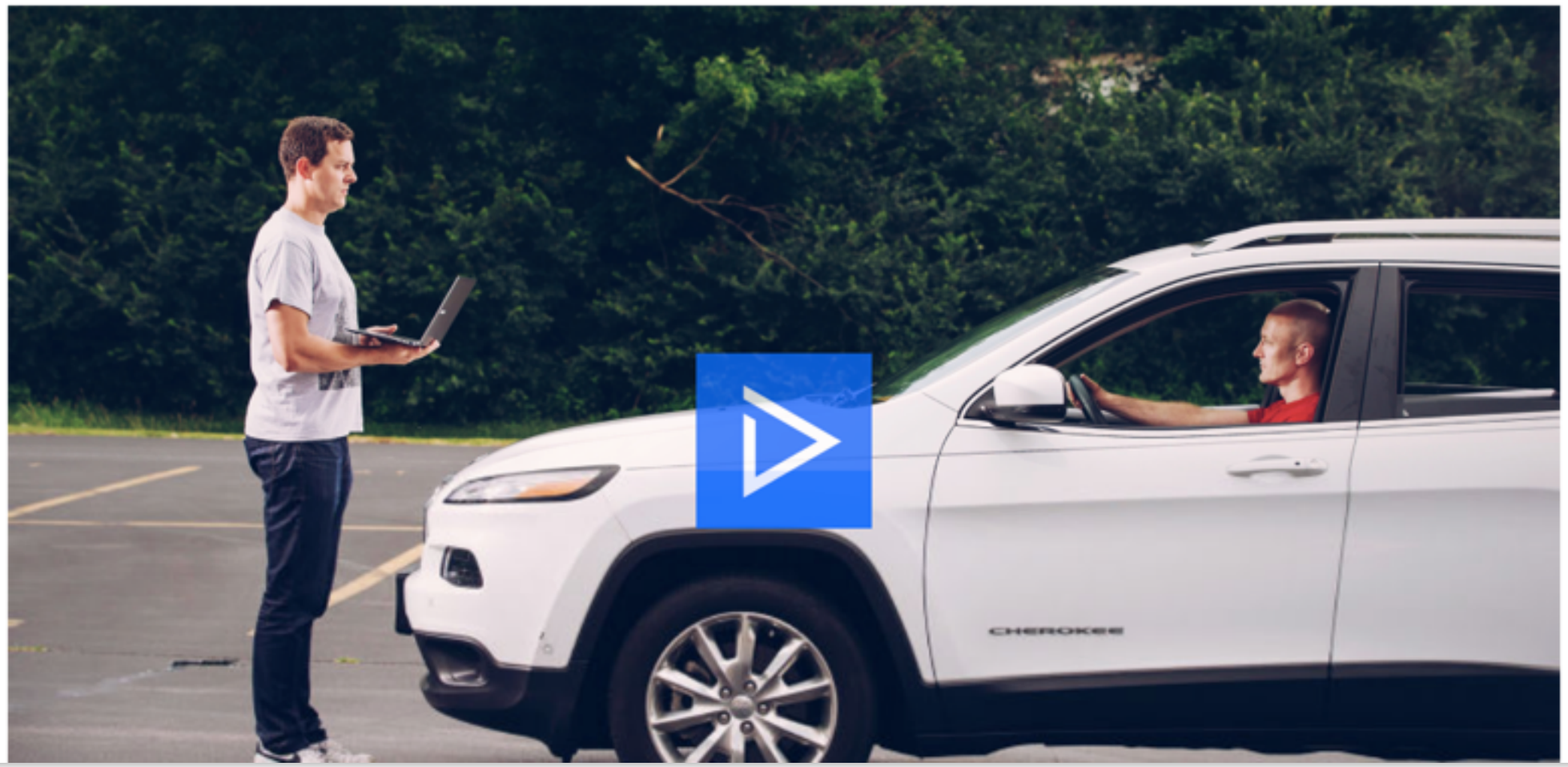
Security Testing

Course + Lab, Spring 2017

Andreas Zeller, Saarland University

ANDY GREENBERG SECURITY 07.21.15 6:00 AM

HACKERS REMOTELY KILL A JEEP ON THE HIGHWAY—WITH ME IN IT



Thermostats can now get infected with ransomware, because 2016

by MATTHEW HUGHES — 29 days ago in GADGETS

Credit: Ken Munro

Recommended

5 reasons why wearables are still ruling our wrists (and everywhere else)

Marie-Anne Leuty · 15 hours ago

Most popular

1. 

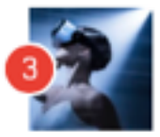
Google Maps now has a 'Catching Pokémon' feature in Timeline

Mix · 1 day ago

2. 

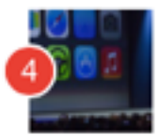
Facebook is testing a new Twitter-like feature to boost conversations

Mix · 22 hours ago

3. 

The world's first VR ballet experience is absolutely stunning

Juan Buis · 1 day ago

4. 

The best Apple Keynotes to watch before Wednesday's iPhone 7 Keynote

Boris Veldhuijzen van Zanten · 1 day ago

5. 

Warner Bros. shoots itself in the foot as it flags its own website for piracy

Mix · 1 day ago

The Internet of ransomware things...

HUNGRY? PAY UP AND I'LL UNLOCK MY DOOR!

ON STRIKE UNTIL YOU SEND MONEY TO MY HACKERS.

20 BUCKS IN MY PAYPAL ACCOUNT OR I'LL ONLY BREW *DECAF!*

I'LL BE BURNING THE TOAST IF YOU DON'T GET ME SOME *DOUGH!*

YOUR DIRTY DISHES CAN WAIT, I'M BUSY MINING BITCOINS.

EXCUSE US WHILE WE PARTICIPATE IN A *DDOS* ATTACK.

30 BUCKS IN BITCOIN, OR NEXT TIME I SMELL SMOKE, I MIGHT JUST LET YOU SLEEP.

THE NEXT TIME YOU LEAVE, IT'LL COST YOU 100 BUCKS TO GET BACK INTO THE HOUSE, UNLESS YOU GIVE ME $75 *NOW!*

MY ALARM SYSTEM IS GOING TO GO OFF RANDOMLY THROUGHOUT THE NIGHT, UNLESS YOU "DONATE".

I'M TURNING OFF THE HEAT UNTIL YOU WARM UP MY BANK ACCOUNT!

WIRE MY HACKER $100 OR I'LL REVERSE MY MOTOR AND BLOW DIRT ALL OVER THIS PLACE!

IF YOU DON'T SEND US *CASH*, YOUR REPUTATION WILL BE IN THE *TRASH.*

I'LL START YOUR CAR, BUT ONLY TO TAKE YOU TO YOUR BANK TO MAKE A TRANSFER.

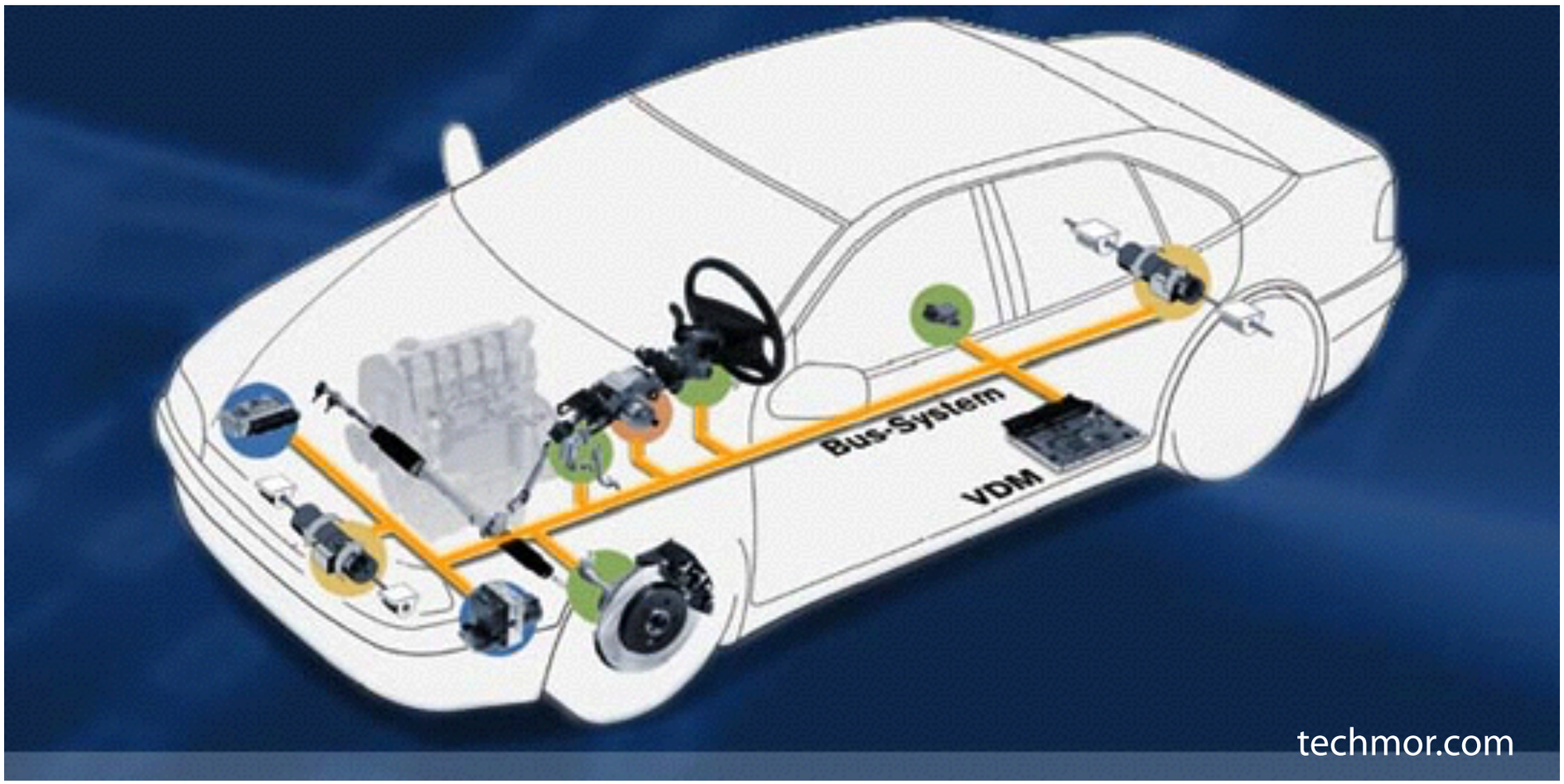
SEND ME $25 OR I'LL TELL EVERYONE ON YOUR SOCIAL NETWORK THAT YOU WERE STUPID ENOUGH TO BUY AN INTERNET-CONNECTED BROOM!

External Attacks

Program

- Some external event causes a *change in program behavior*

Highjacking a Car



Highjacking a Car

- All car components are connected via a bus system (CAN bus)
- Includes engine control, power steering, controls, entertainment system
- Hardware controls tight *access rules* – e.g. entertainment system can only read, not write

Highjacking a Car

1. Connect to *entertainment system* via *public WiFi access*
2. *Exploit vulnerability* to get control over system
3. *Flash* chip that controls CAN bus access to get full writing capabilities
4. Voilá! Full control over car.

A Simple Vulnerability

```
while ((cc = getch()) != c)
{
        name[j++] = cc;
        ...
}
```

- No checking for length of buffer `name`
- Can overwrite stack with *code* and new *return address* that jumps into code
- Any simple test would find that!

Security by Proof

Systems that are *provably secure* ensure that

- specific attacks are *impossible*
  e.g. no buffer overflows, or no SQL injection
- they will always *behave as designed*
  e.g. will always produce a correct result

Requires (expensive) mathematical proof

Security by Testing

Systems that are thoroughly *tested* ensure

- *Low probability* of attack success
  because several attacks already have been tested
- *High complexity* of remaining attacks
  because simple attacks already have been tested
- Cost-efficient if highly *automated*

But *no guarantee* of absence of bugs

Security Testing

- Introduces you to **automated techniques for security testing**
- Enables you to **implement** and **use** such techniques
- Aim: **Smart ways to break systems**

Course Contents

- Simple **fuzzing** techniques
  generating *random inputs* to programs
- Simple **reduction** techniques
  to determine *failure-inducing inputs*
- **Mutation** techniques
  changing existing (valid) inputs

Course Contents

- **Structured** fuzzing techniques
  using *grammars* and models
- **Adaptive** fuzzing techniques
  driven by *code coverage*
- Automatic **inference of input structure**
  so you can effectively fuzz arbitrary programs

Course Format

- **Lecture** in the morning (09:00–10:30)
- **Programming Lab** for the rest of the day
- Runs for **two weeks** (starting today)
- At end, two weeks for **individual project**

Assignments

- Over the course, you build *four projects* that implement course content
- We provide *sample code* from lecture as starting point
- Will be graded by their efficiency on a set of (buggy) *subjects*
- We provide *sample subjects* for training

Individual Project

- After two weeks, use the course content to **create your own security tester**
- Choose domain, techniques as you like
- Submission due after two more weeks
- Will be graded for creativity and efficiency

Programming Language

By Doc Searls – 2006oscon_203.JPG, CC BY-SA 2.0
https://commons.wikimedia.org/w/index.php?curid=4974869

- Compact, easy to **read**, easy to **learn**
  You can learn basic Python in 1–2 hours
- Great libraries for **string manipulation**
  Creating, parsing, manipulating is very easy
- Great features for **dynamic analysis**
  You can write a debugger in ~10 lines

fuzzer.py

```
import random

def fuzzer():
    # Strings up to 1024 characters long
    string_length = int(random.random() * 1024)

    # Fill it with ASCII 32..128 characters
    out = ""
    for i in range(0, string_length):
        out += chr(int(random.random() * 96 + 32))
    return out

if __name__ == "__main__":
    print(fuzzer())
```

Fuzzer Output

[;x1-GPZ+wcckc];,N9J+?#6^6\e?]9lu2_%'4GX"0VUB[E/r
~fApu6b8<{%siq8Zh.6{V,hr?;{Ti.r3PIxMMMv6{xS^+'Hq!
AxB"YXRS@!Kd6;wtAMefFWM(`lJ_<1~o}z3K(CCzRH JIlvHz>_*.
\>JrlU32~eGP?IR=bF3+;y$3lodQ<B89!5"W2fK*vE7v{')KC-
i,c{<[~m!]o;{.'}Gj\(X}EtYetrpbY@aGZ1{P!AZU7x#4(Rtn!
q4nCwqol^y6}0lKo=*JK~;zMKV=9Nai:wxu{J&UV#HaU)*BiC<),`
+t*gka<W=Z.%T5WGHZpI30D<Pq>&]BS6R&j?#tP7iaV}-}`\?
[_[Z^LBMPG-FKj'\xwuZ1=Q`^`5,NQ@[!CuRzJ2DlvBy!
^zkhdf3C5PAkR?V hnl3='i2Qx]D
$qs4O`1@fevnG'2\11Vf3piU37@55ap\zIyl"'f,
$ee,J4Gw:cgNKLie3nx9(`efSlg6#[K"@WjhZ}r[Scun&sBCS,T[/
vY'pduwgzDlVNy7'rnzxNwI)(ynBa>%lb`;`9fG]P_0hdG~$@6
3]KAeEnQ7lU)3Pn,0)G/6N-wyzj/MTd#A;r

Fuzzing

Random Testing at the System Level

Security Testing

- Introduces you to **automated techniques for security testing**
- Enables you to **implement** and **use** such techniques
- Aim: **Smart ways to break systems**

- **Lecture** in the morning (09:00–10:30)
- **Programming Lab** for the rest of the day
- Runs for **two weeks** (starting today)
- At end, two weeks for **individual project**

Course Contents

- Simple **fuzzing** techniques
  generating *random inputs* to programs
- Simple **reduction** techniques
  to determine *failure-inducing inputs*
- **Mutation** techniques
  changing existing (valid) inputs
- **Structured** fuzzing techniques
  using *grammars* and models
- **Adaptive** fuzzing techniques
  driven by *code coverage*
- Automatic **inference of input structure**
  so you can effectively fuzz arbitrary programs

Assignments

- Over the course, you build four test *generators* that implement course content
- We provide *sample code* from lecture as starting point
- Will be graded by their efficiency on a set of (buggy) *subjects*
- We provide *sample subjects* for training

Individual Project

- After two weeks, use the course content to **create your own security tester**
- Choose domain, techniques as you like
- Submission due after two more weeks
- Will be graded for creativity and efficiency