PL Group

Jeremy Gibbons

Geraint Jones

Daniel Kroening

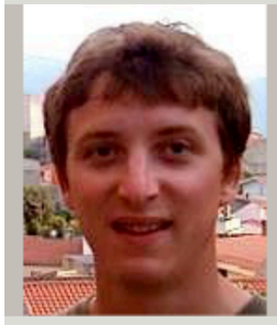
Anthony W. Lin

Andrzej Murawski

Hanno Nickau

Luke Ong

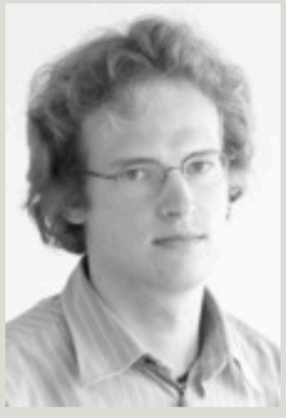
Mike Spivey



Sam Staton

Bernard Sufrin

String Analysis and Applications to Software Security



Daniel Kroening

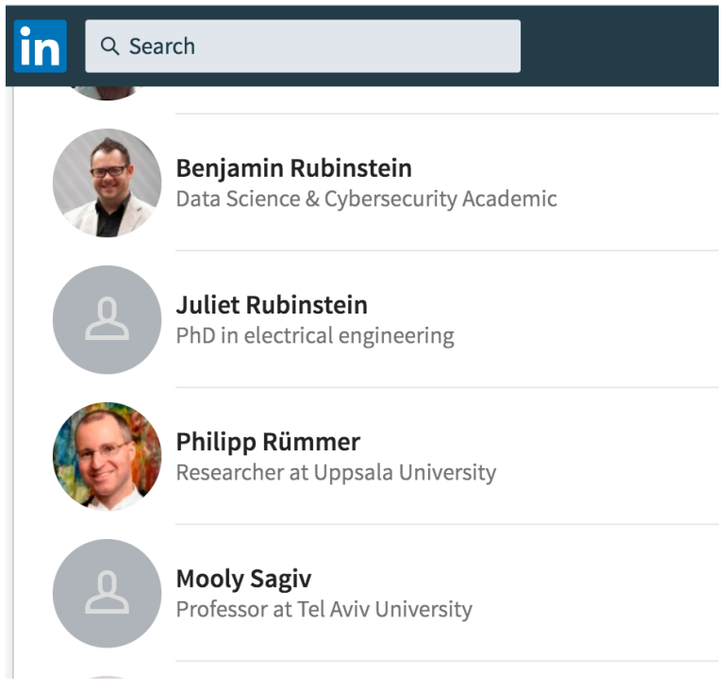
Anthony W. Lin

String Data Type

Prevalent in today's software

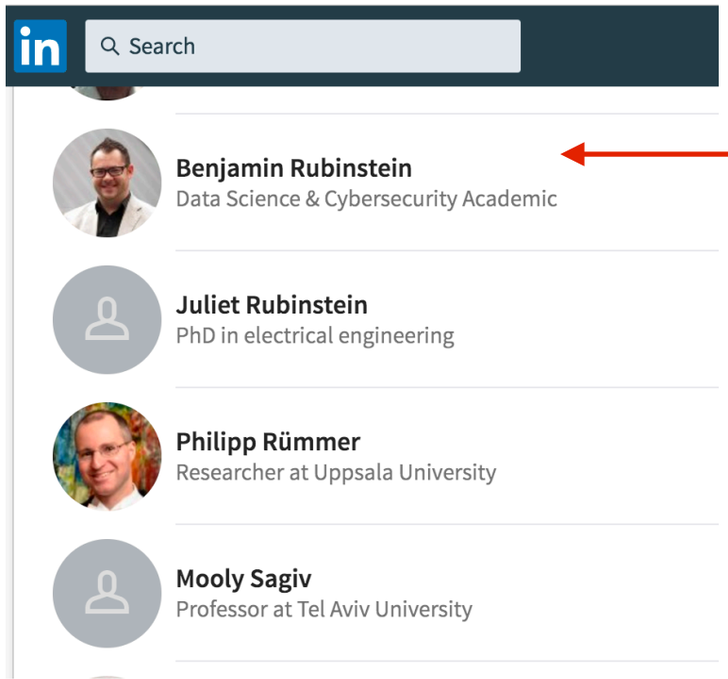
String Data Type

Prevalent in today's software



String Data Type

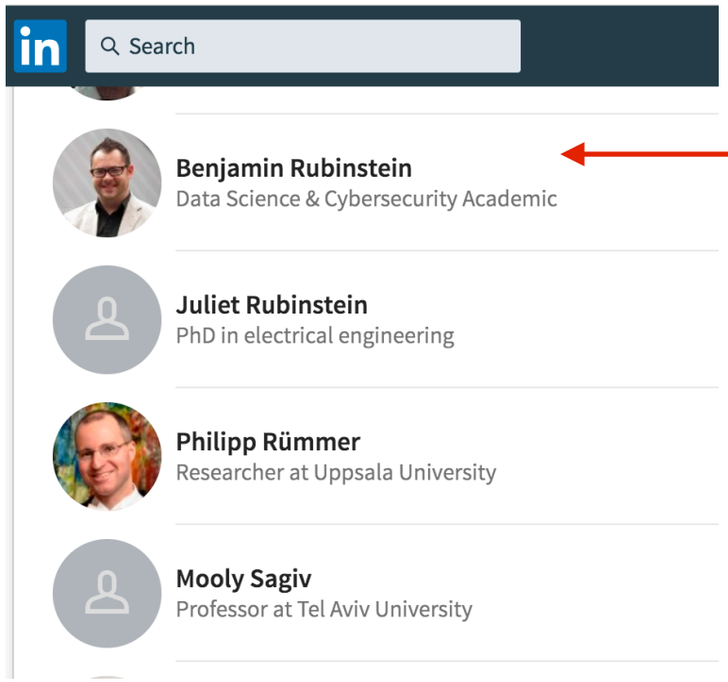
Prevalent in today's software



Ben

String Data Type

Prevalent in today's software



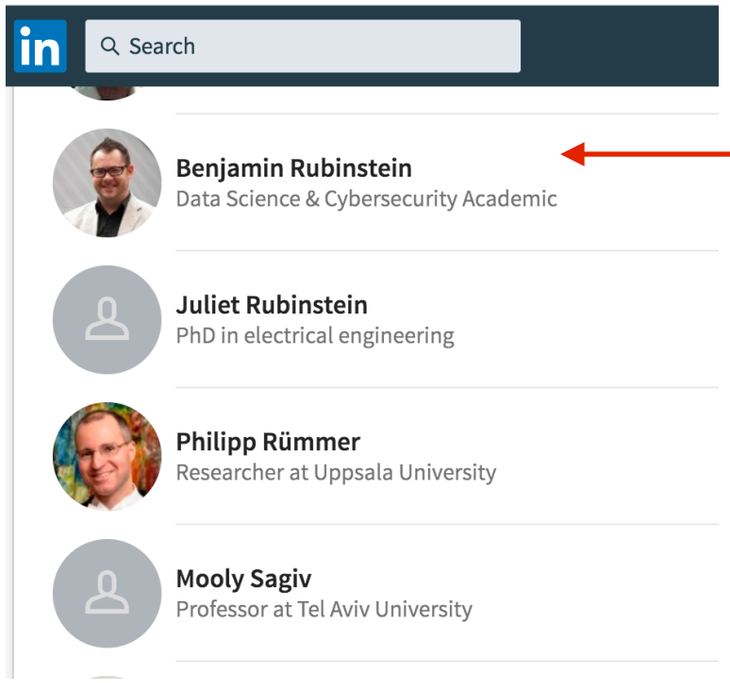
Ben

Dynamically generated by

```
var x = htmlEscape(name);
var y = escapeString(x);
nameElem.innerHTML = '<a onclick=' +
    '"viewPerson(\'' + y + '\')">' + x + '</a>';
```

String Data Type

Prevalent in today's software



Ben

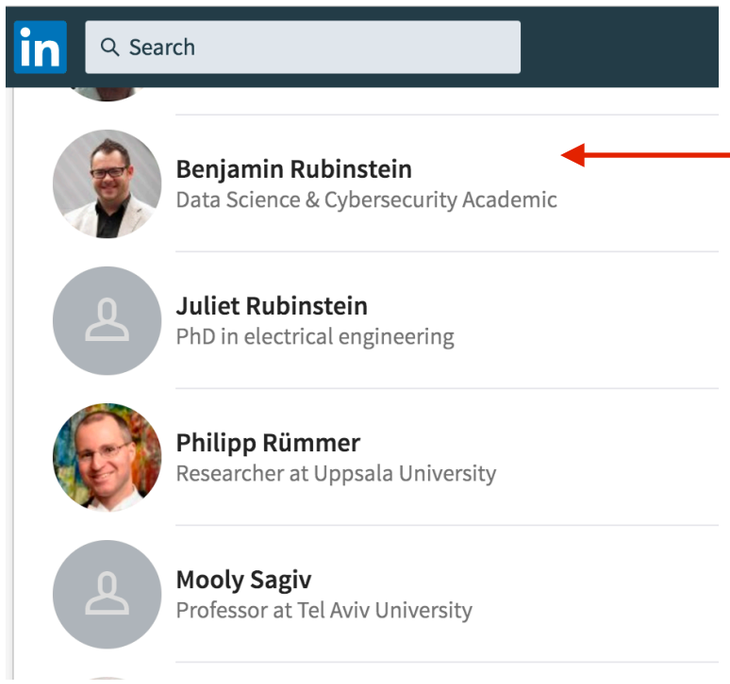
Dynamically generated by

```
var x = htmlEscape(name);
var y = escapeString(x);
nameElem.innerHTML = '<a onclick=' +
    '"viewPerson(\'' + y + '\')">' + x + '</a>';
```

Many string-related bugs — hard to find by random testing

String Data Type

Prevalent in today's software



Ben

Dynamically generated by

```
var x = htmlEscape(name);
var y = escapeString(x);
nameElem.innerHTML = '<a onclick=' +
    '"viewPerson(\'' + y + '\')">' + x + '</a>';
```

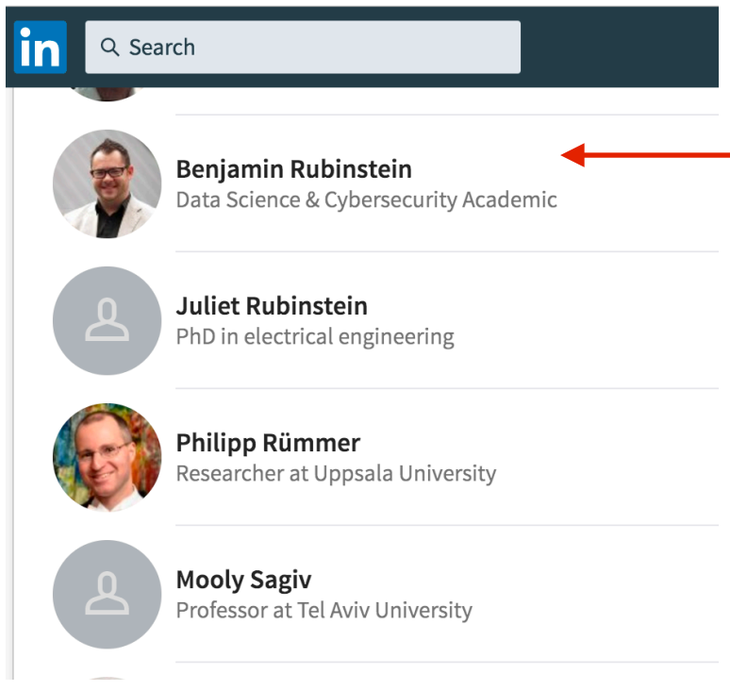
Many string-related bugs — hard to find by random testing

 ……

XSS

String Data Type

Prevalent in today's software



Ben

Dynamically generated by

```
var x = htmlEscape(name);
var y = escapeString(x);
nameElem.innerHTML = '<a onclick=' +
    '"viewPerson(\'' + y + '\')">' + x + '</a>';
```

Many string-related bugs — hard to find by random testing

 ……

XSS

Q: Is the program safe?

Approach from Computational Logic

Approach: view program analysis as constraint solving

Approach from Computational Logic

Approach: view program analysis as constraint solving

Program

```
void foo(String x,String y):
   String v := x + ‘ab’ + y
   String w := y + ‘ba’ + x
   if ( v == w ) { ... }
   …
```

Find x,y that satisfies v == w

Approach from Computational Logic

Approach: view program analysis as constraint solving

Program

```
void foo(String x,String y):
   String v := x + ‘ab’ + y
   String w := y + ‘ba’ + x
   if ( v == w ) { ... }
   …
```

Find x,y that satisfies v == w

Constraint

```
(v = x + ‘ab’ + y) AND
(y = y + ‘ba’ + x) AND
(v = w)
```

Find string substitutions to match equations (satisfiability problem)

Key Challenge

Develop effective **logics** and **solvers** for string analysis problem

Which string operations to permit?

Which ones lead to decidability?

If undecidable but important operations, what can we do?

Key Challenge

Develop effective **logics** and **solvers** for string analysis problem

Which string operations to permit?

Concatenation, regex matching, length functions, …

Which ones lead to decidability?

If undecidable but important operations, what can we do?

Key Challenge

Develop effective **logics** and **solvers** for string analysis problem

Which string operations to permit?

Concatenation, regex matching, length functions, …

Which ones lead to decidability?

Many Long-standing open problems

If undecidable but important operations, what can we do?

Key Challenge

Develop effective **logics** and **solvers** for string analysis problem

Which string operations to permit?

Concatenation, regex matching, length functions, ...

Which ones lead to decidability?

Many Long-standing open problems

If undecidable but important operations, what can we do?

We address all aspects of these challenges

A 50-year-old open problem

Practical string analysis requires concatenation and length constraints

(v = x + ‘ab’ + y) AND
(y = y + ‘ba’ + x) AND
(v = w)

len(x) = len(y)

<u>Question</u>: Is decidable to check satisfiability of such constraints?