Will Computers Ever Be Secure?

Achim D. Brucker

a.brucker@sheffield.ac.uk http://www.brucker.uk/

Department of Computer Science, The University of Sheffield, Sheffield, UK

Pint of Science
Accessing The World’s Information
The Roco, 338 Glossop Road, Sheffield, S10 2HW, United Kingdom
Will Computers Ever Be Secure?:

Abstract

These days, it feels like news reports about data security breaches are commonplace. It looks like as if the attackers won and securing IT systems is a Sisyphean task. In this talk, I will motivate the challenges of building secure systems and provide insights into the (fundamental) questions if we can build a computer program that decides if a system secure or not.
Information of more than 550,000,000 accounts leaked.
Information of more than 550,000,000

Example (LinkedIn, May 2016)

- 164 million email addresses and passwords
- from an attack in 2012, offered for sale May 2016
- Compromised data:
  - email addresses
  - passwords
nearly 157,000 customer records leaked
nearly 16,000 records included bank details
more than 150,000 customers lost
(home services market share fall by 4.4 percent in terms of new customers)
Costs for TalkTalk: around any £60 million
Example (Ashley Madison, July 2015)

- More than 30 million email addresses and much more
- Compromised data:
  - Dates of birth
  - Email addresses
  - Ethnicities, Genders
  - Sexual preferences
  - Home addresses, Phone numbers
  - Payment histories
  - Passwords, Usernames, Security questions and answers
  - Website activity

**Similar Leak:** Mate1 in February 2016:
27 million records with even more personal details
(e.g., drinking/drug habits, parenting plans, political views)
How Would You Attack An IT Systems?

Example

Let's look at two different technical attacks...
How Would You Attack An IT Systems?

Social

“Just ask kindly enough for username and password”
How Would You Attack An IT Systems?

Social

“Just ask kindly enough for username and password”

Technical

“Use the system in an unintended way (or take advantage of a flaw/bug)”
How Would You Attack An IT Systems?

Social

“Just ask kindly enough for username and password”

Technical

“Use the system in an unintended way (or take advantage of a flaw/bug)”

Example

Let’s look at two different technical attacks . . .
Example 1: How To Log Into A System Without Password?

Internal program:

```
SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
```

Using the combination "test" and "secret":

```
SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
```

Now let's try something different

```
SELECT * FROM 'users' WHERE;
```

disabling the

WHERE-clause (condition)
Example 1: How To Log Into A System Without Password?

- Internal program:

```sql
SELECT * FROM `users` WHERE `name` = 'Username' AND `pwd` = 'Password';
```

Using the combination "test" and "secret":

```sql
SELECT * FROM `users` WHERE `name` = 'test' AND `pwd` = 'secret';
```

Now let's try something different

```sql
SELECT * FROM `users` WHERE ;
```

disabling the `WHERE`-clause (condition)
Example 1: How To Log Into A System Without Password?

- **Internal program:**
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- Using the combination “test” and “secret”:
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```
Example 1: How To Log Into A System Without Password?

- Internal program:

```
SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
```

- Using the combination “test” and “secret”:

```
SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
```
Example 1: How To Log Into A System Without Password?

- **Internal program:**

  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- **Using the combination “test” and “secret”:**

  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```
Example 1: How To Log Into A System Without Password?

- **Internal program:**
  
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- Using the combination “test” and “secret”:
  
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```

- Now let’s try something different
Example 1: How To Log Into A System Without Password?

- **Internal program:**

  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- **Using the combination “test” and “secret”:**

  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```

- **Now let’s try something different**

  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = ' OR '1'='1';
  ```
Example 1: How To Log Into A System Without Password?

- **Internal program:**
  
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- **Using the combination “test” and “secret”:**
  
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```

- **Now let’s try something different**
  
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = ' ' OR TRUE;
  ```
Example 1: How To Log Into A System Without Password?

- **Internal program:**
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'Username' AND 'pwd' = 'Password';
  ```

- **Using the combination “test” and “secret”:**
  ```sql
  SELECT * FROM 'users' WHERE 'name' = 'test' AND 'pwd' = 'secret';
  ```

- **Now let’s try something different**
  ```sql
  SELECT * FROM 'users' WHERE TRUE;
  ```

- **disabling the WHERE-clause (condition)**
Example 2: How To Pretend To Be Somebody Else?

- Alice wants to be sure that she talks to Bob (authenticity)

\[
\text{Alice} \quad \text{Bob}
\]
Example 2: How To Pretend To Be Somebody Else?

- Alice wants to be sure that she talks to Bob (authenticity)

A Nonce is a fresh secret only known to the person generating it.
Example 2: How To Pretend To Be Somebody Else?

- Alice wants to be sure that she talks to Bob (authenticity)

Needham and Schroeder proposed in 1978 the following protocol (NSPK):

\[ \{A, N_A\}_{B} \]

"I am Alice and here is my Nonce \( N_A \)."

\[ \{N_A, N_B\}_A \]

"Here is your Nonce \( N_A \) and I have one for you."

\[ \{N_B\}_B \]

"I got it! It is \( N_B \)."

A Nonce is a fresh secret only known to the person generating it.
Example 2: How To Pretend To Be Somebody Else? (Correctness)

Goal

After executing the protocol successfully, Alice and Bob can be sure to talk to each other (and not to somebody else).

Correctness argument (informal):

{A, N_A} \rightarrow_B \text{ “This is Alice and I have chosen a nonce } N_A.”

{N_A, N_B} \rightarrow_A \text{ “Here is your nonce } N_A.
Since I could read it, I must be Bob. I also have a challenge } N_B \text{ for you.”}

{N_B} \rightarrow_B \text{ “You sent me } N_B. \text{ Since only Alice can read this and I sent it back, you must be Alice.”}
Example 2: How To Pretend To Be Somebody Else? (Correctness)

Goal

After executing the protocol successfully, Alice and Bob can be sure to talk to each other (and not to somebody else).

Correctness argument (informal):

\[ \{A, N_A\} \rightarrow_{B} \text{“This is Alice and I have chosen a nonce } N_A.\text{”} \]

\[ \{N_A, N_B\} \rightarrow_{A} \text{“Here is your nonce } N_A. \text{ Since I could read it, I must be Bob. I also have a challenge } N_B \text{ for you.”} \]

\[ \{N_B\} \rightarrow_{B} \text{“You sent me } N_B. \text{ Since only Alice can read this and I sent it back, you must be Alice.”} \]

Protocols are typically small and convincing . . .
Example 2: How To Pretend To Be Somebody Else? (Correctness)

**Goal**

After executing the protocol successfully, Alice and Bob can be sure to talk to each other (and not to somebody else).

**Correctness argument (informal):**

- \(\{A, N_A\}_B\) → “This is Alice and I have chosen a nonce \(N_A\).”
- \(\{N_A, N_B\}_A\) → “Here is your nonce \(N_A\). Since I could read it, I must be Bob. I also have a challenge \(N_B\) for you.”
- \(\{N_B\}_B\) → “You sent me \(N_B\). Since only Alice can read this and I sent it back, you must be Alice.”

Protocols are typically small and convincing . . . and often wrong!
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \xrightarrow{\{A, N_A\}_E} E$
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \xrightarrow{\{A, N_A\}_E} E$

2. $E_A \xrightarrow{\{A, N_A\}_B} B$
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \{A, N_A\} \rightarrow E$
2. $E_A \{A, N_A\} \rightarrow B$
3. $E_A \{N_A, N_B\} \leftarrow B$

A.D. Brucker The University of Sheffield
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \xrightarrow{\{A, N_A\}_E} E$

2. $E_A \xrightarrow{\{A, N_A\}_B} B$

3. $E_A \xleftarrow{\{N_A, N_B\}_A} B$

4. $A \xleftarrow{\{N_A, N_B\}_A} E$
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \{A, N_A\} \rightarrow E$
2. $E \{A, N_A\} \rightarrow B$
3. $E \{N_A, N_B\} \leftarrow B$
4. $A \{N_B\} \rightarrow E$
5. $A \{N_B\} \rightarrow E$

A.D. Brucker  The University of Sheffield

Will Computers Ever Be Secure?

May 25, 2016 8
Example 2: How To Pretend To Be Somebody Else? (Attack)

1. $A \xrightarrow{\{A, N_A\}_{E}} E$

2. $E_A \xleftarrow{\{A, N_A\}_{B}} B$

3. $E_A \xrightarrow{\{N_A, N_B\}_{B}} A$

4. $A \xleftarrow{\{N_A, N_B\}_{A}} E$

5. $A \xrightarrow{\{N_B\}_{E}} E$

6. $E_A \xrightarrow{\{N_B\}_{B}} B$

Bob believes he is speaking with Alice!
My Research Vision
My Research Vision

Tool Support for Building Secure, Safe, and Reliable IT Systems

A machine that takes
- a computer program as input and provides as output
- an answer if the program is secure (correct, safe, reliable)
- concrete instruction how to “fix the program” (if necessary)
Let’s start with a simpler case:
Detect if an app crashes (or freezes your phone)
Question: Can We Realise My Dream?
https://thorehusfeldt.net/2012/06/25/the-freeze-app-does-not-exist/

- Let’s start with a simpler case: Detect if an app crashes (or freezes your phone)
- Let’s call this app Freeze, it
  - allows to select an app
  and “computes” if
Let’s start with a simpler case:
Detect if an app crashes (or freezes your phone)
Let’s call this app Freeze, it
- allows to select an app
and “computes” if
- an app is safe to use
Let’s start with a simpler case:
Detect if an app crashes (or freezes your phone)
Let’s call this app Freeze, it
- allows to select an app
and “computes” if
- an app is safe to use
- an app freezes your phone
Let’s start with a simpler case: Detect if an app crashes (or freezes your phone)

Let’s call this app Freeze, it
- allows to select an app
- and “computes” if
  - an app is safe to use
  - an app freezes your phone

Can you build Freeze?
Question: Can We Realise My Dream?
https://thorehusfeldt.net/2012/06/25/the-freeze-app-does-not-exist/

Let’s start with a simpler case:
Detect if an app crashes (or freezes your phone)

Let’s call this app Freeze, it
■ allows to select an app
and “computes” if
■ an app is safe to use
■ an app freezes your phone

Can you build Freeze?

I cannot, but I can build “Paradox”, which
1 runs Freeze and ask it to inspect Paradox.
2 if Freeze returns “OK” then freeze the phone (e.g., by computing 5/0)
3 if Freeze returns “Not OK” then print “Freeze detected that Paradox freezes” and terminate gracefully

---

```appscript
set Result to
    tell application "Freeze" to
        check application "Paradox"
    end tell
if Result is "OK" then
display 5/0
else display "Freeze detected that Paradox freezes"
end if
```
Is the “Paradox” app malicious? Let’s test it with Freeze:

```
set Result to
tell application "Freeze" to
    check application "Paradox"
end tell
if Result is "OK" then
    display 5/0
else display "Freeze detected that Paradox freezes"
end if
```
Is the “Paradox” app malicious? Let’s test it with Freeze:

- Let’s assume Paradox freezes the phone
  - Freeze will detect this
  - Paradox’s computations continues
    - printing the result
    - and terminates
Is the “Paradox” app malicious? Let’s test it with Freeze:

- Let’s assume Paradox freezes the phone
  - Freeze will detect this
  - Paradox’s computations continues
    - printing the result
    - and terminates

- Let’s assume Paradox does not freeze the phone
  - Freeze will detect this
  - Paradox’s computations continues
    - and freezes the phone (computing 5/0)
Is the “Paradox” app malicious? Let’s test it with Freeze:

- Let’s assume Paradox freezes the phone
  - Freeze will detect this
  - Paradox’s computations continues
    - printing the result
    - and terminates

- Let’s assume Paradox does not freeze the phone
  - Freeze will detect this
  - Paradox’s computations continues
    - and freezes the phone
      (computing 5/0)

There is no app for that.
This is a negative result, right?
This is a negative result, right?
No, it is not. It
  - motivates and
  - provides opportunities for research
This is a negative result, right?
No, it is not. It motivates and provides opportunities for research.

Possible solutions:
- Analyses that approximate, i.e., missing a few problems
  - i.e., reporting a few spurious issues
- Interactive analysis methods
- Make it easier to build secure systems
Thank you for your attention!

Any questions or remarks?

Contact:
Dr. Achim D. Brucker
Department of Computer Science
University of Sheffield
Regent Court
211 Portobello St.
Sheffield S1 4DP
UK

Phone: +44 114 22 21806
https://de.linkedin.com/in/adbrucker
https://www.brucker.uk
https://www.logicalhacking.com
a.brucker@sheffield.ac.uk
Fixing NSPK

Problem in step 2:

Fix (proposed by Lowe):
Agent B should also give his name: \( \{B, N_A, N_B\} \):

1. \( A \rightarrow B: \) \( \{A, N_A\} \)
2. \( B \rightarrow A: \) \( \{B, N_A, N_B\} \)
3. \( A \rightarrow B: \) \( \{N_B\} \)

\[ B \rightarrow A : \{N_A, N_B\} \]