
Spezielle Kapitel aus Betriebssysteme: Secure Code - LVA 353.013 Part 2

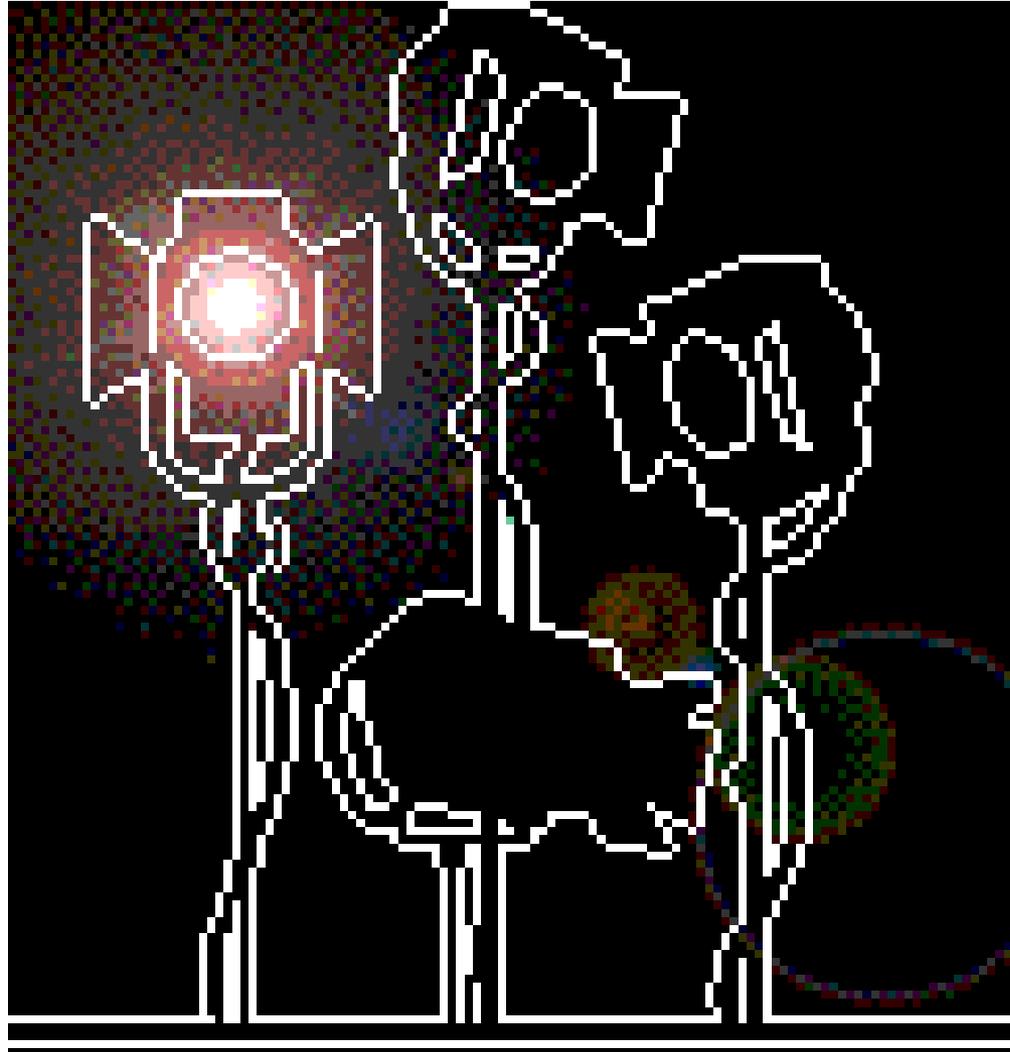
secure: [si-'kyur]

1: free from danger

2: free from risk of loss

3: affording safety

What happens till now?



Secure Code - LVA 353.013

Security is one of the top issues in today's IT

CERT/CC Statistics 1988-2006 - Microsoft Internet Explorer

Adresse <http://www.cert.org/stats/>

Vulnerabilities reported

1995-1999

Year	1995	1996	1997	1998	1999
Vulnerabilities	171	345	311	262	417

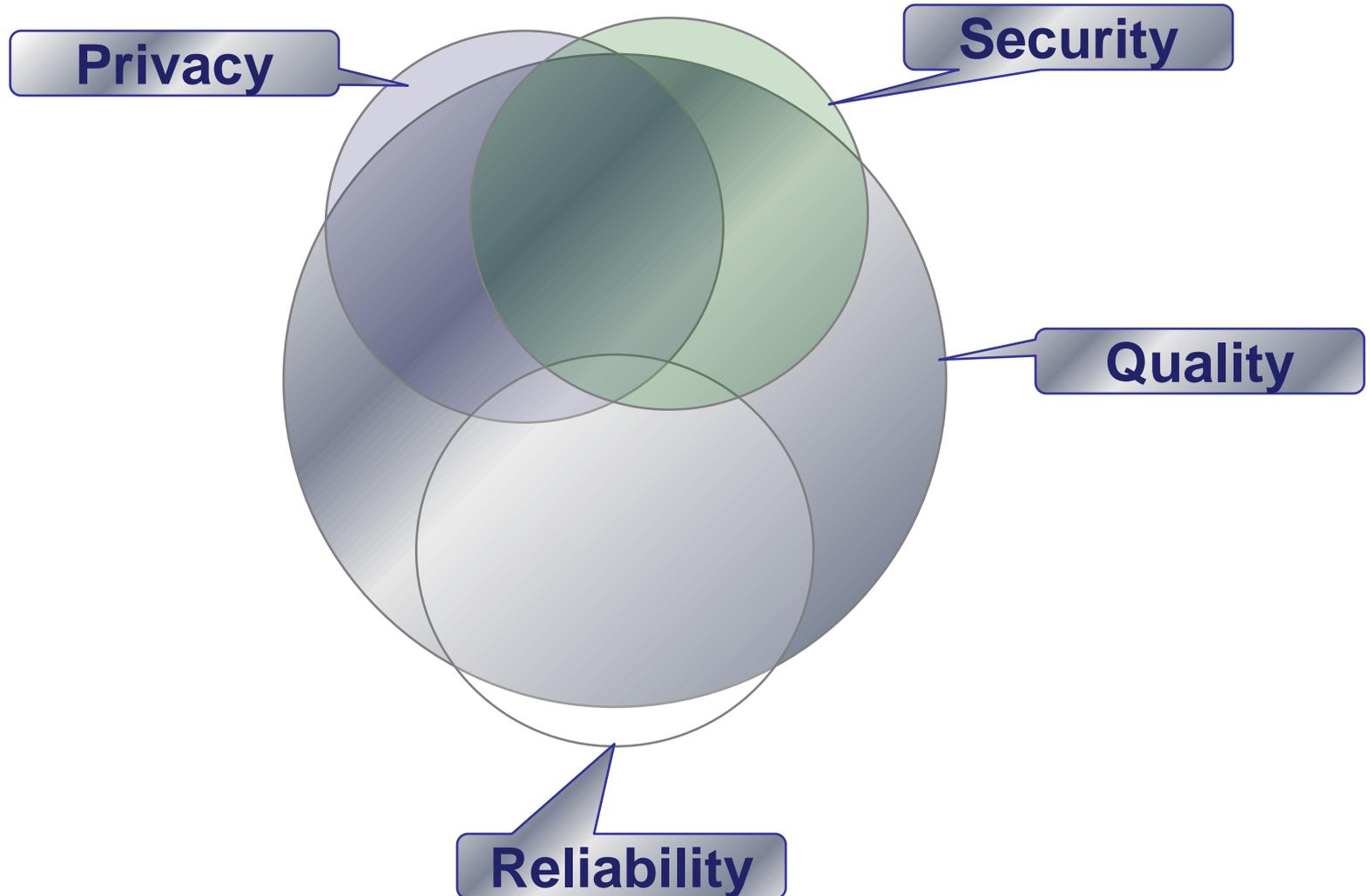
2000-2006

Year	2000	2001	2002	2003	2004	2005	Q1-Q2,2006
Vulnerabilities	1,090	2,437	4,129	3,784	3,780	5,990	3,997

Total vulnerabilities reported (1995-Q2,2006): **26,713**

Secure Code - LVA 353.013

It's Really about Quality



Ridiculous Excuses We've Heard

- Excuse: No one will do that!
- Excuse: Why would anyone do that?
- Excuse: We've never been attacked
- Excuse: We're secure - we use crypto
- Excuse: We're secure - we use ACL
- Excuse: We're secure - we use a firewall
- Excuse: We've reviewed the code

Ridiculous Excuses We've Heard

- Excuse: We know it's the default...
- Excuse: If we don't run as admin
- Excuse: But we'll slip the schedule
- Excuse: It's not exploitable!
- Excuse: But that's the way we've always done it
- Excuse: If only we had better tool ...

Assignment - Sample 1

```
#include <iostream>
void SomeFunction() {
    int someLocalVar = 17;
    int someOtherLocalVar = 33;
}

void SomeOtherFunction() {
    int someLocalVar;
    int someOtherLocalVar;

    std::cout << "someLocalVar: " << someLocalVar << std::endl;
    std::cout << "someOtherLocalVar: " << someOtherLocalVar <<
        std::endl;
}

void main(void) {
    SomeFunction();
    SomeOtherFunction();
}
```

Assignment - Sample 2

```
#include <iostream>

void foo() {
    ...
}

void main(void) {
    int i = 0;

    foo();
    i++;
    std::cout << "i = " << i << std::endl;
}
```

The Attacker's Advantage ...

- The defender must defend all points; the attacker can choose the weakest point.
- The defender can defend only against known attacks; the attacker can probe for unknown vulnerabilities.
- The defender must be constantly vigilant; the attacker can strike at will.

- Technology alone will not solve your problem
- Nobody believes anything bad can happen to them, until it does
- Security works only if the secure way also happens to be the easiest way
- In you do not keep up with security fixes, your network will not be yours for long
- There really is someone out there trying compromise your systems
- Your data and systems are of value to someone
- Security is not about risk elimination; it is about risk management

Security is only as good
as its weakest link

Security Features \neq Secure Features

Why Security Vulnerabilities Occur

Security Professionals Don't Know the Applications

"As a Network Security Professional, I don't know how my company's applications are supposed to work so I deploy a protective solution...but don't know if it's protecting what it's supposed to."

The Application Security Gap



Application Developers and QA Professionals Don't Know Security

"As an Application Developer, I can build great features and functions while meeting deadlines, but I don't know how to build security into my applications."

“Since human beings themselves are not fully debugged yet, there will be bugs in your code no matter what you do”

Chris Mason

Why Writing Secure Code is a Challenge

- Reasons developers give for not building secure applications
 - Security is boring
 - Security is often seen as a functionality disablement – gets in the way
 - Security is difficult to measure
 - Don't know how

Why Writing Secure Code is a Challenge

- Security vulnerabilities are expensive to fix
 - Coordination, finding bug, fixing the code, testers, and PR
 - Cost of lost productivity
 - Cost of lost trust of consumers

- All software has security defects!
 - Yes, EVERYONE!
- Present development models cannot deliver secure software

A Security Framework: SD³ + C

Secure by Design

- Threat modeling
- Code inspection
- Process Improvement

Secure by Default

- Unused features off by default
- Reduce attack surface area
- Least Privilege

Secure by Deployment

- Prescriptive Guidance
- Security Tools
- Training and Education

Communications

- Community Engagement
- Transparency
- Clear policy

Security Development Lifecycle



Traditional Microsoft Software Product Development Lifecycle Tasks and Processes



Readings

- (Howard and LeBlanc 2003) Howard, Michael, and David LeBlanc. *Writing Secure Code, Second Edition*. Redmond, WA: Microsoft Press, 2003.
- (Howard et. al. 2005) Howard, Michael, David LeBlanc, and JohnVega. *19 Deadly Sins of Software Security*. Emeryville, CA: mcGraw-Hill 2005.
- (Howard and Lipner 2006) Howard, Michael, and Steve Lipner. *The Security Development LIFECYCLE*. Redmond, WA: Microsoft Press, 2006.