HTTP Request Signing with AWS Signature Version 4

Annabelle Backman, AWS

IETF 105 – July, 2019

Request Signing in AWS

- Why?
  - Authentication
  - Message integrity
  - Replay prevention
- When?
  - Almost all requests to AWS APIs
- What?
  - Method, endpoint, path, timestamp, query string parameters, body
  - Some headers
- How?
  - HMAC-SHA256 with key derived from shared secret

Signature Version 4

1. Create the canonical request

2. Create the string to sign

3. Calculate the signature

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T0530Z
```

```
CanonicalRequest = HTTPRequestMethod + '\n' +
  CanonicalURI + '\n' +
  CanonicalQueryString + '\n' +
  CanonicalHeaders + '\n' +
  SignedHeaders + '\n' +
  HexEncode(Hash(RequestPayload))
```

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T053000Z
```

```
POST
CanonicalURI
CanonicalQueryString
CanonicalHeaders
SignedHeaders
HexEncode(Hash(RequestPayload))
```

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T053000Z
```

```
POST
/long/path%2520name/
CanonicalQueryString
CanonicalHeaders
SignedHeaders
HexEncode(Hash(RequestPayload))
```

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T053000Z
```

```
POST
/long/path%2520name/
C=abc&C=def&a=1&b=2
CanonicalHeaders
SignedHeaders
HexEncode(Hash(RequestPayload))
```

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T053000Z
```

```
POST
/long/path%2520name/
C=abc&C=def&a=1&b=2
content-type:application/x-www-form-urlencoded; charset=utf8
host:example.amazonaws.com
x-amz-date:20190722T053000Z

content-type;host;x-amz-date
HexEncode(Hash(RequestPayload))
```

SigV4 Step 1: Create the Canonical Request

```
POST /a/../long/path%20name/?a=1&b=2&C=def&C=abc HTTP/1.1
Host: example.amazonaws.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Date: 20190722T053000Z
```

```
POST
/long/path%2520name/
C=abc&C=def&a=1&b=2
content-type:application/x-www-form-urlencoded; charset=utf8
host:example.amazonaws.com
x-amz-date:20190722T053000Z

content-type;host;x-amz-date
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

SigV4 Step 2: Create the String to Sign

```
StringToSign = Algorithm + '\n' +
  RequestDateTime + '\n' +
  CredentialScope + '\n' +)
  HexEncode(Hash(CanonicalRequest))
```

```
AWS4-HMAC-SHA256
20190722T053000Z
20190722/us-east-1/example/aws4_request
f536975d06c0309214f805bb90ccff089219ecd68b2577efef23edd43b7e1a59
```

SigV4 Step 3: Calculate the Signature

```
kSecret    = SecretKey
kDate      = HMAC('AWS' + kSecret, Date)
kRegion    = HMAC(kDate, Region)
kService   = HMAC(kRegion, Service)
kSigning   = HMAC(kService, "aws4_request")

Signature = HexEncode(HMAC(kSigning, StringToSign))
```