OWASP
AppSec Europe
London 2nd-6th July 2018

Usable Security for Developers: A Nightmare

Achim D. Brucker | @adbrucker

{* LogicalHacking *}.com

The University Of Sheffield.

About Me

- Security Expert/Architect at SAP SE
  - Member of the central security team, SAP SE (Germany)
    - Security Testing Strategist
  - Work areas at SAP included:
    - Defining the risk-based Security Testing Strategy
    - Evaluation of security testing tools (e.g., SAST, DAST)
    - Roll-out of security testing tools
    - Secure Software Development Life Cycle integration
    - . . .
- Since December 2015:
  - Associate Professor, The University of Sheffield, UK
  - Head of the Software Assurance & Security Research Team
  - Available as consultancy & (research) collaborations

https://www.brucker.ch/

Outline

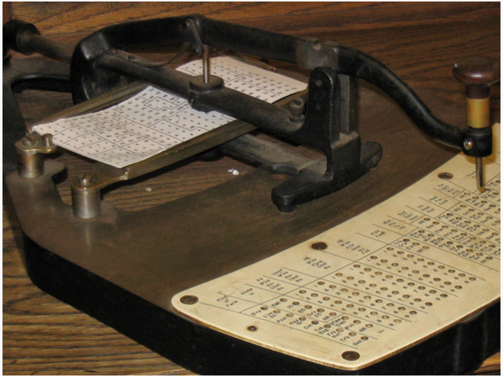
1 Security experts and developers

2 Secure programming cant' be that difficult . . .

3 The most common "fixes"

4 What we should do

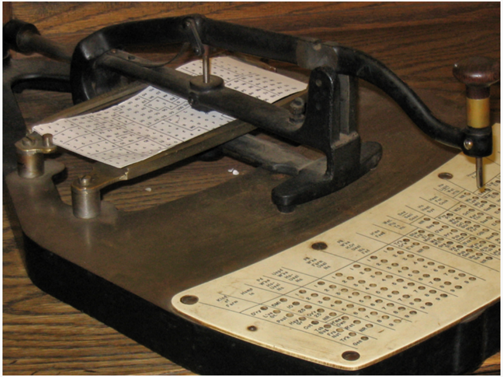
70 years of software development



Since the late 1940ies, we

computer systems.

70 years of software development

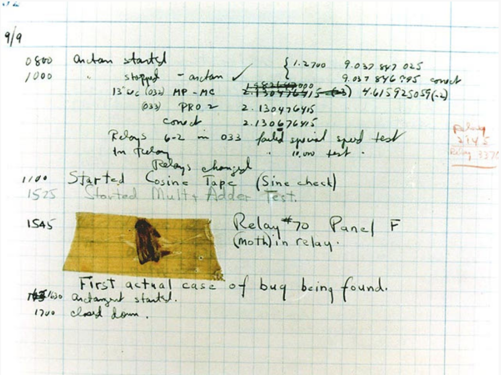


Since the late 1940ies, we

- program,

computer systems.

70 years of software development



9/9

0800 Antran started
1000 " stopped - antran ✓ {1.2700 9.037 847 025
9.037 846 995 conect
13" uc (032) MP - MC ~~1.982147000~~ ~~2.130476415 (-3)~~ 4.615925059(-2)
(033) PRO 2 2.130476415
conect 2.130676415
Relays 6-2 in 033 failed special speed test
In relay " 11.000 test .
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multy Adder Test.
1545 Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1630 Antangent started.
1700 closed down.

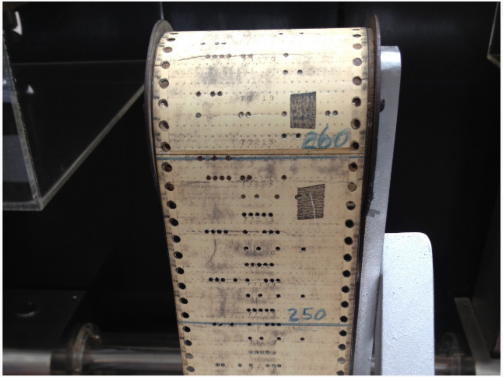
Relay 2145
Relay 3370

Since the late 1940ies, we

- program,
- debug, and

computer systems.

70 years of software development

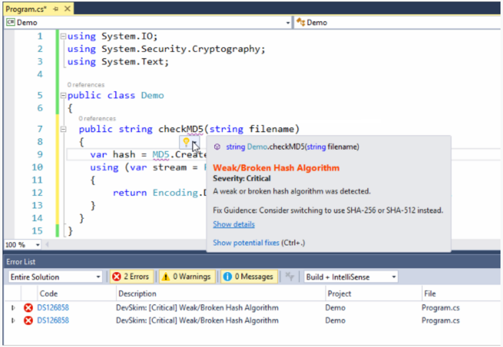


Since the late 1940ies, we

- program,
- debug, and
- patch

computer systems.

70 years of software development



```
using System.IO;
using System.Security.Cryptography;
using System.Text;

0 references
public class Demo
{
  0 references
  public string checkMD5(string filename)
  {
    var hash = MD5.Create
    using (var stream = F
    {
        return Encoding.[
    }
  }
}
```

string Demo.checkMD5(string filename)

Weak/Broken Hash Algorithm
Severity: Critical
A weak or broken hash algorithm was detected.

Fix Guidance: Consider switching to use SHA-256 or SHA-512 instead.

Show details

Show potential fixes (Ctrl+.)

Code	Description	Project	File
DS126858	DevSkim: [Critical] Weak/Broken Hash Algorithm	Demo	Program.cs
DS126858	DevSkim: [Critical] Weak/Broken Hash Algorithm	Demo	Program.cs

Since the late 1940ies, we

- program,
- debug, and
- patch

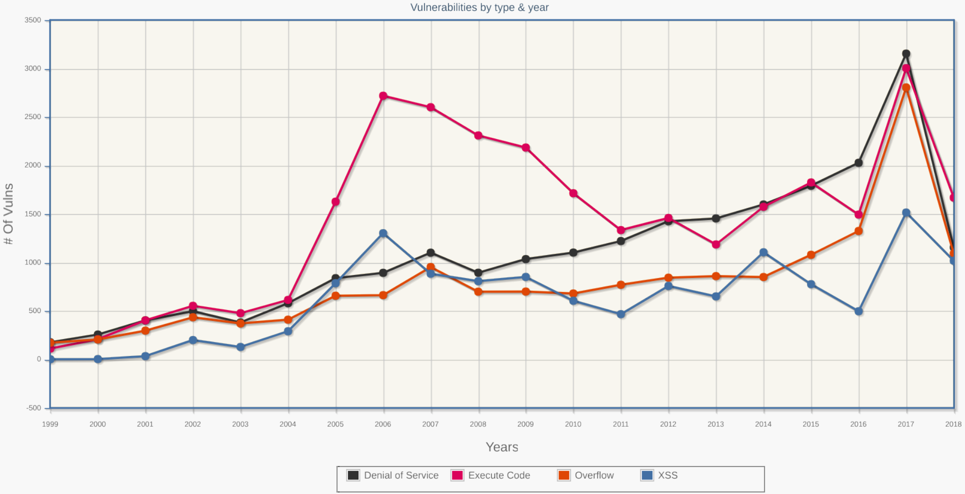
computer systems.

- we do not use punch cards anymore . . .

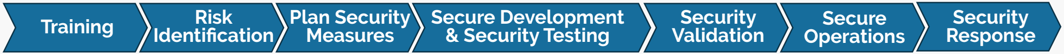
We build software since 70 years

**We build software since 70 years
and still make the same old (security) mistakes**

Usable Security for Developers: A Nightmare



The common “silver bullet”: The SDLC



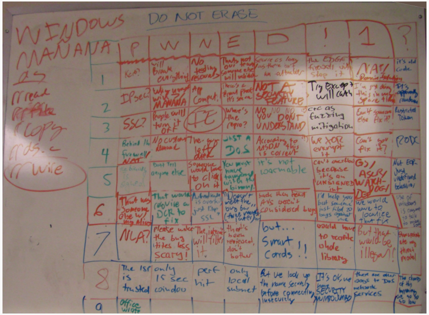
- Central **security experts** (SDLC owner)
  - Organizes security trainings
  - Defines product standard “Security”
  - Defines security testing strategy
  - Validates products
  - . . .
- Development teams
  - Select technologies
  - Select development model
  - Design and execute security testing plan
  - . . .

Works nicely

Works nicely
in theory – let’s move to reality

Security Expert
Developer

Introducing the SDLC: View of the security experts



WINDOWS MANANA

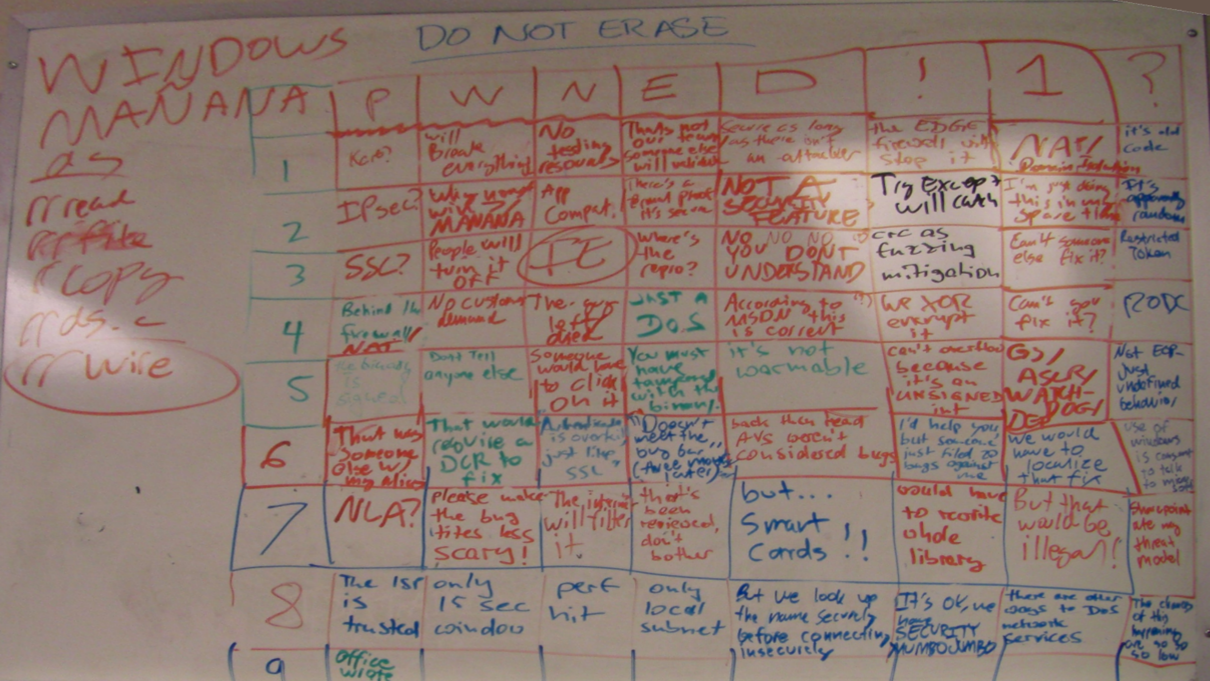
DO NOT ERASE

as

rread
~~rfile~~
rcopy
rds.c
rwire

	P	W	N	E	D	!	1	?
1	Kerb?	will Break everything	No testing resources	Thats not our team someone else will validate	Secure as long as there isn't an attacker	the EDGE firewall will stop it	NAT/ Domain Isolation	it's old code
2	IPsec?	[illegible] MANANA	App Compat.	There's a [illegible] proof it's secure	NOT A SECURITY FEATURE	Try Except will catch	I'm just doing this in my spare time	It's apparently random
3	SSL?	People will turn it OFF	FE	Where's the repro?	NO NO NO YOU DONT UNDERSTAND	CRC as fuzzing mitigation	Can't someone else fix it?	Restricted Token
4	Behind the firewall/ NAT	No customer demand	The guy left [illegible]	JUST A DoS	According to MSDN this is correct	We XOR encrypt it	Can't you fix it?	RODC
5	the binary is signed	Don't Tell anyone else	Someone would have to click on it	You must have tampered with the binary.	it's not wormable	can't overflow because it's an UNSIGNED int	GS/ ASLR/ WATCH-DOG/ DEP	Not EoP- Just undefined behavior
6	That was someone else w/ [illegible]	That would require a DCR to fix	Authenticode is overkill just like SSL	"Doesn't meet the bug bar" (three months later)	back then [illegible] AVs weren't considered bugs	I'd help you, but someone just filed 20 bugs against me	we would have to localize that fix	Use of windows is consent to talk to microsoft
7	NLA?	please make the bug titles less scary!	The internet will filter it.	that's been reviewed, don't bother	but... Smart Cards!!	would have to rewrite whole library	But that would be illegal!	Sharepoint are my threat model
8	The ISP is trusted	only 15 sec window	perf hit	only local subnet	But we look up the name securely before connecting insecurely	It's OK, we have SECURITY MUMBOJUMBO	there are other ways to DoS network services	The chances of this happening are so so so low
9	Office wrote							

- The whiteboard is from the Microsoft's security team
- I confess, I am guilty too:
  We also had a board with "embarrassing developers quotes"



DO NOT ERASE

WINDOWS MANANA

as

//read

//file

//copy

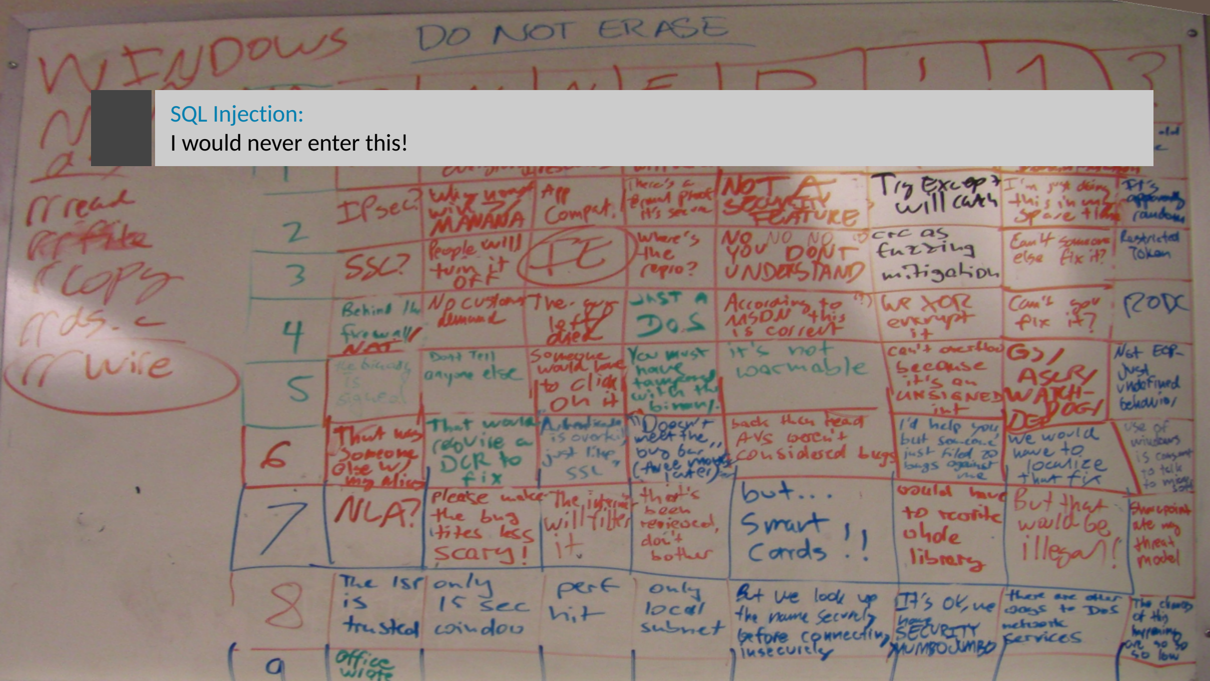
//ds.c

//wire

	P	W	N	E	D	!	1	?
1	Kerb?	will Break everything	No testing resources	Thats not our team someone else will validate	Secure as long as there isn't an attacker	the EDGE firewall will stop it	NAT/ Domain Isolation	it's old code
2	IPsec?	Why wont with MANANA	App Compat.	There's a brand phot it's secure	NOT A SECURITY FEATURE	Try Except will catch	I'm just doing this in my spare time	It's apparently random
3	SSL?	People will turn it OFF	IE	Where's the repro?	NO NO NO YOU DONT UNDERSTAND	crc as fuzzing mitigation	Can't someone else fix it?	Restricted Token
4	Behind the firewall/ NAT	No customer demand	The guy left/ died	JUST A DOS	According to MSDN this is correct (?)	We XOR encrypt it	Can't you fix it?	RODC
5	the binary is signed	Don't Tell anyone else	Someone would have to click on it	You must have tampered with the binary.	it's not wormable	can't overflow because it's an UNSIGNED int	GS/ ASLR/ WATCH-DOG/ DEP	Not EOP- Just undefined behavior
6	That was someone else w/ my alias	That would require a DCR to fix	Authenticode is overkill, just like SSL	"Doesn't meet the bug bar" (three months later)	back then read AVs weren't considered bugs	I'd help you but someone just filed 20 bugs against me	We would have to localize that fix	use of windows is consent to talk to microsoft
7	NLA?	please make the bug titles less scary!	The internet will filter it.	that's been reviewed, don't bother	but... Smart Cards !!	would have to rewrite whole library	But that would be illegal!	Sharepoint ate my threat model
8	The ISP is trusted	only 15 sec window	perf hit	only local subnet	But we look up the name securely before connecting insecurely	It's OK, we have SECURITY MUMBOJUMBO	there are other ways to DoS network services	The chances of this happening are so so so low
9	Office wrote							

SQL Injection:

I would never enter this!



WINDOWS

DO NOT ERASE

rread

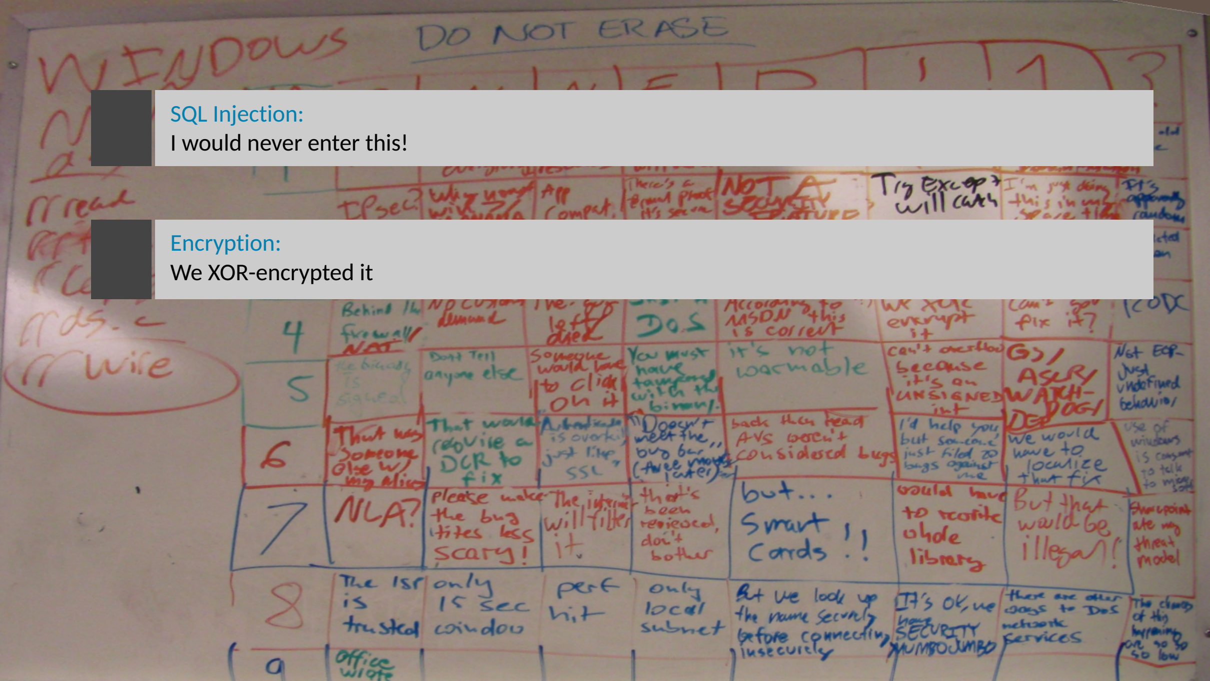
rfile

rcopy

rds.c

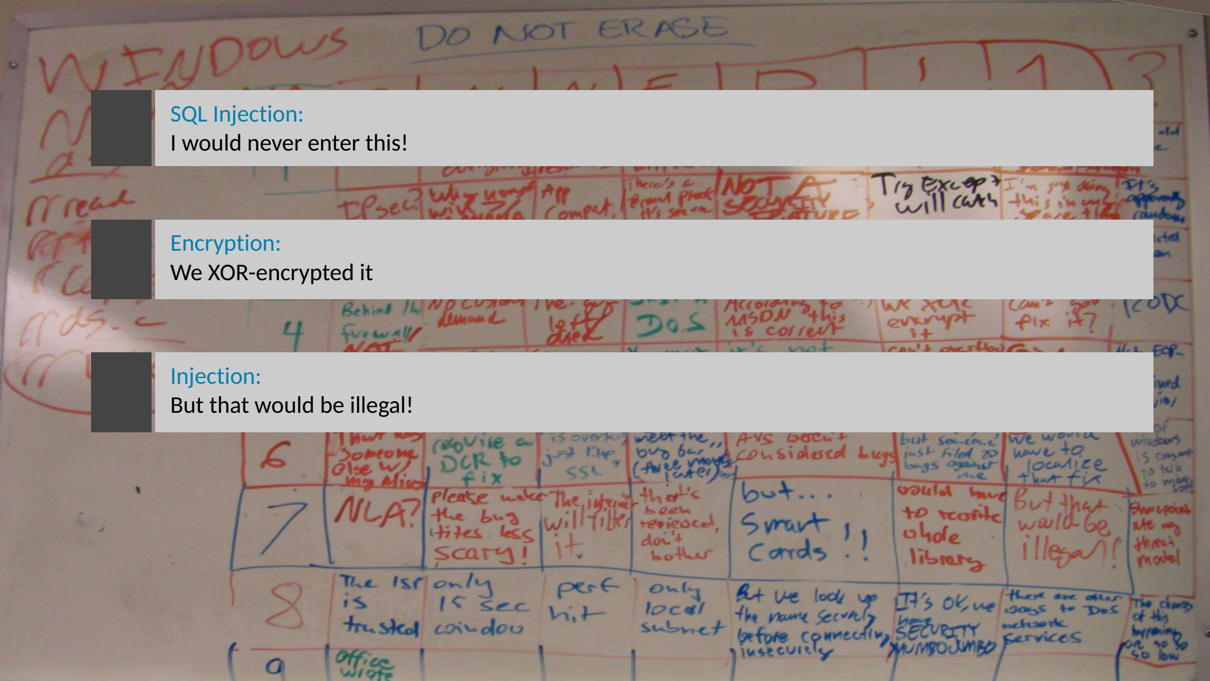
rwire

1								old c
2	IPsec?	Why would we ... MANANA	App Compat.	There's a formal proof it's secure	NOT A SECURITY FEATURE	Try Except will catch	I'm just doing this in my spare time	It's apparently random
3	SSL?	People will turn it OFF	IE	Where's the repro?	NO NO NO YOU DONT UNDERSTAND	CRC as fuzzing mitigation	Can't someone else fix it?	Restricted Token
4	Behind the firewall NOT	No customer demand	The guy left died	JUST A DoS	According to MSDN this is correct	We XOR encrypt it	Can't you fix it?	RODC
5	the binary is signed	Don't Tell anyone else	Someone would have to click on it	You must have tampered with the binary.	it's not wormable	can't overflow because it's an UNSIGNED int	GS/ ASLR/ WATCH-DOG/ DEP	Not EOP- Just undefined behavior
6	That was someone else w/ my alias	That would require a DCR to fix	Authenticode is overkill, just like SSL	"Doesn't meet the bug bar" (three months later)	back then read AVs weren't considered bugs	I'd help you, but someone just filed 20 bugs against me	We would have to localize that fix	Use of windows is consent to talk to microsoft
7	NLA?	Please make the bug titles less scary!	The internet will filter it.	that's been reviewed, don't bother	but... Smart Cards!!	would have to rewrite whole library	But that would be illegal!	Sharepoint ate my threat model
8	The ISR is trusted	only 15 sec window	perf hit	only local subnet	But we look up the name securely before connecting insecurely	It's OK, we have SECURITY MUMBOJUMBO	there are other ways to DoS network services	The chances of this happening are so so so low
9	Office wrote							


WINDOWS
DO NOT ERASE
read
rds.c
wire
4
5
6
7
8
9
IPsec
App Compat.
NOT A SECURITY FEATURE
Try Except will catch
I'm just doing this in user space tho
It's apparently random
Behind the firewall NAT
No customer demand
The guy left died
DoS
According to MSDN this is correct
We XOR encrypt it
Can't you fix it?
The binary is signed
Don't tell anyone else
Someone would have to click on it
You must have tampered with the binary.
it's not wormable
can't overflow because it's an UNSIGNED int
GS / ASLR WATCHDOG DEP
Not EOP. Just undefined behavior
That was someone else w/ my alias
That would require a DCR to fix
Authenticode is overkill just like SSL
"Doesn't meet the bug bar" (three months later)
back then read AVs weren't considered bugs
I'd help you but someone just filed 20 bugs against me
We would have to localize that fix
NLA?
Please make the bug titles less scary!
The internet will filter it.
that's been reviewed, don't bother
but... Smart Cards!!
would have to rewrite whole library
But that would be illegal!
Sharepoint ate my threat model
The ISP is trusted
only 15 sec window
perf hit
only local subnet
But we look up the name securely before connecting insecurely
IT's OK, we have SECURITY MUMBOJUMBO
there are other ways to DoS network services
The chances of this happening are so so low
Office wrote

SQL Injection:
I would never enter this!

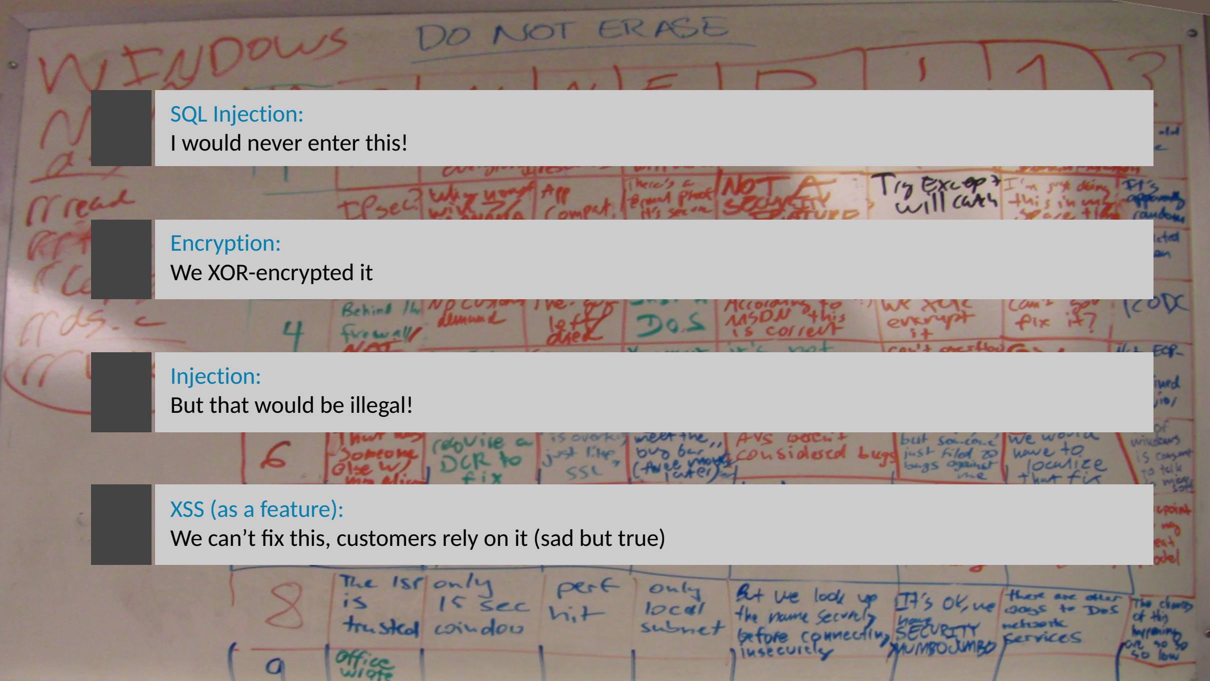
Encryption:
We XOR-encrypted it


WINDOWS
DO NOT ERASE
IPsec
App Compat.
NOT A SECURITY FEATURE
Try Except will catch
Behind the firewall
No customer demand
DoS
According to MSDN this is correct
We XOR encrypt it
Can't you fix it?
6
require a DCR to fix
just like SSL
meet the bug bar
considered bugs
but someone just filed 20 bugs against me
we would have to localize that fix
7
NLA?
Please make the bug titles less scary!
The internet will filter it.
that's been reviewed, don't bother
but... Smart Cards!!
would have to rewrite whole library
But that would be illegal!
Sharepoint ate my threat model
8
The ISP is trusted
only 15 sec window
perf hit
only local subnet
But we look up the name securely before connecting insecurely
It's OK, we have SECURITY MUMBOJUMBO
there are other ways to DoS network services
9
Office wrote

SQL Injection:
I would never enter this!

Encryption:
We XOR-encrypted it

Injection:
But that would be illegal!


WINDOWS
DO NOT ERASE
IPsec
App Compat.
NOT A SECURITY FEATURE
Try Except will catch
Behind the firewall
DoS
According to MSDN this is correct
We XOR encrypt it
Can't you fix it?
6
Someone else
Provide a DCR to fix
just like SSL
AVs doesn't considered bugs
but someone just filed 20 bugs against me
we would have to localize that fix
8
The ISP is trusted
only 15 sec window
perf hit
only local subnet
But we look up the name securely before connecting insecurely
It's OK, we have SECURITY MUMBOJUMBO
there are other ways to DoS network services
9
Office

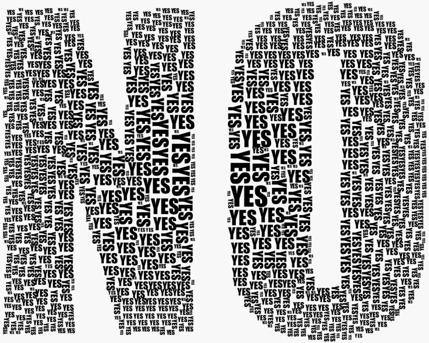
SQL Injection:
I would never enter this!

Encryption:
We XOR-encrypted it

Injection:
But that would be illegal!

XSS (as a feature):
We can’t fix this, customers rely on it (sad but true)

Introducing the SDLC: View of the developers



- Experience security as “The Department of No”
- Confronted with a strange & complex language (there are over 1024 CWEs – and counting)

Example of unfriendly APIs: Buffer overflow

```
> man gets
GETS(3S)                                            GETS(3S)

NAME
     gets, fgets - get a string from a stream

SYNOPSIS
     #include <stdio.h>

     char *gets(s)
     char *s;

DESCRIPTION
     Gets reads a string into s from the standard input
     stream stdin. The string is terminated by a newline
     character, which is replaced in s by a null character.
     Gets returns its argument.
```

Let’s travel back in time

- Unix V7 (1979)
- Reading strings
- Gets returns a string of arbitrary length

Is there a secure use of gets?

Example of unfriendly APIs: Buffer overflow

- Wait, let’s check the man page on a modern Unix/Linux:

```
NAME
   gets - get a string from standard input (DEPRECATED)



BUGS
   Never use gets(). Because it is impossible to tell without knowing the
   data in advance how many characters gets() will read, and because
   gets() will continue to store characters past the end of the buffer,
   it is extremely dangerous to use.  It has been used to break computer
   security.  Use fgets() instead.
```

Example of unfriendly APIs: Buffer overflow

- OK, that’s sounds easy:

```
void f() {
    char buf[20];
    gets(buf);
}
```

Example of unfriendly APIs: Buffer overflow

- OK, that’s sounds easy: Use `fgets(buf, n, stdin)` instead of `gets(buf)`:

```
void f() {
    char buf[20];
    fgets(buf,20,stdin) // NOT: gets(buf);
}
```

Example of unfriendly APIs: Buffer overflow

- OK, that’s sounds easy: Use `fgets(buf, n, stdin)` instead of `gets(buf)`:

```
void f() {
    char buf[20];
    fgets(buf,20,stdin) // NOT: gets(buf);
}
```

- Is this now secure?

Example of unfriendly APIs: Buffer overflow

- OK, that’s sounds easy: Use `fgets(buf, n, stdin)` instead of `gets(buf)`:

```
void f() {
    char buf[20];
    fgets(buf,20,stdin) // NOT: gets(buf);
}
```

- Is this now secure? No, fgets does **not** always null-terminate

Example of unfriendly APIs: Buffer overflow

- OK, that’s sounds easy: Use `fgets(buf, n, stdin)` instead of `gets(buf)`:

```
void f() {
    char buf[20];
    fgets(buf,20,stdin) // NOT: gets(buf);
}
```

- Is this now secure? No, fgets does **not** always null-terminate
- we need to manually null terminate the buffer (and reserve space for the null character)

```
void f() {
    char buf[21];
    fgets(buf,20,stdin);
    buf[20]='\0';
  }
```

- C-Programming has a lot in comming with (insurance) contracts: allways read the small print

Example of unfriendly APIs: Error handling

> “Most OpenSSL functions will return an integer to indicate success or failure. Typically a function will **return 1 on success or 0 on error**. All return codes should be checked and handled as appropriate. Note that not all of the libcrypto functions return 0 for error and 1 for success. There are exceptions which can trip up the unwary. For example **if you want to check a signature with some functions you get 1 if the signature is correct, 0 if it is not correct and -1 if something bad happened** like a memory allocation failure.”
>
> (OpenSSL)

- Recall the common C convention:
  - 0 indicates success
  - any non-zero value indicates failure

Example of unfriendly APIs: Error handling

- Which one is correct:
  1. Consider

     ```
     if (some_verify_function())
       /* signature successful *?
     ```

  2. Consider

     ```
     if ( 1 != some_verify_function())
       /* signature successful *?
     ```

Example of unfriendly APIs: Error handling

- Which one is correct:
  1. Consider

```
if (some_verify_function())
  /* signature successful *?
```

  2. Consider

```
if ( 1 != some_verify_function())
  /* signature successful *?
```

- The last one is correct

Example of unfriendly APIs: The Java 8 Crypto API

Just a nightmare:

- Many configurations to choose from
  - algorithm
  - mode of operation
  - padding scheme
  - right keys and sizes
  - . . .
- Most ciphers are oudated/broken. Only two can still be recommended
  - AES (symmetric)
  - RSA (asymmetric)
- Many providers use insecure defaults (e.g., ECB mode)

Using the Java crypto API, is already hard for somebody who understands crypto . . .

Example of unfriendly APIs: XSS (Java)

- Most Web Frameworks for Java do not provide input/output encoding as default
- Developers need to include third party encoding libraries
  (e.g., OWASP Java Encoder: https://github.com/OWASP/owasp-java-encoder)
- **and** add calls to the encoder manually:

```
PrintWriter out = ....;
out.println("<textarea>"+Encode.forHtml(userData)+"</textarea>");
```

- You need to insert the **right** (there are many) encoder **each time**.

Common mitigations

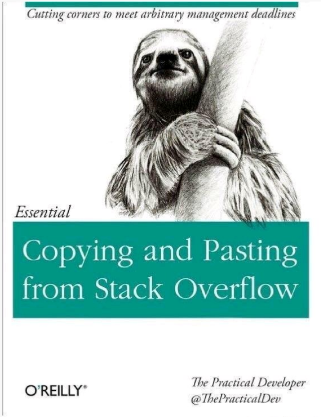
- Provide training
  - Do we really expect that our developers understand all these details?
- Write (coding) guidelines
  - Guidelines without tool support are (mostly) worthless
- Use generic application security testing tools
  - without configuration, these tools are prone to both high false-positive rates and high false-negative rates
  - many tools are developed for security experts (and not for developers)
  - penetration tests

In their generality, these actions are often not very effective!

Security experts and developers need to work together to achieve thecommon goal: secure software!
(Disclaimer: security experts might need to learn how to code)!

Think positive: security enables developers to produce high-quality and secure software!

- Start early in the development:
  - Select frameworks and/or programming languages that are secure by design
  - Develop custom APIs-Wrappers that are easy to use and require only little security knowledge
  - To consider
    - Configure your DAST/IAST/SAST tool to support your custom APIs
    - In the fix recommendations of your DAST/IAST/SAST tool, point developers to the recommended frameworks
    - If you develop APIs, make your examples secure by default


Cutting corners to meet arbitrary management deadlines
Essential
Copying and Pasting from Stack Overflow
O'REILLY®
The Practical Developer
@ThePracticalDev

If you do not support your developers, they will seek for help elsewhere!

Let’s close with a good example: Modern Rails

- Modern versions of Rails are pretty secure by default
- Input/output encoding is enabled by default and, in exceptional cases, needs to be disabled explicitly:

```
   <%= account.balance.html_safe % >
```

  (one can argue, if `html_safe` is a good name denoting un-sanitized (trusted) channels)
- Suddenly, a simple `grep` becomes a powerful static analysis tool

Call for action

Let’s build framework and APIs are easy to use securely!

Call for action

Let’s build framework and APIs are easy to use securely!

Call for action

Let’s build framework and APIs are easy to use securely!

Call for action

Let’s build framework and APIs are easy to use securely!

Call for action

Let’s build framework and APIs are easy to use securely!

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

✉ a.brucker@sheffield.ac.uk
@adbrucker
https://de.linkedin.com/in/adbrucker/
https://www.brucker.ch/
https://logicalhacking.com/blog/

Document Classification and License Information