HTTP Message Signatures

IETF 114

July 28, 2022

Justin Richer and Annabelle Backman

What is it?

- Detached signature mechanism for generic HTTP messages
  - Can sign request and response
  - Works decently across HTTP versions
- Robust against expected changes, e.g.
  - Proxy injection of header fields
  - Partial signature of stable aspects of message
- Allows multiple signatures
  - Including adding signatures over time
- Uses HTTP-native technologies
  - Structured Fields for encoding

```
POST /foo?param=value&pet=dog HTTP/1.1
Host: example.com
Date: Tue, 20 Apr 2021 02:07:55 GMT
Content-Type: application/json
Content-Digest: sha-256=:X48E9qOokqqrvdts8nOJRJN3OWDUoyWxBf7kbu9DBPE=:
Example-Dict: a=(1 2), b=3, c=4;aa=bb, d=(5 6);valid
Content-Length: 18

{"hello": "world"}
```

HTTP Message

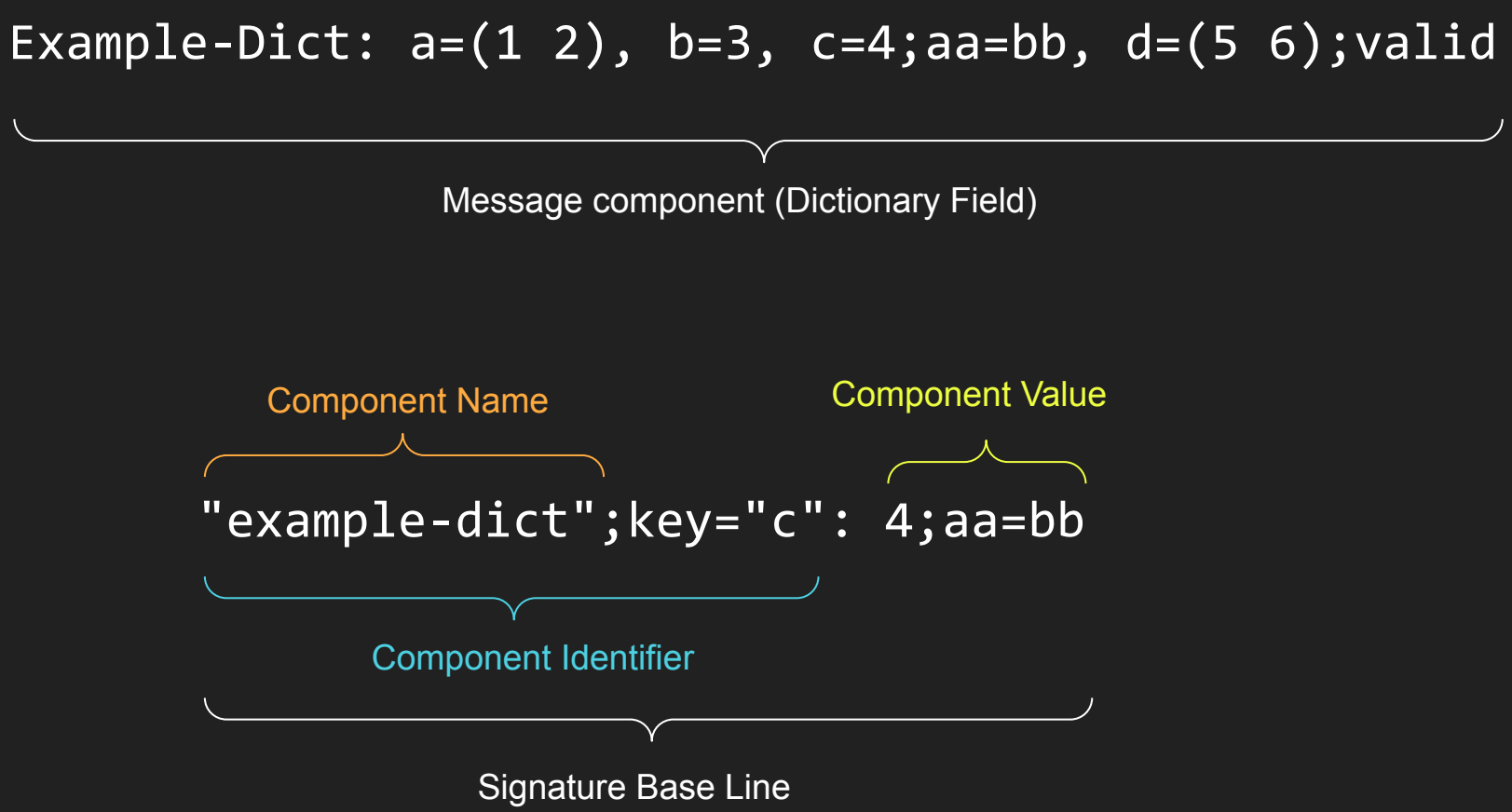
```
"content-type": application/json
"content-digest": sha-256=:X48E9qOokqqrvdts8nOJRJN3OWDUoyWxBf7kbu9DBPE=:
"content-length": 18
"@target-uri": https://example.com/foo?param=value&pet=dog
"@signature-params": ("content-type" "content-digest" "content-length" "@target-uri");created=1657842663
```

Signature Base

```
Signature-Input: ietf114=("content-type" "content-digest" "content-length" "@target-uri");created=1657842663
Signature: ietf114=:O38apFVMx3kUGuNg5tX/4j6ufffch7Uk6KSreWxNzcbs9xRxwbRMaigl6qAnollQ3aeHGuRnePykB1eJmsVMOQ==:
```

Signature Output

Example generated at https://httpsig.org/


Example-Dict: a=(1 2), b=3, c=4;aa=bb, d=(5 6);valid
Message component (Dictionary Field)
Component Name
Component Value
"example-dict";key="c": 4;aa=bb
Component Identifier
Signature Base Line

HTTP Signature Process

- Inputs:
  - HTTP Message
  - Key material
  - Required components
- Functions:
  - Cryptographic primitives: HTTP_SIGN (M, Ks) -> S
  - Key derivation (where needed)
  - Message hashing (where needed)
  - Binary encoding (where needed)
- Outputs:
  - Message signature (byte sequence)
  - Signature parameters (ordered set with parameters)

HTTP Signature Verification Process

- Inputs:
  - HTTP Message
  - Key material
  - Signature parameters (ordered set of covered components with parameters)
  - Message signature
- Functions:
  - Cryptographic primitives: HTTP_VERIFY (M, Kv, S) -> V
  - Key derivation (where needed)
  - Message hashing (where needed)
  - Binary encoding (where needed)
- Outputs:
  - Boolean verification status

Draft Status: From -08 to -11

- More security and privacy considerations added to document
  - Relationship to Digests
  - Dealing with weird things like Set-Cookie
- Significant editorial clarifications
  - Applied HTTP editor guidelines throughout
  - Clarified terms: “component name”, “component identifier”, “component value”
- Updated and expanded examples
- Added “`req`” flag for request-response binding
  - Removed “@request-response” derived component
- Added “`bs`” flag for byte-sequence encoded values

Implementation Status

- Java library (at least two)
- Python library (behind httpsig.org and in-doc examples)
- Scala library
- JavaScript (in-browser)
- Rust library (update of Cavage-draft implementation)
- Go library (from scratch)
- We can add these to httpsig.org as they become usable

HTTP Signatures Roadshow!

GNAP

- Main key proofing method in GNAP

7.3.1. HTTP Message Signing

This method is indicated by the method value `httpsig`. The signer creates an HTTP Message Signature as described in [I-D.ietf-httpbis-message-signatures]. This method defines the following parameters:

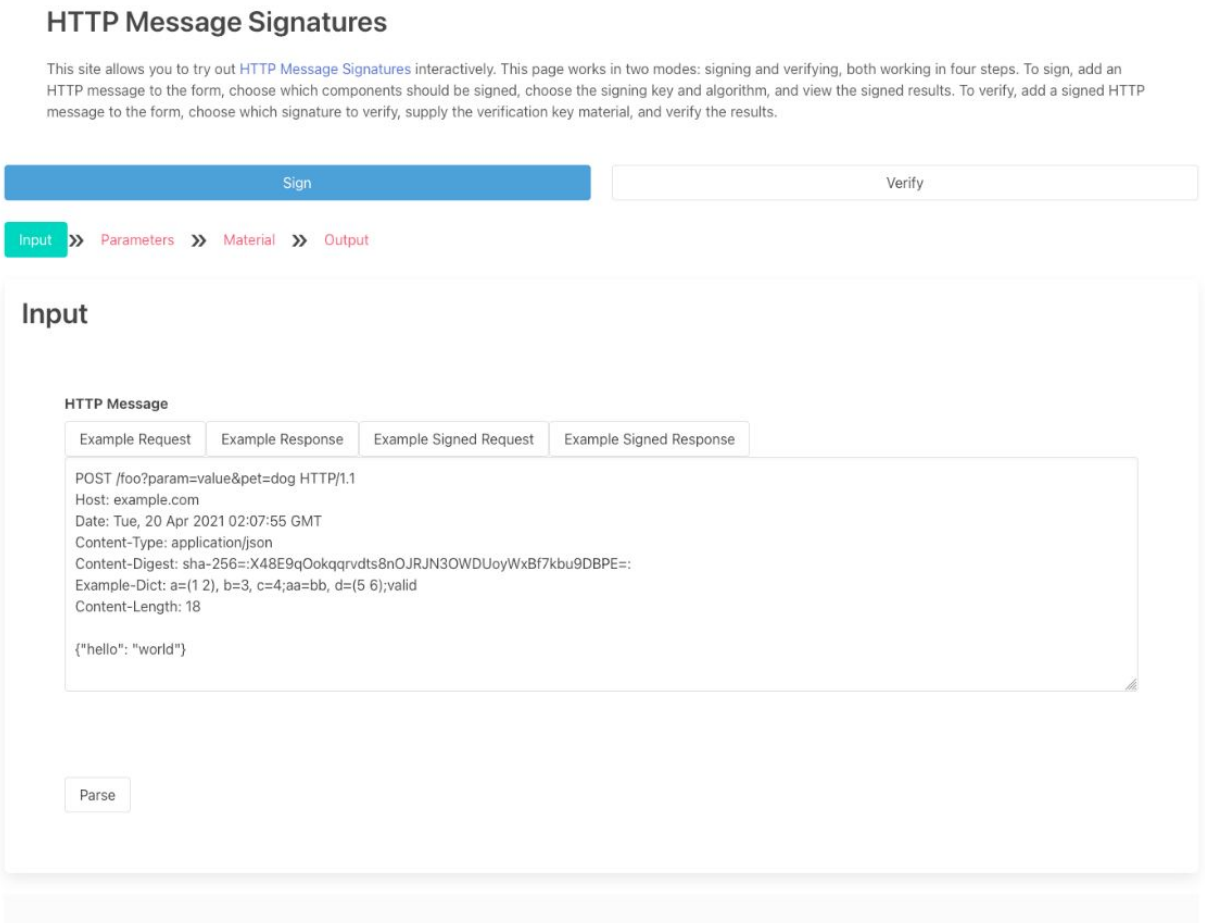
FAPI

- Referenced in OpenID Foundation's "Financial Grade API" draft specification

2.2.4. HTTP Message Signing

To support non-repudiation for NR7, NR8 and NR9 in the [attackermodel], HTTP requests and responses can be signed.

This profile supports HTTP Message Signing using the *HTTP Message Signatures* specification being developed by the IETF HTTP Working Group.



https://httpsig.org/

Working Group Last Call

- **The editors believe this draft is ready for WGLC**
- Core has been stable for a long time
- There are a growing number of implementations
- Other work is depending on this
- We can probably close out the last issues quickly
  - Some can probably be closed without action
  - Last few have proposed solutions or need WG input, could be part of WGLC

Probably-Closable Issues

- Relation with Signed Exchanges (#1206)
- Support for signing specific cookies (#1197)
- Support expected authority changes (#1196)

Open Issue #2133: Signature Context

- Opaque singer-chosen string to indicate “target application” of signature
- Mitigation for oracle attacks
- Similar feature in TLS 1.3
- Proposal in issue: add REQUIRED signature parameter, don’t describe contents (to be filled in by applications)
- Editors proposal: add OPTIONAL signature parameter with descriptions of use, no restrictions on content (PR#2222)

Open Issue #2134: Cache

- Someone could cache the response from one signed request and replay it to another signed (or unsigned) request
- Someone could cache a signed response
- Editors have added some text;

What additional guidance do we need to give?

Open Issue #2144: Server Push

- Draft focuses on traditional request/response
- Components are taken from context of message (request or response with optional request that caused it)
- Server push has a potentially different context
  - One request multiple responses
  - Maybe no requests at all except something the server made up?
- Editors have no idea what to do with this one, need experts
  - Is the description we have enough?
  - Do we need to mention server push explicitly?
  - Are there security considerations with server push?