Introduction to Security

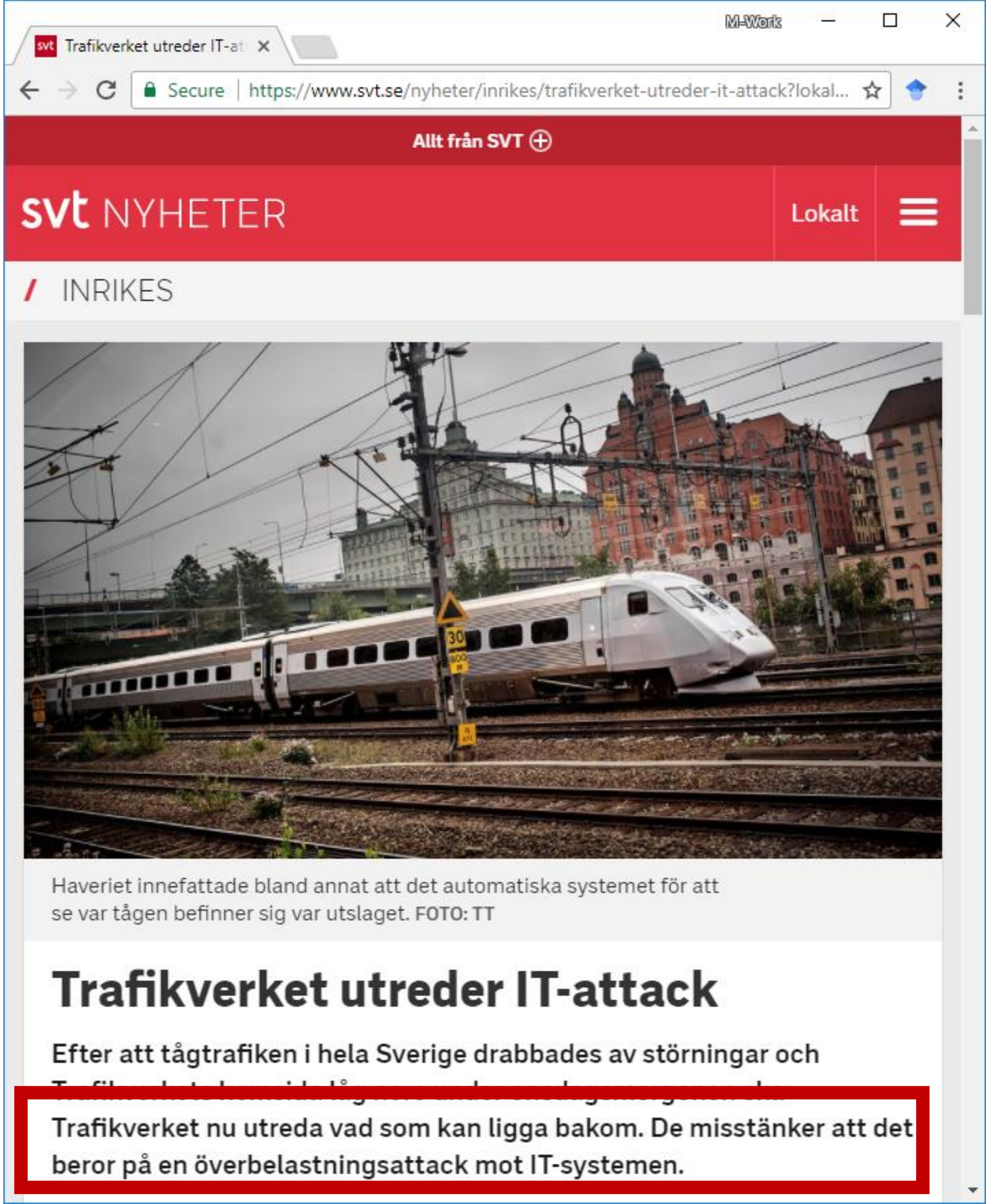
Magnus Almgren

Security Quiz

- Connect to kahoot.it
  - Enter Pin: xxxx (will come when I start the quiz)
- FAQ
  - Questions appear on full screen
  - You press the answer (based on color, symbol) on your device
  - The faster you press the correct answer, the more points
  - Sometimes, several answers may be correct
- Good luck!

Why security?

- De misstänker att det beror på en över-belastnings-attack mot IT-systemen.


Trafikverket utreder IT-at
M-Work
Secure
https://www.svt.se/nyheter/inrikes/trafikverket-utreder-it-attack?lokal...
Allt från SVT ⊕
svt NYHETER
Lokalt
/ INRIKES

Haveriet innefattade bland annat att det automatiska systemet för att se var tågen befinner sig var utslaget. FOTO: TT

Trafikverket utreder IT-attack

Efter att tågtrafiken i hela Sverige drabbades av störningar och Trafikverkets hemsida låg nere under onsdagskvällen ska Trafikverket nu utreda vad som kan ligga bakom. De misstänker att det beror på en överbelastningsattack mot IT-systemen.

https://www.youtube.com/watch?v=fpGR1J4prfs&t=38s
Democratic processes attacked …

Story: The Morris Worm

Robert T. Morris

- November 3, 1988: launch of worm
  → 6,000 computers shut down (in the U.S. only)
- Internet like a small town – 100,000 computers (?) where people knew and trusted each other.
- Many features not built with security in mind.
  - ”doors left unlocked”
  - Internet security – mostly theoretical problem
  - What was there to protect?
- The worm changed the landscape!
  - Wakeup call that security is important!
  - Creation of CERT:s, demand for security experts (academia, industry)
- Over 25 years later, some of the same strategies still work …

http://www.washingtonpost.com/blogs/the-switch/wp/2013/11/01/how-a-grad-student-trying-to-build-the-first-botnet-brought-the-internet-to-its-knees/

The Morris Worm – Steps

Principle for function

A. Intrusion

B. Transfer of main program

C. Settling down and establishing (cracking accounts, hiding, etc)

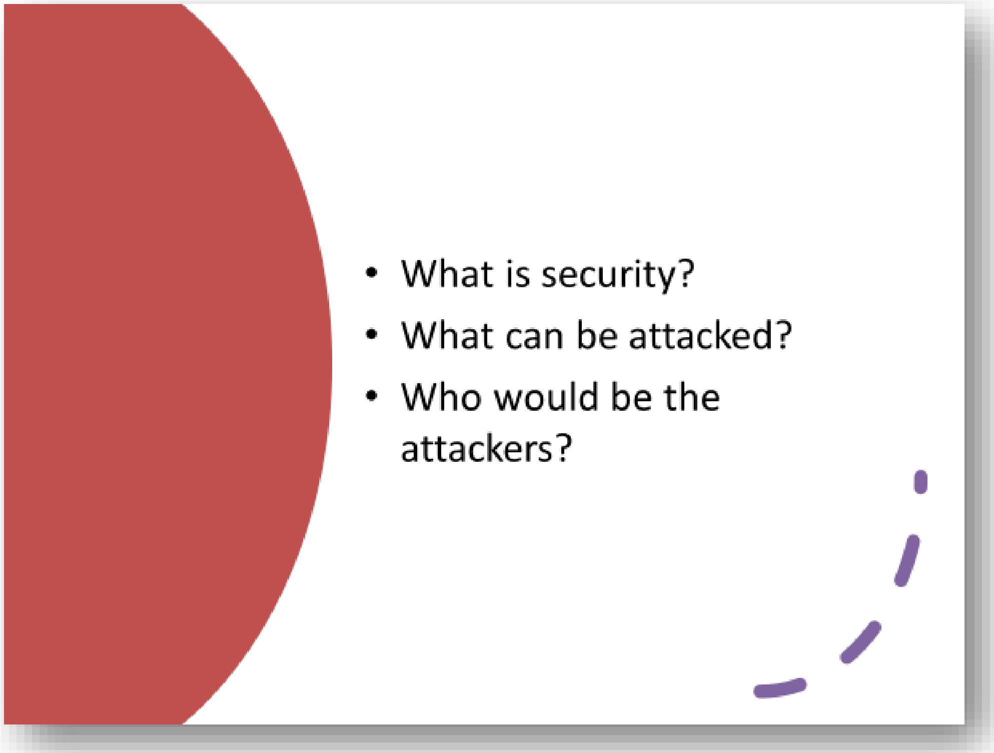
D. Continued intrusions

***Details** (4 well-known attacks)*

1. finding trust relations
2. guess/crack passwords
3. use debug facility in the sendmail mail handler
4. exploit bug in finger program
(buffer overflow)

Stop! What is Security?

- What is security?



- What is security?
- What can be attacked?
- Who would be the attackers?

Breakout rooms

Courses Chalmers/GU

1.

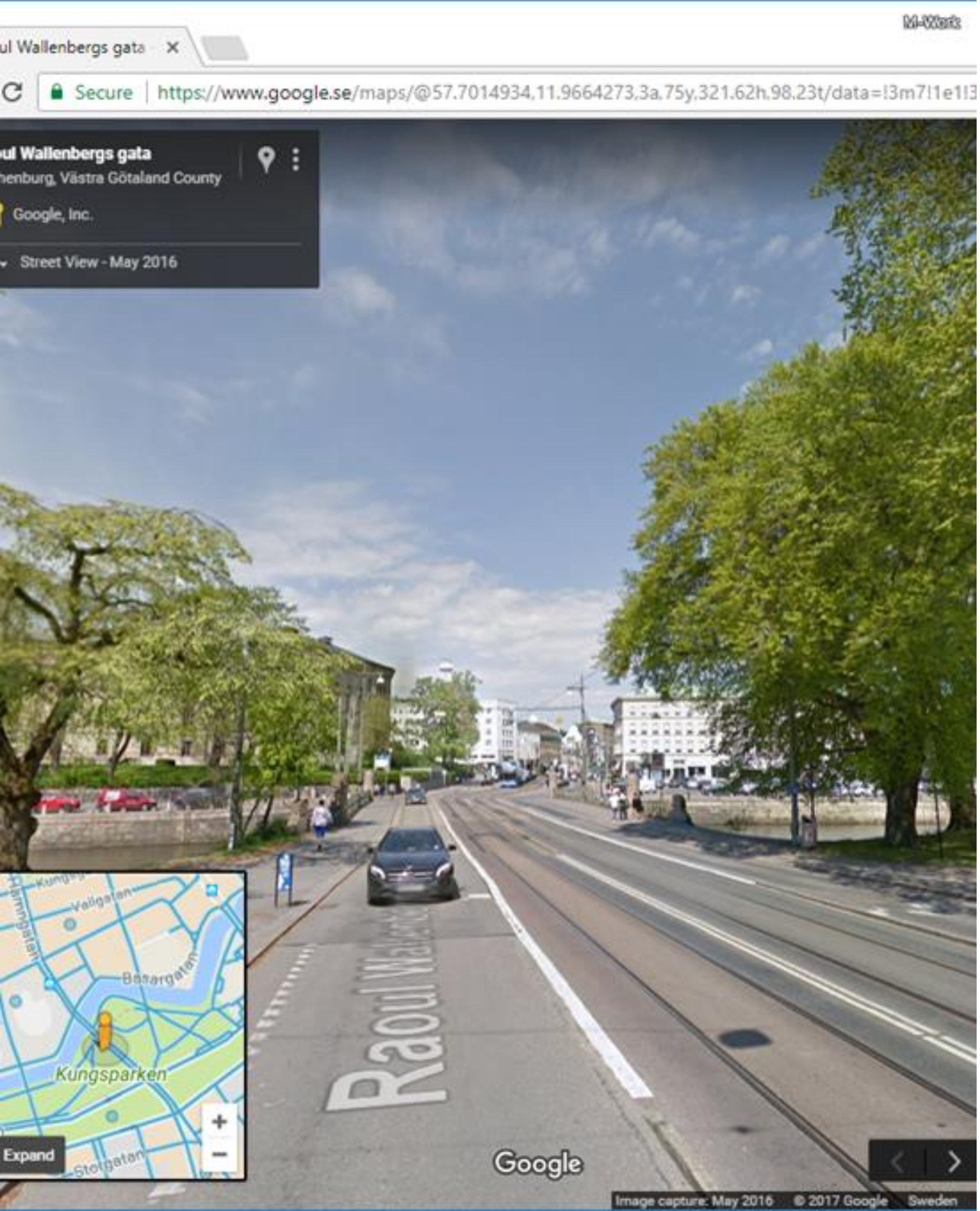
2. Cryptography

3.

4.

5.

Design
vs
Reality

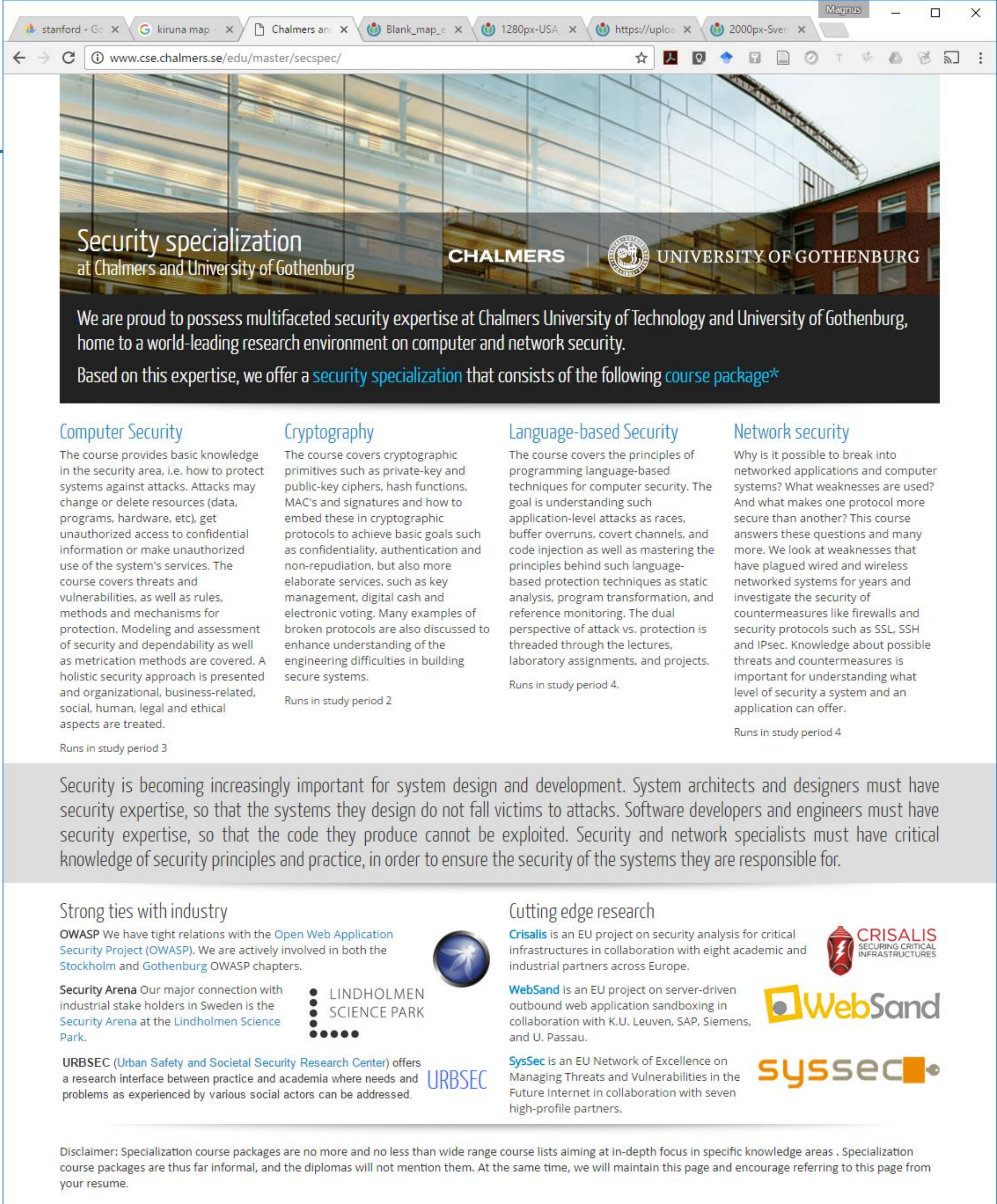

M-Work
ul Wallenbergs gata
Secure
https://www.google.se/maps/@57.7014934,11.9664273,3a,75y,321.62h,98.23t/data=!3m7!1e1!3
oul Wallenbergs gata
henburg, Västra Götaland County
Google, Inc.
Street View - May 2016
Raoul Wallenbergs
Vallgatan
Basargatan
Kungsparken
Storgatan
Expand
Google
Image capture: May 2016 © 2017 Google Sweden

30
Lambertsson

Hej!
Ramudden
Ramudden
Implenia
Implenia
SORG.

Courses Chalmers/GU

1. Computer Security
2. Cryptography
3. Language-based Security
4. Network security
5. ICT Support for Adaptiveness and (Cyber)security in the Smart Grid



Security specialization
at Chalmers and University of Gothenburg

We are proud to possess multifaceted security expertise at Chalmers University of Technology and University of Gothenburg, home to a world-leading research environment on computer and network security.

Based on this expertise, we offer a security specialization that consists of the following course package*

Computer Security

The course provides basic knowledge in the security area, i.e. how to protect systems against attacks. Attacks may change or delete resources (data, programs, hardware, etc), get unauthorized access to confidential information or make unauthorized use of the system's services. The course covers threats and vulnerabilities, as well as rules, methods and mechanisms for protection. Modeling and assessment of security and dependability as well as metrication methods are covered. A holistic security approach is presented and organizational, business-related, social, human, legal and ethical aspects are treated.

Runs in study period 3

Cryptography

The course covers cryptographic primitives such as private-key and public-key ciphers, hash functions, MAC's and signatures and how to embed these in cryptographic protocols to achieve basic goals such as confidentiality, authentication and non-repudiation, but also more elaborate services, such as key management, digital cash and electronic voting. Many examples of broken protocols are also discussed to enhance understanding of the engineering difficulties in building secure systems.

Runs in study period 2

Language-based Security

The course covers the principles of programming language-based techniques for computer security. The goal is understanding such application-level attacks as races, buffer overruns, covert channels, and code injection as well as mastering the principles behind such language-based protection techniques as static analysis, program transformation, and reference monitoring. The dual perspective of attack vs. protection is threaded through the lectures, laboratory assignments, and projects.

Runs in study period 4.

Network security

Why is it possible to break into networked applications and computer systems? What weaknesses are used? And what makes one protocol more secure than another? This course answers these questions and many more. We look at weaknesses that have plagued wired and wireless networked systems for years and investigate the security of countermeasures like firewalls and security protocols such as SSL, SSH and IPsec. Knowledge about possible threats and countermeasures is important for understanding what level of security a system and an application can offer.

Runs in study period 4

Security is becoming increasingly important for system design and development. System architects and designers must have security expertise, so that the systems they design do not fall victims to attacks. Software developers and engineers must have security expertise, so that the code they produce cannot be exploited. Security and network specialists must have critical knowledge of security principles and practice, in order to ensure the security of the systems they are responsible for.

Strong ties with industry

OWASP We have tight relations with the Open Web Application Security Project (OWASP). We are actively involved in both the Stockholm and Gothenburg OWASP chapters.

Security Arena Our major connection with industrial stake holders in Sweden is the Security Arena at the Lindholmen Science Park.

URBSEC (Urban Safety and Societal Security Research Center) offers a research interface between practice and academia where needs and problems as experienced by various social actors can be addressed.

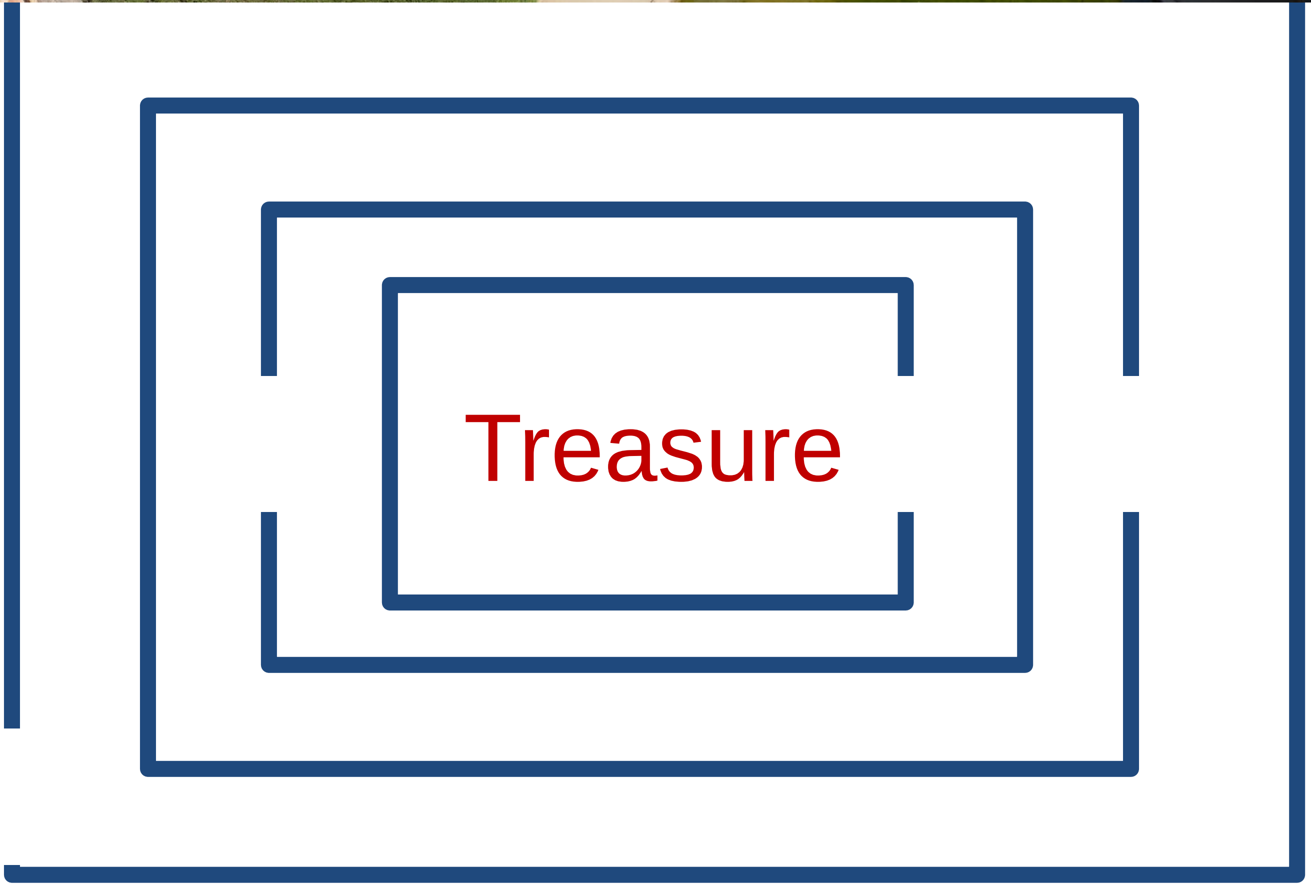
Cutting edge research

Crisalis is an EU project on security analysis for critical infrastructures in collaboration with eight academic and industrial partners across Europe.

WebSand is an EU project on server-driven outbound web application sandboxing in collaboration with K.U. Leuven, SAP, Siemens, and U. Passau.

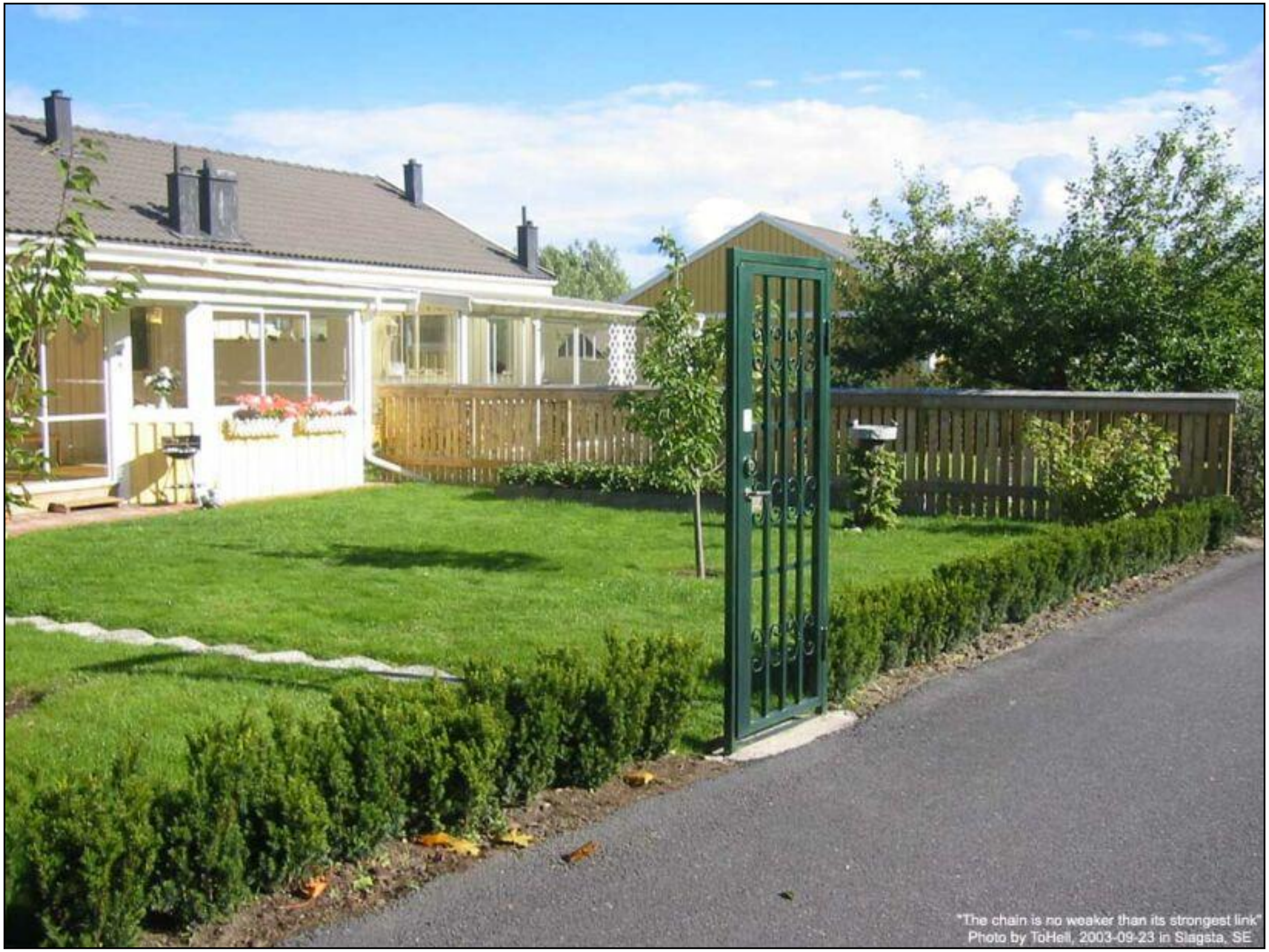
SysSec is an EU Network of Excellence on Managing Threats and Vulnerabilities in the Future Internet in collaboration with seven high-profile partners.

Disclaimer: Specialization course packages are no more and no less than wide range course lists aiming at in-depth focus in specific knowledge areas . Specialization course packages are thus far informal, and the diplomas will not mention them. At the same time, we will maintain this page and encourage referring to this page from your resume.



Chat!

Security is the lack of insecurity!



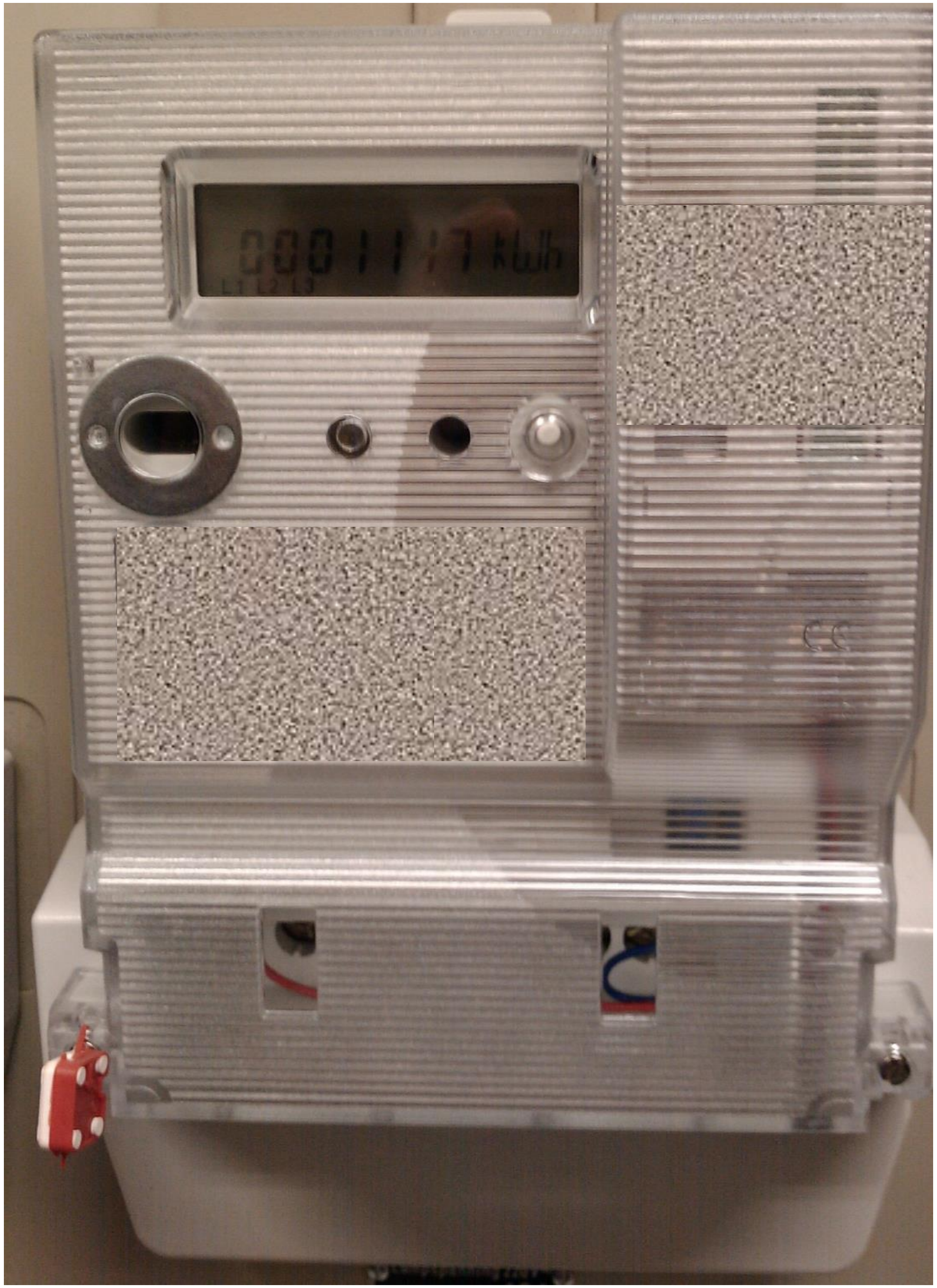
"The chain is no weaker than its strongest link"
Photo by ToHell, 2003-09-23 in Slagsta, SE

The Challenges of Computer Security

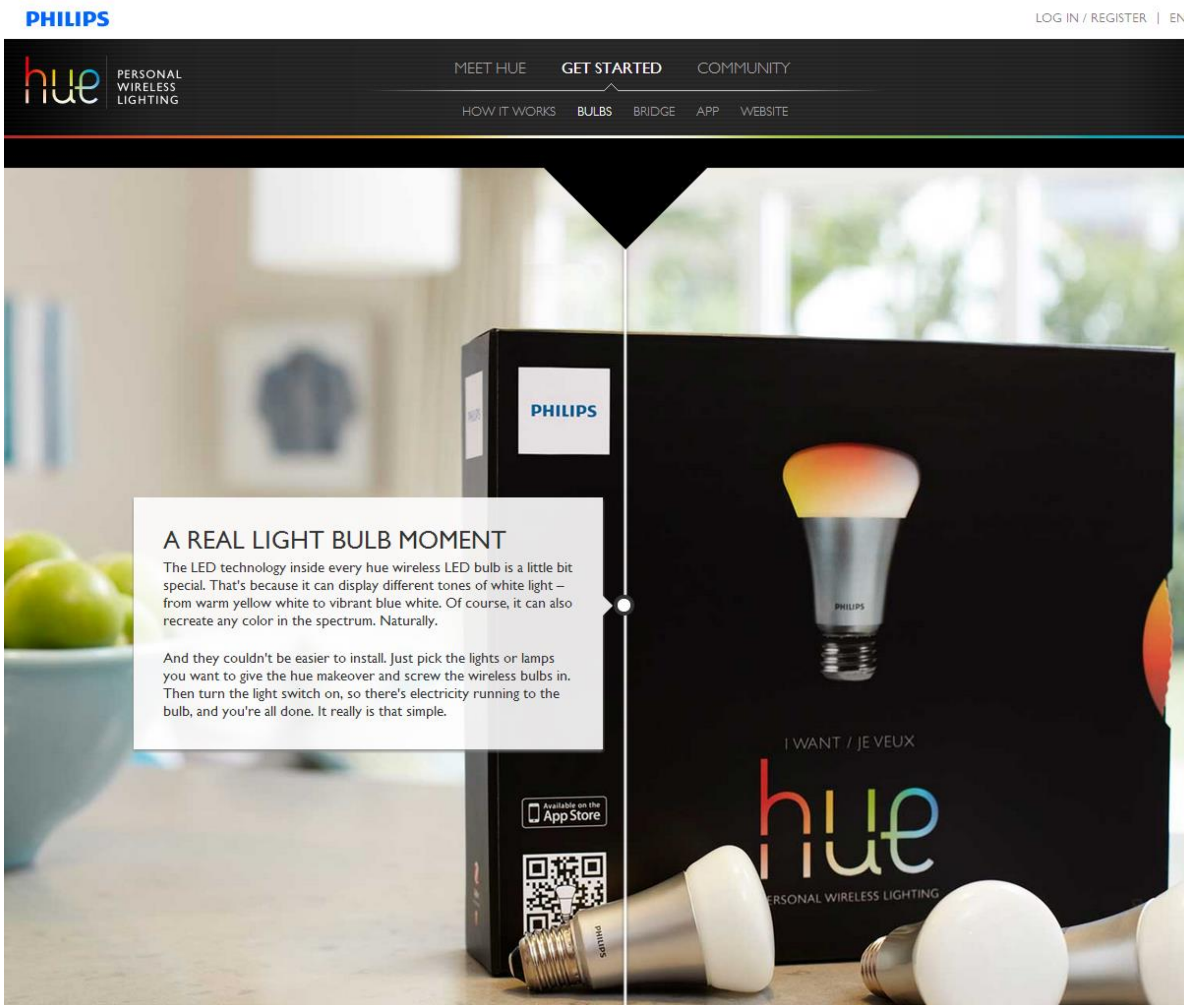
1. Security is not as simple as it may appear to the novice.
   - Possible to attack the security mechanism?
   - Security is not done in isolation from the rest of the system.
2. Security is a “chess game” between the attacker and the security administrator:
   - The attacker only needs to find a ***single*** vulnerability to penetrate the system, while the administrator needs to patch ***all*** holes to ensure system security.
3. Natural tendency to disregard security problems ***until*** a security failure occurs.
4. Security is a process ➔ constant monitoring, long-term perspective.
5. Security is often an afterthought – added after the system has been designed.
6. Some users think security is restricting them in their job.

See Stallings & Brown: Computer Security, Pearson 2008, ISBN: 978-0-13-513711-6, page 11

[redacted] [redacted]


000 1117 kWh
L1 L2 L3

And even lamps need security



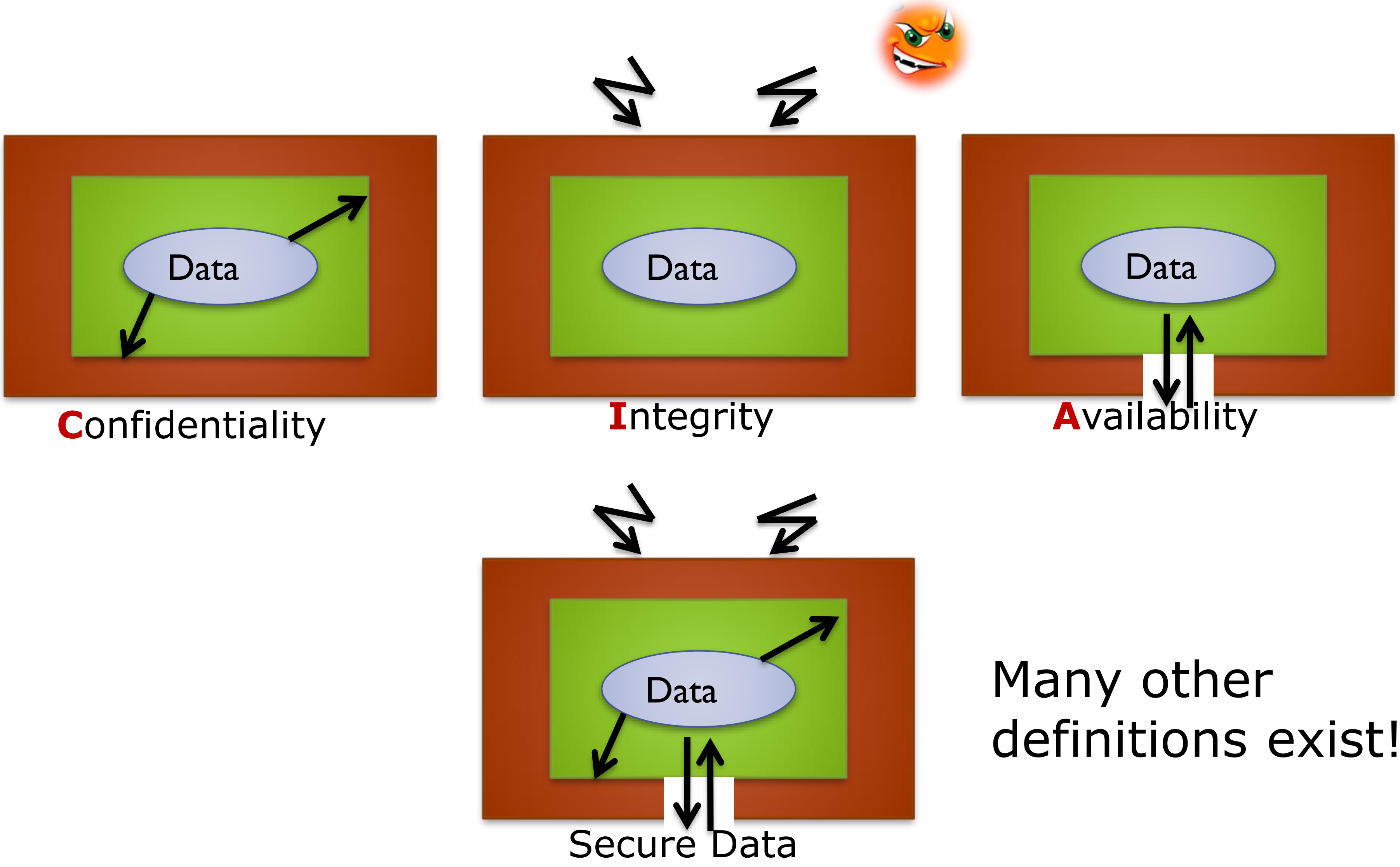
A REAL LIGHT BULB MOMENT

The LED technology inside every hue wireless LED bulb is a little bit special. That's because it can display different tones of white light – from warm yellow white to vibrant blue white. Of course, it can also recreate any color in the spectrum. Naturally.

And they couldn't be easier to install. Just pick the lights or lamps you want to give the hue makeover and screw the wireless bulbs in. Then turn the light switch on, so there's electricity running to the bulb, and you're all done. It really is that simple.

http://www.meethue.com/en-US

Security of Data – “CIA”



Data

Confidentiality

Data

Integrity

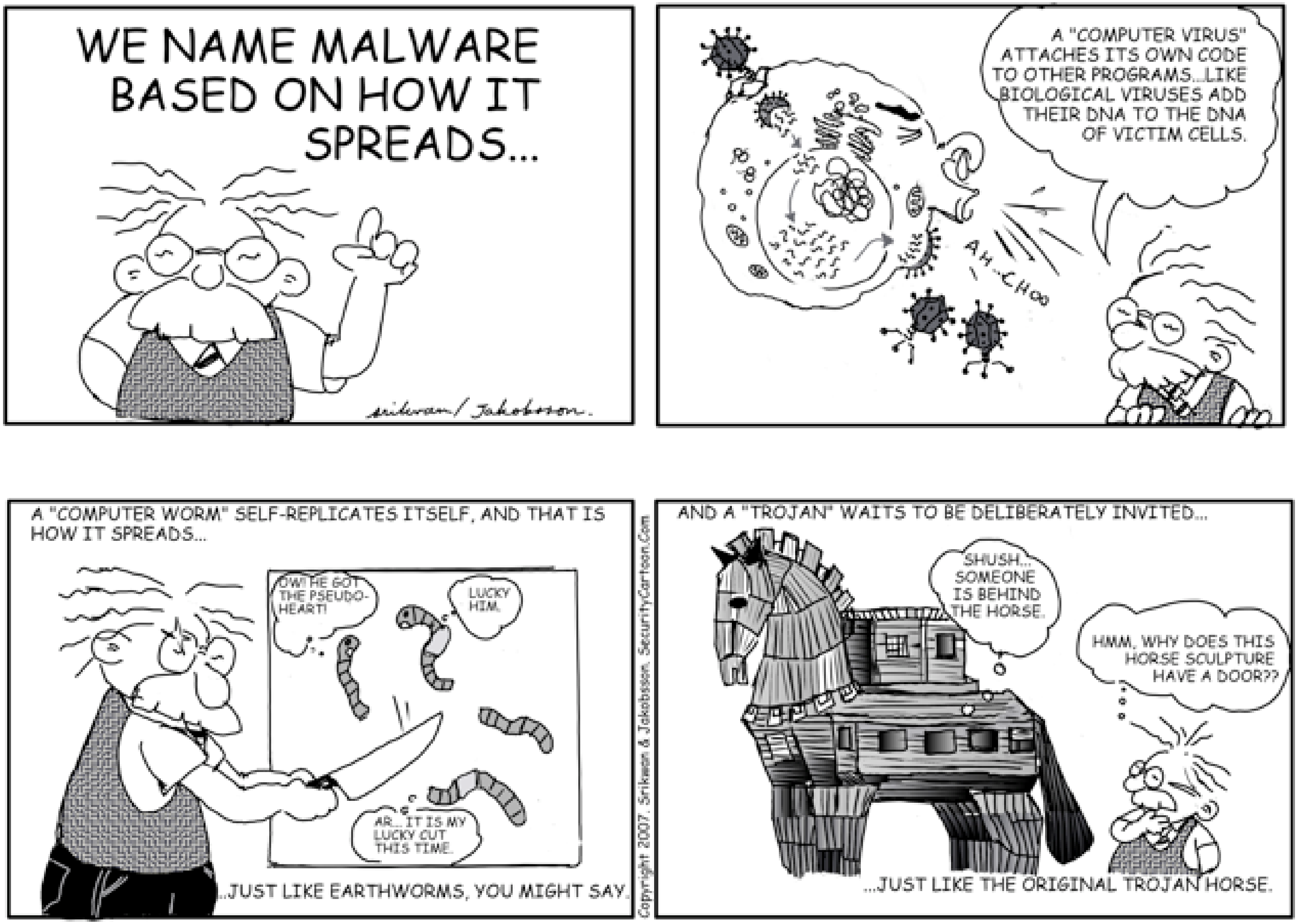
Data

Availability

Data

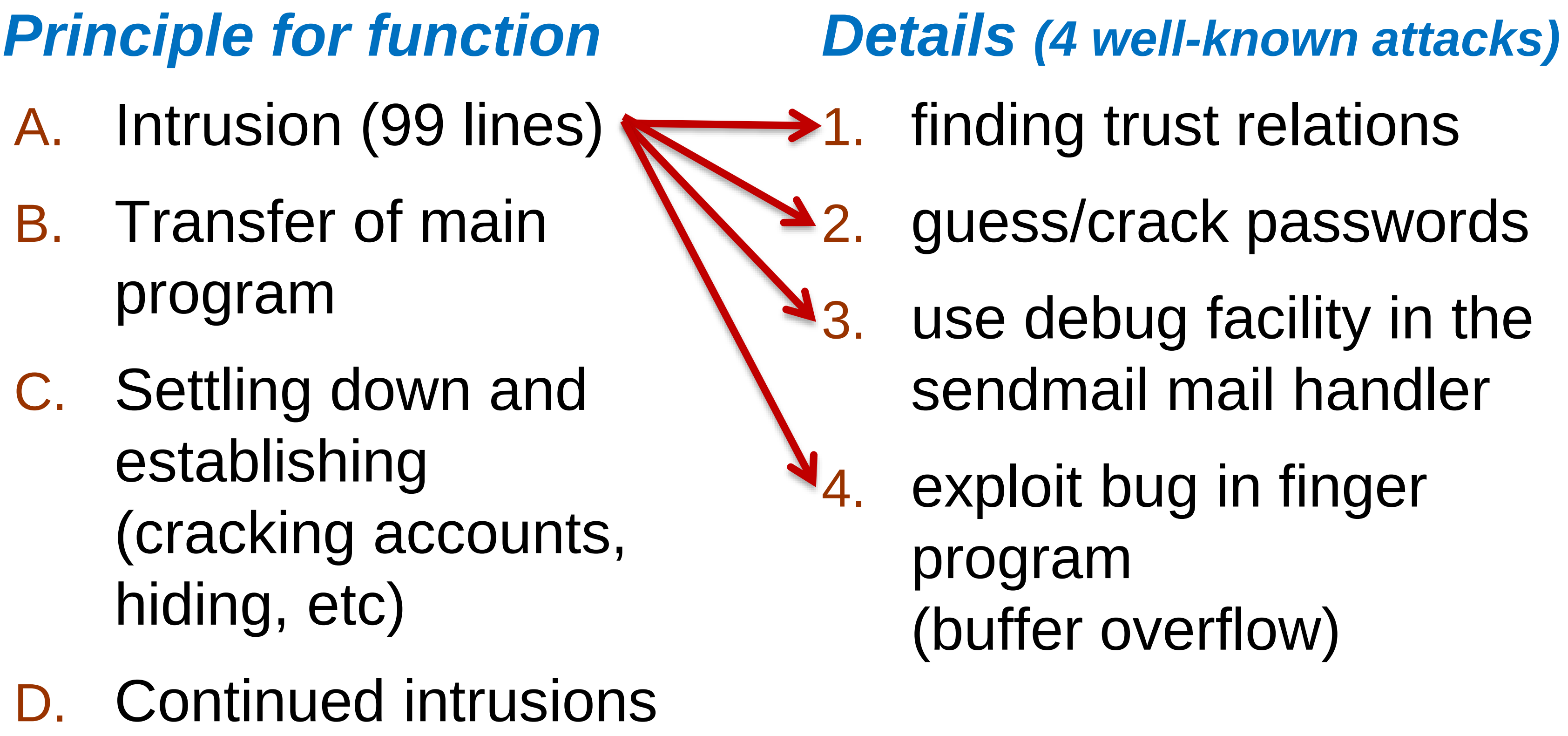
Secure Data

Many other definitions exist!


WE NAME MALWARE BASED ON HOW IT SPREADS...
A "COMPUTER VIRUS" ATTACHES ITS OWN CODE TO OTHER PROGRAMS...LIKE BIOLOGICAL VIRUSES ADD THEIR DNA TO THE DNA OF VICTIM CELLS.
AH-CHOO
A "COMPUTER WORM" SELF-REPLICATES ITSELF, AND THAT IS HOW IT SPREADS...
OW! HE GOT THE PSEUDO-HEART!
LUCKY HIM.
AR... IT IS MY LUCKY CUT THIS TIME.
...JUST LIKE EARTHWORMS, YOU MIGHT SAY.
Copyright 2007, Srikwan & Jakobsson, SecurityCartoon.Com
AND A "TROJAN" WAITS TO BE DELIBERATELY INVITED...
SHUSH... SOMEONE IS BEHIND THE HORSE.
HMM, WHY DOES THIS HORSE SCULPTURE HAVE A DOOR??
...JUST LIKE THE ORIGINAL TROJAN HORSE.

Reproduced with permission. Please visit www.SecurityCartoon.com for more material.

The Morris Worm – Steps



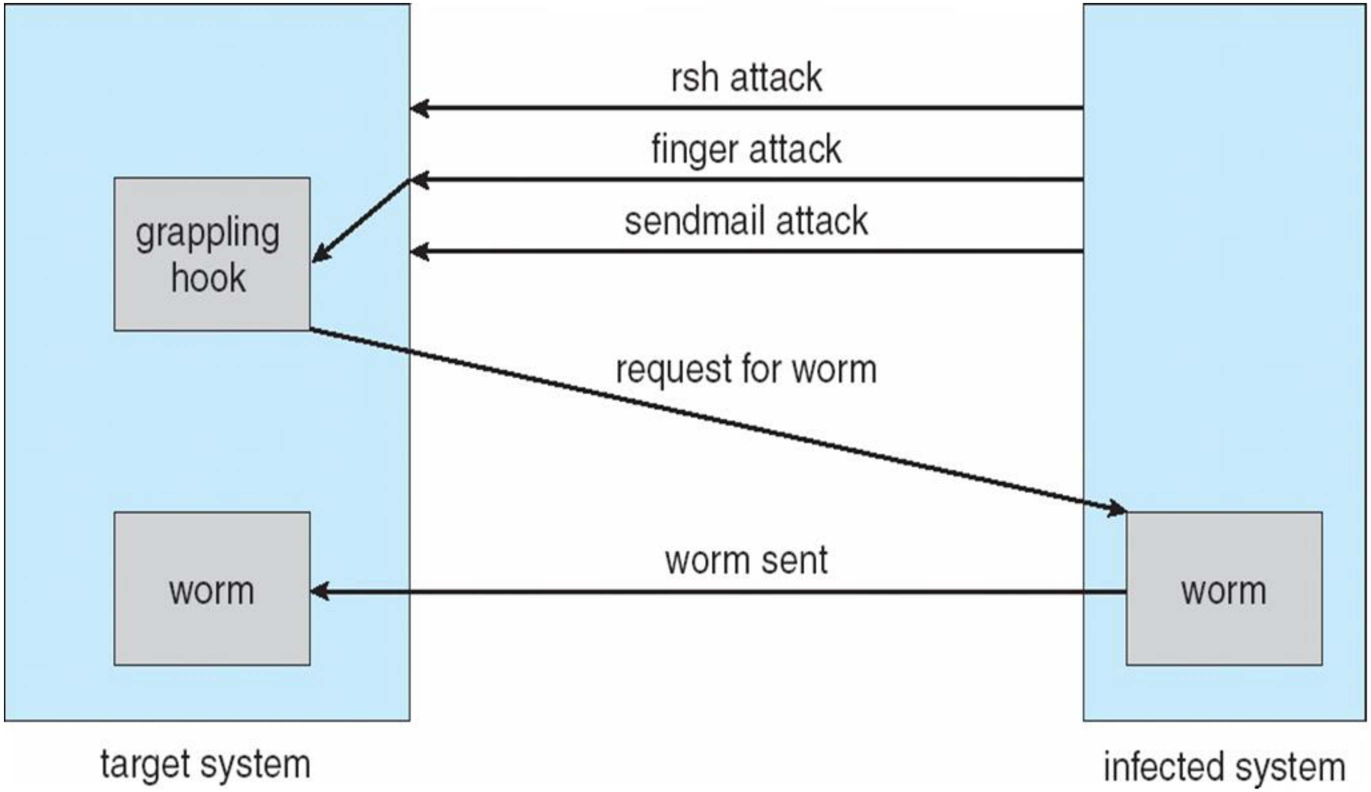
Principle for function

A. Intrusion (99 lines)
B. Transfer of main program
C. Settling down and establishing (cracking accounts, hiding, etc)
D. Continued intrusions

***Details** (4 well-known attacks)*

1. finding trust relations
2. guess/crack passwords
3. use debug facility in the sendmail mail handler
4. exploit bug in finger program (buffer overflow)

The Morris Internet Worm



Finding trust relations

- The worm obtains host addresses by examining
  - the system tables */etc/hosts.equiv* and
  - */.rhosts*,
  - user files like *.forward*
  - dynamic routing information produced by the *netstat*, and finally
  - randomly generated host addresses on local networks.
- It ranks these by order of preference, but what does it mean?

The /etc/hosts.equiv File
The /etc/hosts.equiv file contains

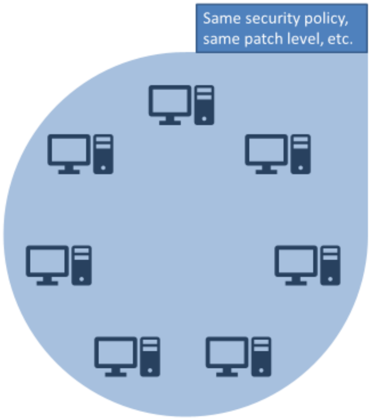
a list of trusted hosts for a remote system.

If a user attempts to log in remotely (using rlogin) from one of the hosts listed in this file, and if the remote system can access the user's password entry, the remote system allows the user to log in

without a password.

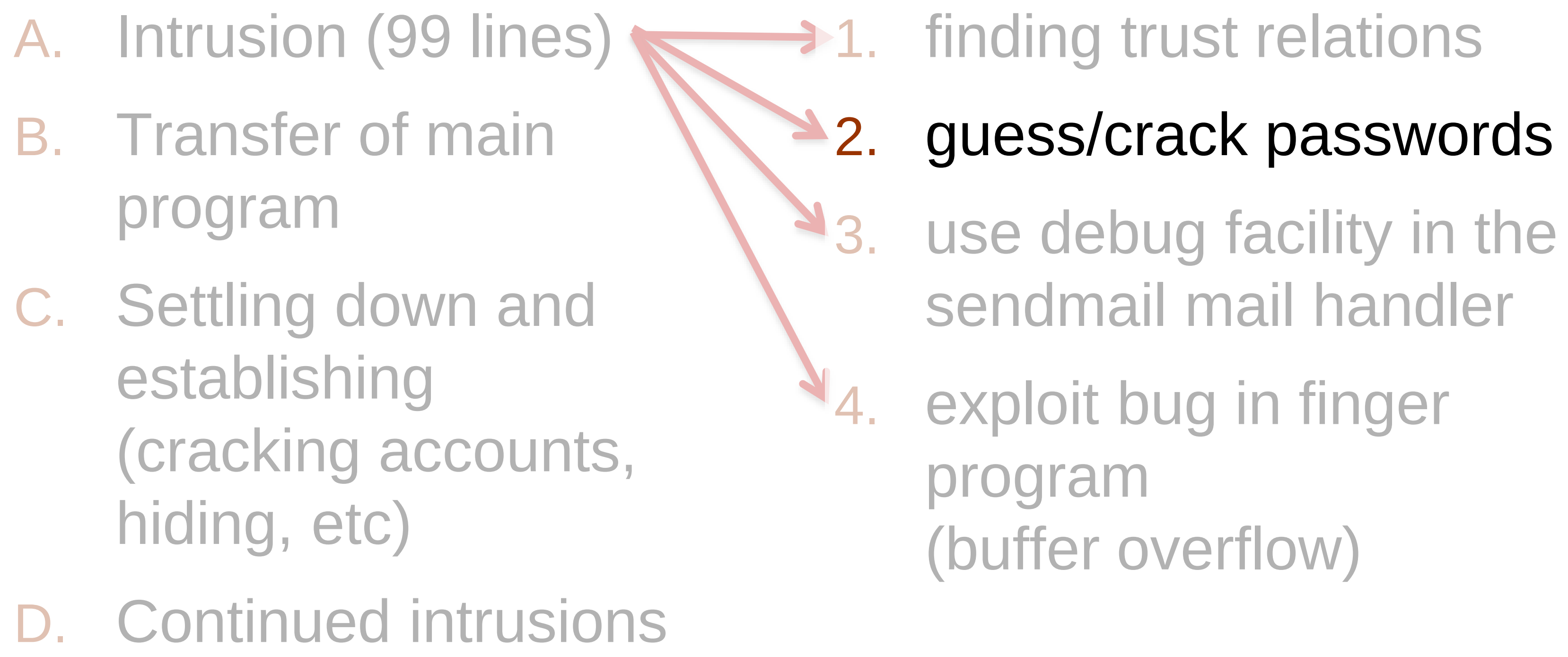
Finding trust relations

- The worm obtains host addresses by examining
  - the system tables */etc/hosts.equiv* and
  - */.rhosts*,
  - user files like *.forward*
  - dynamic routing information produced by the *netstat*, and finally
  - randomly generated host addresses on local networks.
- It ranks these by order of preference, but what does it mean?
- **It contains names of local machines that are likely to permit unauthenticated connections.**



Can you setup the same trust relations with modern programs?

The Morris Worm – Steps



Principle for function

A. Intrusion (99 lines)

B. Transfer of main program

C. Settling down and establishing (cracking accounts, hiding, etc)

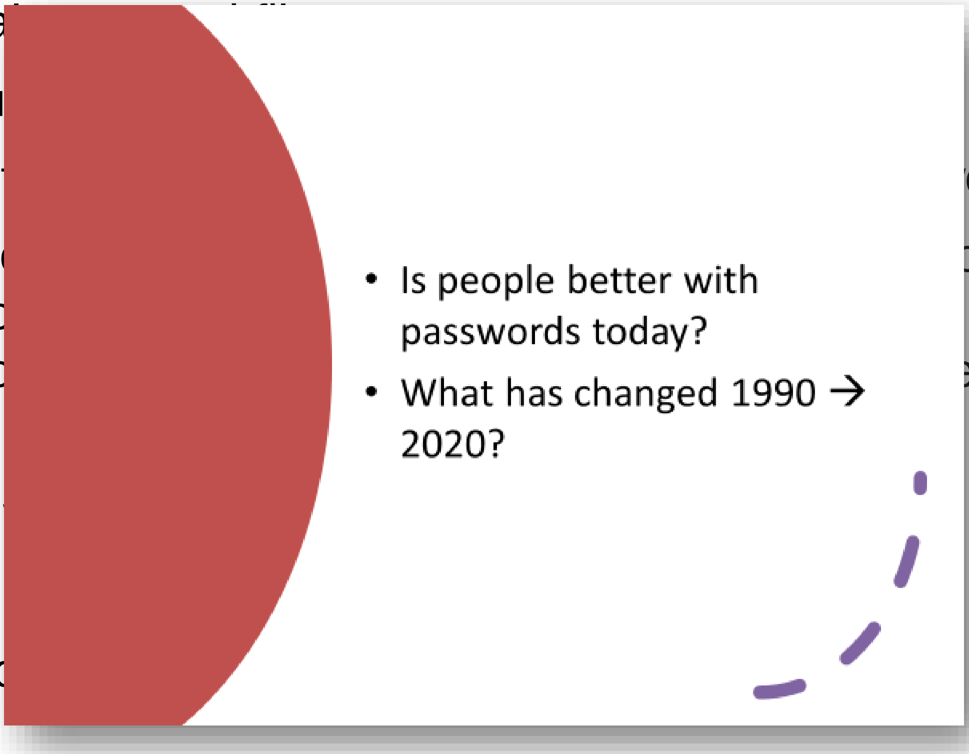
D. Continued intrusions

***Details** (4 well-known attacks)*

1. finding trust relations
2. guess/crack passwords
3. use debug facility in the sendmail mail handler
4. exploit bug in finger program (buffer overflow)

Guess/crack passwords

- **Assumption**: *A user is using the same passwords on all systems*
- Crack loca
  - Each u of it
  - A list o ould be likely
    - aaa c ouscous hacker nutritic singer airplane creosc etin happening ocelot
  - All the
- So are pe



- Is people better with passwords today?
- What has changed 1990 → 2020?

Breakout rooms

Guess/crack passwords

s	1487
123456	1290
hejhej	671
hejsan	580
fotboll	389
123456789	368
bajskorv	328
sommar	322
blomma	285
123123	248
mamma	239
dinmamma	220
johanna	188
1234	169
12345	164
amanda	161
smulan	154
hejhejhej	145
bajs	143
kalleanka	143
qwerty	142
hemligt	136
abc123	136
sverige	135

Thanks to Martin Holst Swende

Guess/crack passwords

s	1487	johanna	188
123456	1290	1234	169
hejhej	671	12345	164
h			161
f			154
1			145
b			143
s			143
blomma	285	qwerty	142
123123	248	hemligt	136
mamma	239	abc123	136
dinmamma	220	sverige	135

hundar (133) > katter (110)

mammamia (118)

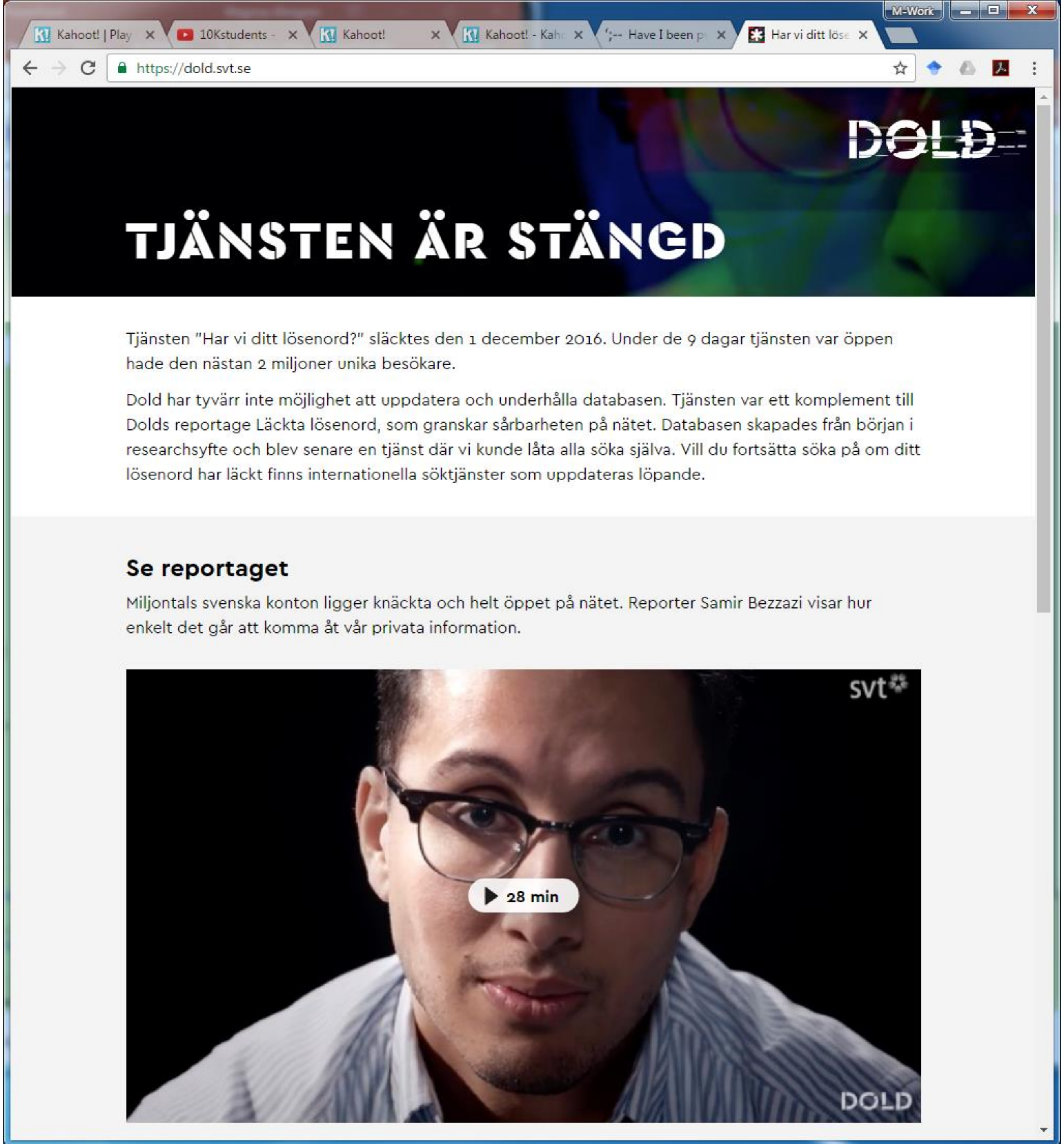
sommarlov (103)

amanda, linnea, sandra, andersson, emelie, matilda

Thanks to Martin Holst Swende

Rainbow table

- A **rainbow table** is a precomputed **table** for reversing cryptographic hash functions, usually for cracking password hashes. **Tables** are usually used in recovering a plaintext password up to a certain length consisting of a limited set of characters.
- Rainbow table - Wikipedia, the free encyclopedia
- en.wikipedia.org/wiki/**Rainbow_table**



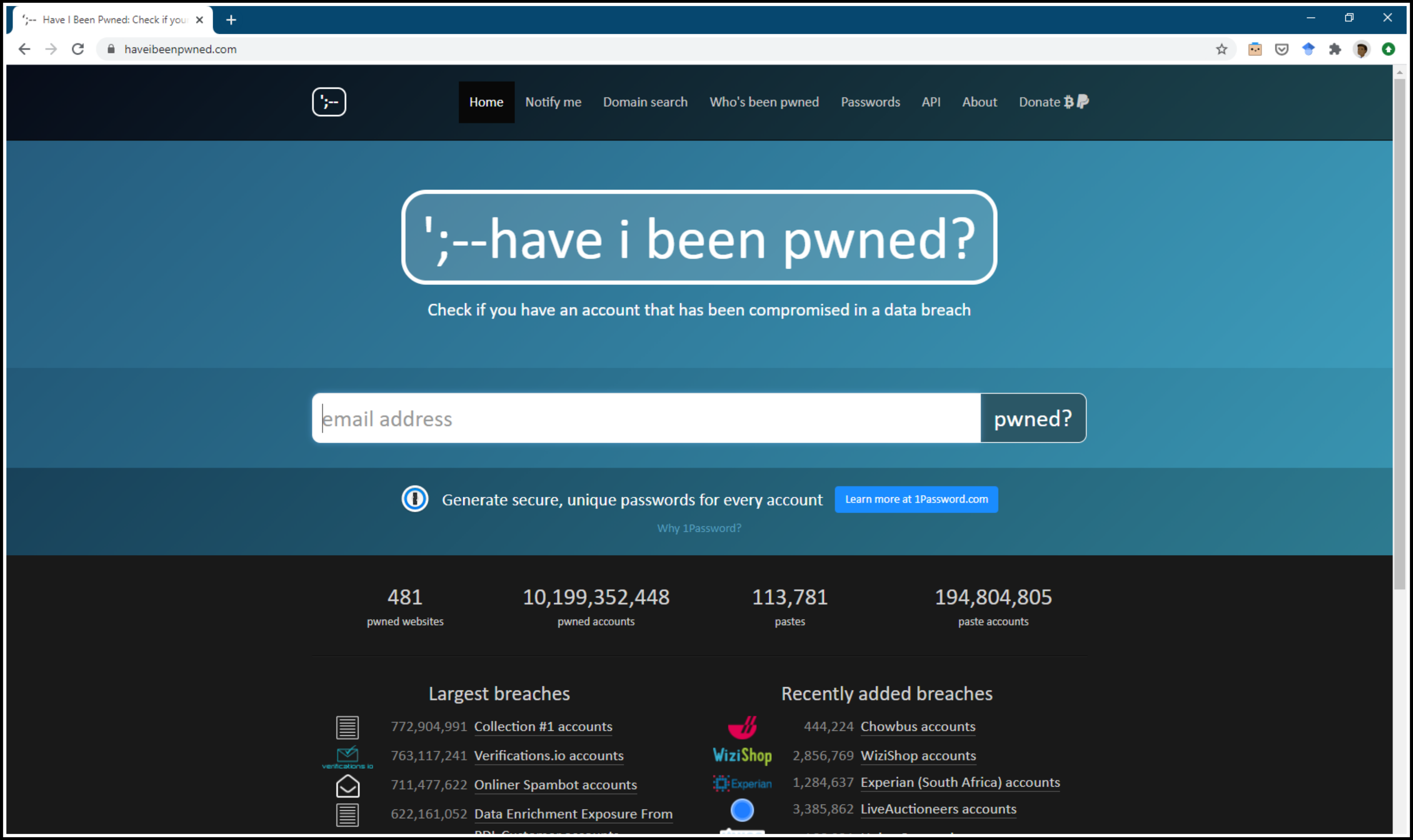
TJÄNSTEN ÄR STÄNGD

Tjänsten "Har vi ditt lösenord?" släcktes den 1 december 2016. Under de 9 dagar tjänsten var öppen hade den nästan 2 miljoner unika besökare.

Dold har tyvärr inte möjlighet att uppdatera och underhålla databasen. Tjänsten var ett komplement till Dolds reportage Läckta lösenord, som granskar sårbarheten på nätet. Databasen skapades från början i researchsyfte och blev senare en tjänst där vi kunde låta alla söka själva. Vill du fortsätta söka på om ditt lösenord har läckt finns internationella söktjänster som uppdateras löpande.

Se reportaget

Miljontals svenska konton ligger knäckta och helt öppet på nätet. Reporter Samir Bezzazi visar hur enkelt det går att komma åt vår privata information.



';--have i been pwned?

Check if you have an account that has been compromised in a data breach

Generate secure, unique passwords for every account

481
pwned websites

10,199,352,448
pwned accounts

113,781
pastes

194,804,805
paste accounts

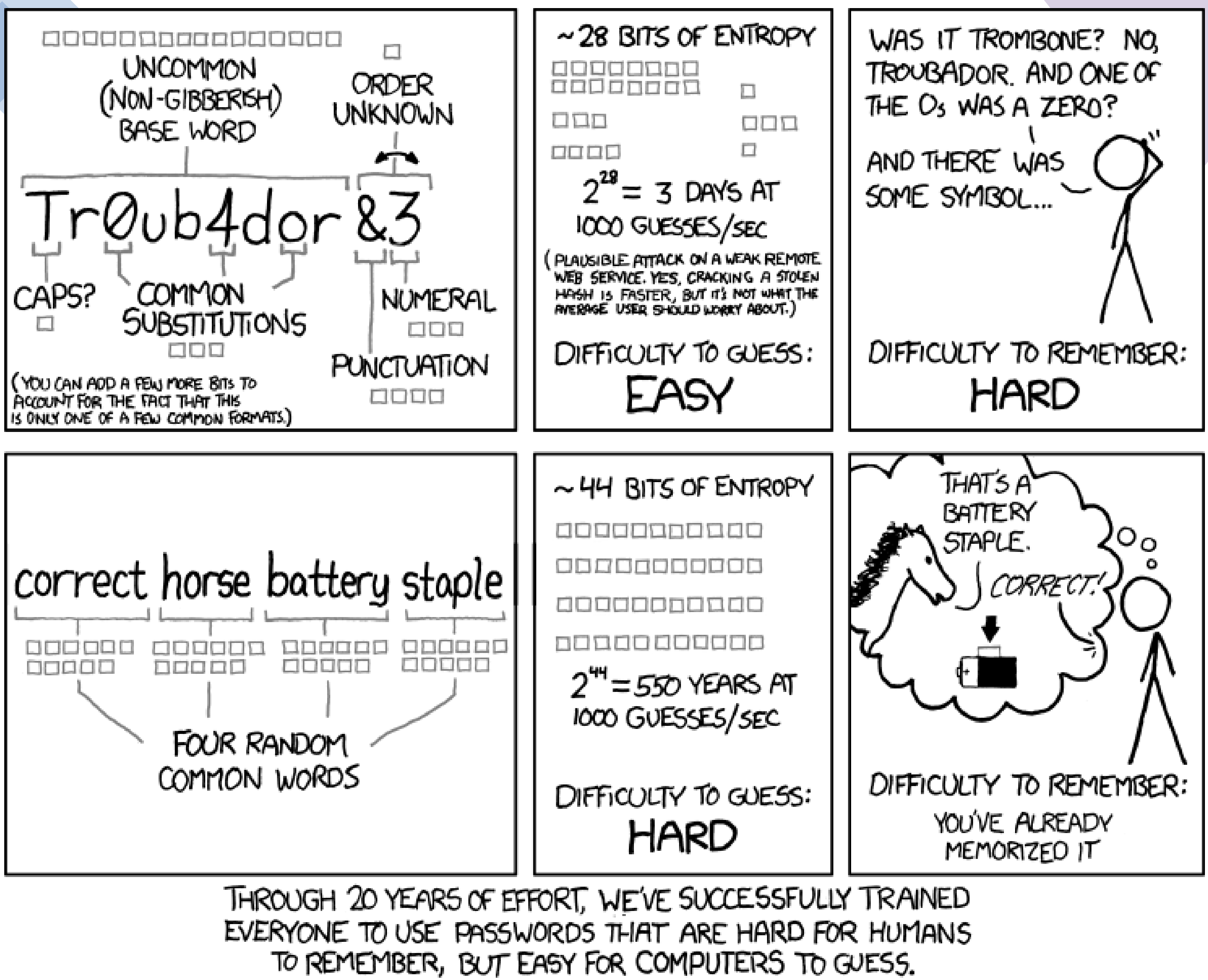
Largest breaches

772,904,991	Collection #1 accounts
763,117,241	Verifications.io accounts
711,477,622	Onliner Spambot accounts
622,161,052	Data Enrichment Exposure From PDL Customer accounts

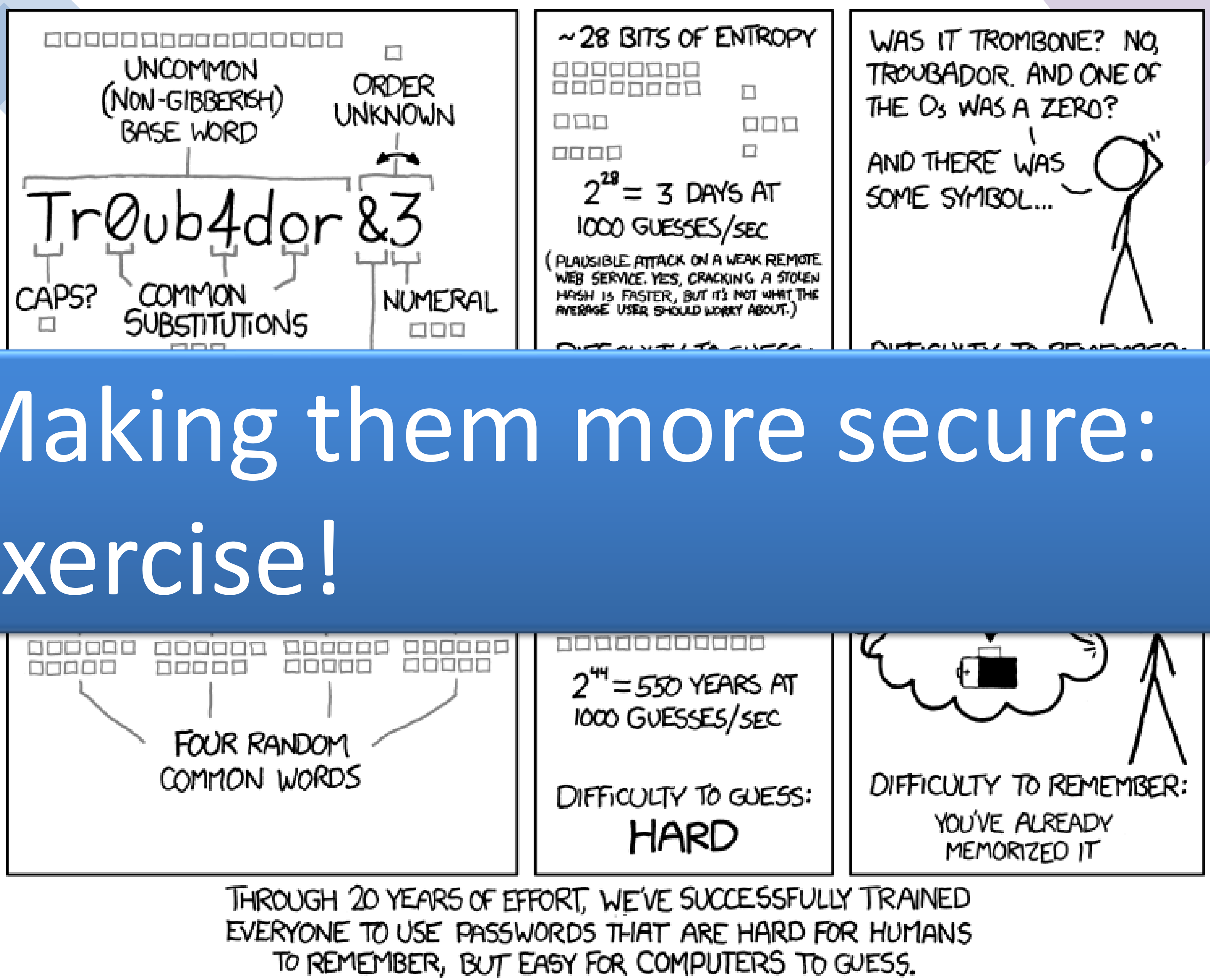
Recently added breaches

444,224	Chowbus accounts
2,856,769	WiziShop accounts
1,284,637	Experian (South Africa) accounts
3,385,862	LiveAuctioneers accounts

https://xkcd.com/936/ ---- Password managers

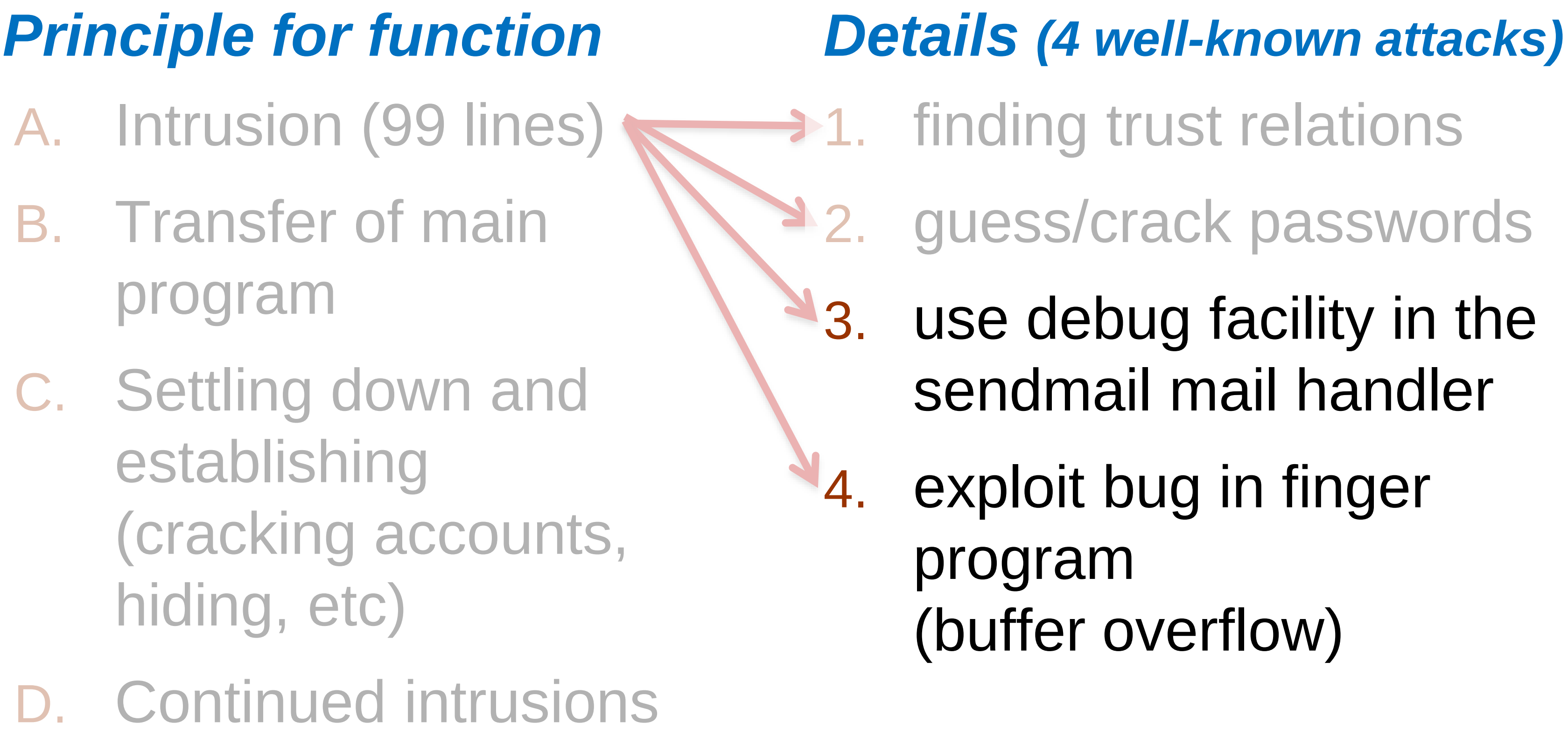

UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Making them more secure: Exercise!


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
~28 BITS OF ENTROPY
2^{28} = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
FOUR RANDOM COMMON WORDS
2^{44} = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

https://xkcd.com/936/ ---- Password managers

The Morris Worm – Steps



Principle for function

A. Intrusion (99 lines)

B. Transfer of main program

C. Settling down and establishing (cracking accounts, hiding, etc)

D. Continued intrusions

Details** **(4 well-known attacks)

1. finding trust relations
2. guess/crack passwords
3. use debug facility in the sendmail mail handler
4. exploit bug in finger program
(buffer overflow)

Use debug facility in the sendmail

- "trap door" in the *sendmail* SMTP mail service,
- A bug in debugging code allows the daemon to to execute a command interpreter and download code across a mail connection.

Exploit bug in finger program
Buffer Overflow

A (really) simple introduction to buffer overflows

Herbert Bos

Vrije Universiteit Amsterdam

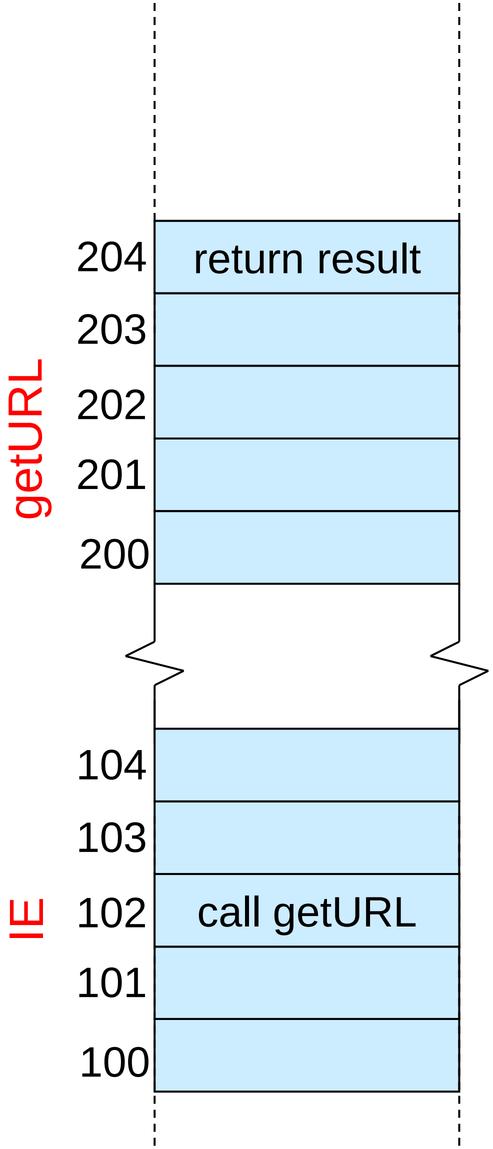
Exploits

- program has a security hole
- exploit = input that abuses the vulnerability

- In this module we will discuss an example: the Buffer overflow

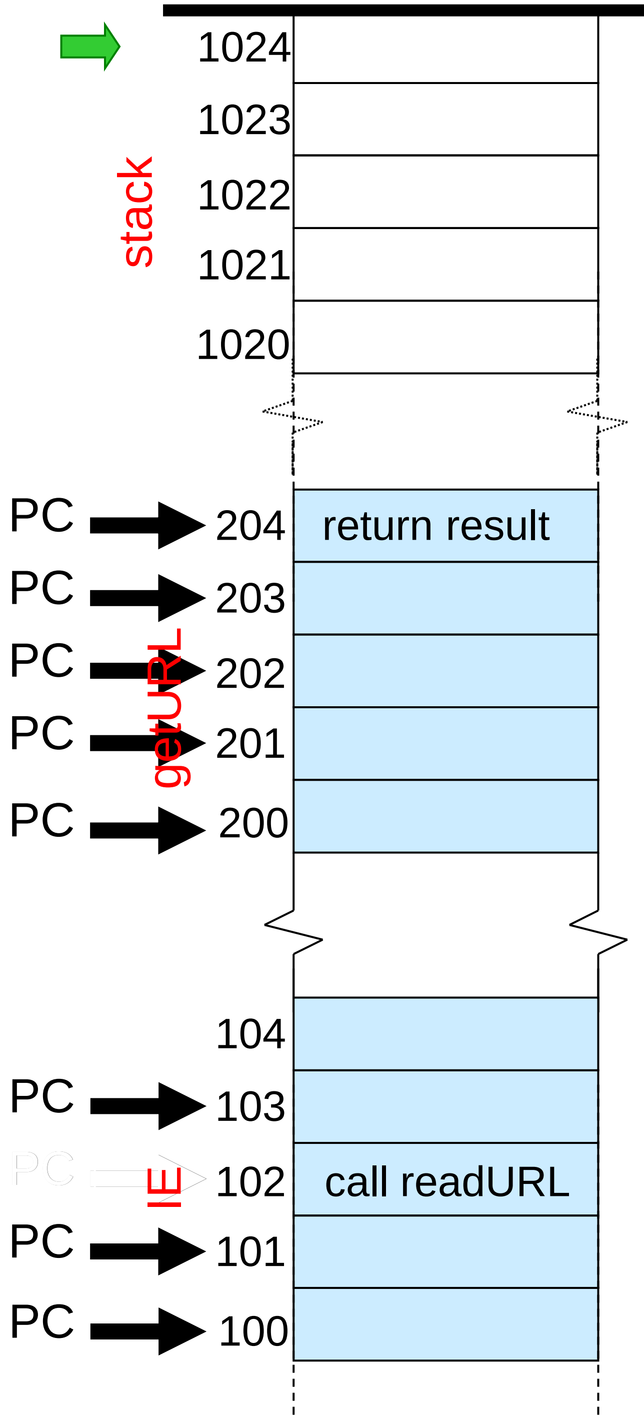
software

- sequence of instructions in memory
- logically divided in functions that call each other
  - function ‘IE’ calls function ‘getURL’ to read the corresponding page
- in CPU, the program counter contains the address in memory of the next instruction to execute
  - normally this is the next address (instruction 100 is followed by instruction 101, etc)
  - not so with function call



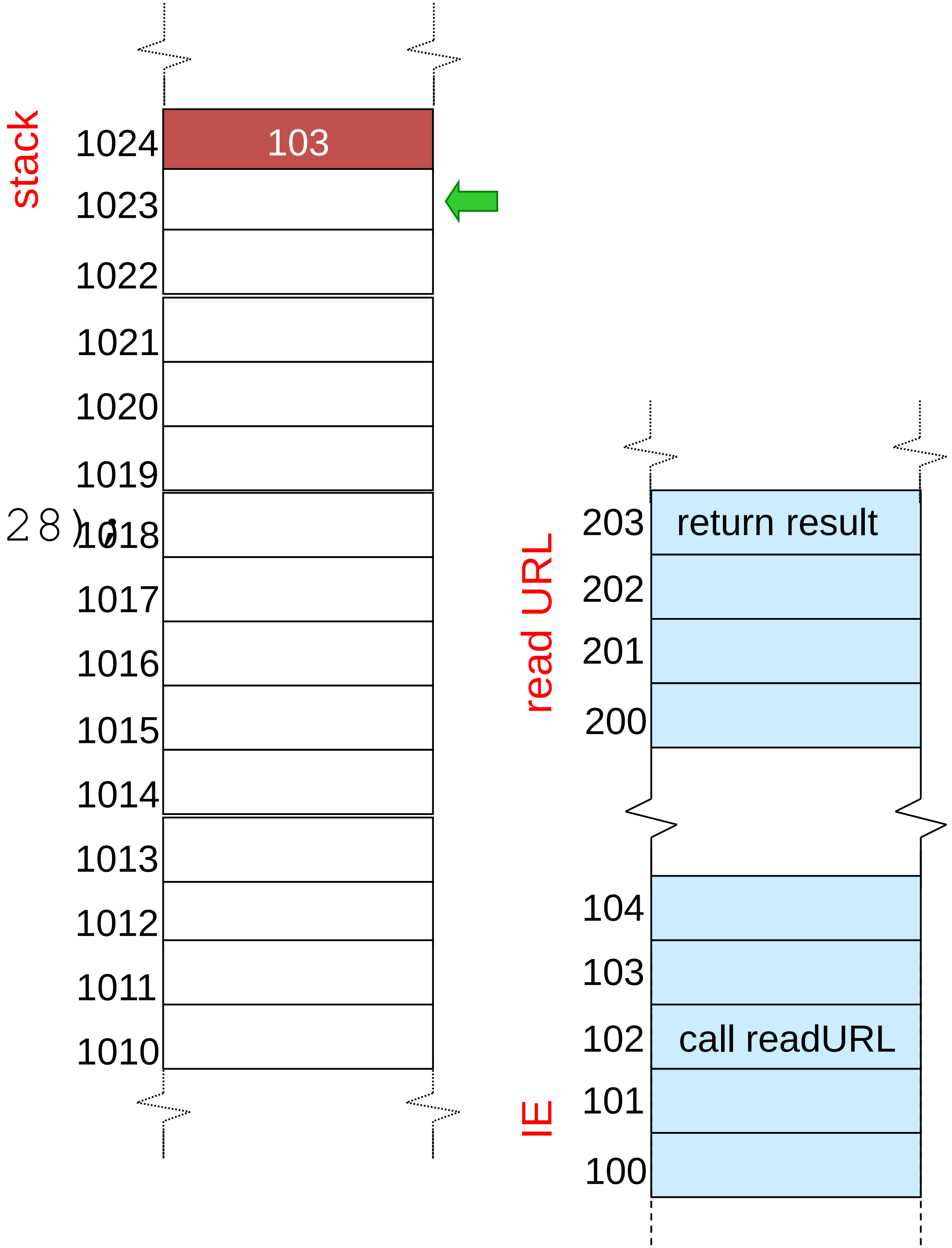
software

- so how does our CPU know where to return?
  - it keeps administration
  - on a 'stack'



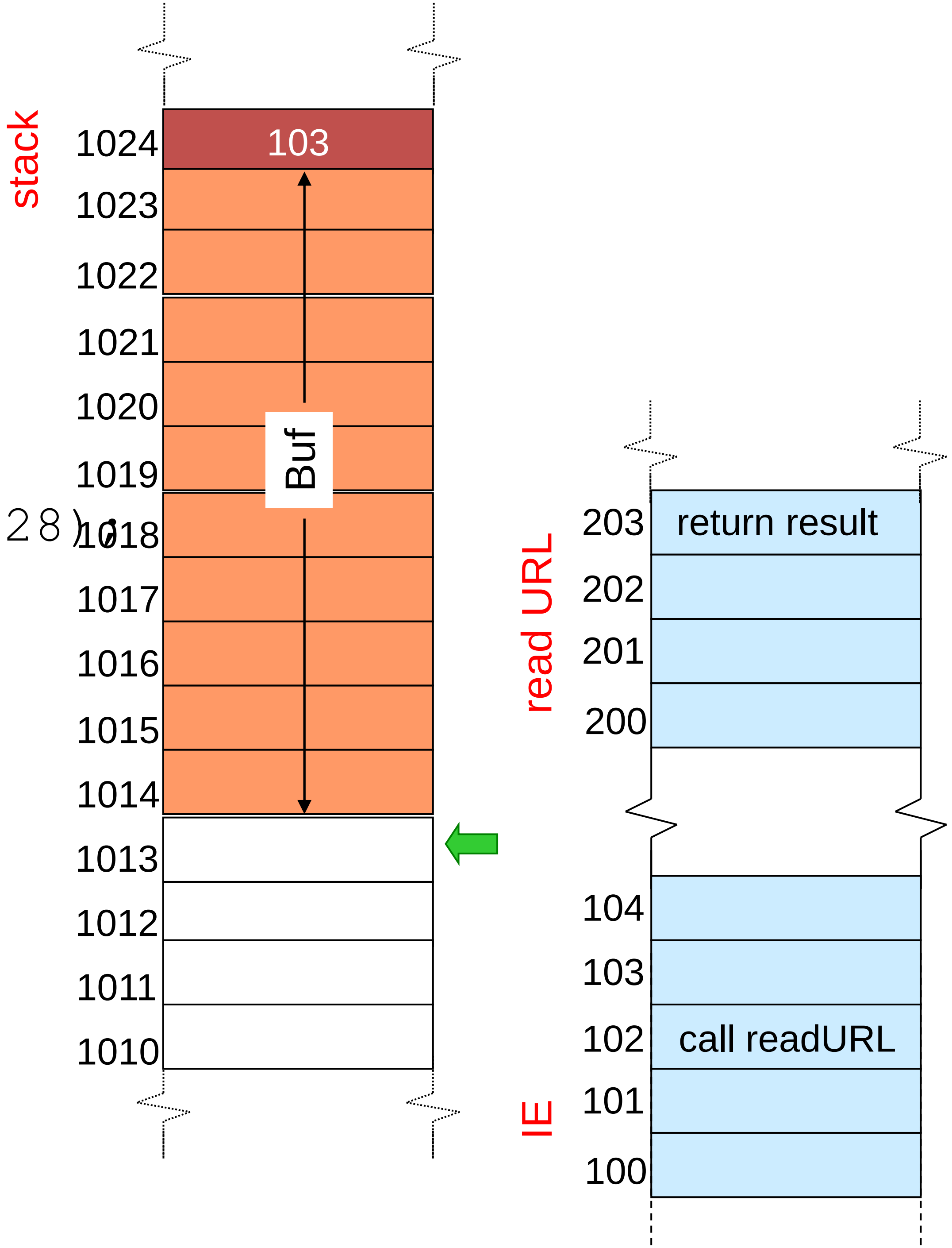
real functions have variables

```
getURL ()
{
   char Buf[10];
   read(keyboard,Buf,128);
   get_webpage (Buf);
}
IE ()
{
   getURL ();
}
```



real functions have variables

```
getURL ()
{
   char Buf[10];
   read(keyboard,Buf,128);
   get_webpage (Buf);
}
IE ()
{
   getURL ();
}
```

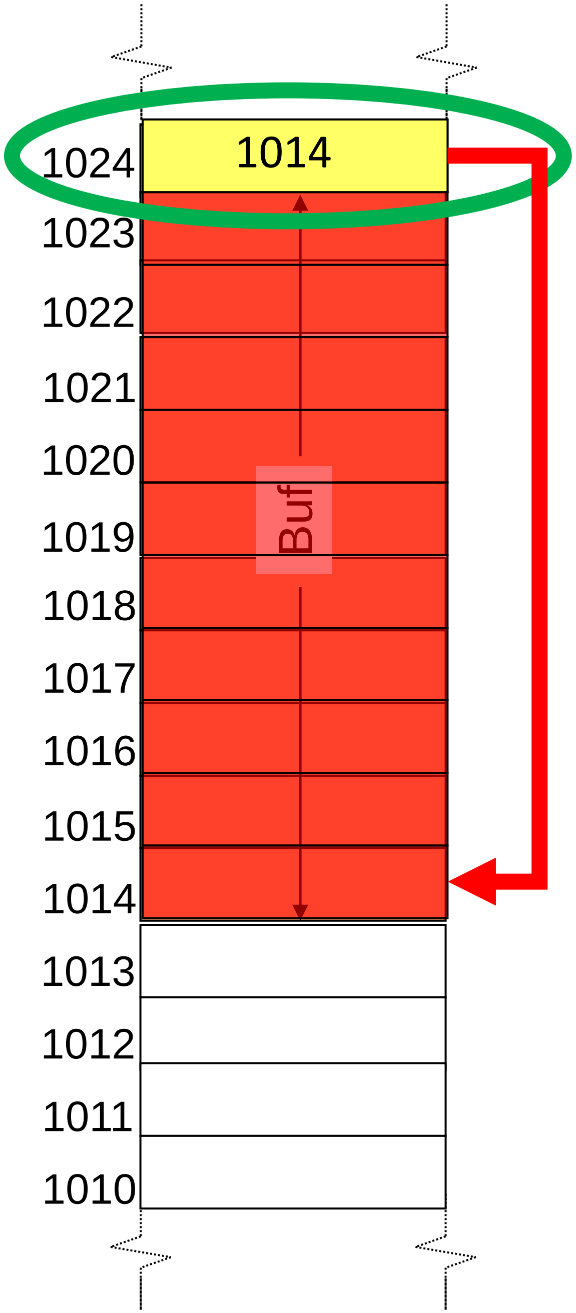


what is next?

- we have learned a lot
- but where are the vulnerabilities?
- and how do we exploit them?

Exploit

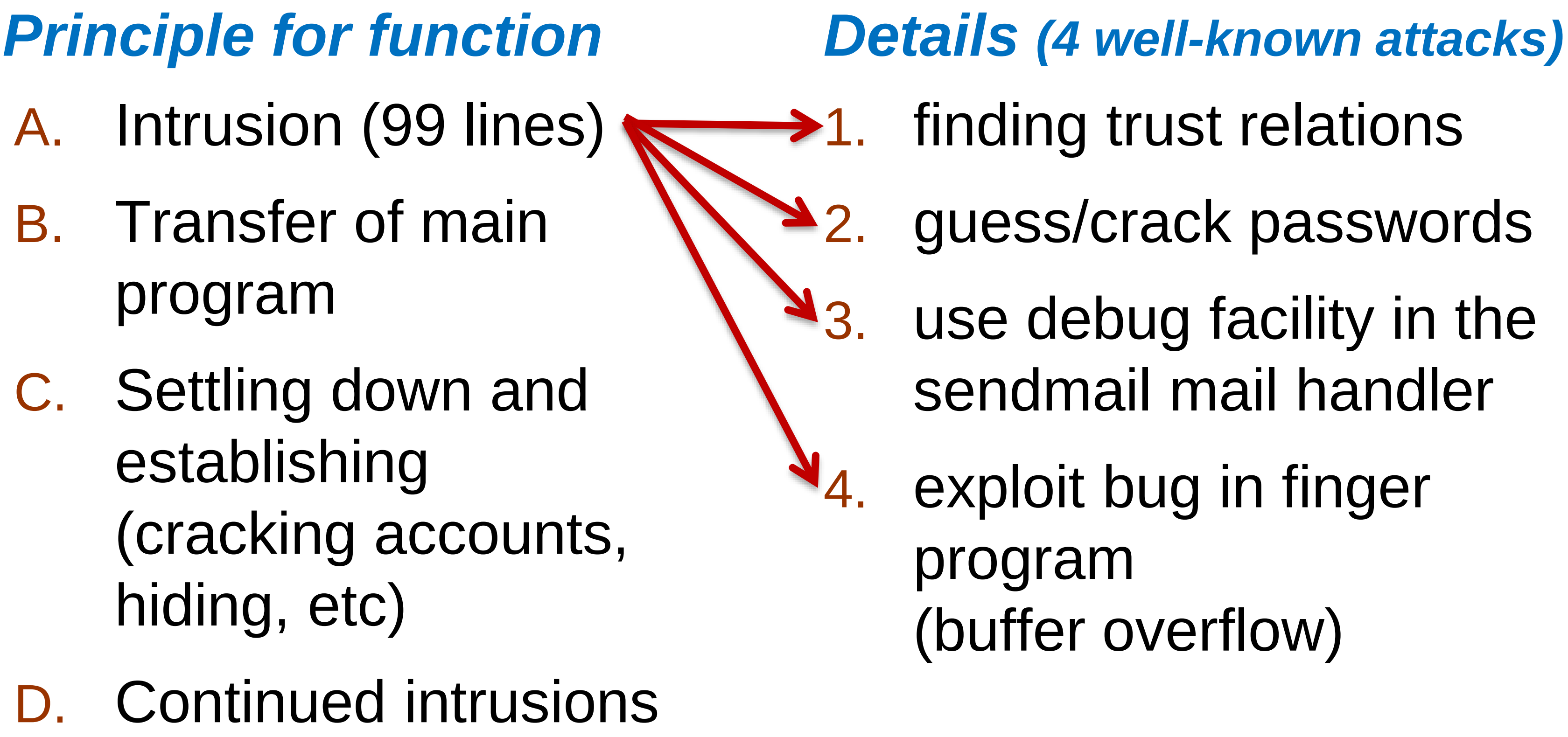
```
getURL ()
{
   char Buf[10];
   read(keyboard,Buf,128);
   get_webpage (Buf);
}
IE ()
{
   getURL ();
}
```



That is it, really

- all we need to do is stick our program in the buffer

The Morris Worm – Steps



Principle for function

A. Intrusion (99 lines)
B. Transfer of main program
C. Settling down and establishing (cracking accounts, hiding, etc)
D. Continued intrusions

Details (4 well-known attacks)

1. finding trust relations
2. guess/crack passwords
3. use debug facility in the sendmail mail handler
4. exploit bug in finger program (buffer overflow)

Internet Worm – Establishing

- **(B) Program transfer**
  - After the intrusion the program (~200 Kbytes) was transferred in a secure way (!)
- **(C) Establishing**
  - guess/crack passwords (root password was not utilised!)
  - camouflage activities (fork, simple EOR-encryption, no copy left on disk)
    Compare with: stealth viruses
  - one-time password for program transfer
- **(D) Continued Intrusions**
  - New machines were infected. There were facilities in the code to avoid multiple infections, but they did not work.

    There can also be bugs in malware…

    Thus, the main result was that the computers/network were overloaded.

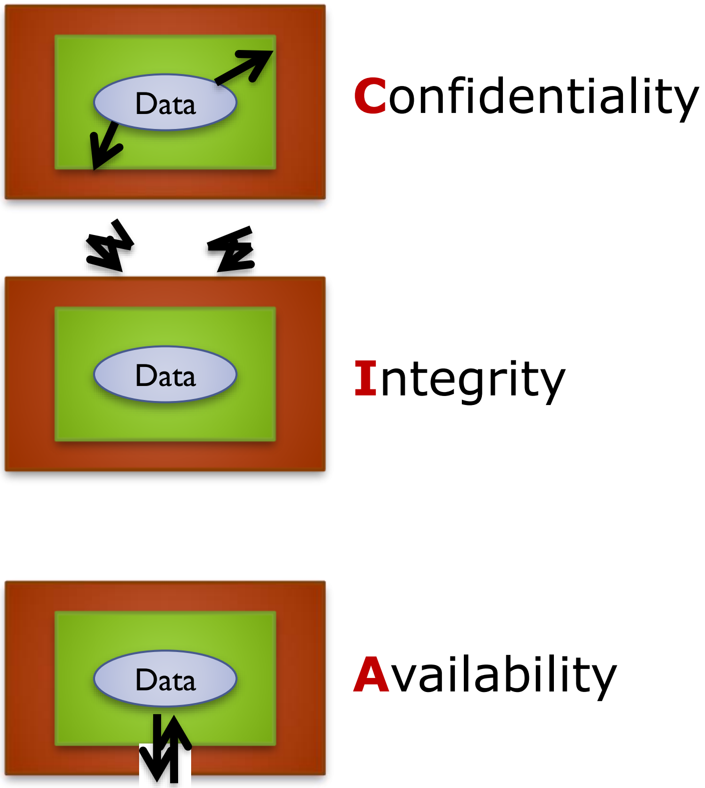
CI**A – an availability failure**

Protection
Access Control

Overview of Access Control

- Central element to computer security
  - Prevent unauthorized users from gaining access to a resource
  - Prevent authorized users from accessing a resource in an unauthorized manner
  - Enable legitimate users to access resources in an authorized manner

- CIA-model



Confidentiality

Integrity

Availability

Actors

- **Subjects →** Generalization to ***domains***
  - Users, processes, etc
- **Objects**
  - Files, Memory locations, users, processes [different modules / type]
  - ***access control matrix***
- **Access Rights**
  - Read, write, execute, delete, create, search
  - Transfer rights, grant rights, create / destroy object

Authentication KEY for access control to work

Garbage in, garbage out

Properties of access control

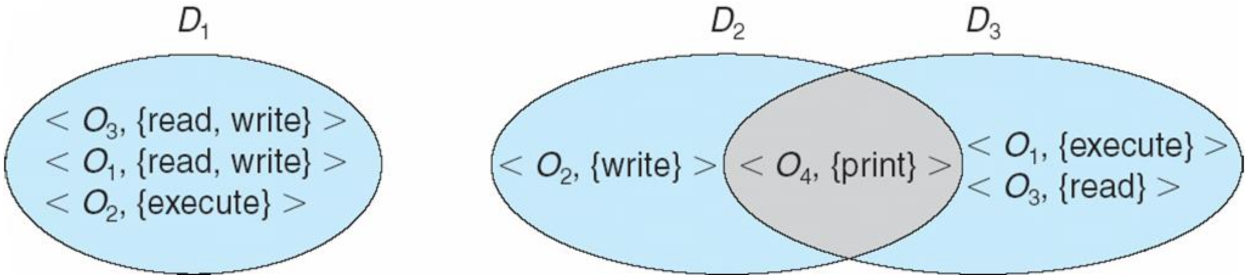
- **Reliable input**
  - A user needs to be authentic! → Authentication needed
- **Least Privilege** (similar to ”need to know”)
  - Only grant minimum authorization to do the job
  - Programs, users and systems should be given just enough **privileges** to perform their tasks
  - Limits damage if entity has a bug, gets abused
  - Can be static (during life of system, during life of process)
  - Or dynamic (changed by process as needed) – **domain switching**, **privilege escalation**
- **Separation of duty**
  - Divide steps in a process so that no single individual can subvert a process.
- **Support for** fine-grained, course-grained specifications

Principles of Protection (Cont.)

- Must consider “grain” aspect
  - Rough-grained privilege management easier, simpler, but least privilege now done in large chunks
    - For example, traditional Unix processes either have abilities of the associated user, or of root
  - Fine-grained management more complex, more overhead, but more protective
    - File ACL lists, RBAC
- Domain can be user, process, procedure

Domain Structure

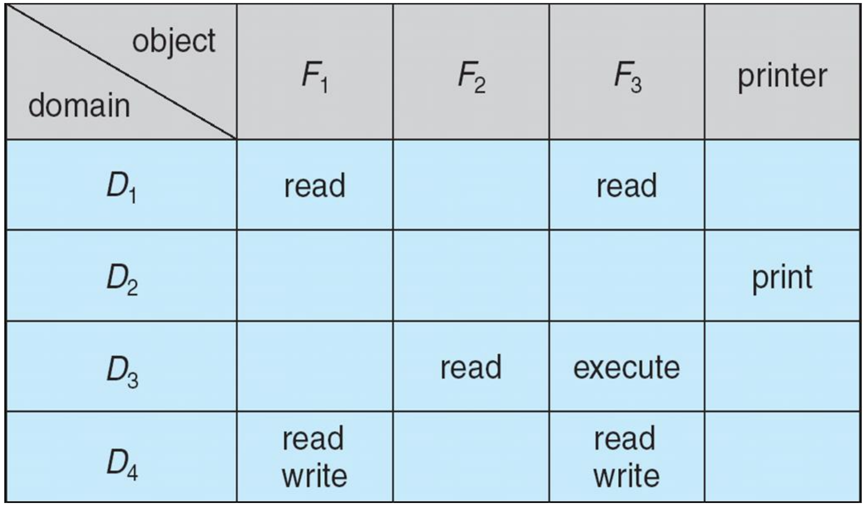
- Access-right = <*object-name*, *rights-set*>
  where *rights-set* is a subset of all valid operations that can be performed on the object
- Domain = set of access-rights



Domain Implementation (UNIX)

- Domain = user-id
- Domain switch accomplished via file system
  - Each file has associated with it a domain bit (setuid bit)
  - When file is executed and setuid = on, then user-id is set to owner of the file being executed
  - When execution completes user-id is reset
- Domain switch accomplished via passwords
  - `su` command temporarily switches to another user's domain when other domain's password provided
- Domain switching via commands
  - `sudo` command prefix executes specified command in another domain (if original domain has privilege or password given)

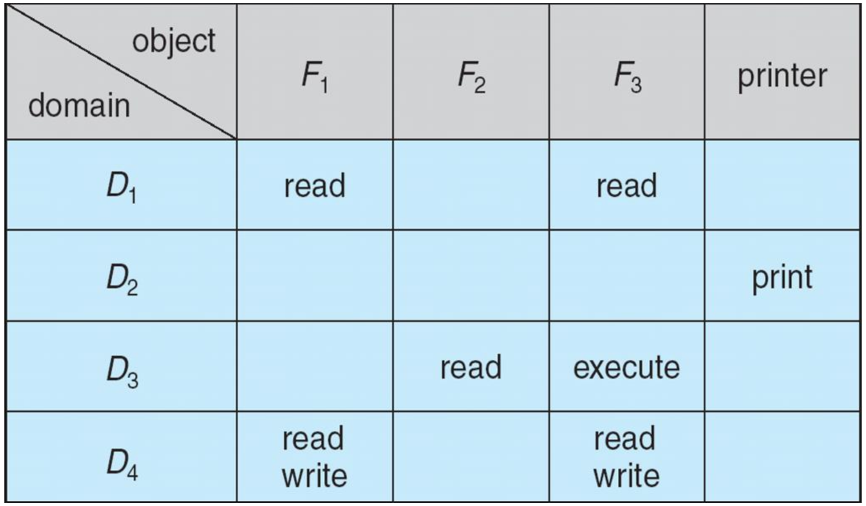
Access Matrix



object domain	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

- View protection as a matrix **(access matrix)**
- Rows represent domains
- Columns represent objects
- `Access(i,j)`
  - the set of operations that a process executing in Domain$_i$ can invoke on Object$_j$

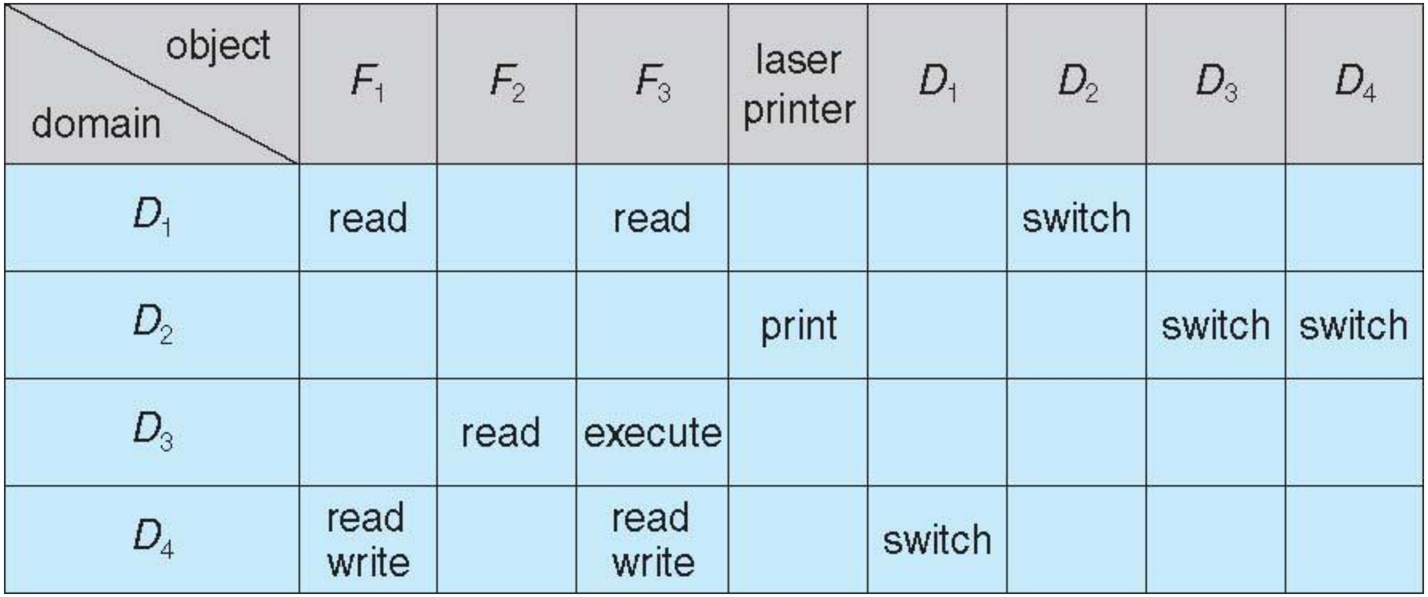
Access Matrix



object domain	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

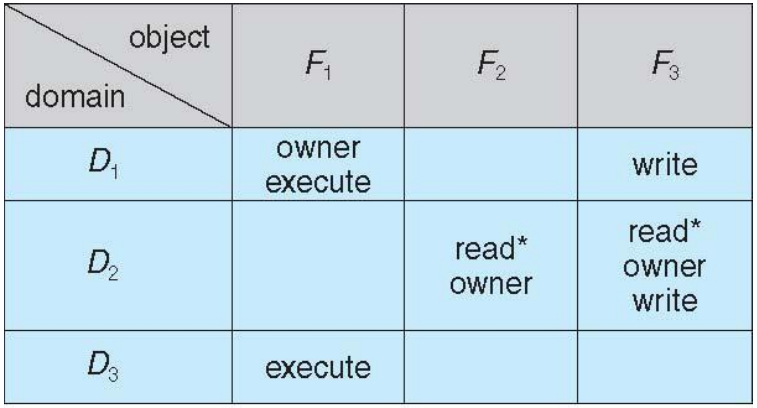
- If a process in Domain D_i tries to do “op” on object O_j, then “op” must be in the access matrix
- User who creates object can define access column for that object
- Can be expanded to dynamic protection
  - Operations to add, delete access rights
  - Special access rights:
    - *owner of O_i*
    - *copy op from O_i to O_j (“*”)*
    - *control – D_i can modify D_j access rights*
    - *transfer – switch from domain D_i to D_j*

Access Matrix of Figure A with Domains as Objects



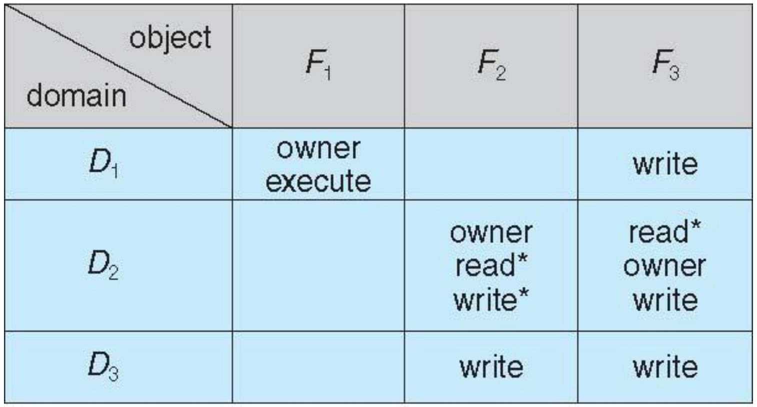
object domain	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			

Access Matrix With *Copy/Owner* Rights



object domain	F_1	F_2	F_3
D_1	owner execute		write
D_2		read* owner	read* owner write
D_3	execute		

(a)



object domain	F_1	F_2	F_3
D_1	owner execute		write
D_2		owner read* write*	read* owner write
D_3		write	write

(b)

Use of Access Matrix (Cont.)

- **Access matrix** design separates *mechanism* from *policy*
  - Mechanism
    - Operating system provides access-matrix + rules
    - If ensures that the matrix is only manipulated by authorized agents and that rules are strictly enforced
  - Policy
    - User dictates policy
    - Who can access what object and in what mode

Implementation of Access Matrix

- Generally, a sparse matrix
- Option 1 – Global table
  - Store ordered triples `<domain, object, rights-set>` in table
  - A requested operation M on object O_j within domain D_i -> search table for < D_i, O_j, R_k >
    - with M ∈ R_k
  - But table could be large -> *will not fit in main memory*
  - Difficult to group objects (consider an object that all domains can read)

Implementation of Access Matrix (Cont.)

- Option 2 – Access lists for objects
  - Each column implemented as an access list for one object
  - Resulting per-object list consists of ordered pairs

    `<domain, rights-set>`

    defining all domains with non-empty set of access rights for the object
  - Easily extended to contain default set
    -> If $M \in$ default set, also allow access

Implementation of Access Matrix (Cont.)

- Option 3 – Capability list for domains
  - Instead of object-based, list is domain based
  - **Capability list** for domain is list of objects together with operations allows on them
  - Object represented by its name or address, called a **capability**
  - Execute operation M on object O_j, process requests operation and specifies capability as parameter
    - Possession of capability means access is allowed
  - Capability list associated with domain but never directly accessible by domain
    - Rather, protected object, maintained by OS and accessed indirectly
    - Like a "secure pointer"
    - Idea can be extended up to applications

Implementation of Access Matrix (Cont.)

- Each column = Access-control list for one object
  Defines who can perform what operation

  Domain 1 = Read, Write
  Domain 2 = Read
  Domain 3 = Read

- Each Row = Capability List (like a key)
  For each domain, what operations allowed on what objects

  Object F1 – Read

  Object F4 – Read, Write, Execute

  Object F5 – Read, Write, Delete, Copy

Comparison of Implementations

- Many trade-offs to consider
  - Global table is simple, but can be large
  - Access lists correspond to needs of users
    - Determining set of access rights for domain non-localized so difficult
    - Every access to an object must be checked
      - Many objects and access rights -> slow
  - Capability lists useful for localizing information for a given process
    - But revocation capabilities can be inefficient

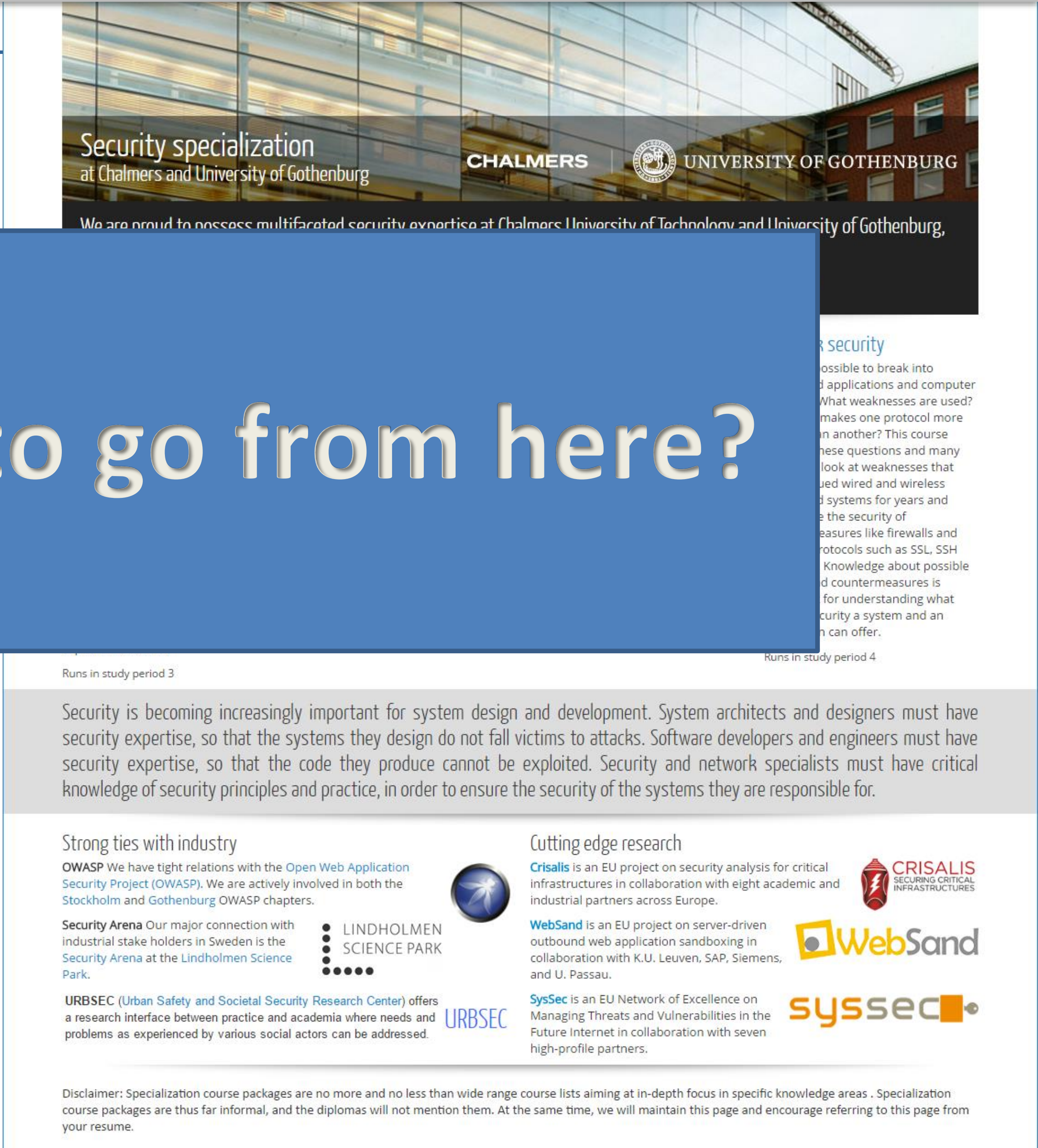
Comparison of Implementations (Cont.)

- Most systems use combination of access lists and capabilities
  - First access to an object -> access list searched
    - If allowed, capability created and attached to process
      - Additional accesses need not be checked
    - After last access, capability destroyed
    - Consider file system with ACLs per file

http://www.cse.chalmers.se/edu/master/secspec/

Courses
Ch

1. C
   S
2. C
3. I
   Security
4. Network security
5. ICT Support for Adaptiveness and (Cyber)security in the Smart Grid



Security specialization
at Chalmers and University of Gothenburg

CHALMERS

UNIVERSITY OF GOTHENBURG

We are proud to possess multifaceted security expertise at Chalmers University of Technology and University of Gothenburg,

security

ossible to break into
d applications and computer
What weaknesses are used?
makes one protocol more
n another? This course
hese questions and many
look at weaknesses that
ued wired and wireless
d systems for years and
e the security of
easures like firewalls and
rotocols such as SSL, SSH
Knowledge about possible
d countermeasures is
for understanding what
curity a system and an
n can offer.

Runs in study period 3

Runs in study period 4

Security is becoming increasingly important for system design and development. System architects and designers must have security expertise, so that the systems they design do not fall victims to attacks. Software developers and engineers must have security expertise, so that the code they produce cannot be exploited. Security and network specialists must have critical knowledge of security principles and practice, in order to ensure the security of the systems they are responsible for.

Strong ties with industry

OWASP We have tight relations with the Open Web Application Security Project (OWASP). We are actively involved in both the Stockholm and Gothenburg OWASP chapters.

Security Arena Our major connection with industrial stake holders in Sweden is the Security Arena at the Lindholmen Science Park.

LINDHOLMEN SCIENCE PARK

URBSEC (Urban Safety and Societal Security Research Center) offers a research interface between practice and academia where needs and problems as experienced by various social actors can be addressed.

URBSEC

Cutting edge research

Crisalis is an EU project on security analysis for critical infrastructures in collaboration with eight academic and industrial partners across Europe.

CRISALIS SECURING CRITICAL INFRASTRUCTURES

WebSand is an EU project on server-driven outbound web application sandboxing in collaboration with K.U. Leuven, SAP, Siemens, and U. Passau.

WebSand

SysSec is an EU Network of Excellence on Managing Threats and Vulnerabilities in the Future Internet in collaboration with seven high-profile partners.

syssec

Disclaimer: Specialization course packages are no more and no less than wide range course lists aiming at in-depth focus in specific knowledge areas . Specialization course packages are thus far informal, and the diplomas will not mention them. At the same time, we will maintain this page and encourage referring to this page from your resume.

Where to go from here?