# Domain Control Validation

Version History

1.00 Original version.
1.01 Minor updates.
1.02 Added: MDCs/UCCs supported by Email DCV.
Added: HTTP and DNS CNAME supported via web-interface.
1.03 Added: MDCs/UCCs supported via HTTP and CNAME DCV.
Updated: Re-issue conditions.
1.04 Added: MDCs/UCCs extended DCV status via CollectSSL.
1.05 Minor updates.
1.06 Minor updates.
1.07 Added: detail on ResendDCVEmail working for non-email DCV too(!)
1.08 (not published) Addition of new (CABF Ballot 169) methods
1.09 Simplified (from 1.08) for implementation and added IP address method (per BR 3.2.2.4.8)

## 1. Introduction

No Comodo server authentication certificate may be issued until we have validated the Applicant's control of the domain(s) to appear in the subject of the certificate.
We refer to this process as DCV (Domain Control Validation).

Up to (and including) version 1.09 of this document, DCV could be performed using one of three methods:

- Email challenge-response
- Creation of a file on the domain's HTTP server
- Creation of a DNS CNAME record for that domain

The details of how these methods are performed have changed as of this version 1.09. These changes are in response to the revisions to the "Validation of Domain Authorization or Control" section of the CA/B Forum's "Baseline Requirements" document, including those changes that are in the ballot process or that are otherwise expected to have been balloted and included by the time our implementation is completed.

This document deals with the automated methods to validate control of a domain. There are other methods which involve legal documents being exchanged that are legal and permitted within the terms of our CPS and CABF Baseline requirements but which are not further mentioned in this document.

## 2. Authorization Domain Name

When you request a certificate from us, you tell us one or more fully qualified domain names (FQDNs) that you want to see in your certificate.
For each FQDN, the “Authorization Domain Name” is the domain name that you use to do the DCV.

The Authorization Domain Name can be the same as the FQDN to be validated, or the Authorization Domain Name can be the FQDN with some labels removed from the beginning.

E.g. If the FQDN to be included in the certificate is internal.example.com, the Authorization Domain Name could be either
internal.example.com or
example.com

One consequence of this is that if you request a certificate for the 2 domains (www.example.com, example.com), an Authorization Domain Name of example.com may be used to do a single validation that will validate both FQDNs, whereas an Authorization Domain Name of www.example.com may not.

An Authorization Domain Name cannot be a registry controlled name and cannot be a public suffix.
That means that ‘co.uk’ or ‘com’ can’t be Authorization Domain Names, and (e.g.) ‘pvt.k12.ma.us’ can’t be an Authorization Domain Name because it is a public suffix.

For E-Mail based DCV (section 3, below), you will tell us explicitly which domain you want to use as the authorization domain.

For DNS-based DCV we will check all of the possible Authorization Domain Names.

The Authorization Domain Name never contains a wildcard ('*') character so even if the FQDN contains a wildcard character, the Authorization Domain Name will not.
E.g. If the FQDN to be included in the certificate is *.service.example.com, the Authorization Domain Name could be either
example.com or
service.example.com

## 3. Email Challenge-Response

When the order is placed, an email address is selected from a shortlist of acceptable options. An email is sent to that address, containing a unique validation code.
The email should be received by someone in control of the domain, where they can follow a link provided in the email and enter the validation code, thus proving domain control.

The list of acceptable email addresses for any given domain are:

- admin@
- administrator@
- hostmaster@
- postmaster@
- webmaster@
- Any Admin, Registrant, Tech or Zone contact email address that appears on the domain's WHOIS record, and is visible to our CA system.

When ordering a certificate through Comodo's web-interface

The web-interface certificate request process will default to offering email challenge-response DCV. The list of acceptable email addresses will be displayed for selection, based on the common name extracted from the CSR.

When ordering a certificate through Comodo's API

Once the CSR is received from the customer, the FQDN of the common name (CN) from the CSR must be extracted. The DecodeCSR API can be used, or any other method you have available.
The FQDN may then be passed to the GetDCVEmailAddressList API, which will return a complete list of acceptable email addresses for that FQDN.
ONLY email addresses returned from GetDCVEmailAddressList are acceptable. If an email address believed to be on the WHOIS record for the domain is not returned, this means our system was unable to extract it from a WHOIS query, and thus the address cannot be used.

Then, once a choice is made from the acceptable email address list, that address can be passed to the AutoApplySSL API, as the dcvEmailAddress parameter. Once the call is made to AutoApplySSL with this parameter, the DCV email will be sent.

Due to our caching of the WHOIS record result, the API call to AutoApplySSL should be made within 24 hours of the GetDCVEmailAddressList API call.
A call to GetDCVEmailAddressList is not required if the dcvEmailAddress selection is from the 5 'default' email addresses (i.e. not one extracted from the domain's WHOIS record).
Even if the email address passed to dcvEmailAddress is to be one from the domain's WHOIS record, we do not insist that you call GetDCVEmailAddressList. Our only requirement is that the email address you choose would be returned from GetDCVEmailAddressList.

*For multi-domain certificates, please see the information in Section 9.*

The unique validation code is only valid for 30 days. I.e. any attempt to use the unique validation code more than 30 days after it was created will fail.

## 4. HTTP Based DCV

HTTP based DCV requires that a HTTP server be running on port 80 or that an HTTPS server be running on port 443 of the Authorization Domain Name.
We follow CNAMEs when completing HTTP based DCV.
We look for the file at every valid Authorization Domain, i.e. we start with the FQDN and then we will strip one or more labels from left to right in the FQDN and will look for the file on each intermediate domain.

Two hashes of the CSR are generated before submission to Comodo. A plain text file is created on the HTTP/S server of the Authorization Domain Name, with one hash as the filename, and one hash within the text file itself.
We call this text file the Request Token. The content of the Request Token is described in more detail in sections 7 and 8, below.

For example:
A CSR is generated with the CN=www.example.com
The Authorization Domain Name will be example.com
The CSR is hashed using both the MD5 and SHA-256 hashing algorithms.

A text file is created, containing the SHA-256 hash and the domain 'comodoca.com' on the next line.

*c9c863405fe7675a3988b97664ea6baf442019e4e52fa335f406f7c5f26cf14f*
*comodoca.com*

The file is then named in the format: <MD5 hash>.txt and placed in the /.well-known/pki-validation directory of the HTTP server, like so:

http://example.com/.well-known/pki-validation/C7FBC2039E400C8EF74129EC7DB1842C.txt

Once the order is received by Comodo and HTTP based DCV is specified, the Comodo CA system checks for the presence of the text file and its content. If the file is found and the hash values match, domain control is proven.

When ordering a certificate through Comodo's web-interface
The hash values are calculated and presented via the web-interface during the order process. They are on the same page as the DCV email-address options.
Both the MD5 and SHA-256 hash values of the CSR are shown, and must be saved to the file served from your HTTP server as above before continuing with the order.

When ordering a certificate through Comodo's API
Generate the hashes from the CSR before the order is submitted to Comodo.
The hashes MUST be generated from the DER-encoded (i.e. binary) version of the CSR – not the base64 PEM encoded version. Variations in the PEM encoding can cause differing hash values, whereas the hashes of the DER encoded version will remain constant.
The file must be created using the UPPERCASE formatting of the MD5 hash, as most HTTP servers are case-sensitive. The Comodo CA system will only look for the UPPERCASE hash filename.
The file must be created with a .txt extension.
The SHA-256 hash within the file is case-insensitive.

When the AutoApplySSL call is made, an additional parameter must be specified to indicate use of HTTP/S based DCV.
This parameter is called 'dcvMethod' and must be set to the UPPERCASE value:
HTTP_CSR_HASH for HTTP on port 80 or
HTTPS_CSR_HASH for HTTPS on port 443.

*For multi-domain certificates, please see the information in Section 9 (Notes).*

## 5. DNS CNAME Based DCV

DNS CNAME based DCV requires the creation of a unique CNAME record, pointed back to Comodo.
We look for the CNAME at every valid Authorization Domain, i.e. we start with the FQDN and then we will strip one or more labels from left to right in the FQDN and will look for the CNAME on each intermediate domain.

E.g. for a certificate request for an FQDN of *.mail.internal.example.com, we would look for the CNAME in these places and in this order:

mail.internal.example.com
internal.example.com
example.com

The Authorization Domain Name is the one we find it on.

Two hashes of the CSR are generated before submission to Comodo. A CNAME DNS record is created under the Authorization Domain Name.
We call the content of the CNAME the Request Token. The content of the Request Token is described in more detail in sections 7 and 8, below.

The format of the CNAME will be:

*'_' <MD5 hash>.Authorization Domain Name CNAME <SHA-256 hash>.[<uniqueValue>.]comodoca.com*

Note the leading underscore at the start of the MD5 hash.

For example:
A CSR is generated with the CN=www.example.com
The CSR is hashed using both the MD5 and SHA-256 hashing algorithms.

MD5: c7fbc2039e400c8ef74129ec7db1842c
SHA-256: c9c863405fe7675a3988b97664ea6baf442019e4e52fa335f406f7c5f26cf14f

To perform DNS CNAME based DCV, the following DNS CNAME record may be created before submitting the order:

*_c7fbc2039e400c8ef74129ec7db1842c.example.com CNAME*
*c9c863405fe7675a3988b97664ea6baf.442019e4e52fa335f406f7c5f26cf14f.comodoca.com*

Then the request is submitted to Comodo, the presence of this CNAME DNS record is checked, and if found, domain control is proven.

<u>When ordering a certificate through Comodo's web-interface</u>
The hash values are calculated and presented via the web-interface during the order process. They are on the same screen as the DCV email-address options.
Both the MD5 and SHA-256 hash values of the CSR are shown, and must be added to DNS as a CNAME record as the above instructions show before continuing with the order.

<u>When ordering a certificate through Comodo's API</u>
The hashes are generated from the CSR before the order is submitted to Comodo.
The hashes MUST be generated from the DER-encoded (i.e. binary) version of the CSR – not the base64 PEM encoded version. Variations in the PEM encoding can cause differing hash values, whereas the hashes of the DER encoded version will remain constant.
The DNS CNAME record is then created in the format above.

When the AutoApplySSL call is made, an additional parameter must be specified to indicate use of CNAME based DCV.
This parameter is called 'dcvMethod' and must be set to the UPPERCASE value 'CNAME_CSR_HASH'.
*For multi-domain certificates, please see the information in Section 9 (Notes).*

## 6. IP Address Based DCV

IP Address based DCV requires that a DNS lookup for A records for the FQDN resolves to an IP address over which the applicant has control.

Comodo will obtain documentation of IP address assignment from the Internet Assigned Numbers Authority (IANA) or a Regional Internet Registry (RIPE, APNIC, ARIN, AfriNIC, LACNIC).

Comodo records the IP ranges over which the applicant has control and initially validates as a manual process that the documentation available confirms that the applicant actually has control of those ranges.
The relevant range(s) of IP address must have been validated and recorded against the applicant's account BEFORE the certificates which are to rely on this DCV method are requested.

When the AutoApplySSL call is made, an additional parameter must be specified to indicate use of IP address based DCV.
This parameter is called 'dcvMethod' and must be set to the UPPERCASE value 'IP_ADDRESS_PRE'.

## 7. Request Tokens

A Request Token is the pieces of text that we ask you to put in a file on your webserver, or in a DNS response, to indicate to us that this server really is under your control.
There are 2 pieces of information that are always in a Comodo validation token and, if you're a high-volume user or if you re-use public keys a lot then there will probably be a third.
a) (required) The SHA-256 hash of your PKCS#10 CSR.
The hash MUST be generated from the DER-encoded (i.e. binary) version of the CSR – not the base64 PEM encoded version. Variations in the PEM encoding can cause differing hash values, whereas the hashes of the DER encoded version will remain constant.
You MUST use a hex (base 16) representation of the hash.
b) (required) the string 'comodoca.com'
This isn't just because we like to see our name in lights. This ensures that the Request Token has been generated for Comodo to validate and thereby helps to prevent a possible replay attack on the validation process.
c) (optional) uniqueValue
This is an alphanumeric value that may be up to 20 characters long.
It is generated by you, the applicant, and you will need to pass this same uniqueValue to us when you submit the CSR.

It is not permissible to re-use the same Request Token for more than one certificate.
This means that a subsequent order, e.g. a renewal, for a certificate which uses exactly the same CSR as the original order will fail.

There are several ways that you can ensure that each CSR (and therefore each Request Token) is unique.

a) You could use a fresh key-pair for every request.
b) You could request a different list of domain names in the CSR. E.g. a CSR for (a.example.com, b.example.com) will be different from a CSR using the same key but with the order of the domains changed to be (b.example.com, a.example.com).
c) You can include other material in the CSR which is not to be part of the certificate but whose presence will be sufficient to ensure the uniqueness of the CSR.
   E.g. a challenge password may be included in the CSR as an attribute. This is trivially done using the OpenSSL command line tool and specifying a challenge password when prompted, but may also be implemented in any other way to include a challenge password (or other content) as an Attribute in accordance with RFC2986.

If none of those work for you then you may specify the additional unique element to the Request Token by incorporating a uniqueValue into the request token as described above.
Remember, you only need add this extra unique element if you intend to use the same CSR with multiple certificates.

## 8. Format of Request Tokens and examples

Please note that the content of the Request Token is defined in the prior section in this document, section 7 "Request Tokens".
Please note that in the following examples the MD5 hash value is always hex encoded and must always be upper-cased.

**a) HTTP based DCV**
When using HTTP based DCV, the Request Token should appear as successive lines in a text file.
The file should be entirely composed of characters in the US 7 bit ASCII set.
There should be no BOM (Byte Order Mark) at the start of the file.
Successive lines in the text file may be separated either by single linefeed (LF, '\n', 0x0A) characters,
or by carriage return + linefeed (CRLF, '\r\n', 0x0D 0x0A) pairs.

The URL to be checked is formed in this way:

http[s]://<Authorization Domain Name>/.well-known/pki-validation/<MD5 hash>.txt

E.g. http://example.com/.well-known/pki-validation/C7FBC2039E400C8EF74129EC7DB1842C.txt might contain:
*c9c863405fe7675a3988b97664ea6baf442019e4e52fa335f406f7c5f26cf14f*
*comodoca.com*
*10af9db9tu*

Note that third line (10af9db9tu), the optional uniqueValue. If you're not supplying a uniqueValue then omit that third line.

**b) DNS CNAME based DCV**
When using DNS CNAME based DCV, the Request Token should appear as successive labels in the CNAME RDATA (i.e. the right hand side of your DNS CNAME definition).
The format is always of the form:

'_' <MD5 hash>.<Authorization Domain Name> CNAME <SHA-256 hash>.[<uniqueValue>.]comodoca.com.

Be aware that a hex (base-16) encoded SHA-256 hash will not fit in a single DNS label because it is too long. The SHA-256 hash should therefore be split into two labels, each 32 characters long.
E.g. #1, using hex (base-16) encoding, and splitting the SHA-256 hash into two labels (i.e. inserting a '.' In the middle of the SHA-256 hash value), a DNS record may be setup as:

_c7fbc2039e400c8ef74129ec7db1842c.example.com. CNAME
c9c863405fe7675a3988b97664ea6baf.442019e4e52fa335f406f7c5f26cf14f.comodoca.com.

E.g. #2, using hex (base-16) encoding, and splitting the SHA-256 hash into two labels (i.e. inserting a '.' In the middle of the SHA-256 hash value), and including a uniqueValue, a DNS record may be setup as:

_c7fbc2039e400c8ef74129ec7db1842c.example.com. CNAME
c9c863405fe7675a3988b97664ea6baf.442019e4e52fa335f406f7c5f26cf14f.10af9db9tu.comodoca.com.

## 9. Notes

All certificate types (single, wildcard, MDC/UCC) can be validated with any of the available DCV mechanisms. Multi-domain certificates can use a different mechanisms for each FQDN in the request.

Re-issuing
Re-issues of the certificates require re-validation.

The re-issue does *not* require revalidation of already-validated FQDNs *if the same private key is used to generate the CSR for re-issue*. If a new private key is used to generate the CSR, then the order must have DCV re-performed by one of the available methods for all FQDNs in the request before the certificate can be issued.
This will also apply to re-issues that facilitate the addition or removal of domains for multi-domain certificates.

Re-sending DCV Emails
DCV emails can be resent from within the web-interface, or via the API 'ResendDCVEmail'.
This will resend the DCV email for single-certificate orders, and for multi-domain certificate orders all outstanding (i.e. unvalidated) FQDNs the emails will be resent.

Where a non-email DCV method has been chosen for a domain (as described in sections 4, and 5), the ResendDCVEmail API may be used to reset the frequency at which the practical control is tested.

www. sub-domains
We no longer consider proof of control of 'www.DOMAIN' as also proving control of 'DOMAIN'.
Previously, if you ordered a certificate from us for the 2 FQDNs (www.example.com and example.com), and validated e.g. using HTTP_CSR_HASH on www.example.com, we took that to also demonstrate control of example.com. That is no longer the case.

It remains the case that validating control of example.com is sufficient for the validation of a certificate to contain both example.com and www.example.com.

Multi-domain certificates
Multi-domain certificates (MDCs, UCCs) require DCV for all domain names in all orders. Any of the available mechanisms (email, HTTP , and DNS CNAME) can be used. The web-based interface for this is provided once the order is placed. Simply login to your account and locate the order.
You will be presented with a screen that allows for selection of any valid email address to validate each FQDN in the certificate. Our system will run WHOIS lookups for all domains to provide the email addresses scraped from those records where possible. You can use the 'Refresh' button to update the data on screen – large numbers of FQDNs in a single certificate will take some time for all WHOIS records to be read.
The web interface can also be used to choose the HTTP, HTTPS, or DNS CNAME mechanisms for each FQDN.

You can remove some FQDNs from the certificate if they are unable to be validated, and issue the certificate with only the domains validated to that point.

Multi-domain certificate API Details
The API can be used to request and validate multi-domain certificates via email-based DCV.
The changes to the existing API and processes for ordering are:

- Instead of the 'dcvEmailAddress' parameter which we use for single-domain certificates, there is a 'dcvEmailAddresses' (note the plural) parameter.

This parameter accepts a list of email addresses to use for DCV. There must be one email address per FQDN in the 'domainNames' parameter, and they must be in exactly the same order.
Unlike single certificate orders via the API, our system will *not* reject orders because of incorrect DCV email addresses. They will be accepted, but no email will be sent. They can then be edited via the web-interface.
It is important to pass only valid email addresses for the DCV email address for each domainName.
Valid email addresses can be obtained by either using one of the default 5 (listed earlier in this document), or by calling the GetDCVEmailAddressList API for that domain.
It is also possible for you to perform and independent WHOIS lookup and send an email you can extract from the output to our API. However, please note that our system can only 'see' email addresses on a WHOIS lookup that is from the command-line. Email addresses visible from web-based WHOIS queries, or any that require human challenge-response systems (e.g. CAPTCHAS) will not be usable by our system, and the order will sit awaiting verification until you login and update the DCV

email address via the web-interface.

Our system will attempt to send as few emails as possible. Where several FQDNs exist within the same registered domain name, one email will be sent.
The API can also be used to validate domains using the HTTP and DNS CNAME mechanisms.
As above, the 'dcvEmailAddresses' parameter should be used. For each domain in the 'domainNames' parameter, in order, you can specify one of the following:

- A DCV email address as detailed in this document.
- The value '**HTTPCSRHASH**' – and for this 'domainName' the HTTP DCV mechanism will be used.
- The value '**HTTPSCSRHASH**' – and for this 'domainName' the HTTPS DCV mechanism will be used.

The value '**CNAMECSRHASH**' – and for this 'domainName' the DNS CNAME DCV mechanism will be used.
In addition, if you wish to have *all* the domains validated by one of the alternative (HTTP or HTTPS or DNS CNAME) mechanisms, then simply pass a single value for the 'dcvEmailAddresses' parameter of:

- **ALLHTTPCSRHASH**
  or
- **ALLHTTPSCSRHASH**
  or
- **ALLCNAMECSRHASH**

This should be the only value passed for the parameter, and will attempt to verify all domains via the same mechanism.

Multi-Domain Extended Status

The 'CollectSSL' API offers extended status for multi-domain certificates, showing all the requested FQDNs on an order, and the current DCV status of each.
Calling the 'CollectSSL' API with the correct authentication details, order ID and an additional parameter: 'showMDCDomainDetails' with the value 'Y' will provide this information in response to a status query.