No Followers? No Botnet? No Problem! Asymmetric Denial of Service Attacks

Bryan Sullivan
Microsoft

Session ID: HT-107

Session Classification: Intermediate

DoS attacks circa 1995

DoS attacks circa 2012

DoS attacks circa 2015

Algorithmic Complexity

RSACONFERENCE
EUROPE 2012

Algorithmic Complexity

Time

Constant O(1)

Items to process

Algorithmic Complexity

Algorithmic Complexity

Time

Items to process

- Constant O(1)
- Linear O(n)
- Quadratic O(n^2)

Algorithmic Complexity

Time

Items to process

- Constant $O(1)$
- Linear $O(n)$
- Quadratic $O(n^2)$
- Exponential $O(2^n)$

Algorithmic Complexity

Time

Items to process

- Constant $O(1)$
- Linear $O(n)$
- Quadratic $O(n^2)$
- Exponential $O(2^n)$
- Factorial $O(n!)$

Hashtable Collision

RSA CONFERENCE EUROPE 2012

Normal Hashtable Operation

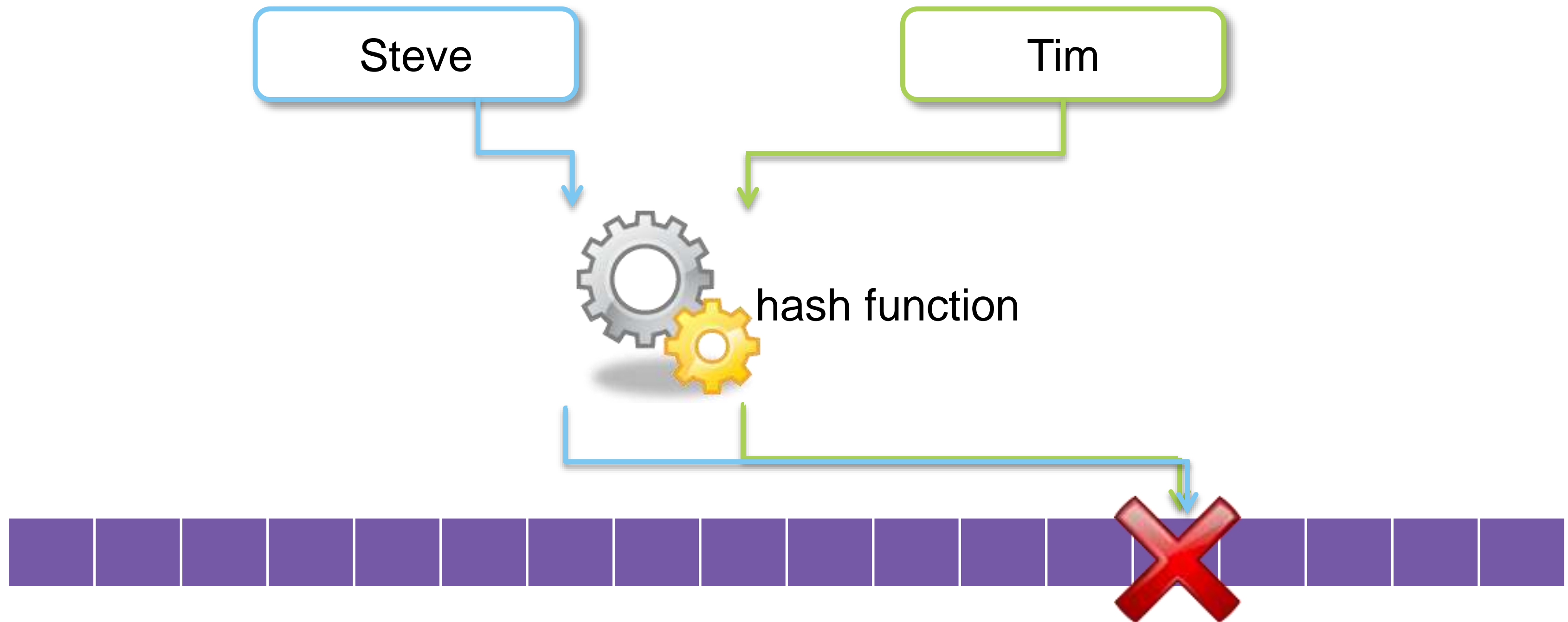
Bryan

Amy

Abigail

hash function

Hashtable Collision



Hashtable Collision

hash function

Steve

Tim

Cryptographic Hash Properties

- Preimage resistance
  - Given a hash h it should be difficult to find any message m such that h = hash(m)

Cryptographic Hash Properties

- Preimage resistance
  - Given a hash h it should be difficult to find any message m such that h = hash(m)
- Second preimage resistance
  - Given an input m1 it should be difficult to find another input m2 - where m1 != m2 - such that hash(m1) = hash(m2)

Cryptographic Hash Properties

- Preimage resistance
  - Given a hash h it should be difficult to find any message m such that h = hash(m)
- Second preimage resistance
  - Given an input m1 it should be difficult to find another input m2 - where m1 != m2 - such that hash(m1) = hash(m2)
- Collision resistance
  - It should be difficult to find two different messages m1 and m2 such that hash(m1) = hash(m2)

Examples

- SHA2 is strong
- CRC32 is weak
  - But very fast
  - Used where cryptographic strength isn't required…

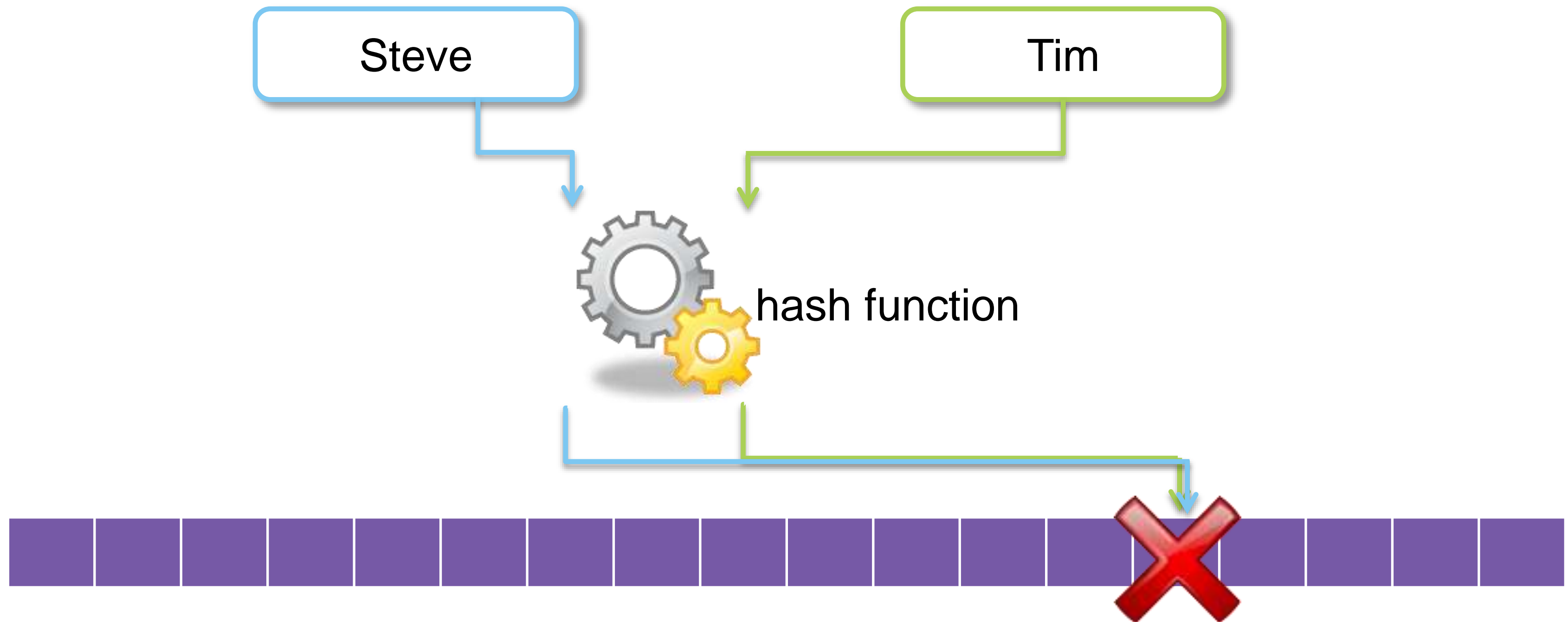
Cryptographic Strength

- When do we need cryptographically strong hashes?

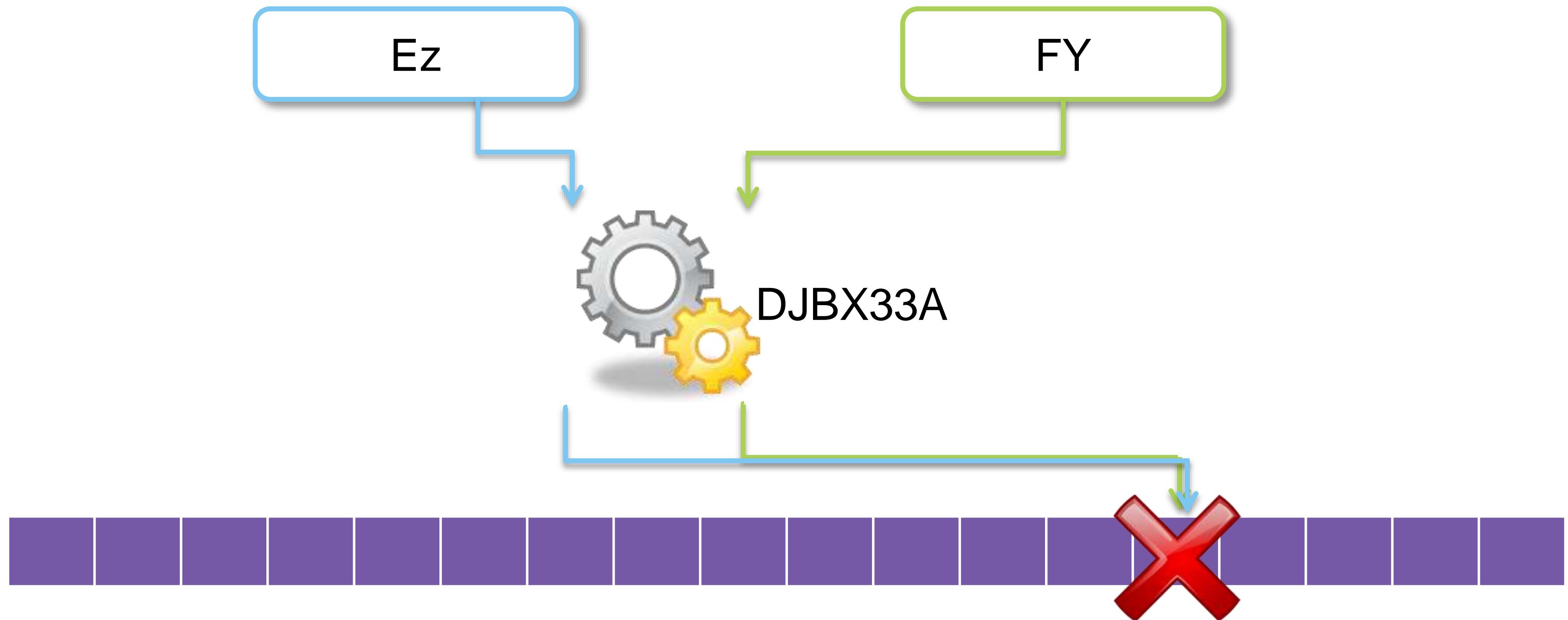
Cryptographic Strength

- When do we need cryptographically strong hashes?
  - Digital signatures
  - Integrity checks
  - Message Authentication Codes
  - Password verification
  - Others…?

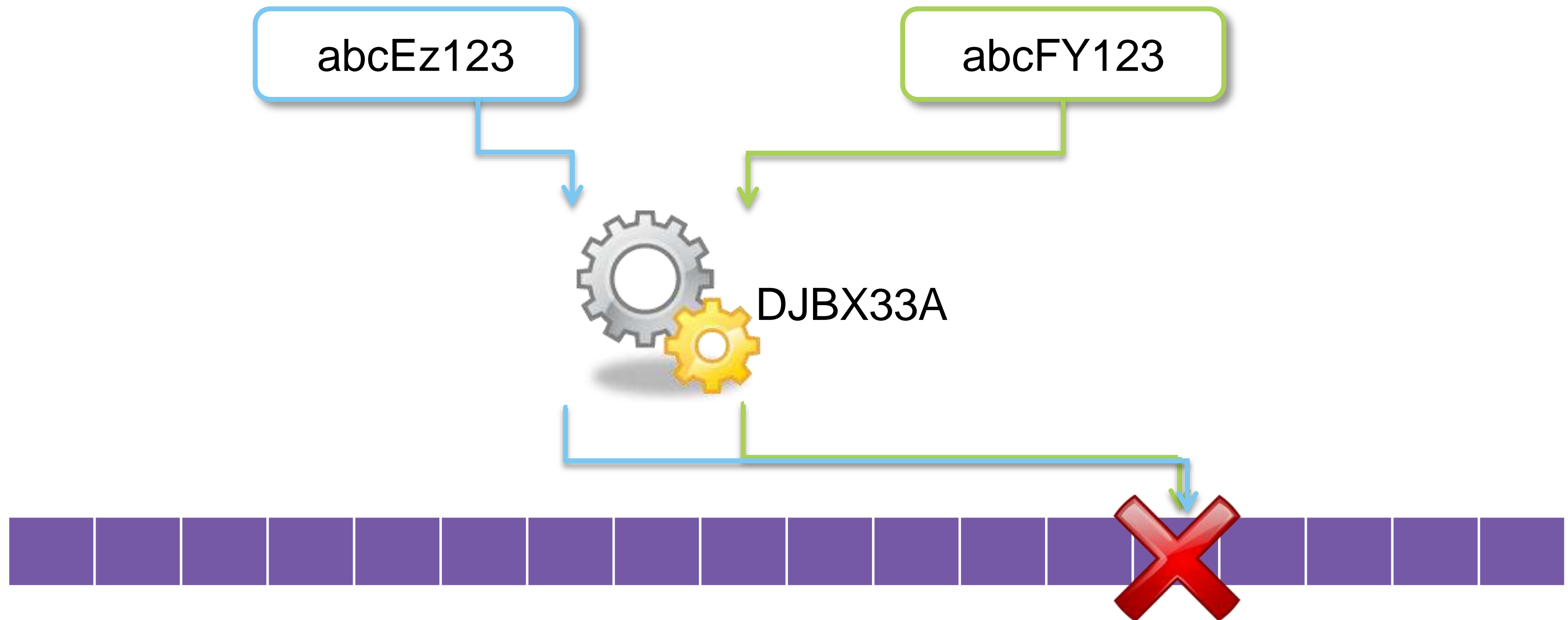
Hashtable Collision



Equivalent Substring Collisions



Equivalent Substring Collisions



Equivalent Substring Collisions

EzEz

EzFY

FYEz

FYFY

DJBX33A

Equivalent Substring Collisions

EzEzEz

EzEzFY

EzFYEz

EzFYFY

FYEzEz

FYEzFY

FYFYEz

FYFYFY

DJBX33A

Cryptographic Hash Properties

- Preimage resistance
  - Given a hash h it should be difficult to find any message m such that h = hash(m)
- Second preimage resistance
  - Given an input m1 it should be difficult to find another input m2 - where m1 != m2 - such that hash(m1) = hash(m2)
- Collision resistance
  - It should be difficult to find two different messages m1 and m2 such that hash(m1) = hash(m2)

Demo: Equivalent Substring Attack

RSACONFERENCE EUROPE 2012

Affected Technologies

- ASP.NET (patched in advisory 2659883)
- PHP (patched in version 5.3.9)
- Java (Tomcat patched in 5.5.35, 6.0.35, 7.0.23)
- Python (patched in 2.6.8, 2.7.3, 3.1.5, 3.2.3)

Defensive Strategies

- Keep your frameworks patched
- Use cryptographically strong hash algorithms for hashtables

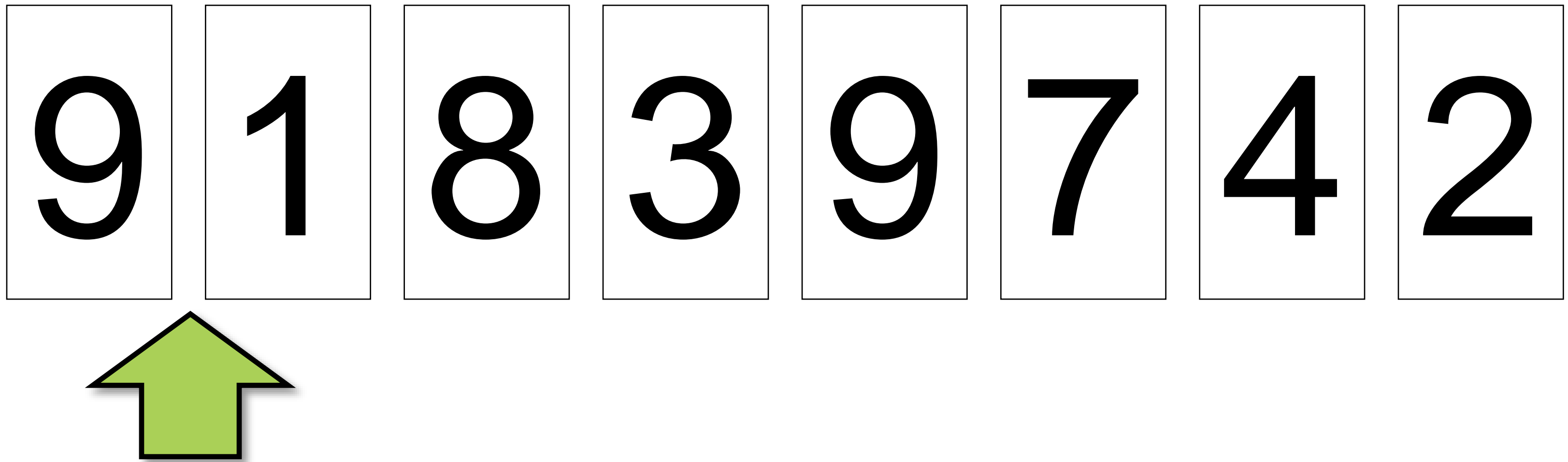
Sorting

RSA CONFERENCE EUROPE 2012

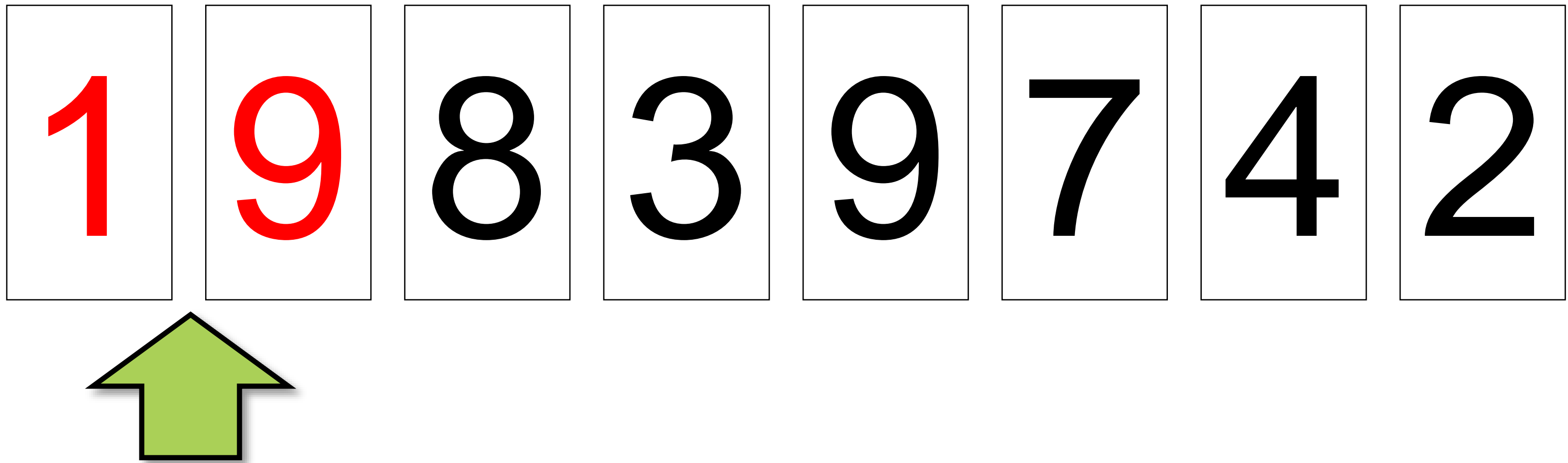
Bubble Sort

9	1	8	3	9	7	4	2

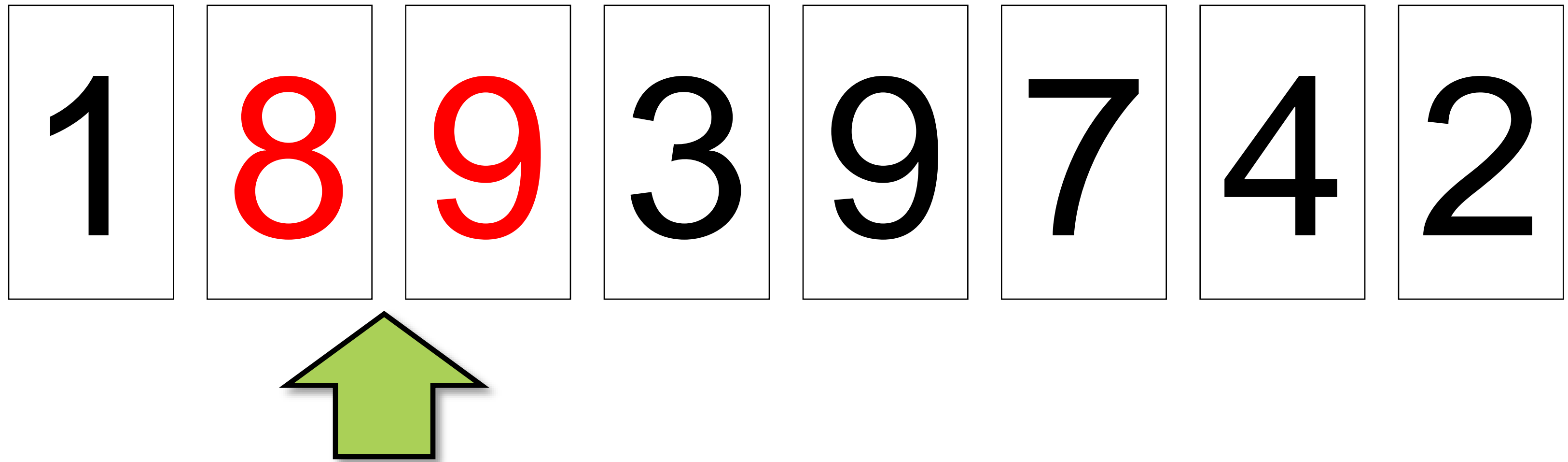
Bubble Sort



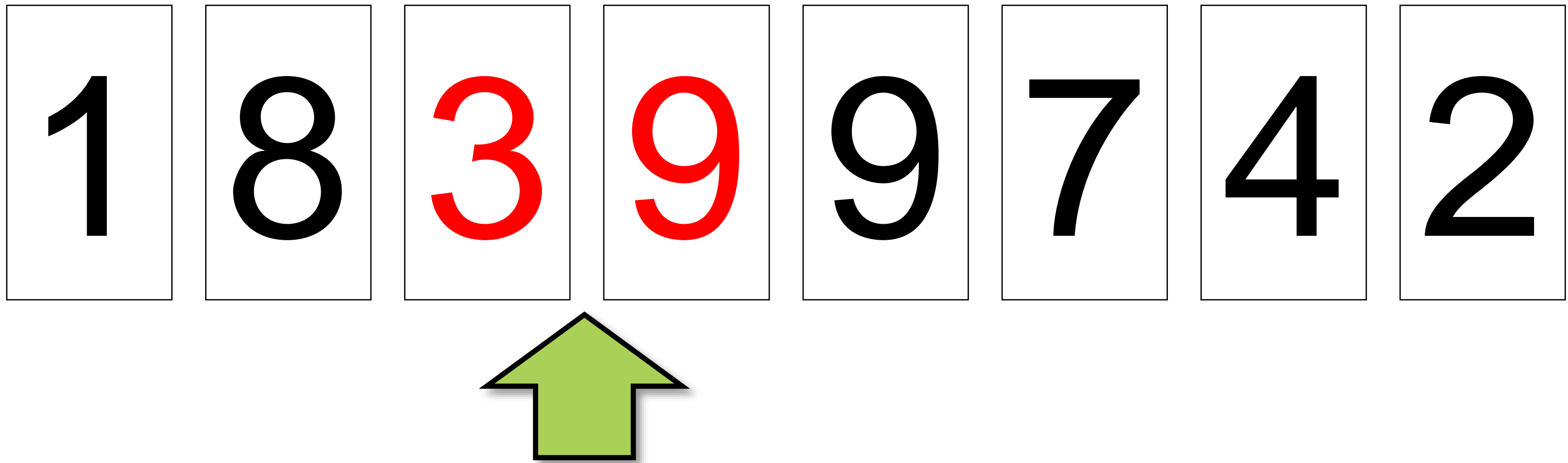
Bubble Sort



Bubble Sort



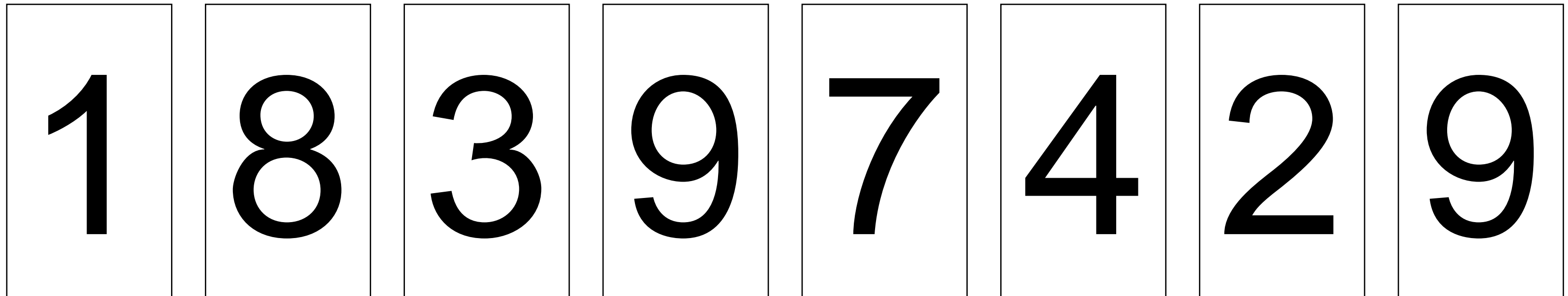
Bubble Sort



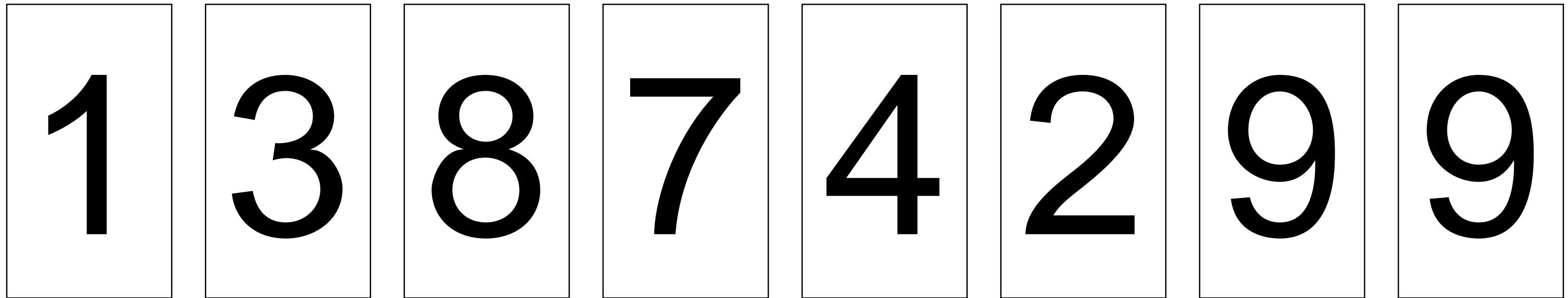
Bubble Sort

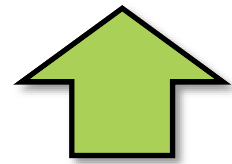
1	8	3	9	7	4	2	9

Bubble Sort

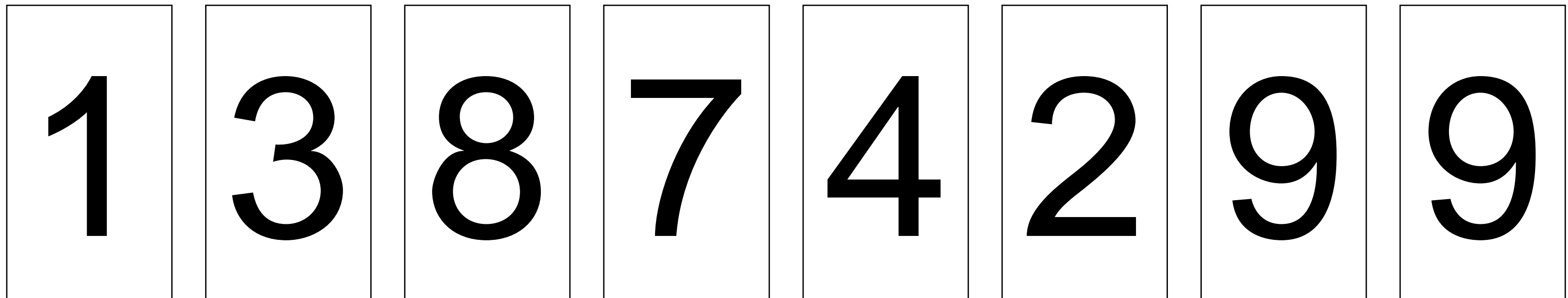


Bubble Sort





Bubble Sort

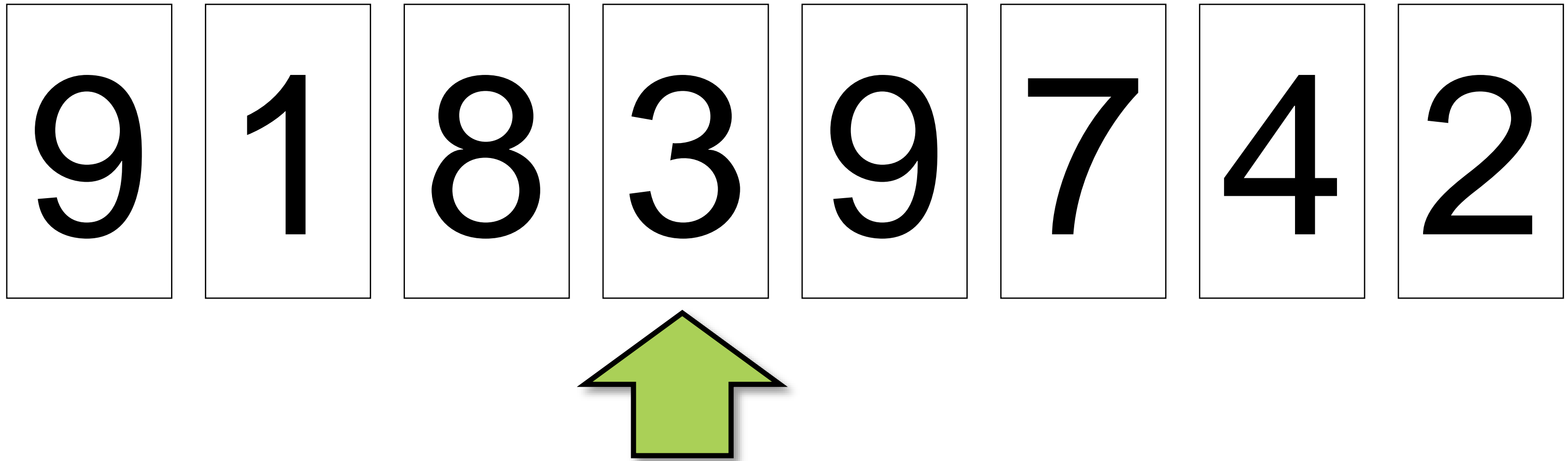


- Runs in $O(n^2)$ time
- Inefficient and vulnerable

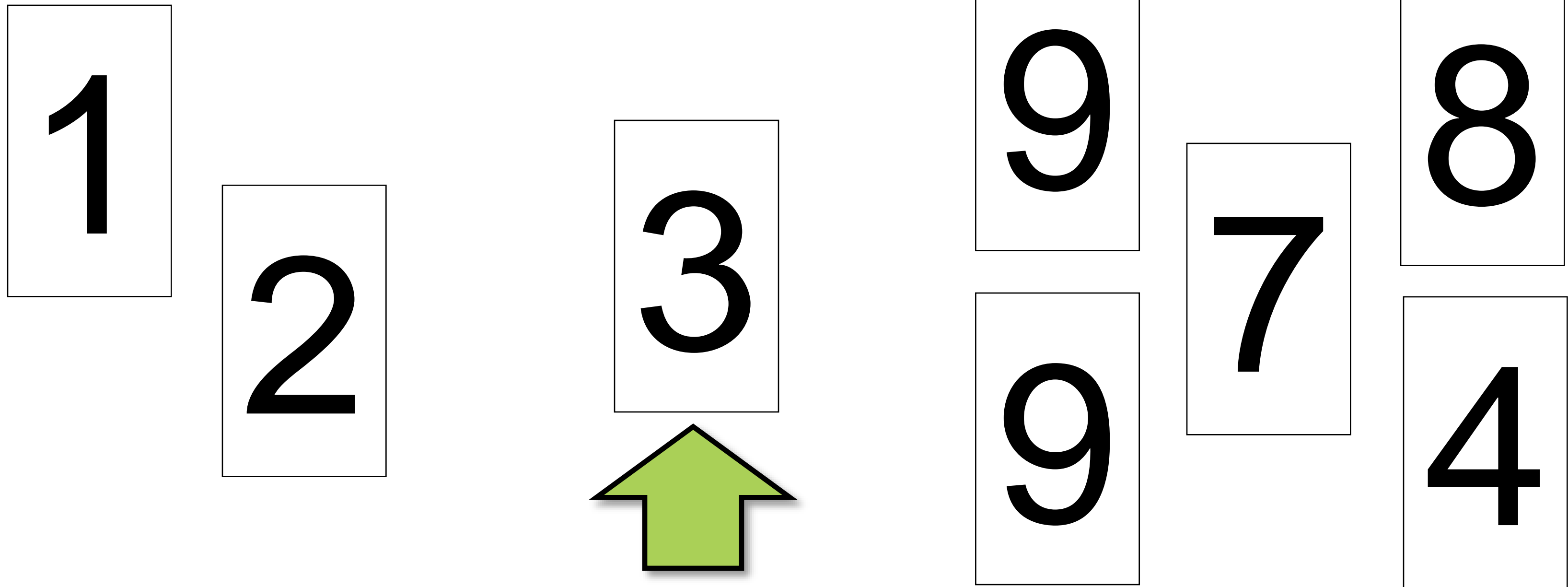
Quick Sort

9	1	8	3	9	7	4	2

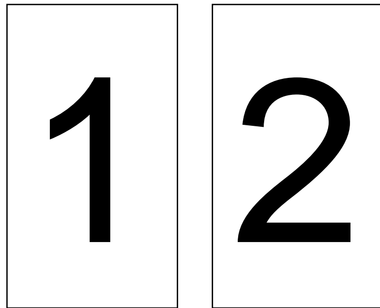
Quick Sort

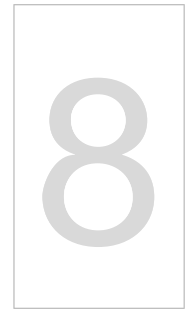


Quick Sort

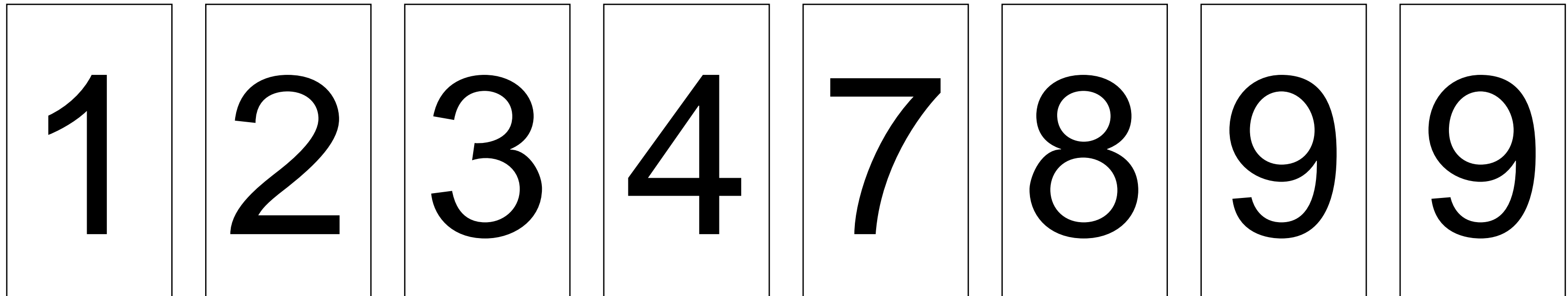


Quick Sort





Quick Sort



- Usually runs in O(n log n) time
- …but what's the worst-case scenario?

Demo: Quicksort Attack

RSACONFERENCE EUROPE 2012

Defensive Strategies

- Use the most efficient sorting algorithm available
  - Judge this on worst-case, not usual behavior
- Limit input size when possible

Regular Expressions

Regex Example 1

^\d+$ vs ^(\d|\d\d)+$

Input length (chars)	32	33	34	35	36	37	38

Time (s)

60
50
40
30
20
10
0

Legend: ^\d+$; ^(\d|\d\d)+$

Regex Example 2

^.*a vs .*a

Input length (chars)	Time (s)

Chart axis labels: Time (s): 0, 50, 100, 150, 200, 250, 300; Input length (chars): 0, 20000, 40000, 60000, 80000, 100000, 120000. Legend: ^.*a, .*a

Expert advice…

- "…the developer should be able to define a very strong validation pattern, usually based on regular expressions, for validating [user] input."

OWASP SQL Injection Prevention Cheat Sheet

- "Regular expressions are a good way to validate text fields such as names, addresses, phone numbers, and other user information."

MSDN Patterns & Practices

- "Regex is a perfect tool for input validation."

Bryan Sullivan, Ajax Security

Demo: ReDoS

RSA CONFERENCE EUROPE 2012

More irony…

- "Just as we perform whitelist input validation on the server for security purposes, developers must perform client-side validation to ensure security of their offline applications."

Ajax Security

Apply Your Knowledge

- Good or bad?

```
^([a-zA-Z0-9_\-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([a-zA-Z0-9\-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$
```

Apply Your Knowledge

- Good or bad?

^([a-zA-Z0-9_\-\.]+) …]{1,3}\.[0-
9]{1,3}\.[… -zA-Z0-9\-
]+\.)+))([a … -9]{1,3})(\]?)$

Apply Your Knowledge

- Good or bad?

```
^(ht|f)p(s?)\\:\\/\\/[0-9a-zA-Z]([-.\\w]*[0-
9a-zA-Z])*(:(0-9)*)*(\\/?)([a-zA-Z0-9\\-
\\.\\?\\,\\:\\'\\/\\\\\\+=&amp;%\\$#_]*)?$
```

Apply Your Knowledge

- Good or bad?

```
^(ht|f)p(s?)\ [0-9a Z]([-.\\w]*[0-
9a-zA-Z])*(:(0 *) )([a-zA-Z0-9\\-
\\.\\?\\,\\:\\'\\/\\\\ mp;%\\$#_]*)?$
```

Apply Your Knowledge

- Good or bad?

```
^((([a-zA-Z\'\.\-]+)?)((,\s*([a-zA-
Z]+))?)|([A-Za-z0-9](([_\.\-]?[a-zA-Z0-
9]+)*)@([A-Za-z0-9]+)(([\.\-]?[a-zA-Z0-
9]+)*)\.([A-Za-z]{2,})))(;{1}(((([a-zA-Z\'\.\-
]+){1})((,\s*([a-zA-Z]+))?))|([A-Za-z0-
9](([_\.\-]?[a-zA-Z0-9]+)*)@([A-Za-z0-
9]+)(([\.\-]?[a-zA-Z0-9]+)*)\.([A-Za-
z]{2,})){1}))*$
```

Apply Your Knowledge

- Good or bad?

```
^((([a-zA-Z\'\.\-]+)?)((,\s*([a-zA-
Z]+))?)|([A[illegible]0-9](([illegible]]?[a-zA-Z0-
9]+)*)@([A-Za[illegible]9][illegible]\.\-]?[a-zA-Z0-
9]+)*)\.([A-Za-z]{2[illegible]1}(((([a-zA-Z\'\.\-
]+){1})((,\s*([a[illegible]?))|([A-Za-z0-
9](([_\.\-]?[a[illegible]Z0-9[illegible]@([A-Za-z0-
9]+)(([\.\-]?[illegible]zA-Z0-9[illegible]*)\.([A-Za-
z]{2,})){1}))*$
```

Demo: SDL Regex Fuzzer

Words of Wisdom?

- “Some people, when confronted with a problem, think ‘I know, I’ll use regular expressions.’ Now they have two problems.”

Jamie Zawinski

Defensive Strategies

- Do keep using regular expressions
- Watch for danger patterns
  - Grouping expressions containing repetition that are themselves repeated, such as (\d+)+
  - Grouping expressions containing alternation where the alternates overlap, such as (\d|\d\d)+
- Test regexes with the SDL Regex Fuzzer
  - Even if (especially if?) you get them from 3rd parties

XML Entity Expansion

RSACONFERENCE EUROPE 2012

XML Entities

```
<!DOCTYPE employees [
<!ENTITY companyname "Contoso, Inc.">
]>
<employees>
  <employee>Amy S, &companyname;</employee>
  <employee>Abigail S, &companyname;</employee>
</employees>
```

Quadratic Entity Expansion

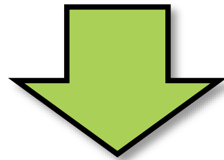
```
<!DOCTYPE employees [
<!ENTITY a "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa…">
]>
<employees>
  <employee>&a;&a;&a;&a;&a;&a;&a;…</employee>
</employees>
```

Exponential Entity Expansion

```
<!DOCTYPE lolz [
<!ENTITY lol1 "lol">
<!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;…">
<!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;…">
<!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;…">
…
<!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;…">
]>
<lolz>&lol9;</lolz>
```

Exponential Entity Explosion…

`<lolz>&lol9;</lolz>`



lollollollollollollollollollollollollollollollollollollollololloll

lollollollollollollollollollollollollollollollollollollollollollol

llollollollollollollollollollollollollollollollollollollollollollo

llollollollollollollollollollollollollollollollollollollollollollo

3GB of LOLs

llollollollollollollollollollollollollollollollollollollollollollo

llollollollollollollollollollollollollollollollollollollollollol…

Demo: XML Entity Expansion Attack

RSA CONFERENCE EUROPE 2012

Defensive Strategies

- Disable inline DTD resolution
- Disable entity resolution
- Disable recursive entity resolution
- Limit the maximum number of characters that can be expanded via entity resolution

- Disable external entity resolution
- Limit the time spent on entity resolution

Algorithmic Complexity

Time

Items to process

- Constant O(1)
- Linear O(n)
- Quadratic O(n^2)
- Exponential O(2^n)
- Factorial O(n!)

Algorithmic Complexity

$O(\infty)$

Time

Items to process

- Constant $O(1)$
- Linear $O(n)$
- Quadratic $O(n^2)$
- Exponential $O(2^n)$
- Factorial $O(n!)$

Infinite Loop Attacks

RSA CONFERENCE EUROPE 2012

PHP/Java "Magic Number"

- Vulnerability occurred when a string was converted to a floating point value:

```
<?php $f = (float)'2.2250738585072012e-308'; ?>
```

```
Double.parseDouble("2.2250738585072012e-308");
```

- Execution path leads to the C function strtod()

Big Data Queries

- NoSQL data is usually unstructured
- No “SELECT column FROM table” equivalent

```
$q =  'function() { var search_date = \'' .
     $_GET['date'] . '\';' .
     'return this.bday == search_date; }';

$collection->find(array('$where' => $q));
```

Defensive Strategies

- Keep your frameworks patched (again)
- Validate user input
  - with, um, regular expressions

Wrapping Up

RSACONFERENCE EUROPE 2012

Respect the Danger of DoS

STRIDE

Apply Your Knowledge

- Stay patched
- Determine your worst-case algorithmic complexity
  - Use analysis tools when possible (like Regex Fuzzer)
  - Swap algorithms to lower complexity if feasible
  - Tightly control user input when not feasible
  - Sandbox processing when you can’t control user input

A Silver Lining

Special thanks to Amit Klein, Alex Roichman, Adar Weidman, Rick Regan, M.D McIlroy, Alexander Klink and Julian Wälde.