finexer

API Reference Support Center

Dashboard

Webhook signatures

Checking the authenticity of a webhook as an extra layer of security

All webhooks are signed with a signature specified in the header as `fx-signature`. This allows you to verify that the webhook is authentic and does come from Finexer rather than a third party.

Before you're able to verify the signature, you will need your signature key. This can be found in your account settings. This signature is unique to your account and should not be shared with anyone. **Treat it like a password.**

The signature format

The signature hash is

calculated using the **HMAC** method with the **SHA256** algorithm. The hash also contains a timestamp. This is the a time in UTC that is generated by Finexer each time an event is sent to your endpoint. If Finexer retries to sent an event (e.g. your endpoint previously replied with a non-`2xx` status code), then a new signature and timestamp is generated for the new delivery attempt.
This will help prevent replay attacks, which is where an attacker intercepts the transmission between Finexer and your endpoint and re-transmits it. As the timestamp is part of this, that isn't possible to timestamp without invalidating the signature.

The timestamp is prefixed with `t=` and appears in ISO 8601 format (e.g. 2020-05-20T00:00:00).
The second part of the signature is a hex-encoded string of the sent JSON (the request body), known as the hash. This is prefixed with `s=`, and is the part to use to verify the request. Finexer derives this

hash as follows:

```
{
    var key =
"bJf4ZJKXZh199oJkfacRWdAkL"
// unique signature key
(found in account
settings)
    var time = "2020-05-
12T14:45:00Z"; // time in
utc (an event is sent)
    var json = "{}"; // an
event data in json
representation
    var hash =
ComputeHash($"{time}.
{json}", key); // use HMAC
with SHA-256 algorithm
    var s = ToHex(hash);
// convert hash to hex
(this is a signature)
}
```

In the receiving request, the header will appear as a single line, in the format of `t={time};s={hash}`.

For example:

```
t=2020-05-
12T14:45:00Z;s=94ee059335e58
7e501cc4bf90613e0814f00a7b08
bc7c648fd865a2af6a22cc2
```

.

Calculating the signature

Step 1: Retrieve the timestamp and signature from the

header

Split the `fx-signature` header value using a semicolon `;` as the delimiter. Then proceed to split each segment using the equals character `=` character to get both the prefix and the value pair. For example:

```
t=2020-05-12T14:45:00Z;s=94ee059335e587
```

splits into:

`t=2020-05-12T14:45:00Z` and `s=94ee059335e587`. And further into `2020-05-12T14:45:00Z` and `94ee059335e587`. This last piece is the signature of the webhook.

Step 2: Prepare the payload to be compared

To calculate the hash, you'll need the timestamp from the header (`t=`) and the JSON in the body of the webhook itself. These should be concatenated by a period:

```
2020-05-12T14:45:00Z.
{"key":"value"}
```

Step 3: Calculate the hash

Using the HMAC method, SHA256 algorithm and your secret as the key, you can now generate the signature hash from `2020-05-12T14:45:00Z.{"key":"value"}`. Remember, the signature hash is based on the hex-encoded string. Once you have this in place, you can check the signature hash you've generated against the hash you've received with the webhook in the header (`s=`).

You should take into consideration the timestamp that the webhook was sent and compare it to the time that it was received, and decide if this is within your own tolerance. The time is in UTC.