GlobalSign Atlas
Certificate Management API Guide

July 2023
API Version 2

Table of Contents

Overview 3

Getting Started 3

Portal Onboarding Process 3

Manual Onboarding Process 4

For Venafi Customers 4

Atlas Account Credentials 4

Login to Atlas 4

Using the APIs 5

Certificate Management 5

Retrieve Account Validation Policy 5

Issue a New Certificate 6

Retrieve a Certificate 6

Revoke a Certificate 7

Rekey (Reissue) a Certificate 7

Retrieve a Certificate Trust Chain 8

Domain Management 8

Adding Domains 9

Initiate a Domain Claim 9

Verify a Domain Using DNS 9

Verify a Domain Using HTTP 10

Verify a Domain Using Email 10

Verify a Domain Using CNAME 10

Managing Domains 11

Retrieve the Status of a Specific Domain 11

Retrieve List of Domains 11

Emails for Domain Approval 12

Delete a Claim 12

Renew a Domain 12

Reporting 13

Certificate Stats 13

Issued Certificates 13

Revoked Certificates 13

Expiring Certificates 13
Account Information 13
Certificate Issuance Quota 13
Appendix A: Validation Policy 14
String Fields 14
List Fields 15
Appendix B: Domain Validation Using DNS CNAME Records 17
Update your DNS 17
Validate a Domain 17
Appendix C: Developer Guidance 19
Issuing Certificates 19
Domain Management 19

Overview

The GlobalSign Atlas platform is GlobalSign's answer to the evolving PKI landscape. Featuring state-of-the-art RESTful APIs, partners and customers can manage certificates through a robust, secure platform that is built for high volume certificate lifecycle management.

Link to API RAML: https://www.globalsign.com/en/resources/apis/api-documentation/globalsign_atlas_certificate_management_api.html

API login URL: https://emea.api.hvca.globalsign.com:8443/v2/login

To support PKI agility, GlobalSign regularly rotates its TLS issuing CAs every quarter. More information can be found on our support article, including CA links: https://support.globalsign.com/atlas/atlas-tls/atlas-tls-ica-rotations

Getting Started

Portal Onboarding Process

Customers seeking standard TLS or SMIME use cases can use this process, all others should use the Manual Onboarding Process described in the next section.

To begin, speak with your GlobalSign Account Manager. They will work with you to define the type of product you want to use and generate a quote.

1. Go to https://atlas.globalsign.com/register to register for an Atlas account. You will need to validate your email address.
2. Once logged in, you will be prompted to enter some business information and select if this account is for your organization or others. In most cases choose for your organization.
3. Your GlobalSign Account Manager will send you the service quote through the portal. Once you accept the quote, create an Atlas Identity, which will be sent to our Vetting department for verification (depending on verification level).
4. Once your identity has been vetted, generate your API credentials. This step will link your Identity and Service to your credentials. Save these credentials in a secure location. Each GlobalSign Service will have its own API credentials.
5. Generate your mTLS certificate and save it to a secure location.

Please refer to these Support articles for more details on the above process: https://support.globalsign.com/atlas/general-category-faqs

Manual Onboarding Process

To begin, speak with your GlobalSign Account Manager. They will collect from you some high-level company and contact information, as well as a public key (for instructions on generating a key pair please read the Atlas Account Credentials section below). This information is sent to the GlobalSign Vetting department and client management database. Vetting new customers takes up to 3 business days to complete and only happens once.

Once vetting is complete, an Atlas account will be created for you. You will receive encrypted API credentials and an mTLS certificate. Use your private key to decrypt the credentials, which will allow you to have full access to the Atlas APIs.

For Venafi Customers

For Venafi customers, create an account on the Venafi Cloud Platform. Then, you will go through the same manual process described above, but with some differences: you do not need to supply us a public key, and when you receive your encrypted GlobalSign Atlas credentials, you will upload them to the Venafi Cloud Platform to use certificates purchased from GlobalSign. Additionally, you do not need an mTLS certificate.

Atlas Account Credentials

To use the Atlas APIs, you need an API key and secret (your API credentials) as well as an mTLS certificate provided by GlobalSign. Note that Venafi customers do not need an mTLS certificate.

To securely provide you with the API credentials, you must first generate an RSA public/private key pair. You provide the public key to GlobalSign as part of the account set-up process, which we will use to encrypt the API credentials and send back to you. Full details are described in our support article: https://support.globalsign.com/ssl/api-plugins/how-obtain-globalsign-restful-api-account-credentials

Your public key is also used to generate an mTLS certificate, which GlobalSign will send to you along with your API credentials. Keep your API credentials and mTLS certificate in a safe place because you need both to access the APIs.

Login to Atlas

`POST /login`

After you decrypt your API credentials and receive your mTLS certificate, use the /login endpoint to access Atlas. The API response will include an access_token, which must be provided when accessing all of the other APIs in Atlas. This token is valid for a default of 10 minutes and when it expires, you must use the /login endpoint to obtain a new one.

Request Example:

```
{
  "key": "e510e289e6cd8947",
  "secret": "a477a8393d17a55ecb2ba6a61f58feb84770b621"
}
```

Response Example:

```
{
  "access_token":
"eyJhbGciOiAiSFMyNTYiLCAidHlwIjogIkpXVCJ9.eyJ1c2VyX2lkIjogMX0.BSf1w1blYKcbxVlyOtUogUsozH2
clY34xxYPd81QI1Q"
}
```

Using the APIs

In a few of the APIs, the results are paginated depending on what is specified in the “page,” “per_page” and “status” objects. The max (and default) per_page value is 100.

Certificate Management

Retrieve Account Validation Policy

`GET /validationpolicy`

To retrieve the validation policy for your user account, make a GET request to the /validationpolicy endpoint. The Account Validation Policy is where certificate subject information is defined. Only one Validation Policy can be linked to a user account. The following fields are defined at this level. This list may change over time, please check the current RAML.

- `validity`
- `subject_dn`
    - `common_name`
    - `given_name`
    - `surname`
    - `organization`
    - `organizational_unit`
    - `organization_identifier`
    - `country`
    - `state`
    - `locality`
    - `street_address`
    - `email`
    - `pseudonym`
    - `jurisdiction_of_incorporation_locality_name`
    - `jurisdiction_of_incorporation_state_or_province_name`
    - `jurisdiction_of_incorporation_country_name`
    - `business_category`
    - `extra_attributes`

- `san`
  - `dns_names`
  - `emails`
  - `uris`
  - `ip_addresses`
  - `other_names`
- `key_usages`
- `extended_key_usages`
- `subject_da`
  - `gender`
  - `date_of_birth`
  - `place_of_birth`
  - `country_of_citizenship`
  - `country_of_residence`
  - `extra_attributes`
- `qualified_statements`
- `ms_extension_template`
- `custom_extensions`
- `signature`
- `hash_algorithm`
- `public_key`
- `public_key_signature`

Issue a New Certificate

`POST /certificates`

Issue a new certificate by making a POST request to the /certificates endpoint. You can issue both wildcard and non-wildcard certificates with this endpoint. The fields that must be supplied to use this API are predicated by your validation policy. Review the Retrieve Account Validation Policy section to understand what fields are required for your certificate request.

When the certificate is issued, you will receive the URL to the certificate which contains the certificate serial number ({certificate}). The certificate serial number is assigned prior to the issuance of the certificate to track its status. You use the serial number in other API commands, such as the GET request to fetch this certificate.

In the event the certificate is never issued, that serial number will not be reused for another certificate or certificate request.

Retrieve a Certificate

`GET /certificates/{certificate}`

Retrieve a certificate by making a GET request to the /certificates/{certificate} endpoint. Use the certificate serial number in the {certificate} object. The response will include the certificate, the status of the certificate (issued or revoked), and when the certificate status was last changed in unix time.

Revoke a Certificate

`PATCH /certificates/{certificate}`

Revoke a certificate by making a PATCH request to the /certificates/{certificate} endpoint. Place the certificate serial number in the {certificates} object. You must include a revocation reason in the request. If the reason for revocation is due to keyCompromise you are encouraged to also include the time the key was compromised in your request. Once revoked, the certificate will display in the GET /stats/revoked endpoint. You can review revoked certificates using the GET /certificates/{certificate} endpoint.

The revocation reason must be one of the following:

- keyCompromise: When there is reason to believe that the private key of the certificate has been compromised, e.g., an unauthorized person has had access to the private key of the certificate.
- cessationOfOperation: When you no longer own all of the domain names in the certificate or when you will no longer be using the certificate because you are discontinuing your website.
- affiliationChanged: When your organization's name or other organizational information in the certificate has changed.
- Superseded: When you request a new certificate to replace an existing certificate.
- Unspecified: When the reason for revocation is not one of the above reasons, you must use this reason code.

Rekey (Reissue) a Certificate

`POST /certificates/{certificate}/rekey`

Reissue an active certificate by making a POST request to the /certificates/{certificate}/rekey endpoint. This will allow you to issue a new certificate using the parameters of an existing certificate. This is useful when you want multiple copies of a certificate with different private keys; for example, when you are operating a load balancer service where all the servers need the same certificate but have different private keys. The newly issued certificate will have the same not_after date as the original.

Use the certificate serial number in the {certificates} object and send a new public key or CSR (depending on your validation policy) with your request. No other field modifications will be allowed. The new certificate will have identical fields to the existing certificate.

Retrieve a Certificate Trust Chain

`GET /trustchain`

Retrieve the chain of trust for the certificates in your account by making a GET request to the /trustchain endpoint. The response is an ordered list of PEM-encoded CA certificates, starting with the Issuing CA and ending with the Root CA.

Domain Management

When you attempt to prove control of a domain, you are making a domain “claim.” Once a domain is verified, it is added to your account, and certificates can then be issued from the domain.

For the /claims endpoints below, results are paginated and filtered based on query parameters. You can define how many pages you want to view, how many results per page you want to view, and the time frame.

Adding Domains

Initiate a Domain Claim

`POST /claims/domains{domain}`

Initiate a domain claim (prove ownership of a domain) by making a POST request to the /claims/domains{domain} endpoint. Enter the fully qualified, IDNA-encoded domain to be validated into the {domain} object.

In the API response, a claimID is returned in the header, and the body includes a unique token and the token's expiration date (30 days from creation). When using either the DNS or HTTP validation methods, you load the token into the domain's DNS TXT record or website, and Atlas uses the claimID to validate the domain against your account.

Request Example:

```
{{httpAddress}}/v2/claims/domains/http-on-hv.globalsign-support.com }
```

Response Example:

```
Header: /v2/claims/domains/0136017F938B709EE80A67B80183B83F

{"token": "globalsign-domain-verification=##########",
 "assert_by": 1543570475
}
```

Verify a Domain Using DNS

`POST /claims/domains/{claimID}/dns`

After you have initiated a claim, you need to load the token you received in the claim response into your domain's DNS TXT record.

1. Go to the DNS Manager of the website where the SSL/TLS certificate will be installed.
2. Add a DNS record: TXT type.
3. Enter the claim token into the TXT value field: globalsign-domain-verification=<token>
4. Save your changes.

Once you have placed the token, make a POST request to the /claims/domains/{claimID}/dns endpoint to verify the domain. Include the website URL in the body of your request (authorization_domain). Atlas will look for the token and should return a 201 result. DNS propagation is not immediate so it might take a few minutes for the DNS record to be available.

Verify a Domain Using HTTP

`POST /claims/domains{claimID}/http`

After you have initiated a claim, you need to load the token you received in the claim response into your website.

1. Copy the claim token and paste it into a blank text editor. Save the file with the name “gsdv.txt”
2. Upload the gsdv.txt file to the directory: <your website>/.well-known/pki-validation/gsdv.txt

Once you have placed the token, make a POST request to the /claims/domains/{claimID}/http endpoint to verify the domain. Include the website URL in the body of your request (authorization_domain), and whether your website is HTTP or HTTPS scheme. Atlas will look for the token and should return a 201 result.

Notes:

1. This validation method does not currently follow http redirects, please be sure to put the token on the site being validated and enter the full domain address into the API request. Redirects are often set up from http to https, or from example.com to www.example.com. In the event the API encounters a redirect, it will reply with an error description similar to:

   “https://domain.com/.well-known/pki-validation/gsdv.txt: status code of the request is not 200: 301”
2. This validation method does not currently support wildcard certificates.

Verify a Domain Using Email

`POST /claims/domains{claimID}/email`

Verify a domain using the Email validation method by making a POST request to the /claims/domains{claimID}/email endpoint. Enter the email address that the validation email will be sent to in the body of your request. When the email arrives, it will contain a link that you must click to verify that you control the domain associated with the claim.

To view a list of approval email addresses, make a GET request to the /claims/domains/{claimID}/email endpoint.

Verify a Domain Using CNAME

You can verify domains by creating subdomain CNAME records for each domain you want to verify. CNAME records begin with “_”. For example, if you want to verify example.com, you need to create a CNAME for _acme-challange.example.com. The CNAME should point to the domain where you will put the DNS TXT record during the DNS domain validation method. For more information, please review the Appendix.

Managing Domains

We recommend you implement an internal process that collects all valid domains and stores them with the claimID locally to your system for future use.

Retrieve the Status of a Specific Domain

`GET /claims/domains{claimID}`

Retrieve the status for a specific domain claim by making a GET request to the /claims/domains{claimID} endpoint. Enter your claim ID into the {claimID} object. The API response includes the status of the claim (PENDING, VERIFIED), when the domain claim was created, when the domain claim expires, and verification log entries.

Retrieve List of Domains

`GET /claims/domains`

Retrieve a list of past domain claims and the total number of those claims by making a GET request to the /claims/domains endpoint. You can filter the list of domains by status (PENDING, VERIFIED) or using wildcard search as described in the below table.

Definitions "_" matches any single character "%" matches any sequence of zero or more characters	
Rule	**Example**
Wildcards at the beginning of the search expression need to be followed by a full-stop.	Correct: %.test.com Incorrect: %test.com
Wildcards at the end of the search expression need to be preceded by a full-stop.	Correct: another.% Incorrect: another.test%
Wildcards can't replace full-stops between subdomains of the domain.	Incorrect: ano%st.com (for "another.test.com")

The response will include information about the claims, such as status, validation method, and number of tries to verify the claim. The following text is an example API response.

```
{
    "assert_by": 1541608353 ,
    "created_at": 1536236966,
    "domain": "globalsign-support.com.",
    "expires_at": 1567843663,
    "id": "0171B257BEC69F171121FBE600205412",
    "log": [

         {
             "description": "no valid DNS records returned",
             "method": "DNS",
             "status": "ERROR",
             "timestamp": 1536252078
         },
         {
             "description": "claim successfully verified",
             "method": "DNS",
             "status": "SUCCESS",
             "timestamp": 1536307663
         }
    ],
    "status": "VERIFIED"
}
```

Emails for Domain Approval

`GET /claims/domains/{claimID}/email`

Retrieve a list of approval emails that can be used to validate domains using the Email Validation method. Make a GET request to the /claims/domains/{claimID}/email endpoint. The response will return a list of emails associated with the account that can be used to receive a validation email from GlobalSign to prove domain ownership.

Delete a Claim

`DELETE /claims/domains{claimID}`

This API removes the specified claim (based on claimID) and any associated information from Atlas.

Renew a Domain

`POST /claims/domains{claimID}/reassert`

This API is used when you want to renew a domain prior to its expiration. The domain will remain valid as you go through the domain verification process. Begin by making a POST request to the /claims/domains{claimID}/reassert endpoint. Enter the original claim ID into the {claimID} object. This will return a new token to validate the claim. Then use one of the domain verification methods to validate the domain. Tokens are valid for 30 days.

Reporting

Certificate Stats

For the /stats endpoints below, results are paginated and filtered based on query parameters. You can define how many pages you want to view, how many results per page you want to view, and the time frame.

Issued Certificates

`GET /stats/issued`
Retrieve a list of certificates issued over a specified time interval from your account by making a GET request to the /stats/issued endpoint. The response will include the total number of issued certificates, each certificate's serial number, and notBefore/notAfter dates. The maximum time supported is 30 days, with the default being 10 minutes.

Revoked Certificates

`GET /stats/revoked`
Retrieve a list of certificates that have been revoked over a specified time interval from your account by making a GET request to the /stats/revoked endpoint. The response will include the total number of revoked certificates, and each certificate's serial number and notBefore/notAfter date. The maximum look-back time supported is 30 days, with the default being 10 minutes.

Expiring Certificates

`GET /stats/expiring`
Retrieve a list of certificates that are going to expire over a specified time interval from your account by making a GET request to the /stats/expiring endpoint. The response will include the total number of expiring certificates within that time period, and each certificate's serial number and notBefore/notAfter date. The maximum time supported is 30 days, with the default being the current time.

Account Information

Certificate Issuance Quota

`GET /quotas/issuance`
Retrieve your remaining certificate issuance quota by making a GET request to the /quotas/issuance endpoint. The response will return the total number of certificates remaining that can be issued from your account.

Appendix A: Validation Policy

Your validation policy provides detailed guidance and rules for the fields required to request certificates using the POST /certificates endpoint. This includes all subjectDN fields, validity period, key usages/extended key usages, and public key information. The next two sections describe the properties of validation policies.

String Fields

A string field is text that is added to a certificate. This is the most common field type and is used for subjectDN fields. The validation policy for string fields consists of two components: Presence and Format.

Presence specifies where the information comes from. There are four possible values:

- REQUIRED: Required fields MUST be supplied by the user, and those validated values will then be placed in the certificate.
- OPTIONAL: Optional fields MAY be supplied by the user, and those validated values (when provided) will be placed in the certificate. If no value is provided, then the field will not be present in the issued certificate.
- FORBIDDEN: Forbidden fields will not be included in the issued certificate. If a value is supplied in the certificate request, the request will be rejected.
- STATIC: Static fields are populated by the user's identity and automatically included in the issued certificate. If there is no value in the corresponding identity field, the certificate request will be rejected.

Format determines the type of value a field can have. There are three possible values:

- Empty: Used when a field has presence FORBIDDEN
- Regex: Used when a field has presence REQUIRED or OPTIONAL. A regex expression is a special text string that defines what characters are allowed in the field. An example is "^[A-Za-z0-9]+$," which allows a field to have all alphabet characters, upper and lower case, and all numbers.
- Text: Used when a field has presence STATIC. It is a value that is placed in the certificate request on the back end, usually through your validated Atlas identity. For example, format = "GB" in the subjectDN country field means that the issued certificate will always have "GB" set in the subjectDN country field, and this value cannot be modified by the user.

For example:

```
"common_name": {
    "presence": "REQUIRED",
    "format": "^[A-Za-z][A-Za-z -]+$"
 },
"organization": {
    "presence": "STATIC",
    "format": "GMO GlobalSign"
 },
"country": {
    "presence": "STATIC",
    "format": "GB"
 },
"state": {
    "presence": "OPTIONAL",
    "format": "^[A-Za-z][A-Za-z \\-]+$"
 }
```

List Fields

A list field is a list of strings, such as DNS names in a SAN extension. The list fields in validation policies consist of four components: Static, List, Mincount, and Maxcount.

- Static:
    - If Static is set to false, the list is provided by the user through the certificate request.
    - If Static is set to true, the list is fixed on the back end and automatically included in the issued certificate.
- List: Based on the Static value, the List field will either contain a list of regexes/suffixes to validate the list sent by the customer, or a fixed list supplied by the back end.
    - Regex: See the String Fields section above
    - Suffix: The ending portion of a text string that must match your validated Atlas identity. For example, suffix ".globalsign.com" will allow "test.globalsign.com" but not "test.example.com"
- Mincount: The minimum number of elements a list can have. For example, min = 1 means you are required to send a list with at least 1 element.
- Maxcount: The maximum number of elements a list can have. For example, max = 0 means you are not allowed to send a list while max = 5 means the list can have up to 5 elements.

For example:

```
"organizational_unit": {
      "static": false,
      "list": [
        "^[A-Za-z][A-Za-z \\-]+$"
      ],
      "mincount": 1,
      "maxcount": 3
 },
"extra_attributes": {
      "1.3.6.1.5.5.7.48.1.5": {
        "static": true,
        "value_type": "PRINTABLESTRING",
        "value_format": "static attribute",
        "mincount": 1,
        "maxcount": 1
}
```

Appendix B: Domain Validation Using DNS CNAME Records

A Canonical Name (CNAME) record is a type of resource record in the Domain Name System (DNS) which maps one domain name (an alias) to another (the CNAME). You can, for example, point ftp.example.com and www.example.com to the DNS entry for example.com, which in turn has an A record which points to the IP address. If the IP address ever changes, you only have to record the change in one place within the network: in the DNS A record for example.com.

CNAME records are handled specially in the DNS and have several restrictions on their use. When a DNS resolver encounters a CNAME record while looking for a regular resource record, it will restart the query using the CNAME instead of the original name. (If the resolver is specifically told to look for CNAME records, the CNAME is returned, rather than restarting the query.) The CNAME that a CNAME record points to can be anywhere in the DNS, whether local or on a remote server in a different DNS zone.

Update your DNS

For each domain you want to verify using CNAME, you need to create a subdomain CNAME record that begins with an underscore character ("_"). For example, if you want to verify example.com, you need to create a CNAME for "_acme-challange.example.com." The CNAME should point to the place you intend to put a DNS TXT record with the verification token. Let's use "verify-example.com."

Validate a Domain

When you want to verify example.com you specify "_acme-challange.example.com" as the Authorization Domain Name (ADN) in the API call as the place you want GlobalSign to look for the token. The API will see that there is a CNAME record to "verify-example.com," so GlobalSign will look there for the DNS TXT record. You don't need to modify the example.com DNS in the process of this domain validation.

1. Using the Atlas API, request a domain to be verified: POST /claims/domains/{domain}
2. Receive the token for the validation.
3. Update your DNS to add a TXT record to verify-example.com with the token (it's OK if there are a few there already).

   Note: As noted above, since this is a different domain name (and different DNS login credentials) than the ones to your production DNS, this validation method is secure because no one can change your production DNS domain names.
4. Using the Atlas API, post to /claims/domains/{claimID}/dns with the ADN of "_acme-challange.example.com."
5. GlobalSign knows this is a valid ADN because of the logic you built previously (can approve domains using a CNAME in a subdomain beginning with "_"). GlobalSign will find the CNAME and then will check for the TXT record at verify-example.com.

6. The valid TXT record is found at verify-example.com, and the domain validation is approved.

You may then delete the TXT record you created.

Appendix C: Developer Guidance

This appendix describes process flows to guide developers in building applications that consume the Atlas APIs.

Issuing Certificates

Follow the below sequence of API calls to issue a certificate, starting with logging into Atlas, then obtaining the certificate, and then confirming the certificate chain of trust.

1. `POST /login`
   Using your Atlas API credentials and mTLS certificate, login to get the JWS token for subsequent actions.
2. `GET /validationpolicy`
   Fetch the list of required and optional fields that are needed to obtain a certificate.
3. `POST /certificates`
   Using the results of the validation policy, supply the necessary fields to request a new certificate.
4. `GET /certificates/{certificate}`
   Fetch your issued certificate. If it has been issued, it will be returned; otherwise a status or error will be provided to indicate next steps.
5. `GET /trustchain`
   Fetch the trust chain for your issued certificate, which will be needed for final configuration and use by the intended application. Without the full chain, the issued certificate may not be trusted by relying parties. It is important to do this for every issuance because issuing CA(s) may change from time to time and having the correct certificate chain prevents service interruption.

Domain Management

If the Atlas product you are issuing certificates from requires domain validation (e.g. SSL/TLS), you will need to follow the below sequence of API calls to validate the domain(s) before issuing certificates. If you attempt to request a certificate prior to domain validation, you will receive an error.

1. `POST /claims/domains/{domain}`
   Generate a new 'claim' to prove ownership of a domain and receive a claimID for subsequent actions.
2. `POST /claims/domains/{claimID}/<validationmethod>`
   Using the claimID, select one of three domain validation methods to verify ownership of the domain. Regardless of validation method, an out of band action is needed to update the applicable HTTP or DNS location, or review and process an email.
3. `GET /claims/domains`

Fetch the status of your domain claim. You can request certificates once the domain has been validated.

4. `POST /claims/domains/{claimID}/reassert`
   Renew a domain prior to its expiration using the original claimID. This API will issue you a new token for domain verification.