Securing Data with Hashing

Josh Cummings
PRINCIPAL SOFTWARE ENGINEER

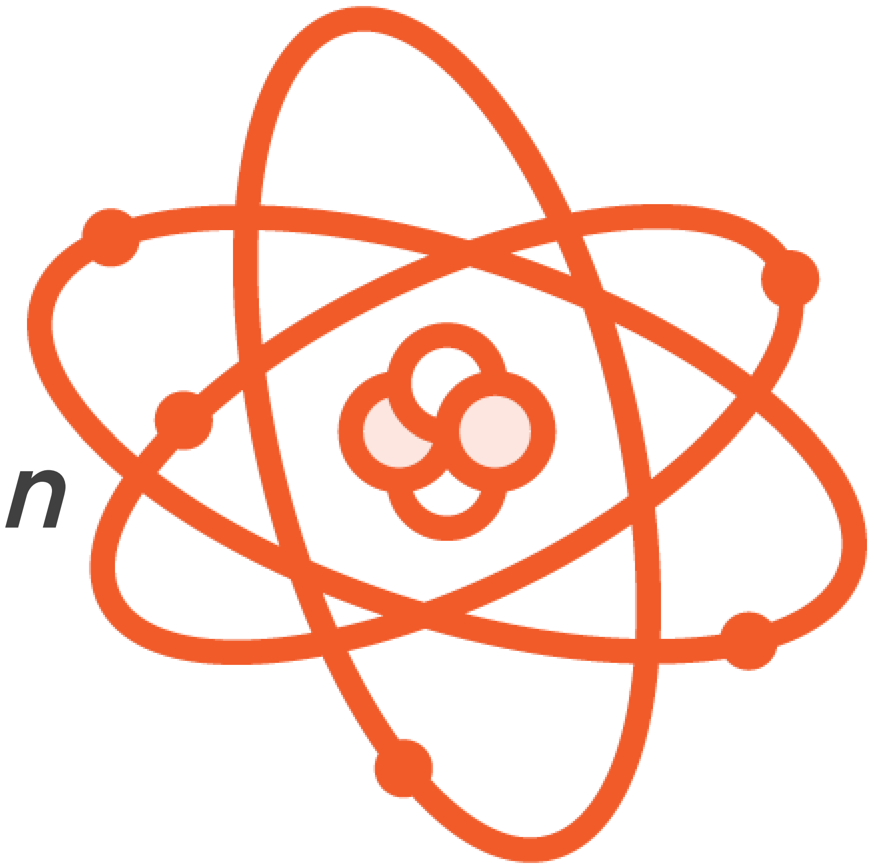
@jzheaux blog.jzheaux.io

Demo

Desmos

zetta-

one billion trillion



1%

```
if (cantChew())
```

```
dontBiteOff()
```

Rules of Secure Data Stewardship

If you don’t need it…

…don’t ask for it

If you don’t need it later on…

…don’t store it

If you need it, but don’t need to know it…

…hash it

Demo

Terracotta Request Logging

Hashes are Queryable

```
select * from requests where session_id = '4321'
```

requestId	sessionId	userId	action	result	duration
1234	4321	(null)	login	success	521
2345	4321	56	search	success	143
3456	5432	57	order	error	32
4567	4321	56	order	success	1024
5678	5432	57	order	success	1057

Hashes are Queryable

```
select * from requests where session_id = ‘4321’
```

requestId	sessionId	userId	action	result	duration
1234	**4321**	**(null)**	**login**	**success**	**521**
2345	**4321**	**56**	**search**	**success**	**143**
3456	5432	57	order	error	32
4567	**4321**	**56**	**order**	**success**	**1024**
5678	5432	57	order	success	1057

Hashes are Queryable

```
select * from requests where hash = hash('4321')
```

requestId	hash	userId	action	result	duration
1234	abcd	(null)	login	success	521
2345	abcd	56	search	success	143
3456	efgh	57	order	error	32
4567	abcd	56	order	success	1024
5678	efgh	57	order	success	1057

Hashes are Queryable

```
select * from requests where hash = hash('4321')
```

requestId	hash	userId	action	result	duration
1234	**abcd**	**(null)**	**login**	**success**	**521**
2345	**abcd**	**56**	**search**	**success**	**143**
3456	efgh	57	order	error	32
4567	**abcd**	**56**	**order**	**success**	**1024**
5678	efgh	57	order	success	1057

Side note: the Servlet spec doesn’t require session ids to be guids

```
MessageDigest md = MessageDigest.getInstance(“SHA-512”);
md.update(“abcd”.getBytes(“UTF-8”))
byte[] hash = md.digest();
```

Hashing in Java

Performs a hash on a given byte array

Supports MD5, SHA-256, SHA-512 and a few others

Cipher

An algorithm applied to text, rendering it unreadable

Cipher Examples

Substitution Ciphers

ROT13: Java is awesome

Wnin vf njrfbzr

Vigenere

Encryption

AES

RSA

```
hash(x, y) = ( x + y ) % 10
// hash(2,3) => 5
// hash(9,6) => 5
// hash(7,8) => 5
```

Hashes

One-way ciphers, computationally difficult to reverse

Information is lost in the process

Typically emits values of the same length

Encoding Examples

Context-sensitive Transmission

URL: http://url?p=http://url2?q=r%26s=t

&s=t

HTML: <textarea>&lt;script&gt;</textarea>

<textarea><script></textarea>

Charset-reducing

Base64

Hex

```
MessageDigest md = MessageDigest.getInstance("SHA-512");
md.update("abcd".getBytes("UTF-8"))
byte[] hash = md.digest();
```

Hashing in Java

~~It's an encryption~~

~~It's an encoding~~

Demo

MessageDigest

Demo

MessageDigest in Terracotta Bank

Demo

Base64 in Terracotta

Wait… Why not just use ISO-8859-1, then?

Byte Encoding

From Bytes to Bytes

Character Encoding

From Bytes to Characters

Demo

MessageDigest -> character encoding

```
String str = “Hello”;

byte[] bytes = str.getBytes(“UTF-8”);

// 01101000 01100101 01101100 01101100 01101111

byte[] hashed = md5Hash(bytes);

// 01011101 01000001 01000000 00101010 …

//   5   D    4   1    4   0    2   A   …
```

Encoding in Hex

No special character mapping

Takes up 50% more space than Base64

Demo

Hex in Terracotta

So, Can I Use MD5?

Typically safer to choose a stronger algorithm

MD5 is not FIPS 140-2 compliant

But, let’s get into some specifics

Hashing Attacks

Collision Resistant: Difficult to create x and x' such that $h(x) = h(x')$

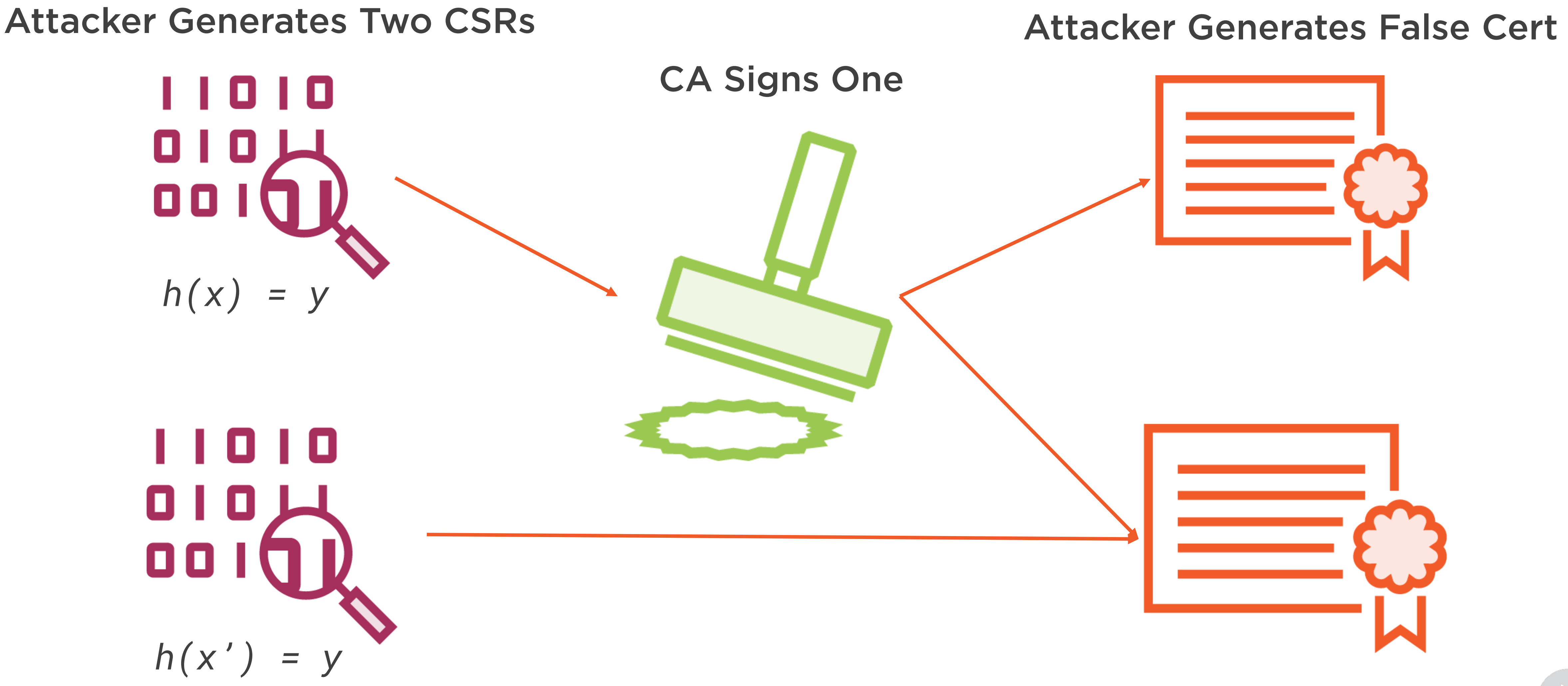
Preimage Resistant: Difficult to find x given only $h(x)$

Second-preimage Resistant: Difficult to find x', given x, such that $h(x) = (x')$

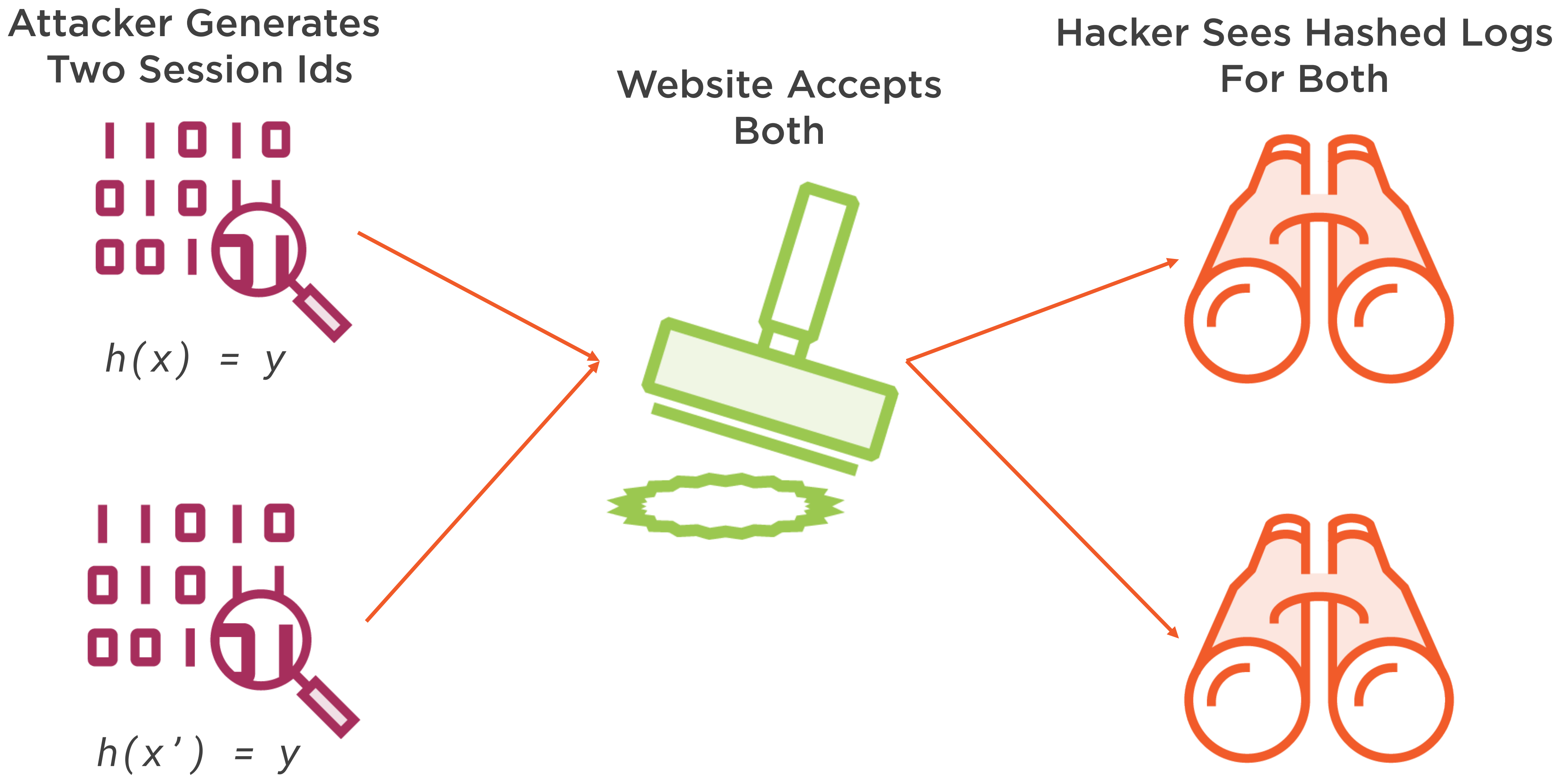
Collision Resistance

Ideal	MD5
An n-bit hash will take 2^(n/2) hashes before a collision is more than 50% likely	MD5 takes 2^24 hashes to form a likely collision, 1 trillion times worse than optimal
A 128-bit hash should take 2^64 hashes before a collision is likely	
	Can’t be used when the user controls both inputs

Hacking SSL Certificates



Hacking Session Ids


Attacker Generates Two Session Ids
$h(x) = y$
$h(x') = y$
Website Accepts Both
Hacker Sees Hashed Logs For Both

Second-preimage Resistance

Ideal	MD5
An n-bit hash will take 2^(n-1) hashes before a second input (preimage) is found that results in the target hash	MD5 takes 2^123 hashes to find a second preimage
A 128-bit hash should take 2^127 hashes before finding a second preimage	
For secure use cases, prefer a work factor of 256 or higher	Can’t be used for secure use cases, but still an option where the user can’t control the input

Demo

JWT

Preimage Resistance

you

639bae9ac6b3e1a84cebb7b403297b79

5d41402abc4b2a76b9719d911017c592

Preimage Resistance

a1e6cd7f9480f01643245e0b648d9fbe

bad

5d41402abc4b2a76b9719d911017c592

Preimage Resistance

ab86a1e1ef70dff97959067b723c5c24

me

5d41402abc4b2a76b9719d911017c592

Preimage Resistance

7d0db380a5b95a8ba1da0bca241abda1

at

5d41402abc4b2a76b9719d911017c592

Preimage Resistance

5d41402abc4b2a76b9719d911017c592

hello

5d41402abc4b2a76b9719d911017c592

Preimage Resistance

Ideal

An n-bit hash will take 2^(n-1) hashes the preimage is found

A 128-bit hash should take 2^127 hashes before finding the preimage

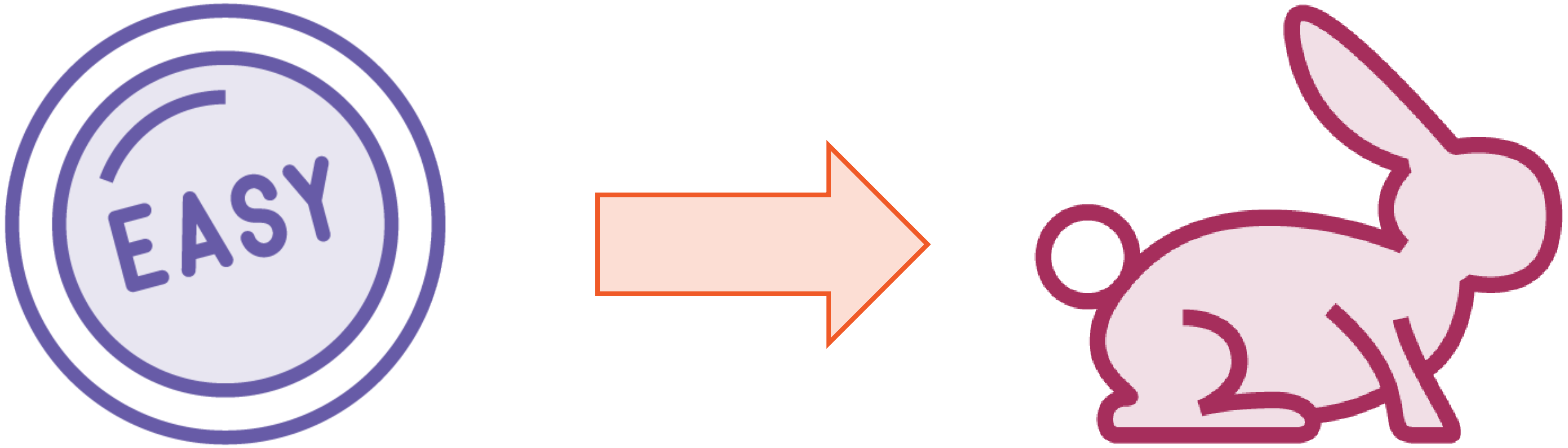
For secure use cases, prefer a work factor of 256 or higher

MD5

MD5 takes 2^123 hashes to find a preimage

Can’t be used for secure use cases, but still an option where the user can’t control the input

But... What About Passwords?



Short, Simple Passwords

Small Search Space

MD5

Don’t use MD5 when the user controls the input

Don’t use MD5 for most secure use cases

MD5 may be used when the user doesn’t control the input

- And possibly when size and speed matter more than integrity

Hashing

The Rules of Secure Data Stewardship

Encrypting, Hashing, Encoding

Java supports MD5 and SHA through MessageDigest

MD5 is not applicable for most security-related scenarios