

UNIZETO



GENERAL  
CERTIFICATION AUTHORITY



**WebNotarius<sup>®</sup>**

# WebNotarius<sup>®</sup> service communication technical specification

ver 1.1

# Contents

1.	INTRODUCTION.....	3
1.1	IMPLEMENTATION OF TRANSACTION IN DVCS PROTOCOL .....	3
2.	SOAP PROTOCOLE .....	4
2.1	<i>Verification of certificate validity (VPKC)</i> .....	4
2.2	<i>Verification of signature validation (VSD)</i> .....	5
3.	DEDICATED API LIBRARIES.....	7
4.	DEDICATED SOFTWARE.....	8

## 1. Introduction

Connection to the WebNotarius<sup>®</sup> service requires a proper verification request, compatible with DVCS protocol described in RFC 3029 (Internet X.509 Public Key Infrastructure - Data Validation and Certification Server Protocols.).

### 1.1 Implementation of transaction in DVCS protocol

DVCS protocol transaction is initiated by a subscriber of WebNotarius<sup>®</sup> service. Subscriber creates a request which contains data for which verification will be performed (it may be the correctness of e-signature or validity of public key certificate). Request is signed by subscriber and in next step is sent to the WebNotarius server using HTTP protocol.

WebNotarius server, after receiving the request, verifies its signature and in next step performs verification (or not in case request signature is not valid). If the request is properly constructed, server performs, suitable for sent request, operations, whose product is a verification evidence. Verification evidence is sent to the subscriber. If the request is invalid, the server sends to the WebNotarius subscriber information about the error that occurred. In both cases the response is signed using a CMS structure - SignedData.

At the moment, connection to the WebNotarius<sup>®</sup> service is possible using web-service through SOAP protocol, APIs or dedicated software. The following paragraphs describes listed ways.

## 2. SOAP protocole

### 2.1 Verification of certificate validity (VPKC)

#### 2.1.1 Input structure

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://soap.dvcs.services.pki/">
<soapenv:Header/>
<soapenv:Body>
  <soap:verifyCertificate>
    <CertificateRequest>
      <certificate>
        <certificateValue> Base64 Certificate </certificateValue>
      </certificate>
    </CertificateRequest>
  </soap:verifyCertificate>
</soapenv:Body>
</soapenv:Envelope>

```

#### 2.1.2 Output structure

```

<env:Envelope
xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
<env:Header>
<wsse:Security> Signature under the response </wsse:Security>
</env:Header>
<env:Body>
<soap:verifyCertificateResponse
xmlns:soap="http://soap.dvcs.services.pki/">
  <Response>
    <certStatus>
      <issuer>String describing certificate issuer
    </issuer>
    <serial>Certificate serial number </serial>
    <status> Certificate status </status>
    <statusInfo>Certificate info <statusInfo>
    <subject> String containing certificate owner
    info</subject>

```

```

    </certStatus>
    <response>Base64 DVCS Server response </response>
    <responseTime>Time of response</responseTime>
    <serialNumber>Serial number of response
</serialNumber>
    <serviceType>Validation of Public Key
Certificates</serviceType>
    <status>Response status</status>
  </Response>
</soap:verifyCertificateResponse>
</env:Body>
</env:Envelope>

```

## 2.2 Verification of signature validation (VSD)

### 2.2.1 Input structure

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://soap.dvcs.services.pki/">
  <soapenv:Header/>
  <soapenv:Body>
    <soap:verifySignature>
      <SignatureRequest>
        <message>
          <message>Optional field for input in Base64 for external
signatures</message>
        </message>
        <signature>
          <value>Field for signature in Base 64</value>
        </signature>
      </SignatureRequest>
    </soap:verifySignature>
  </soapenv:Body>
</soapenv:Envelope>

```

## 2.2.2 Output structure

```

<env:Envelope
xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
<env:Header>
  <wsse:Security>Signature under response</wsse:Security>
</env:Header>
<env:Body>
  <soap:verifySignatureRespons
xmlns:soap="http://soap.dvcs.services.pki/">
    <Response>
      <certStatus>
        <issuer> String describing certificate issuer </issuer>
        <serial>Certificate serial number</serial>
        <status> Obtained certificate status </status>
        <statusInfo> Certificate status info </statusInfo>
        <subject> String containing certificate owner info
        </subject>
      </certStatus>
      <response> Base64 DVCS Server response </response>
      <responseTime> Response time </responseTime>
      <serialNumber> Response serial number </serialNumber>
      <serviceType>Validation of Digitally Signed
      Document</serviceType>
      <status> Status of response </status>
    </Response>
  </soap:verifySignatureRespons>
</env:Body>
</env:Envelope>

```

### 3. Dedicated API libraries

With DVCS service there a possibility to connect using a dedicated API libraries based on the Java Virtual Machine (JDK 1.3 onwards). Classes supporting the creation of requests, issuing credentials, sending requests for interpreting services, and server responses are in the package `pki.services.dvcs.structures.api`. These includes in particular `VSDService` and `VPKCSservice`, responsible for handling verification services requests for validity of electronic signatures and certificates. Below is presented an example of using class `VSDService`:

```
VSDService service = new VSDService(
    "http://localhost:8080/dvcserver", null, null);

int res = service.executeVSDService(xml, null);

if (res == PKIStatus.PKISTATUS_GRANTED) {
    System.out.println("STATUS: CORRECT!");
} else if (res == PKIStatus.PKISTATUS_REJECTION) {
    System.out.println("STATUS: REJECTED!");
    ArrayList errors = service.getErrorMessage();
    for (Iterator iter = errors.iterator(); iter.hasNext();) {
        String element = (String) iter.next();
        System.out.println(element);
    }
} else if (res == VSDService.DVCS_ERROR_RESPONSE) {
    System.out.println("WRONG REQUEST!");
    ArrayList errors = service.getErrorMessage();
    for (Iterator iter = errors.iterator(); iter.hasNext();) {
        String element = (String) iter.next();
        System.out.println(element);
    }
} else {
    System.out.println("SYSTEM ERROR!");
    ArrayList errors = service.getErrorMessage();
    for (Iterator iter = errors.iterator(); iter.hasNext();) {
        String element = (String) iter.next();
        System.out.println(element);
    }
}
```

## 4. Dedicated Software

Connection to the WebNotarius service can be conducted using software proCertumDVCSClient. The software is a standalone application, requiring the installation on your computer. More information on this product can be obtained after contacting CERTUM through [www.webnotarius.pl](http://www.webnotarius.pl) and [www.webnotarius.eu](http://www.webnotarius.eu) websites.

In addition, verification can be conducted using [www.webnotarius.pl](http://www.webnotarius.pl) and [www.webnotarius.eu](http://www.webnotarius.eu) websites through WebNotarius web application (available on all most popular web browsers). This solution allows for processing verification using intuitive and easy in use web interface, which allows to receive verification status and electronic evidence of verification from WebNotarius<sup>®</sup> service.