Software & Hardware Security

Erik Poll

Digital Security group

Radboud University

Nijmegen

The Netherlands

Nijmegen

Digital Security group

Rigorous & formal methods to design & analyse secure ICT systems

Incl. societal impact, esp. on privacy

Also looking at concrete applications

DigiD

software security

attacks

- buffer overflows in C(++)
- web problems:

  SQL inj, XSS, CSRF,..

defenses

- security testing
- static analysis

  for Java & C

online privacy &
cybercrime

hardware security

- smartcards & RFID

- attacks
- bank cards

- e-passport

The problem

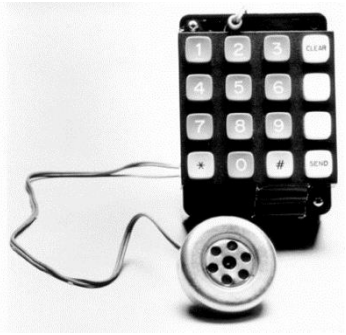
pre-history of hacking

In 1950s, Joe Engressia showed the telephone network could be hacked by phone phreaking:

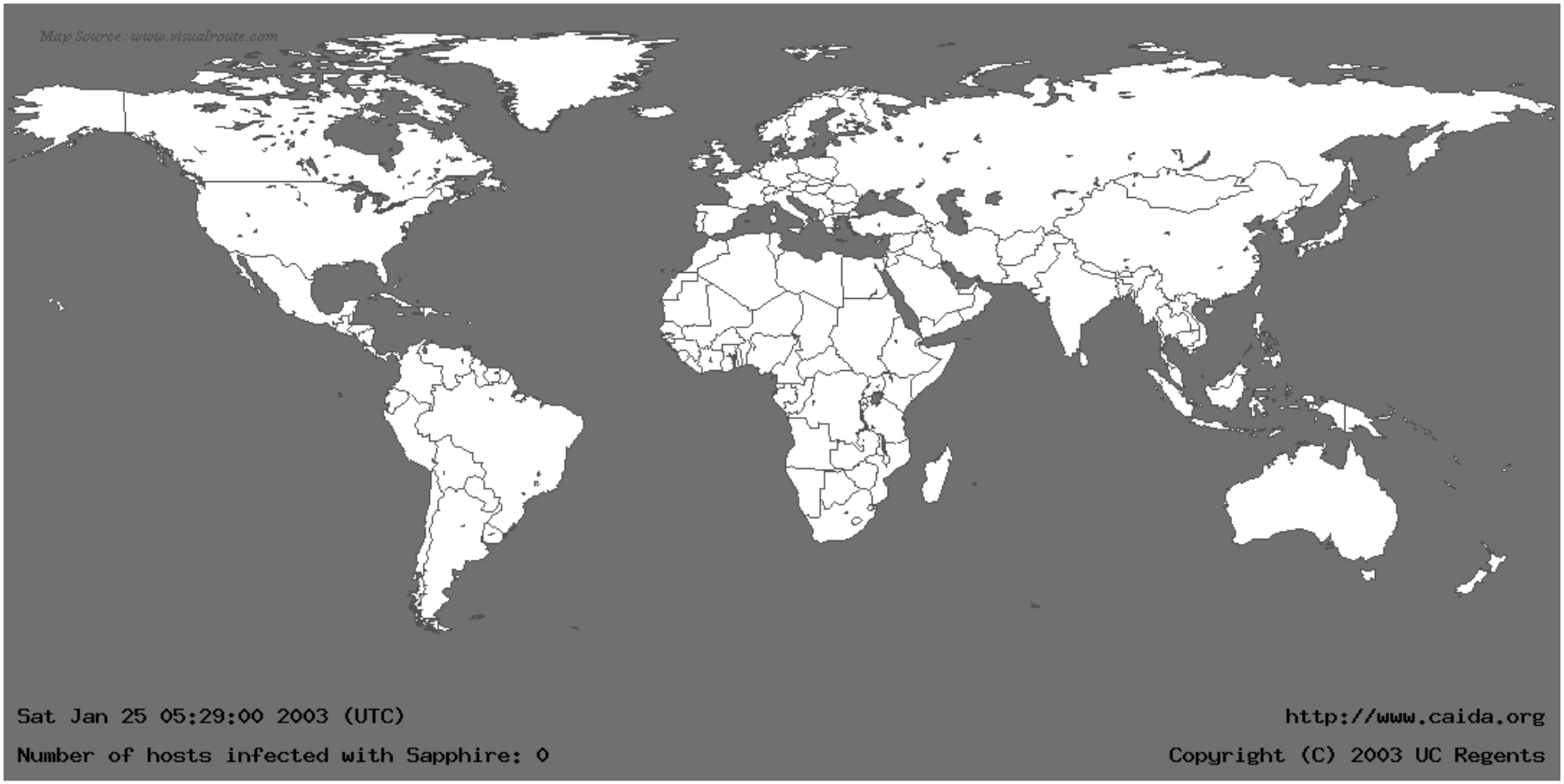
ie. whistling at right frequencies

http://www.youtube.com/watch?v=vVZm7I1CTBs

In 1970s, before founding Apple together with Steve Jobs, Steve Wozniak sold Blue Boxes for phone phreaking at university

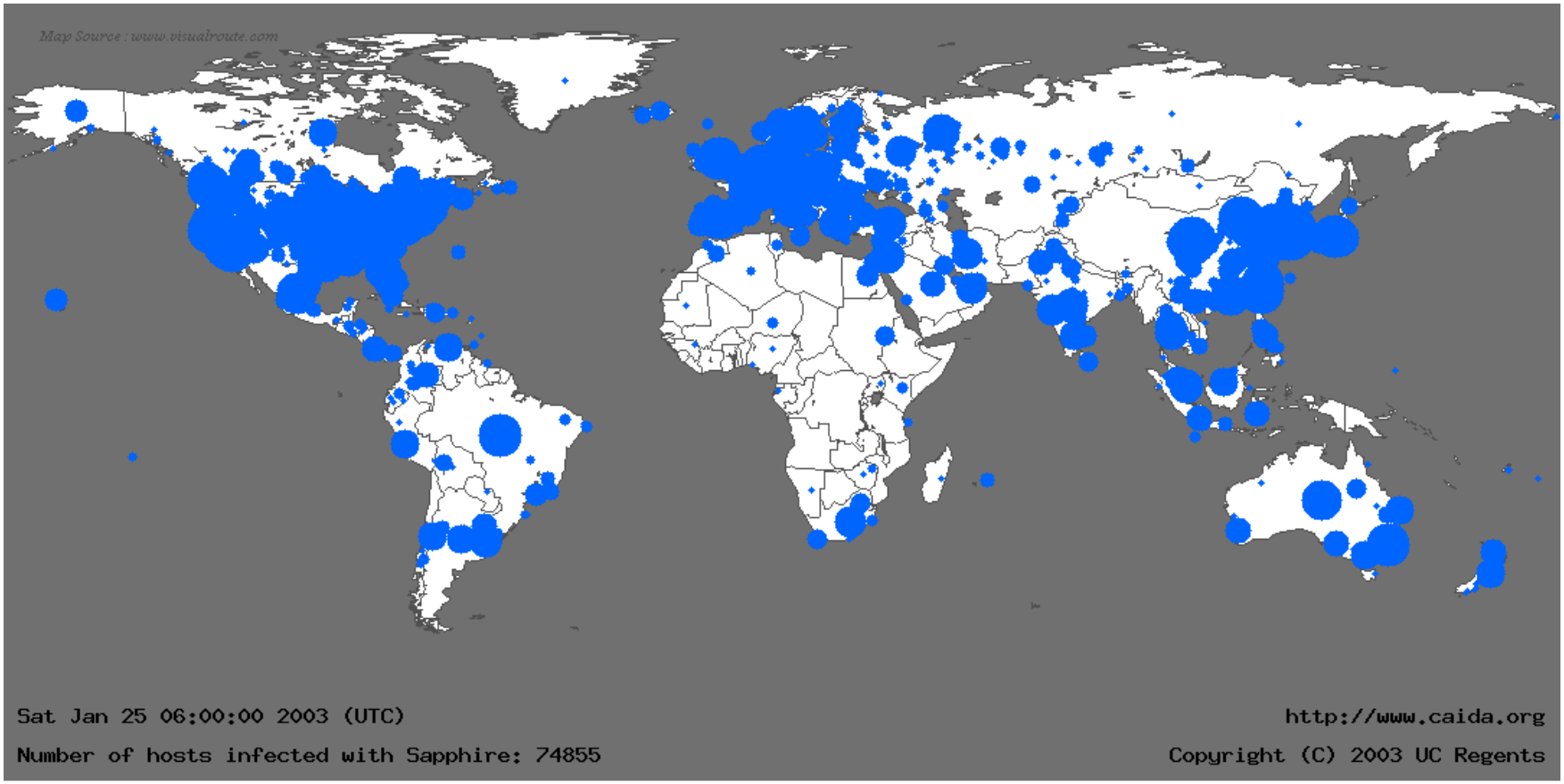


Slammer Worm (2003)



Pictures taken from *The Spread of the Sapphire/Slammer Worm*, by David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, Nicholas Weaver

Slammer Worm (2003)



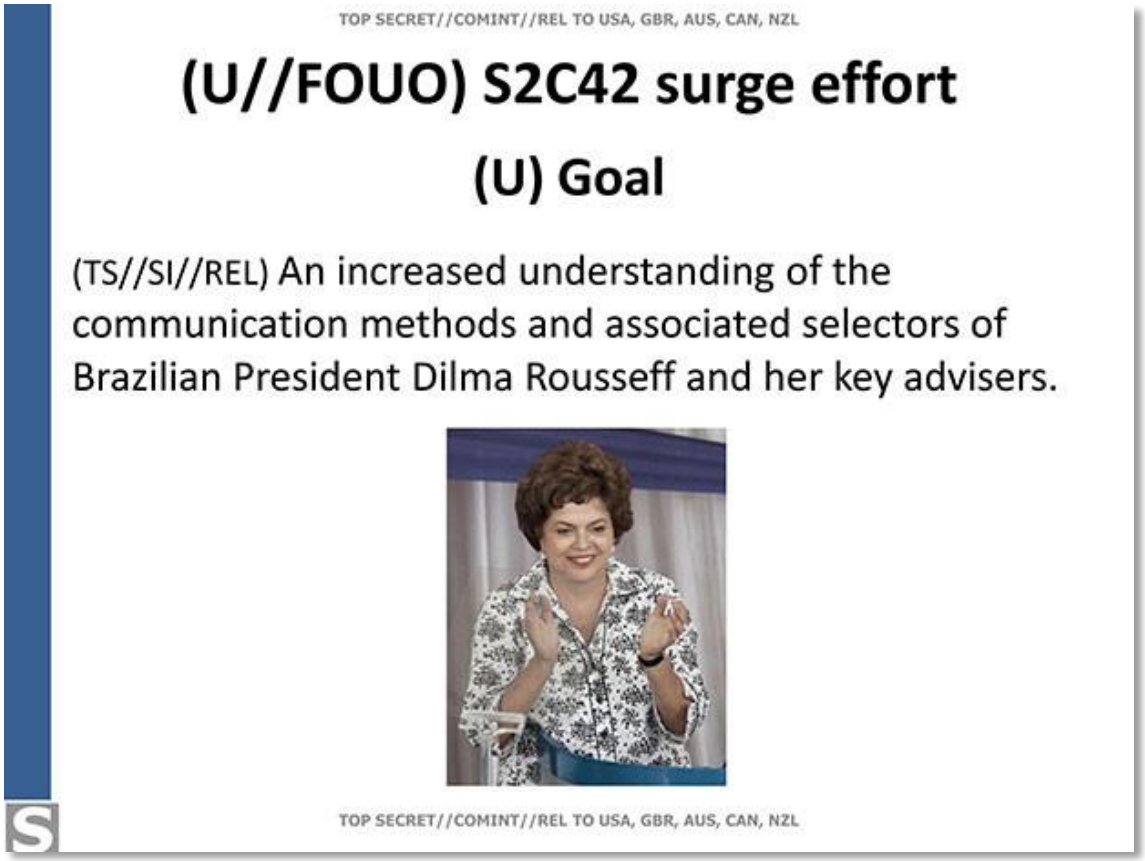
Pictures taken from *The Spread of the Sapphire/Slammer Worm*, by David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, Nicholas Weaver

The Washington Post | PostTV | Politics | Opinions | Local | Sports | National | World

The Americas

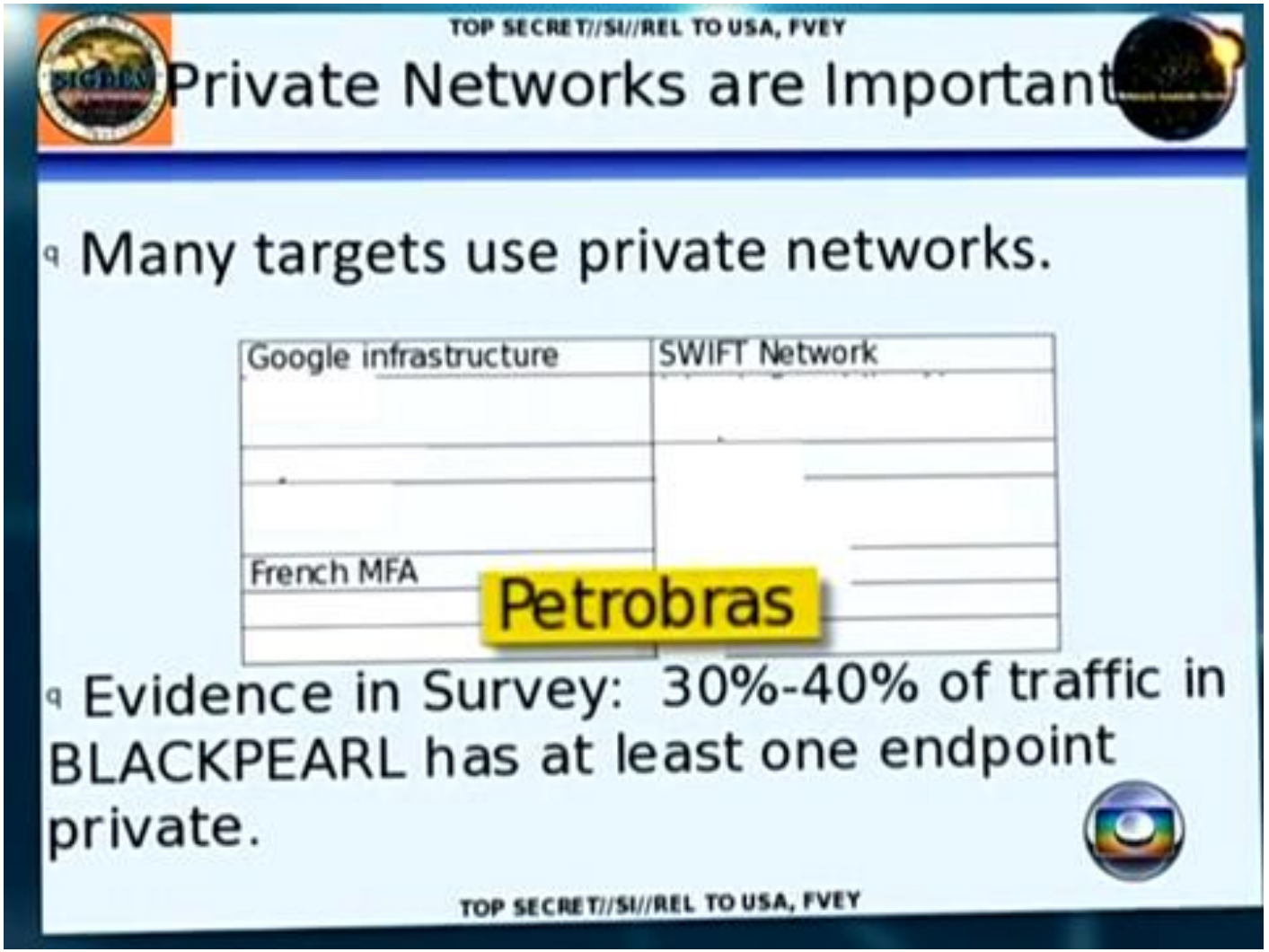
US-Brazil tensions flaring after report that NSA program targeted Brazil's president

'Tensões entre EUA-Brasil se incendeiam depois de reportagem sobre programa da NSA que teve presidente do Brasil como alvo', diz título da matéria do "Washington Post" Reprodução / The Washington Post



Top secret NSA slides leaked by Edward Snowden

More info at http:// leaksource.info and
http:// www.theguardian.com/us-news/the-nsa-files


TOP SECRET//SI//REL TO USA, FVEY
Private Networks are Important
Many targets use private networks.
Google infrastructure
SWIFT Network
French MFA
Petrobras
Evidence in Survey: 30%-40% of traffic in BLACKPEARL has at least one endpoint private.
TOP SECRET//SI//REL TO USA, FVEY

Notícias
Baboo
Bahia
Brasil
Brasil Escola
Carros
Cidades
Dino
Distrito Federal
Downloads
Duelos
Economia
Contábeis
Notícias
Fotos
Glossário
Enquetes
Mural
Vídeos
Educação
R7 Coursera
Eleições 2014
Empregos

Economia

Imposto de Renda 2014 | Greve dos bancários | 20 anos do Real | Empreende

3/12/2014 às 01h28 (Atualizado em 3/12/2014 às 21h10)

Apesar de erro, compradores de passagens baratas da KLM têm direito à viagem

Segundo Procon, artigos do Código de Defesa do Consumidor permitem embarque dos clientes

Alexandre Garcia, do R7

Os consumidores que **adquiriram passagens aéreas da KLM a preços promocionais** para a Europa na última segunda-feira (1º) têm direito a viajar, de acordo com o Procon. Segundo o órgão, os artigos 31 e 35 do CDC (Código de Defesa do Consumidor) dão margem para que o cliente embarque nos voos.

Enquanto o artigo 31 do CDC informa que toda empresa ofertante de determinado produto ou serviço deve honrar com o proposto, o de número 35 reconhece que o consumidor lesado com a situação pode exigir o cumprimento da obrigação, aceitar outro serviço equivalente no lugar ou receber o valor do pagamento de volta.

Security problems of past days…

To get an impression of the scale of the problem, have a look at

http://www.securityfocus.com/vulnerabilities

http://www.us-cert.gov/ncas/alerts
http://www.us-cert.gov/ncas/bulletins

http://www.securitytracker.com/

Quiz

What do laptops, tablets, mobile phones, wifi access points, network routers, bank cards, e-passports, eID cards, smartphone apps, web sites, web browsers, web servers, operating systems, firewalls, intrusion detection systems, cars, and airplanes have in common?

Why can all these things be hacked, if we are not very careful?

There is <u>SOFTWARE</u> inside them!

Software (in)security

- Software is the main source of security problems.
  - Software is *the* weakest link in the security chain, with the possible exception of “the human factor”

- Software security does (did?) not get much attention
  - in other security courses, or
  - in programming courses,

  or indeed, in much of the security literature!

Computer security courses traditionally focus on cryptography

“if you think your problem can be solved by cryptography,
then you do not understand cryptography
and you do not understand your problem”

[Bruce Schneier]

Superficial analysis of the problem

Observation 1

All these problems are due to *(bad) software*

Namely software in

- the Linux/Windows/Mac operating system (OS)
- web servers
- web browsers
- the router software
- ...

Because of these software bugs constant patching of system is needed to keep them secure

Observation 2

All these problems are due to bad software that

- can be executed/addressed over the network
  - eg. in case of Slammer worm
- executes on (untrusted) input obtained over the network

or both

With ever more network connectivity,

ever more software can be attacked.

Changing target of attacks

- Traditionally, focus of attacks was on operating system and network
  "Solutions"
  - regular patching of OS
  - firewalls
  - virus scanners
- Increasingly, focus on
  - web applications
  - web browser
  - mobile devices
    - smartphones, tablet, that pass through firewalls
  - embedded software
    - software in cars, factories, infrastructure...

  and *targetted* attacks on specific organisation or person
  (known as ATP = Advanced Persistent Threat)

Changing nature of attackers

Traditionally, hackers were amateurs motivated by fun

- publishing attacks for fame & glory
- attacks creating lots of publicity

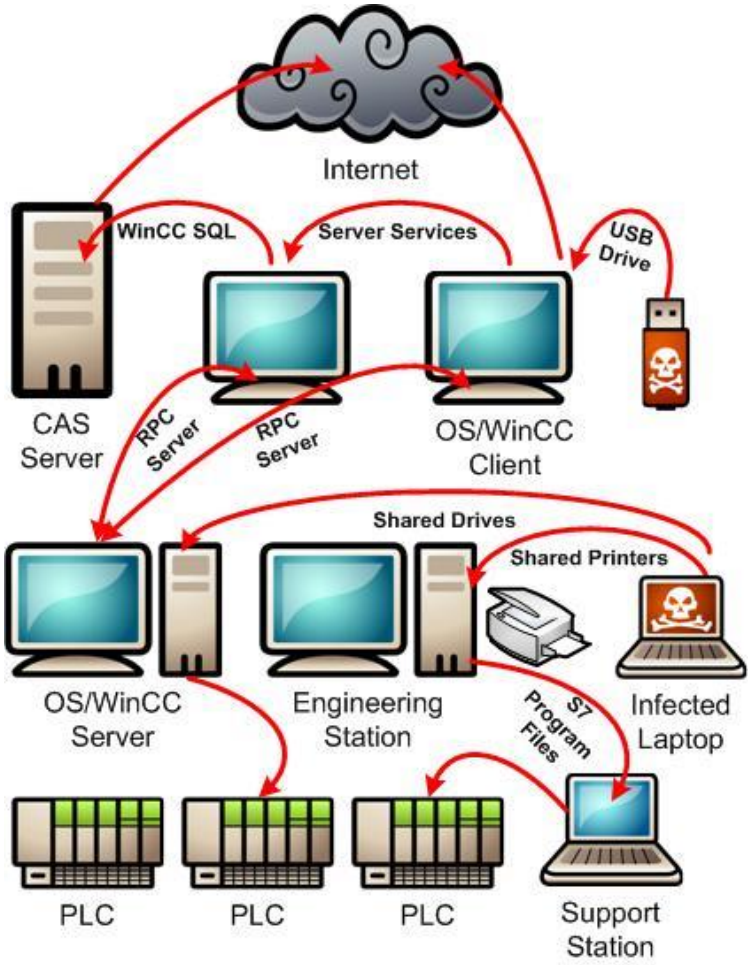
Increasingly, hackers are professional

- attackers go underground
  - zero-day exploits are worth a lot of money

Attackers increasingly include

- organized crime
  with lots of money and (hired) expertise
- government agencies:
  with even more money & in-house expertise

stuxnet attack



Malware (by US and Israel?) attacking nuclear enrichment facility in Iran

http://www.ted.com/talks/ralph_langner_cracking_stuxnet_a_21st_century_cyberweapon.html

Software (in)security: crucial facts

- No silver bullets!

  crypto or special security features do not magically solve all problems

- Security is emergent property of entire system
  - just like quality

- (Non-functional) security aspects should be integral part of the design, right from the start

We focus on software security now, but don't forget

that security is about

people (users, employees, sys-admins, programmers,...), and

their laziness, mistakes, stupidity, incompetence, confusion,

software, bugs, verification, hackers, viruses, testing,

operating systems, networks, databases, hardware,

access control, passwords, smartcards, biometrics, cryptology,

security protocols, security policies & their enforcement,

monitoring, auditing, risk management, *complexity*,

legislation, persecution, liability, public relations

public perception, conventions, standards, …..

The causes of the problem

Quick audience poll

- How many of you learned to program in C or C++?
- How many had it as a first programming language?
- How many of your C(++) courses
  - warned you about buffer overflows?
  - explained how to avoid them?

Major causes of problems are

- lack of awareness
- lack of knowledge
- irresponsible teaching of dangerous programming languages

Quick audience poll

- How many of you have built a web-application?
  - in which programming languages?
- What is the *secure* way of doing a SQL query in this language? (to avoid SQL injection flaws)

Major causes of problems are

- lack of awareness
- lack of knowledge

1. Security is always a secondary concern

- Security is always a secondary concern
  - primary goal of software is to *provide* some functionality or services;
  - *managing* associated risks is a derived/secondary concern

- There is often a trade-off/conflict between
  - security
  - functionality & convenience

  where security typically looses out
    - more examples of this later...

DOCTOR FUN

16 Jan 2006

Functionality vs security

- Functionality is about what software *should do*,
  security is (also) about what it *should not do*

Unless you think like an attacker,
you will be unaware of any potential threats

Functionality vs security: Lost battles?

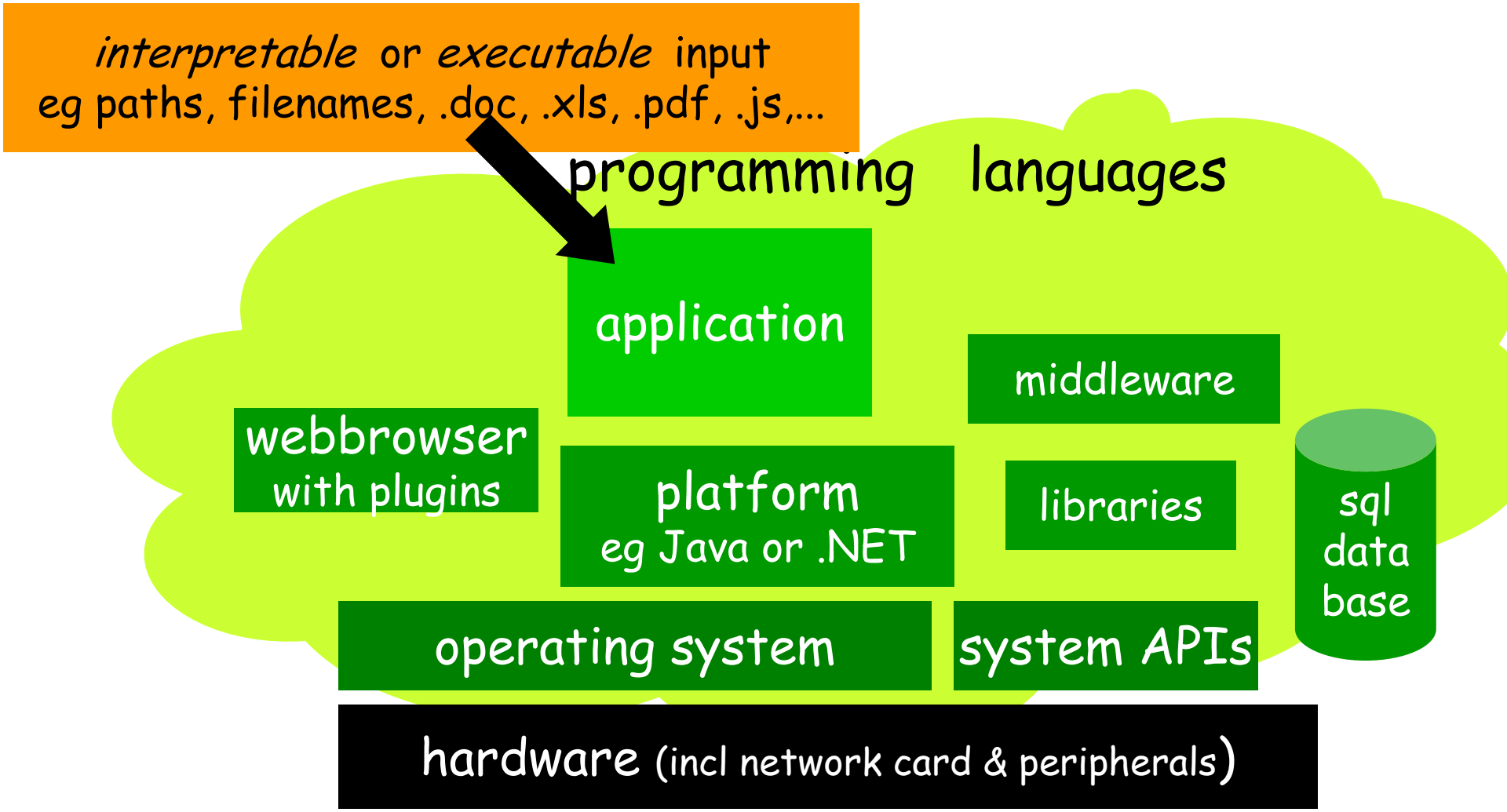
- operating systems (OSs)
  - with huge OS, with huge attack surface
- programming languages
  - with easy to use, efficient, but very insecure and error-prone mechanisms
- web browsers
  - with plug-ins for various formats, javascript, ActiveX, Ajax ...
- email clients
  - which automatically cope with all sorts of formats & attachments..

Functionality vs security : PHP

"After writing PHP forum software for three years now, I've come to the conclusion that it is basically impossible for normal programmers to write secure PHP code. It takes far too much effort. .... PHP's raison d'etre is that it is simple to pick up and make it do something useful. There needs to be a major push ... to make it safe for the likely level of programmers - newbies. Newbies have zero chance of writing secure software unless their language is safe. ... "

[Source http://www.greebo.cnet/?p=320]

2. Weakness in depth



2. Weakness in depth

Software

- runs on a huge, complicated infrastructure
  - OS, platforms, webbrowser, lots of libraries & APIs, ...
- is built using complicated languages & formats
  - programming languages, but also SQL, HTML, XML, ...
- using various tools
  - compilers, IDEs, preprocessors, dynamic code downloads

These may have security holes, or may make the introduction of security holes very easy & likely

Recap

Problems are due to

- lack of awareness
  - of threats, but also of what should be protected
- lack of knowledge
  - of potential security problems, but also of solutions
- compounded by complexity
  - software written in complicated languages, using large APIs , and running on huge infrastructure
- people choosing functionality over security

Security concepts & goals

Security

- Security is about regulating access to assets
  - assets can be *information, functionality*, or *physical assets*
  -
- Software provides functionality
  - eg on-line exam results
- This functionality comes with certain risks
  - eg what are risks of on-line exam results?
- (Software) security is about managing these risks

Starting point for ensuring security

- Any discussion of security should start with an inventory of
  - the stakeholders – ie. who is involved
  - their assets, and
  - the threats to these assets

  by possible attackers
  - employees, clients, script kiddies, criminals

Any discussion of security without understanding these issues is meaningless:

You have to know **what** you want to secure,

against **what type of attacks**, and against **who**

Security concepts

Goal of security is to reduce risks to *acceptable* levels,

- Security is never 100%

So you have to know **what** you want to secure,
against **what type of attacks**, against **who,**
and **at what cost**

Security Objectives: CIA

- Confidentiality
  - unauthorised users cannot *read* information
- Integrity
  - unauthorised users cannot *alter* information
- Availability
  - authorised users *can* access information
  - ie. preventing DoS (Denial of Service) attacks

- Non-repudiation or accountability
  - authorised users *cannot deny* actions

Security objectives

- Integrity nearly always more important than confidentiality

  Eg think of

  - your bank account information
  - your medical records
  - *all* the software you use, incl. the entire OS

How to realise security objectives? AAAA

- Authentication
  - who are you?
- Access control/Authorisation
  - control who is allowed to do what
  - this requires a specification of who is allowed to do what
- Auditing
  - check if anything went wrong
- Action
  - if so, take action

How to realise security objectives?

Other names for the last three A's

- **Prevention**
  - measures to stop breaches of security goals
- **Detection**
  - measures to detect breaches of security goals
- **Reaction**
  - measures to recover assets, repair damage, and persecute (and deter) offenders

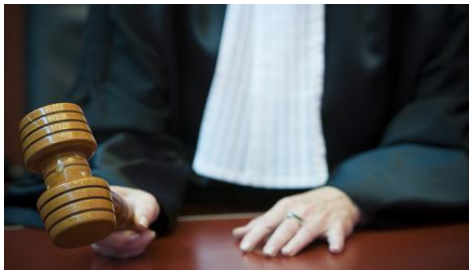
Try to prevent, *but also* detect and react

Never think that good prevention
makes detection & reaction superfluous.

Eg. breaking into house or office is often easy;
only detection & reaction seriously deters burglars.

Detection of digital break-in is harder
who noticed a break-in on his computer recently?

Reaction (incl. prosecution) is even harder
how to find the person responsible,
somewhere on the internet?



Software security

warning: confusing terminology

Common use of terminology can be very confused & confusing:
(security) weakness, flaw, vulnerability, bug, error, coding defect...

We can make a distinction between

- a security weakness/flaw:
  something that is wrong or could be better
- a security vulnerability
  a weakness/flaw that can actually be exploited by an attacker,
  which requires the flaw to be
  - accessible: attacker has to be able to get at it
  - exploitable: attacker has to be able to do some damage with it

Eg by unplugging your network connection,
some (many?) vulnerabilities become flaws.

software vulnerabilities

Software vulnerabilities can be introduced at two “levels”

- design flaws
  vulnerability in the design
- bugs aka implementation flaws or code-level defects

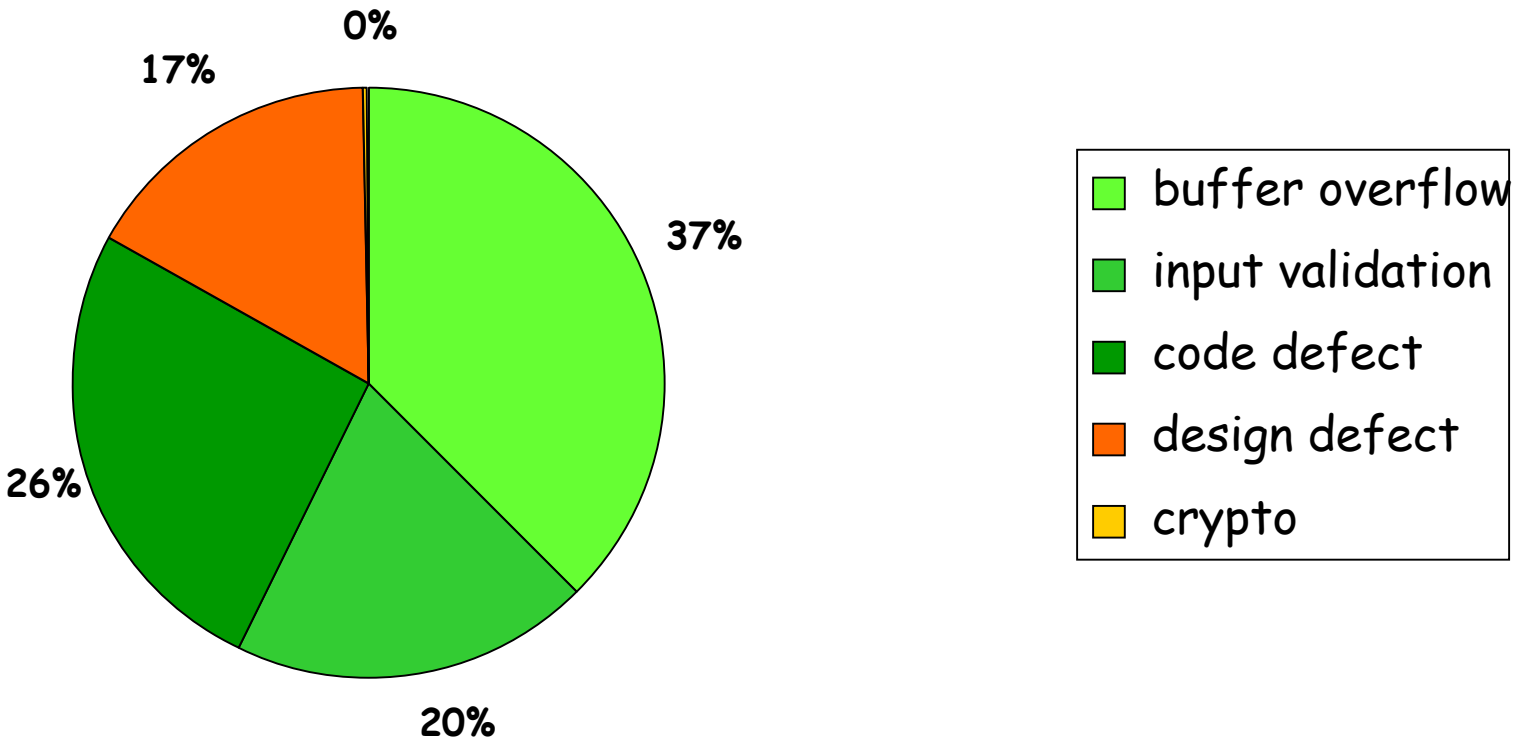
  vulnerability in the software introduced when implementing a system

Rough consensus: bugs and design flaws are equally common

Vulnerabilities also arise on other levels (out of scope for now)

- configuration flaw when installing software on a machine
- the user
- unforeseen consequence of the *intended* functionality (eg. spam)

Typical software security vulnerabilities



Security bugs found in Microsoft bug fix month (2002)

bugs aka implementation flaws aka code-level defects

There are roughly two kinds of implementation flaws

1. bugs that can be understood looking at the program itself (and understanding what it is meant to do!)
   - eg. , simple typos, confusing two program variables, off-by-one error in array access, ...
   - sometimes called logic errors, as opposed to syntax errors,

     or an errors in the program logic
2. lower-level problems that can only be spotted if you understand the underlying platform of the program in execution, eg
   - buffer overflow,integer overflow,... in binaries compiled from C(++)
   - SQL injection, XSS, CSRF,.... in web-applications

The big problem of software security

The *bad* news

people keep making the same (types of) mistakes

The *good* news

people keep making the same (types of) mistakes

…… so we can do something about it!

“Every advantage has its disadvantage ” -- Johan Cruijff

security in the software development life cycle

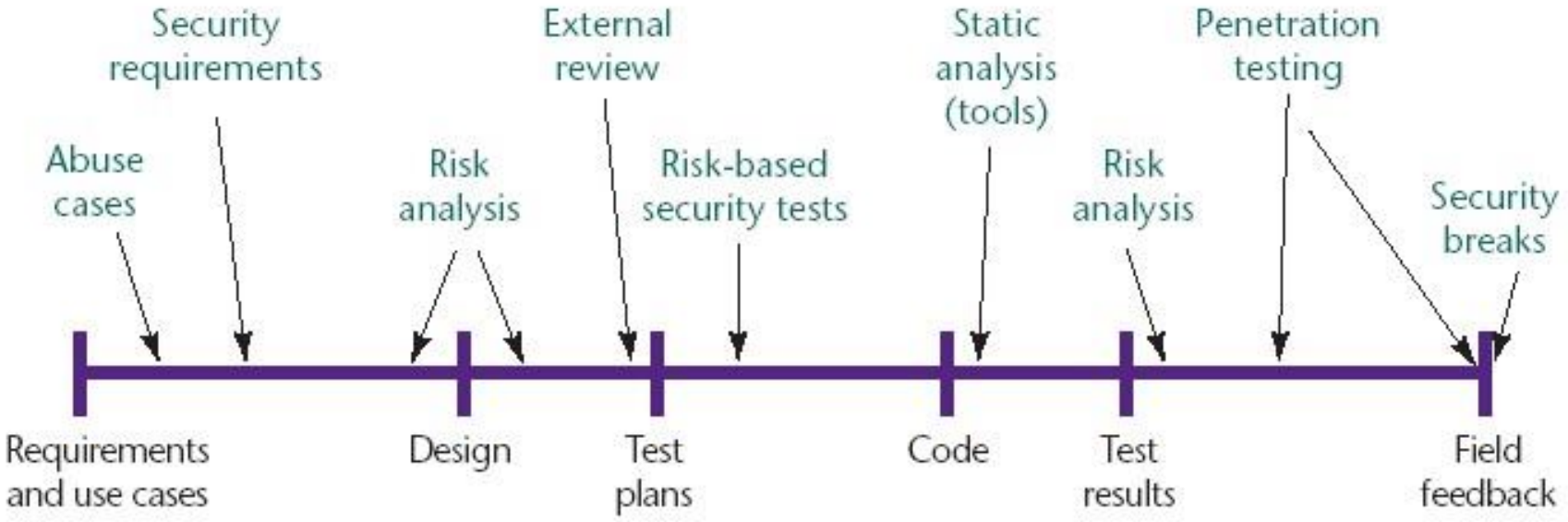
Tackling Software Insecurity

- Knowledge about standard mistakes is crucial in preventing them
  - these depends on the programming language, the "platform" (OS, database systems, web-application framework,…), and the type of application
  - lots of info available on this now

- But this is not enough: security to be taken into account from the start, throughout software development life cycle
  - several ideas & methodologies to do this

Security in Software Development Life Cycle

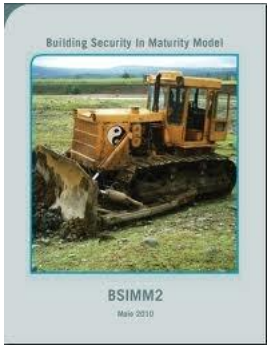
McGraw's Touchpoints



[Gary McGraw, *Software security*, Security & Privacy Magazine, IEEE, Vol 2, No. 2, pp. 80-83, 2004.]

Methodologies for security in development life cycle

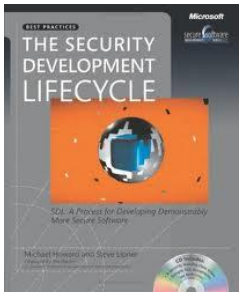
Common/best practices, with methods for assessments, and roadmaps for improvement



- McGraw's Touchpoints

  BSIMM Building Security In – Maturity Model

  http://bsimm.com



- Microsoft SDL Security Development Lifecycle

- OpenSAMM Software Assurance Maturity Model

  http://opensamm.org

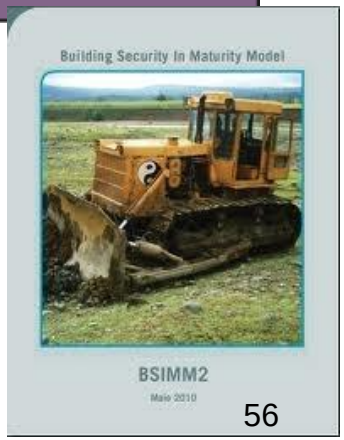
OPENSAMM

Microsoft's SDL Optimisation Model

BSIMM

Governance	Intelligence	SSDL Touchpoints	Deployment
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Manage-ment

Based on data collected from large enterprises



Spot the (security) flaws in electronic_purse.c

```
int balance;

void decrease(int amount)
 { if (balance <= amount)
         { balance = balance - amount; }
   else { printf("Insufficient funds\n"); }
 }

void increase(int amount)
 { balance = balance + amount;
 }
```

<= should be >=

what if amount is negative?

what if this sum is too large for an `int`?

Different kinds of implementation flaws

what if amount is negative?

- lack of input validation of (untrusted) user input
  - could be a design flaw rather than an implementation flaw?
  - more “fundamental” than the flaws below

<= should be >=

- simple mistake in the program logic

what if this sum is too large for an `int`?

- potential problem depending on how the underlying platform work, eg. in case of an integer overflow;
  - “lower level” than the flaws above

More info

- Gary McGraw,

  Software security,

  Security & Privacy Magazine, IEEE, Vol 2, No. 2, pp. 80-83, 2004.

- Check out websites

  http://www.us-cert.gov/ncas/alerts/

  http://www.us-cert.gov/ncas/bulletins/

  http://www.securitytracker.com/

  http://www.securityfocus.com/vulnerabilities

  for security alerts in the past week