Follow this presentation:
http://bit.ly/nodeclub-crypto

GitHub project:
http://github.com/walling/nodeclub-crypto

Crypto and Session Management

in node.js

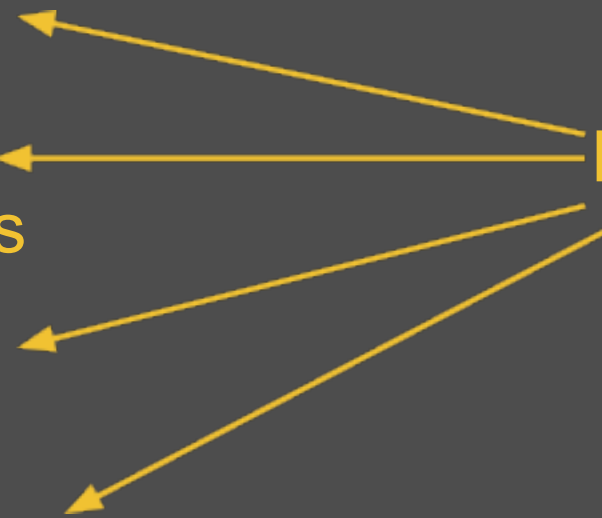
digroli

Agenda

Follow this presentation:
http://bit.ly/nodeclub-crypto

GitHub project:
http://github.com/walling/nodeclub-crypto

- Crypto basics
- MACs
- Managing sessions
- More exciting stuff
- Questions?



digroli

Agenda

Follow this presentation:
http://bit.ly/nodeclub-crypto

GitHub project:
http://github.com/walling/nodeclub-crypto

- Crypto basics
- MACs
- Managing sessions
- More exciting stuff
- Questions?

Live demo coding

live demo

digroli

Crypto Basics

"Back to School"

digroli

Security Terminology

- Authentication

  “verifying a claim made by a subject that it should be allowed to act on behalf of a given principal”

Security Terminology

- Authentication

  “verifying a claim made by a subject that it should be allowed to act on behalf of a given principal”

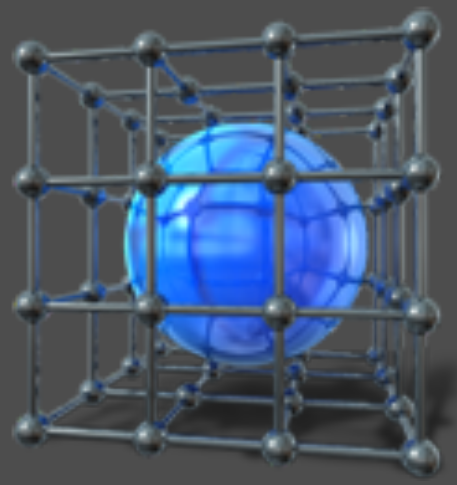
- Authorization

  “verifying that an authenticated subject has permission to perform certain operations or access specific resources”

— Wikipedia

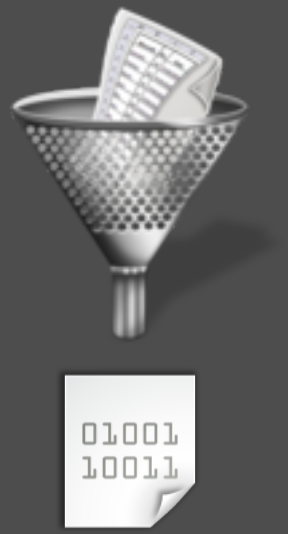
Security Terminology

- Confidentiality
- Integrity
- Availability
- Non-repudiation



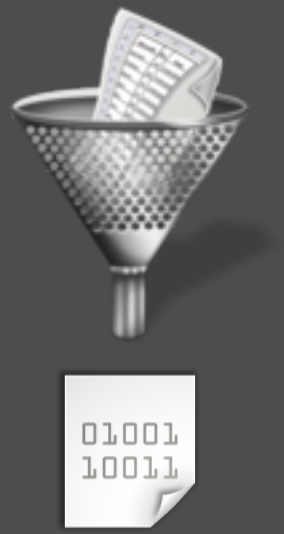
Hash Functions

- Makes a fingerprint of a document



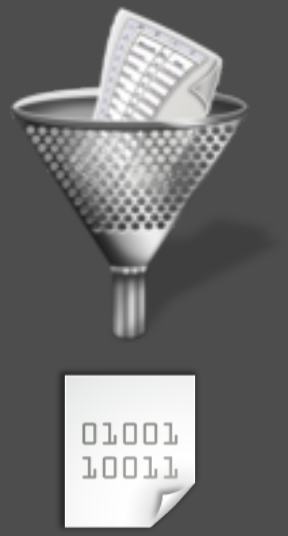
Hash Functions

- Makes a fingerprint of a document
- If two hashes differ the input documents differ



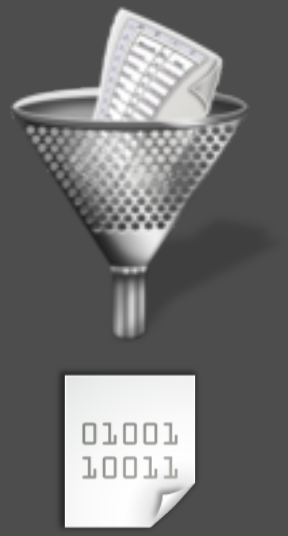
Hash Functions

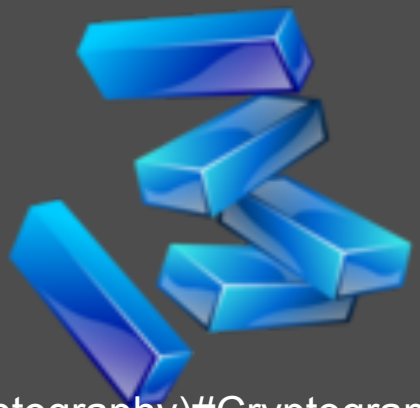
- Makes a fingerprint of a document
- If two hashes differ the input documents differ
- Two distinct documents results in two different hashes with high probability



Hash Functions

- Makes a fingerprint of a document
- If two hashes differ the input documents differ
- Two distinct documents results in two different hashes with high probability





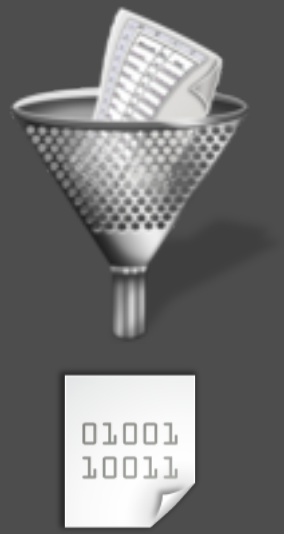
MD5
SHA-1
SHA-256
SHA-512

Check http://en.wikipedia.org/wiki/Hash_function_(cryptography)#Cryptographic_hash_algorithms

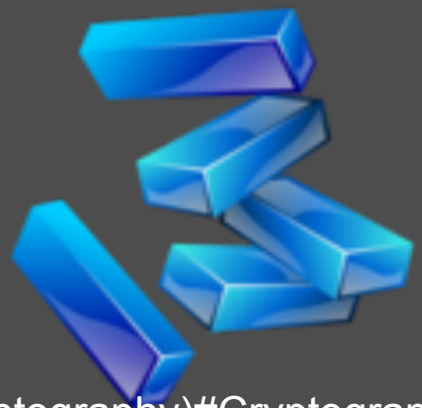
digroli

Hash Functions

self-sha1.js



- Makes a fingerprint of a document
- If two hashes differ the input documents differ
- Two distinct documents results in two different hashes with high probability

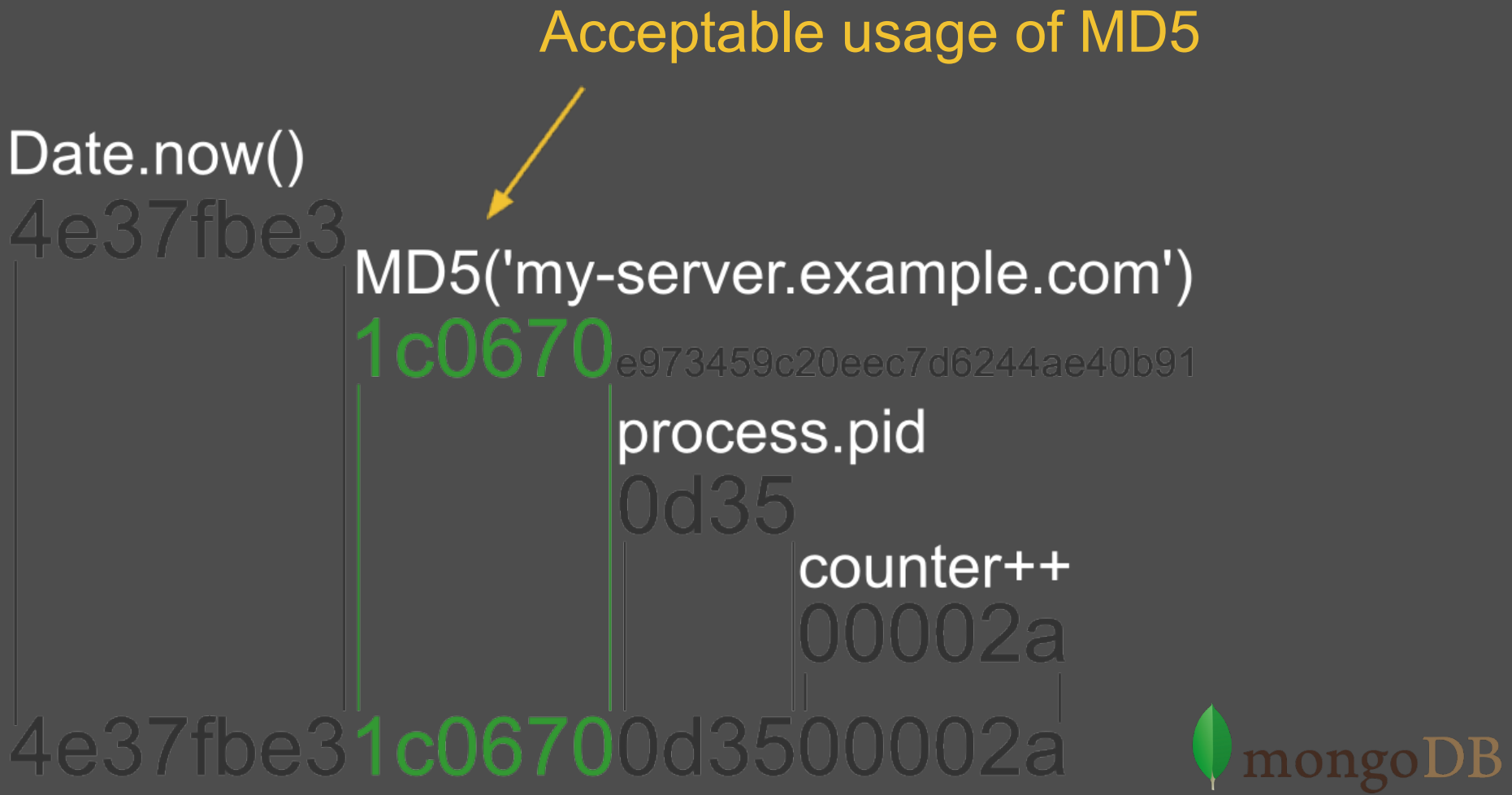


MD5
SHA-1
SHA-256
SHA-512

Check http://en.wikipedia.org/wiki/Hash_function_(cryptography)#Cryptographic_hash_algorithms

digroli

MongoDB Object IDs



digroli

Cryptographic Secure Randomness

- The better randomness the better the cryptographic algorithms perform

Cryptographic Secure Randomness

- The better randomness the better the cryptographic algorithms perform
- Usually /dev/urandom is sufficient (and non-blocking)

Cryptographic Secure Randomness

buffer-random.js

- The better randomness the better the cryptographic algorithms perform
- Usually /dev/urandom is sufficient (and non-blocking)

digroli

MACs

Not your grandma's laptop!

Message Authentication Code

Attach it to a document and it verifies:

- Authentication
- Integrity

Message Authentication Code

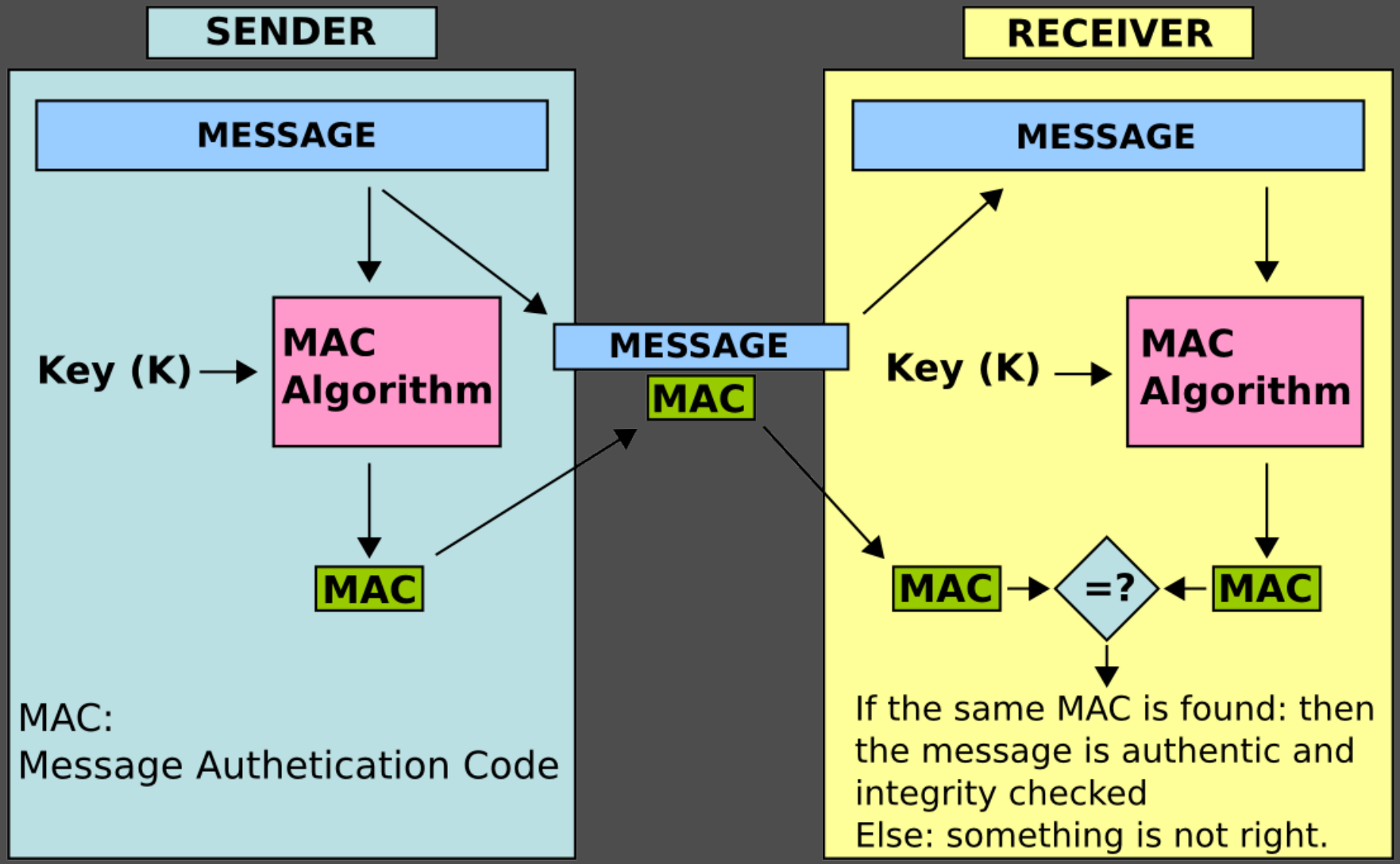
Attach it to a document and it verifies:

- Authentication
- Integrity

Recipe:

- 1 Message
- 1 Secret Key
- 1 MAC algorithm

Message Authentication Code



— Wikipedia

Näive implementation of a MAC

Easy, just use hash function:

SHA-1(key + secret)

Näive implementation of a MAC

Easy, just use hash function:

SHA-1(key + secret)

The only problem is the many possible attacks!

Näive implementation of a MAC

Easy, just use hash function:

SHA-1(key + secret)

The only problem is the many possible attacks!

Better to use what the crypto exports built ...

MAC, the real way

$$HMAC_H(\text{key}, \text{msg}) =$$

$$H((\text{key} \oplus \text{pad}_1) + H((\text{key} \oplus \text{pad}_2) + \text{msg}))$$

(*H* is any cryptographic hash function, fx. SHA-1)

MAC, the real way

$$HMAC_H(\text{key}, \text{msg}) =$$

$$H((\text{key} \oplus \text{pad}_1) + H((\text{key} \oplus \text{pad}_2) + \text{msg}))$$

(*H* is any cryptographic hash function, fx. SHA-1)

... a bit insane maybe, but no known attacks.

MAC, the real way

$$HMAC_H(\text{key}, \text{msg}) =$$

$$H((\text{key} \oplus \text{pad}_1) + H((\text{key} \oplus \text{pad}_2) + \text{msg}))$$

(*H* is any cryptographic hash function, fx. SHA-1)

... a bit insane maybe, but no known attacks.

We will use $HMAC_{\text{SHA-1}}$ for this presentation.

digroli

MAC, the real way

hmac.js

$$HMAC_H(\text{key}, \text{msg}) =$$

$$H((\text{key} \oplus \text{pad}_1) + H((\text{key} \oplus \text{pad}_2) + \text{msg}))$$

(*H* is any cryptographic hash function, fx. SHA-1)

... a bit insane maybe, but no known attacks.

We will use $HMAC_{\text{SHA-1}}$ for this presentation.

digroli

What can MACs be used for?

- Signing and verifying content given to clients

What can MACs be used for?

- Signing and verifying content given to clients

  You need a shared secret key for the servers.

What can MACs be used for?

- Signing and verifying content given to clients

  You need a shared secret key for the servers.

- Makes your webservice more RESTful

What can MACs be used for?

- Signing and verifying content given to clients

  You need a shared secret key for the servers.

- Makes your webservice more RESTful

Examples of content:

- Access Tokens
- URLs
- Cookies

What can MACs be used for?

mac-examples.js

- Signing and verifying content given to clients

  You need a shared secret key for the servers.

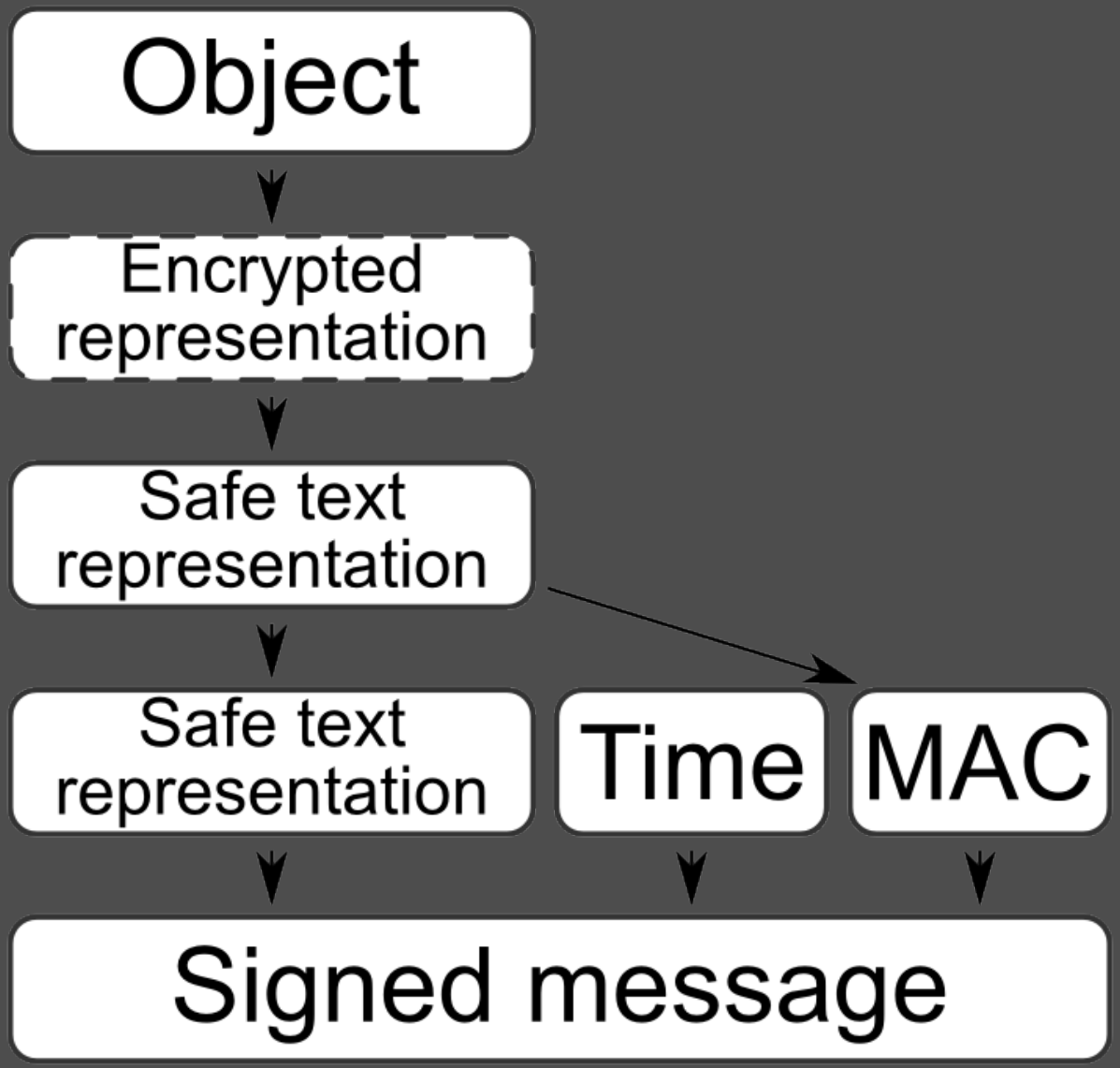
- Makes your webservice more RESTful

Examples of content:

- Access Tokens
- URLs
- Cookies

digroli

How message signing works in detail



Choosing a message

For access token it can be:

- ID of the authenticated user
- Boolean whether it is admin user
- Information about access control

Choosing a message

For access token it can be:

- ID of the authenticated user
- Boolean whether it is admin user
- Information about access control

For URL it can be:

- URL to sign
- Allowed HTTP methods
- URL parts that can be changed

Choosing a message

For access token it can be:

- ID of the authenticated user
- Boolean whether it is admin user
- Information about access control

For URL it can be:

- URL to sign
- Allowed HTTP methods
- URL parts that can be changed

Depending on non-message context.

digroli

Timeouts

Include a timeout in your message.

Timeouts

Include a timeout in your message.

- Otherwise you state that the message is true forever

Timeouts

Include a timeout in your message.

- Otherwise you state that the message is true forever
- For an access token this would mean the user is authenticated forever

Timeouts

Include a timeout in your message.

- Otherwise you state that the message is true forever
- For an access token this would mean the user is authenticated forever

You don't want that!

Timeouts

Include a timeout in your message.

- Otherwise you state that the message is true forever
- For an access token this would mean the user is authenticated forever

You don't want that!

A good timeout depends on the domain.

Timeouts

Include a timeout in your message.

- Otherwise you state that the message is true forever
- For an access token this would mean the user is authenticated forever

You don't want that!

A good timeout depends on the domain.

I choose half an hour for access tokens.

Timeouts

timestamp-test.js

Include a timeout in your message.

- Otherwise you state that the message is true forever
- For an access token this would mean the user is authenticated forever

You don't want that!

A good timeout depends on the domain.

I choose half an hour for access tokens.

Managing Sessions

"Learn to Love the Cookie"

digroli

The old-school way of sessions

- Store session ID in cookie

The old-school way of sessions

- Store session ID in cookie
- Store session state on server

The old-school way of sessions

- Store session ID in cookie
- Store session state on server

The problem remains ...

The old-school way of sessions

- Store session ID in cookie
- Store session state on server

The problem remains ...

The session state is not distributed by default.

The old-school way of sessions

- Store session ID in cookie
- Store session state on server

The problem remains ...

The session state is not distributed by default.

Usually solved by using memcached.

The New Way

Sign session state and send it to the client.

The New Way

Sign session state and send it to the client.

It is distributed and RESTful by default.

The New Way

Sign session state and send it to the client.

It is distributed and RESTful by default.

You need a shared key between all your servers.

Server Shared Key

Make it as long as the block size of your hash function.

Server Shared Key

Make it as long as the block size of your hash function.

For SHA-1 that is 512 bits = 64 bytes:

Server Shared Key

Make it as long as the block size of your hash function.

For SHA-1 that is 512 bits = 64 bytes:

```
new Buffer(64).randomize().toString('base64')
```

Server Shared Key

Make it as long as the block size of your hash function.

For SHA-1 that is 512 bits = 64 bytes:

```
new Buffer(64).randomize().toString('base64')
```

Depend on NODE_DEPLOYMENT environment variable:

Server Shared Key

Make it as long as the block size of your hash function.

For SHA-1 that is 512 bits = 64 bytes:

```
new Buffer(64).randomize().toString('base64')
```

Depend on NODE_DEPLOYMENT environment variable:

- For “development” use hard-coded key

Server Shared Key

Make it as long as the block size of your hash function.

For SHA-1 that is 512 bits = 64 bytes:

```
new Buffer(64).randomize().toString('base64')
```

Depend on NODE_DEPLOYMENT environment variable:

- For “development” use hard-coded key
- For “production” load key from disk

More Exciting Stuff

Wake Up!

Image Credits

http://en.wikipedia.org/wiki/File:MAC.svg
http://www.thebuzzmedia.com/mongodb-single-server-data-durability-guide/
http://www.iconfinder.com/icondetails/1403/128/cloud_sun_weather_icon
http://www.iconfinder.com/icondetails/24736/128/key_keys_login_password_private_secure_security_icon
http://www.iconfinder.com/icondetails/25041/128/binary_executable_fs_gnome_icon
http://www.iconfinder.com/icondetails/37070/128/blueprint_build_hammer_tool_xcode_icon
http://www.iconfinder.com/icondetails/37767/128/antique_clock_pocket_watch_time_timepiece_watch_icon
http://www.iconfinder.com/icondetails/41429/128/black_jack_casino_dice_slots_icon
http://www.iconfinder.com/icondetails/43350/128/business_business_man_consultancy_consultant_man_staff_user_icon
http://www.iconfinder.com/icondetails/44568/128/collision_detection_icon
http://www.iconfinder.com/icondetails/45534/128/filter_funnel_icon
http://www.iconfinder.com/icondetails/6613/128/box_firewall_hosting_network_package_security_icon
http://www.iconfinder.com/icondetails/6624/128/passport_password_icon

Other images are made by Bjarke Walling.

digroli

Tweet me: @walling
E-mail me: bwp@bwp.dk

Questions?

Thanks for dropping by!

digroli