I ♥ APIS

ADVANCED SECURITY EXTENSIONS IN APIGEE EDGE: HMAC, HttpSignature

Dino Chiesa,
Vinit Mehta

apigee

1

I ♥ APIS

SECURITY EXTENSIONS IN APIGEE EDGE: HMAC, HttpSignature

Mehta, Chiesa

2

Who is using Signatures?

AWS - custom HMAC-SHA256 signature over the [verb . host . uri . queryparams]
Google Maps API for Work - HMAC-SHA1 over URL+query (includes clientid)
Twitter - OAuth1.0a signatures (HMAC-SHA1) over headers + params
Azure storage - HMAC-SHA256 over [verb . contentmd5 . contenttype . date . url]
Github's outbound webhooks - HMAC-SHA1 over the body
(prospect) Automaker - XML DSIG over payload
(prospect) WW Retailer - HttpSignature (Signature applied to select headers)
(customer) Payeezy - custom HMAC-SHA256 over select headers and body
(customer) BBC – HMAC on calls from Salesforce.com, $$ Video-on-demand
(customer) Vodafone - HMAC
(customer) O2 - HMAC

apigee

Why are these people using Message-level security?
Why are they requiring Signatures on their payloads?

apigee

Let’s talk about MITM, Non-repudiation, Auditing

```
POST /accounts/12334566/update
API-Key: ryJIXSIAMjuyDii8
Host: api.example.com
Content-Type: application/json
Content-Length: 132

{
  "rty": "MRA",
  "n": "5SGw1jcqyFYEZaf3VduzmRk_jcBNFFLQgOf9U",
  "e": "AQAB",
  "gla": "Rm425",
  "use": "sig",
  "kid": "196"
}
```

apigee 5

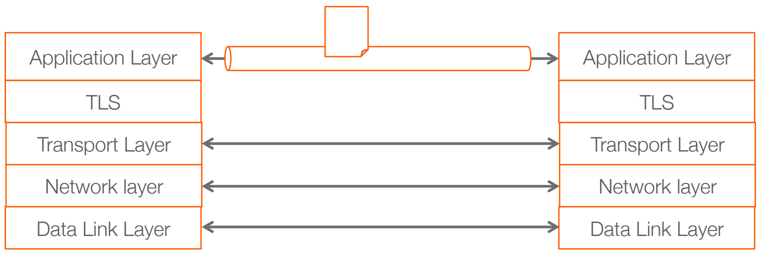
Let’s talk about MITM, Non-repudiation, Auditing

```
POST /accounts/12334566/update
API-Key: ryJIXSIAMjuyDii8
Host: api.example.com
Content-Type: application/json
Content-Length: 132

{
  "rty": "MRA",
  "n": "5SGw1jcqyFYEZaf3VduzmRk_jcBNFFLQgOf9U",
  "e": "AQAB",
  "gla": "Rm425",
  "use": "sig",
  "kid": "196"
}
```

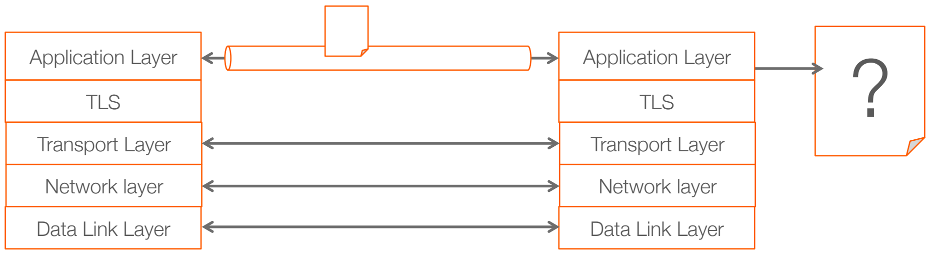
- TLS encrypts all of this, except the hostname
- TLS is point-to-point
- What happens if you relay this message beyond the TLS-protected entry point?
- If this message gets archived, how to guarantee its integrity?

apigee

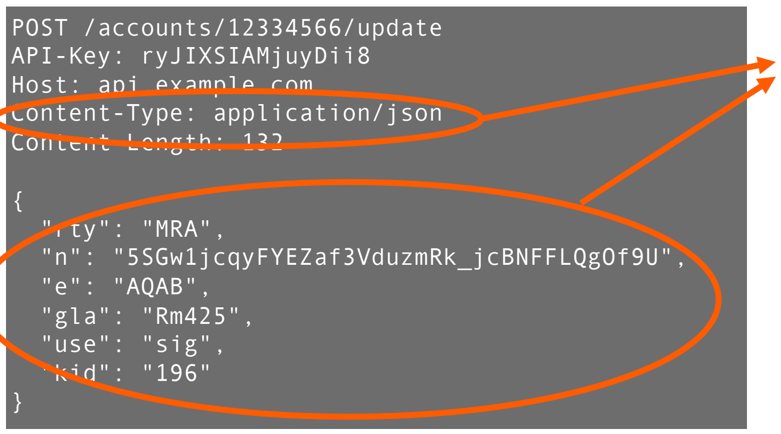
Transport-layer security secures data in transit



Message-layer security is independent of transport



Message-level Signatures protect the payload



```
POST /accounts/12334566/update
API-Key: ryJIXSIAMjuyDii8
Host: api.example.com
Content-Type: application/json
Content-Length: 132

{
  "kty": "MRA",
  "n": "5SGw1jcqyFYEZaf3VduzmRk_jcBNFFLQgOf9U",
  "e": "AQAB",
  "gla": "Rm425",
  "use": "sig",
  "kid": "196"
}
```

- Message-level signatures can protect these parts of the message, independently of transport
- Can optionally perform message-level encryption, encryption of selected fields, etc.

apigee

MAC = Message Authentication Code
HMAC = keyed-hash MAC

MAC may be used to verify the integrity of a message.

HMAC injects a key into the normal hashing function. HMAC may be used to simultaneously verify both the integrity and the authentication of a message.

apigee

You are Smart People,
So Obviously…

You want to verify HMACs in API Proxies.

Today, in Apigee Edge, You can't.

apigee

Apigee Edge includes standard policies for many security tasks.

But it does not include a policy to generate or validate HMACs. You're stuck.

apigee

Apigee Edge includes standard policies for many security tasks.

But it does not include a policy to generate or validate HMACs. You're stuck.

Or Maybe you're not?

apigee

Code + Configure !

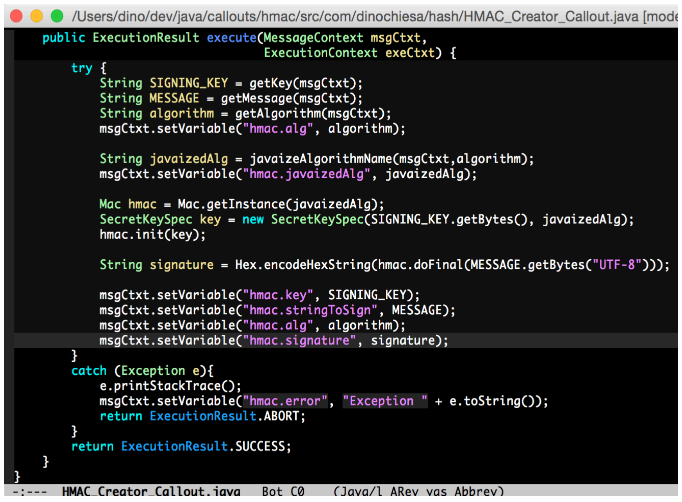
apigee

What are Java Callouts?

- Embed your Java code as a policy in Apigee Edge
- One Interface, one method, 2 parameters
- Can read policy configuration
- Can read and write context variables
- …anchor anywhere in Edge policy flow

- One of the ways to extend Edge with custom code. Also JavaScript, Python, nodejs.

- RTFM: http://apigee.com/docs/api-services/reference/java-callout-policy



/Users/dino/dev/java/callouts/hmac/src/com/dinochiesa/hash/HMAC_Creator_Callout.java [mode

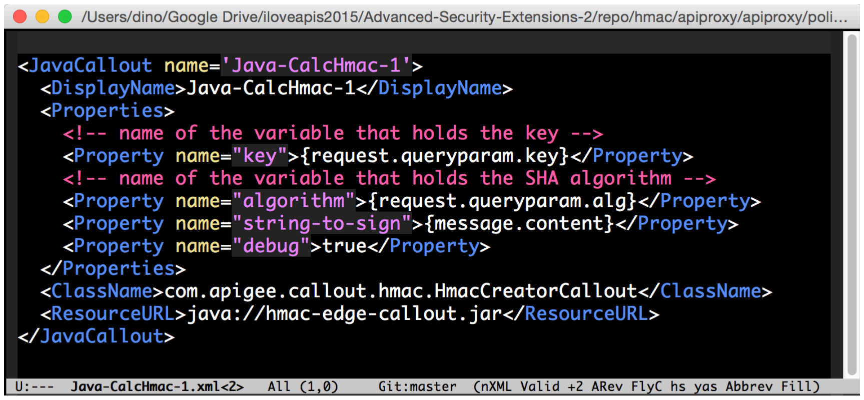
```
    public ExecutionResult execute(MessageContext msgCtxt,
                                   ExecutionContext exeCtxt) {
        try {
            String SIGNING_KEY = getKey(msgCtxt);
            String MESSAGE = getMessage(msgCtxt);
            String algorithm = getAlgorithm(msgCtxt);
            msgCtxt.setVariable("hmac.alg", algorithm);

            String javaizedAlg = javaizeAlgorithmName(msgCtxt,algorithm);
            msgCtxt.setVariable("hmac.javaizedAlg", javaizedAlg);

            Mac hmac = Mac.getInstance(javaizedAlg);
            SecretKeySpec key = new SecretKeySpec(SIGNING_KEY.getBytes(), javaizedAlg);
            hmac.init(key);

            String signature = Hex.encodeHexString(hmac.doFinal(MESSAGE.getBytes("UTF-8")));

            msgCtxt.setVariable("hmac.key", SIGNING_KEY);
            msgCtxt.setVariable("hmac.stringToSign", MESSAGE);
            msgCtxt.setVariable("hmac.alg", algorithm);
            msgCtxt.setVariable("hmac.signature", signature);
        }
        catch (Exception e){
            e.printStackTrace();
            msgCtxt.setVariable("hmac.error", "Exception " + e.toString());
            return ExecutionResult.ABORT;
        }
        return ExecutionResult.SUCCESS;
    }
}
-:---  HMAC_Creator_Callout.java    Bot C0     (Java/l ARev yas Abbrev)
```

apigee

15

Java Callout for HMAC Verification or Generation

- Re-usable now in any of your Proxies
- Configure it with XML as any other policy
- Verify integrity of any payload
- Can read HMAC generated by third party libraries
- Relies on secret key or public/private key pair



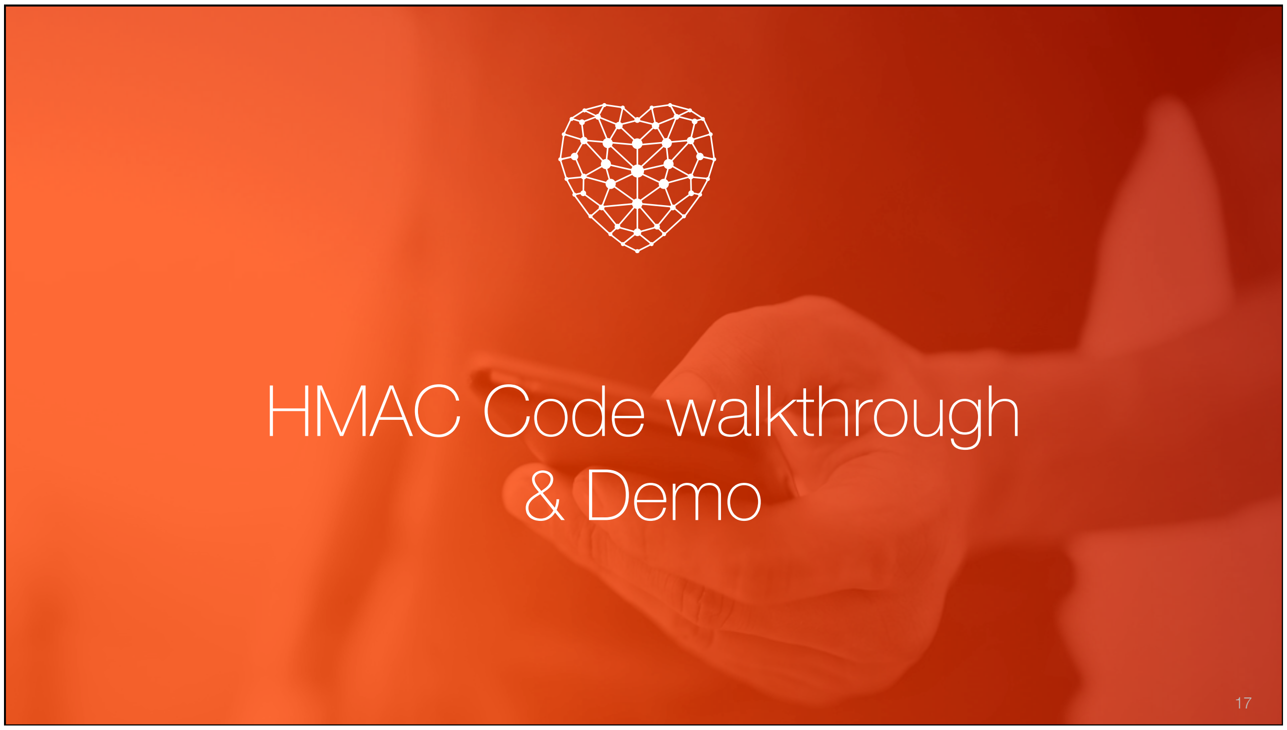
/Users/dino/Google Drive/iloveapis2015/Advanced-Security-Extensions-2/repo/hmac/apiproxy/apiproxy/poli…

```
<JavaCallout name='Java-CalcHmac-1'>
  <DisplayName>Java-CalcHmac-1</DisplayName>
  <Properties>
    <!-- name of the variable that holds the key -->
    <Property name="key">{request.queryparam.key}</Property>
    <!-- name of the variable that holds the SHA algorithm -->
    <Property name="algorithm">{request.queryparam.alg}</Property>
    <Property name="string-to-sign">{message.content}</Property>
    <Property name="debug">true</Property>
  </Properties>
  <ClassName>com.apigee.callout.hmac.HmacCreatorCallout</ClassName>
  <ResourceURL>java://hmac-edge-callout.jar</ResourceURL>
</JavaCallout>
U:---  Java-CalcHmac-1.xml<2>   All (1,0)    Git:master  (nXML Valid +2 ARev FlyC hs yas Abbrev Fill)
```

https://github.com/apigee/iloveapis2015-hmac-httpsignature

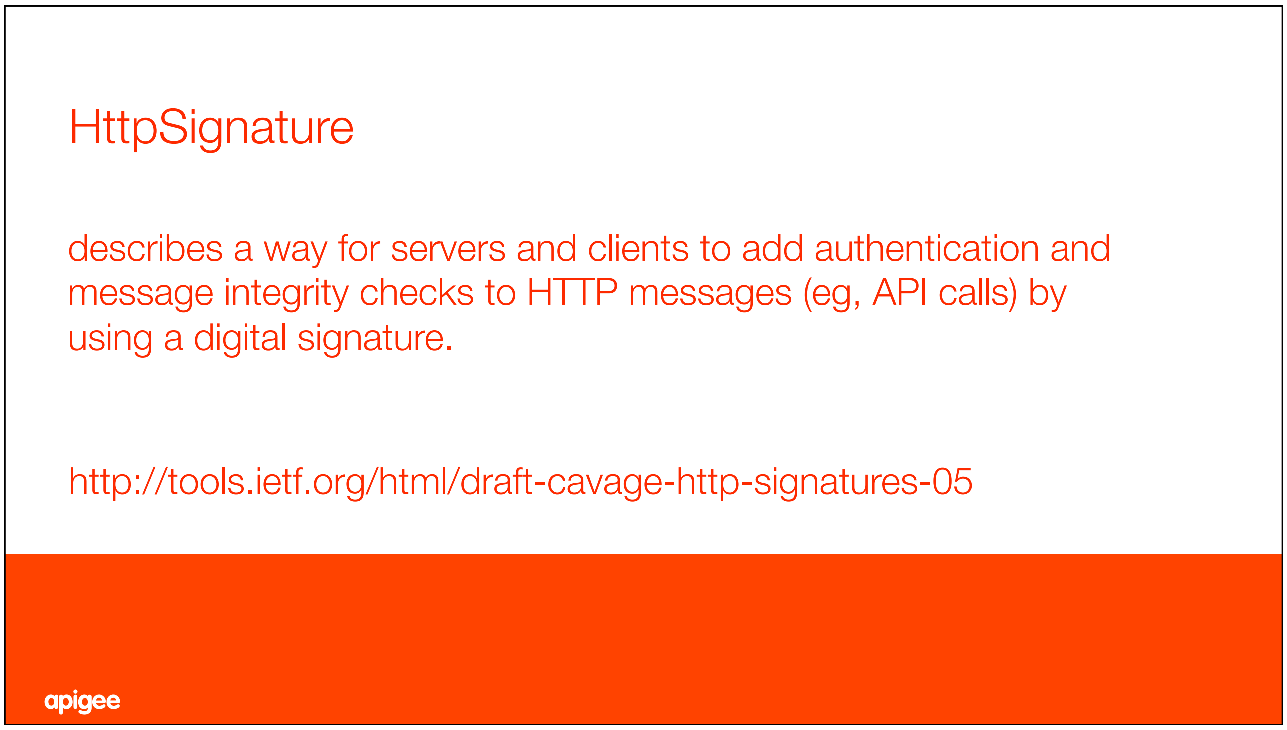
apigee

16



HMAC Code walkthrough & Demo

17



HttpSignature

describes a way for servers and clients to add authentication and message integrity checks to HTTP messages (eg, API calls) by using a digital signature.

http://tools.ietf.org/html/draft-cavage-http-signatures-05

apigee

HttpSignature

Complements API Key or Token-based authentication.

apigee

HttpSignature

The client sends in a Signature header that contains 4 things:

- keyid - identifying the key used by the client. The meaning is app dependent.
- algorithm - can be RSA-SHA (public/private key) or HMAC-SHA (shared key)
- list of HTTP headers - optional; space delimited; these are included in the signing base
- a computed signature of those headers

apigee

HttpSignature

Each element is formed as key="value" and they are separated by commas.
This must be passed in an HTTP header named "Signature".
The resulting header might look like this:

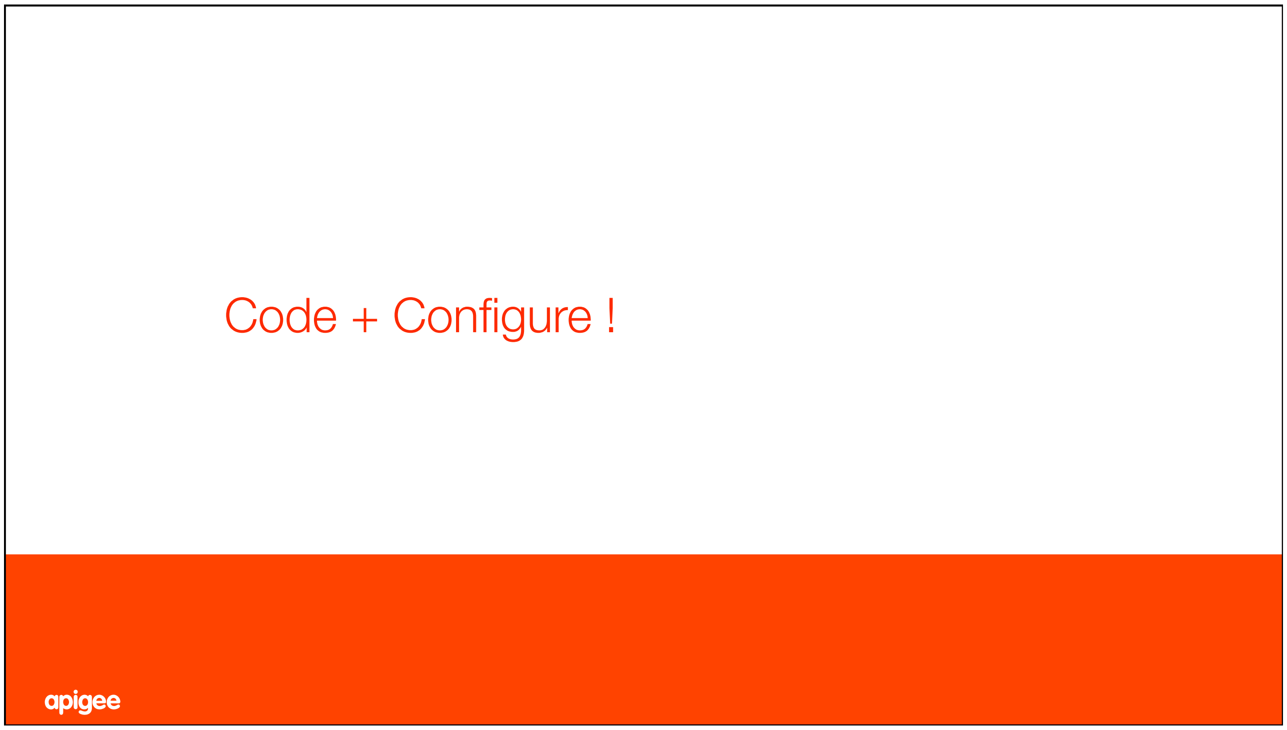
```
Signature: keyId=”mykey",algorithm="hmac-sha256",headers="(request-
target) date",signature="udvCIHZAafyK+szbOI/KkLxeIihexHpHpvMrwbeoErI="
```

apigee

Apigee Edge includes standard policies for many security tasks.

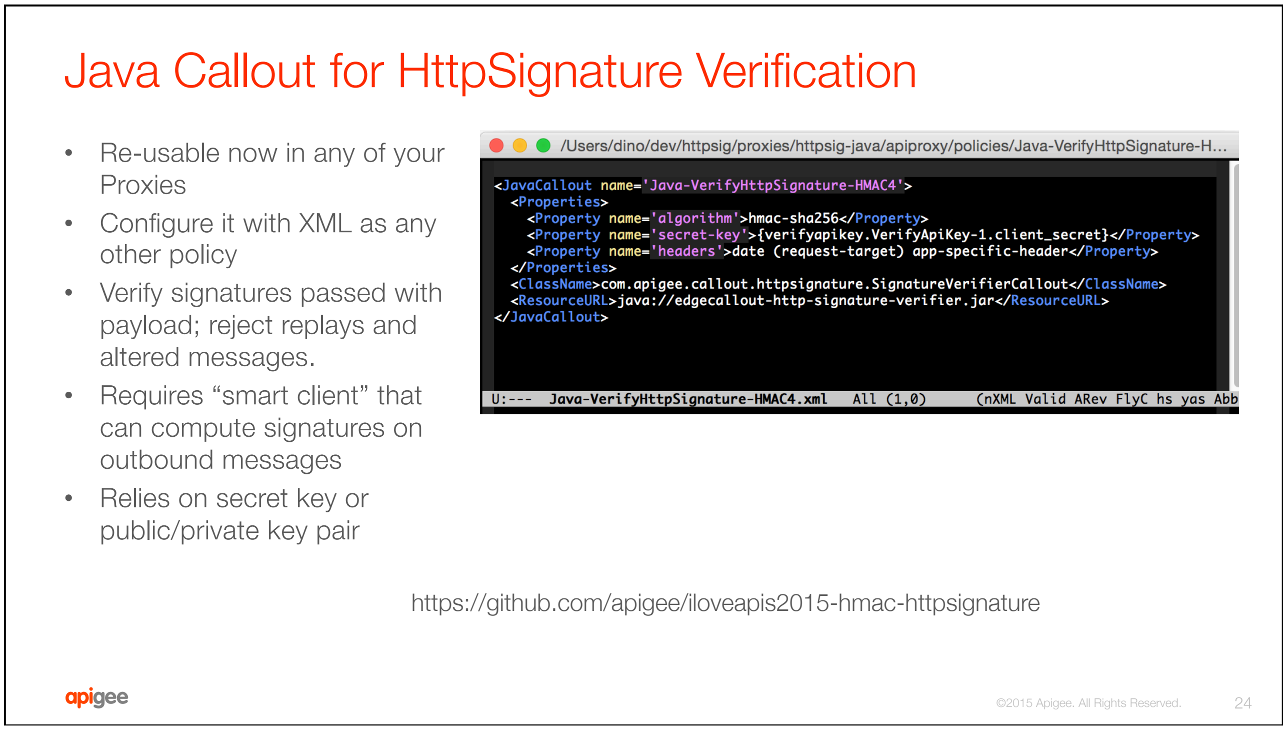
But it does not include a policy to validate HttpSignature. Sound Familiar?

apigee



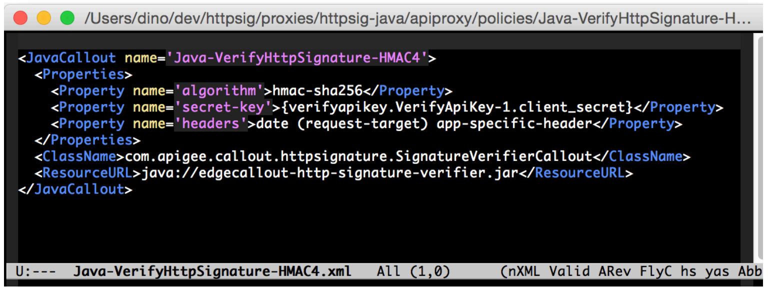
Code + Configure !

apigee



Java Callout for HttpSignature Verification

- Re-usable now in any of your Proxies
- Configure it with XML as any other policy
- Verify signatures passed with payload; reject replays and altered messages.
- Requires “smart client” that can compute signatures on outbound messages
- Relies on secret key or public/private key pair



/Users/dino/dev/httpsig/proxies/httpsig-java/apiproxy/policies/Java-VerifyHttpSignature-H…

```
<JavaCallout name='Java-VerifyHttpSignature-HMAC4'>
  <Properties>
    <Property name='algorithm'>hmac-sha256</Property>
    <Property name='secret-key'>{verifyapikey.VerifyApiKey-1.client_secret}</Property>
    <Property name='headers'>date (request-target) app-specific-header</Property>
  </Properties>
  <ClassName>com.apigee.callout.httpsignature.SignatureVerifierCallout</ClassName>
  <ResourceURL>java://edgecallout-http-signature-verifier.jar</ResourceURL>
</JavaCallout>
```

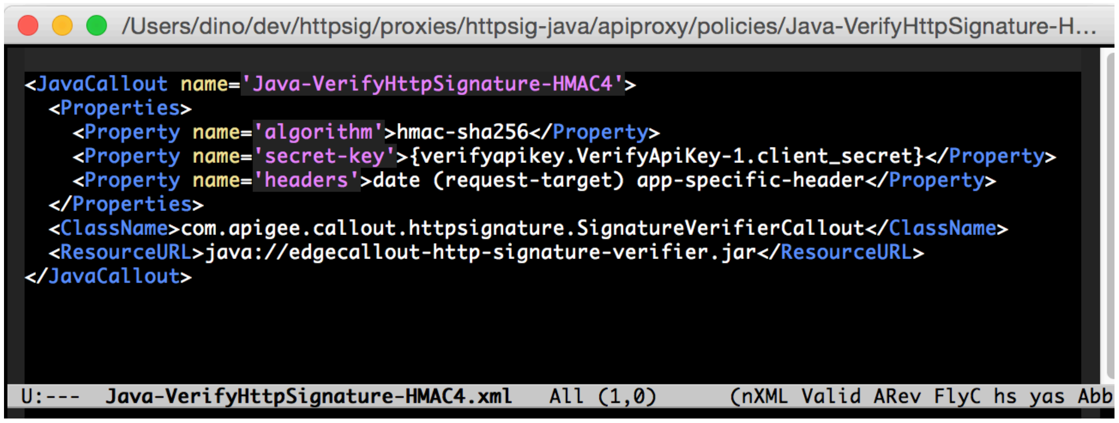
```
U:---  Java-VerifyHttpSignature-HMAC4.xml   All (1,0)     (nXML Valid ARev FlyC hs yas Abb
```

https://github.com/apigee/iloveapis2015-hmac-httpsignature

apigee

©2015 Apigee. All Rights Reserved.   24

Java Callout for HttpSignature Verification

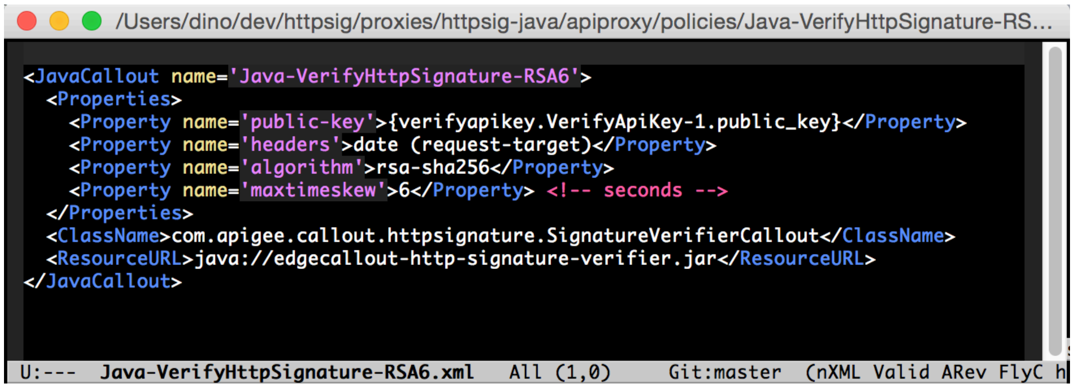

/Users/dino/dev/httpsig/proxies/httpsig-java/apiproxy/policies/Java-VerifyHttpSignature-H…

```
<JavaCallout name='Java-VerifyHttpSignature-HMAC4'>
  <Properties>
    <Property name='algorithm'>hmac-sha256</Property>
    <Property name='secret-key'>{verifyapikey.VerifyApiKey-1.client_secret}</Property>
    <Property name='headers'>date (request-target) app-specific-header</Property>
  </Properties>
  <ClassName>com.apigee.callout.httpsignature.SignatureVerifierCallout</ClassName>
  <ResourceURL>java://edgecallout-http-signature-verifier.jar</ResourceURL>
</JavaCallout>
```

```
U:---  Java-VerifyHttpSignature-HMAC4.xml   All (1,0)     (nXML Valid ARev FlyC hs yas Abb
```

apigee

Java Callout for HttpSignature Verification


/Users/dino/dev/httpsig/proxies/httpsig-java/apiproxy/policies/Java-VerifyHttpSignature-RS…

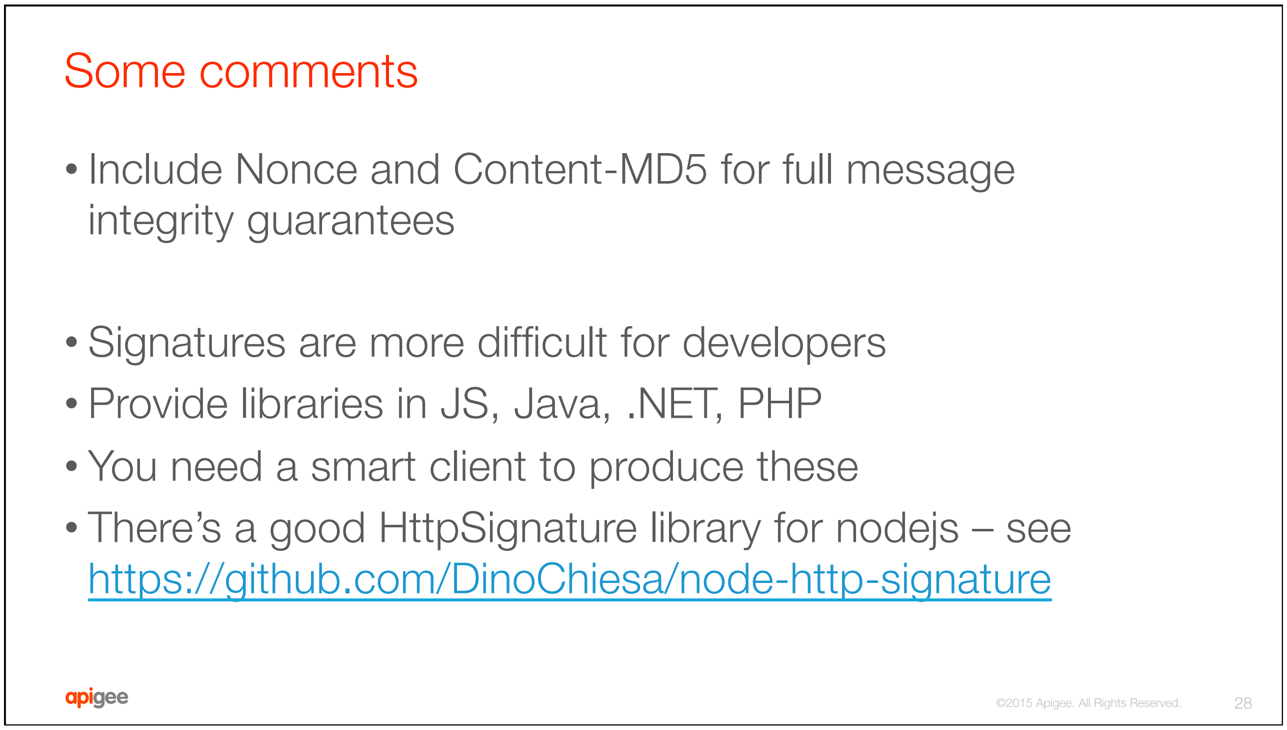
```
<JavaCallout name='Java-VerifyHttpSignature-RSA6'>
  <Properties>
    <Property name='public-key'>{verifyapikey.VerifyApiKey-1.public_key}</Property>
    <Property name='headers'>date (request-target)</Property>
    <Property name='algorithm'>rsa-sha256</Property>
    <Property name='maxtimeskew'>6</Property> <!-- seconds -->
  </Properties>
  <ClassName>com.apigee.callout.httpsignature.SignatureVerifierCallout</ClassName>
  <ResourceURL>java://edgecallout-http-signature-verifier.jar</ResourceURL>
</JavaCallout>
```

```
U:---  Java-VerifyHttpSignature-RSA6.xml   All (1,0)    Git:master  (nXML Valid ARev FlyC h
```

apigee

HttpSignature Code walkthrough & Demo

27



Some comments

- Include Nonce and Content-MD5 for full message integrity guarantees
- Signatures are more difficult for developers
- Provide libraries in JS, Java, .NET, PHP
- You need a smart client to produce these
- There's a good HttpSignature library for nodejs – see https://github.com/DinoChiesa/node-http-signature

apigee

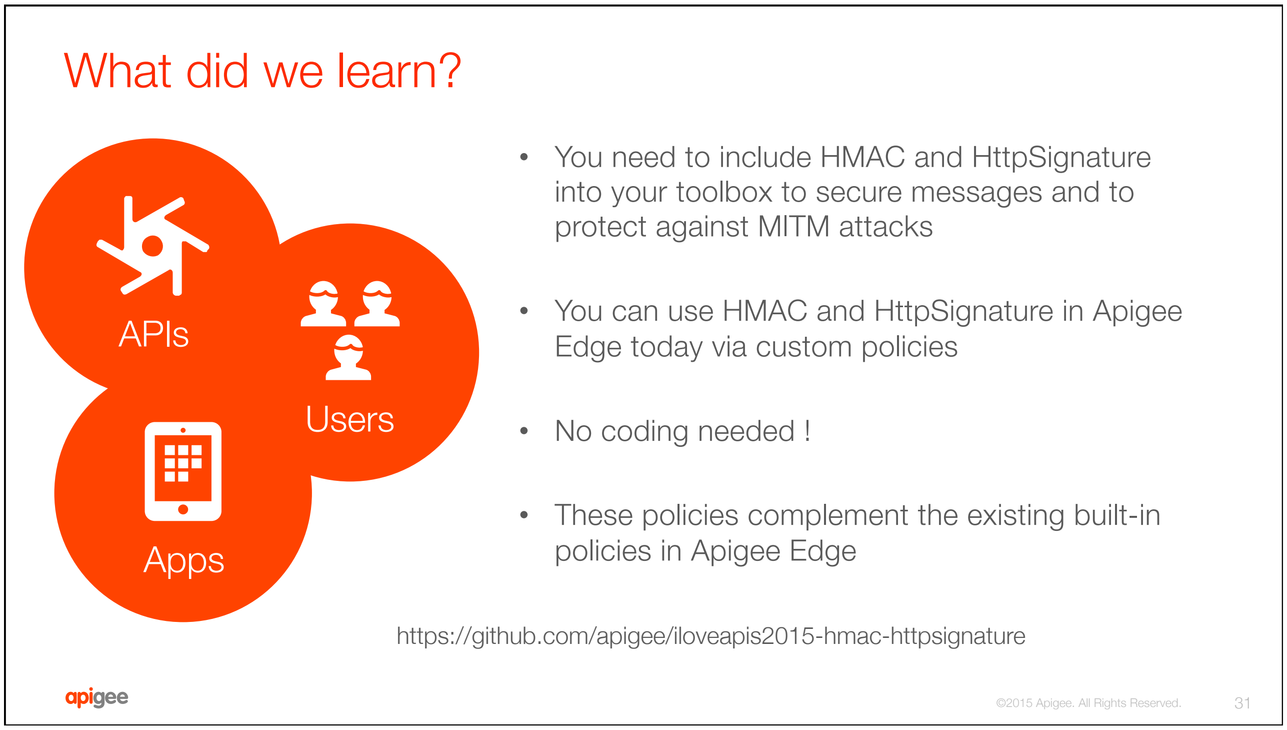
28

When to use HMAC, HttpSignature?

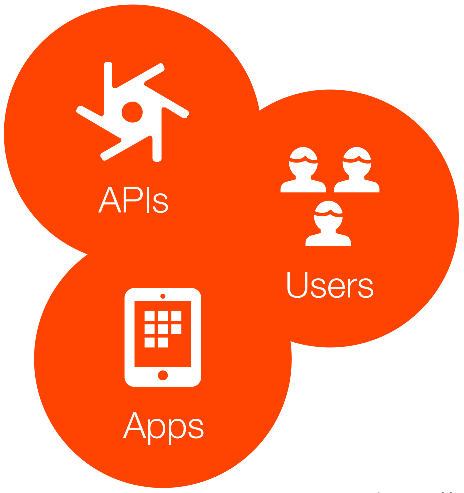
apigee

- Use these callouts whenever you want to add HMAC or HttpSignature verification to your proxies
- To avoid MITM risks
- To Layer Message-level protection on top of TLS
- Scenarios :
  Non-Repudiation and archival
      eg, medical records release consent
  Message-layer Integrity

apigee



What did we learn?



- You need to include HMAC and HttpSignature into your toolbox to secure messages and to protect against MITM attacks
- You can use HMAC and HttpSignature in Apigee Edge today via custom policies
- No coding needed !
- These policies complement the existing built-in policies in Apigee Edge

https://github.com/apigee/iloveapis2015-hmac-httpsignature

apigee