同じフォルダ（C:¥rqst¥server）に上書きコピーします。また、クライアント上で認証局の証明書（ca.crt）、クライアントの秘密鍵・証明書（client.key, client.crt）をバイナリと同じフォルダ（C:¥rqst¥client）に上書きコピーします。

3.4 証明書検証モードでの RQST の起動

サーバ上でコマンドプロンプトを開き、次を入力して実行します。

```
C:\rqst\server>vpn-server
```

続いてクライアント上でもコマンドプロンプトを開き、次のコマンドを入力して実行します。なお、接続先のサーバ名はサーバ証明書作成時に指定した FQDN のどれかにする必要があります。

```
C:\rqst\client>vpn-client vpn://rqst.example.com:3456
```

3.5 サービスとしてのサーバの起動

RQST サーバは次のように設定することで Windows のサービスとしても実行することができます。

まず、管理者権限のあるコマンドプロンプトを開き、次のように入力してサービスとしてインストールします。

```
C:\rqst\server>vpn-server.exe install
```

そしてコマンドプロンプトに次のように入力をして、サービスを起動します。

```
C:\rqst\server>sc start quic_vpn_server
```

停止する場合は次のように入力します。

```
C:\rqst\server>sc stop quic_vpn_server
```

4. バイナリのビルド

RQST は Rust で書かれており、以下のようにビルドすることができます。ソースコードは `rqst-src-v010.zip` を適切なフォルダに展開して配置してください（以下では、`C:\¥rqst-src` を使用）。

4.1 Visual Studio Community 2019 のインストール

Visual Studio Community 2019 を [MS のサイト](#) からダウンロードしてインストールします。インストールの際、ワークロードでは"C++によるデスクトップ開発"を選択し、個別のコンポーネントでは"Windows 用 C++ Cmake ツール"を追加し、言語パックでは"日本語"と"英語"を追加します。

4.2 Rust のインストール

`rustup` を [公式サイト](#) からダウンロードし、実行してインストールを行います。

4.3 NASM のインストール

NASM を [公式サイト](#) からダウンロードしてインストールします。また、バイナリのインストール先を環境変数の Path に追加します。

4.4 ビルドの実行

スタートメニューから"x64 Native Tools Command Prompt for VS 2019"を選択して開発環境のコマンドプロンプトを立ち上げます。その後、RQST のソースコードを展開したフォルダに移動した後、次のコマンドを入力してビルドを行います。

```
C:\¥rqst-src>cargo build --release
```