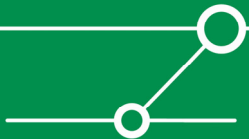


Application Security?

Dominick Baier
www.ernw.de



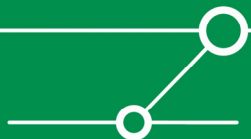
About Me



- Dipl. Ing. Informations-Technik (BA)
- Certified BS 7799 Lead Auditor

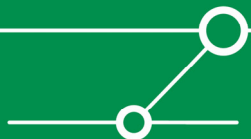
- Focus on Application Security / .NET Security
- Developer-Coaching / Training / Research
- Auditing and Penetration Testing

- Contributor to the Developmentor Security Curriculum
- Speaker (WinDev, DevWeek, ADC...)
- Upcoming Book about Penetration Testing (Vieweg Verlag Germany)



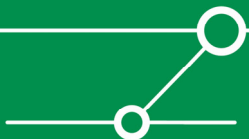
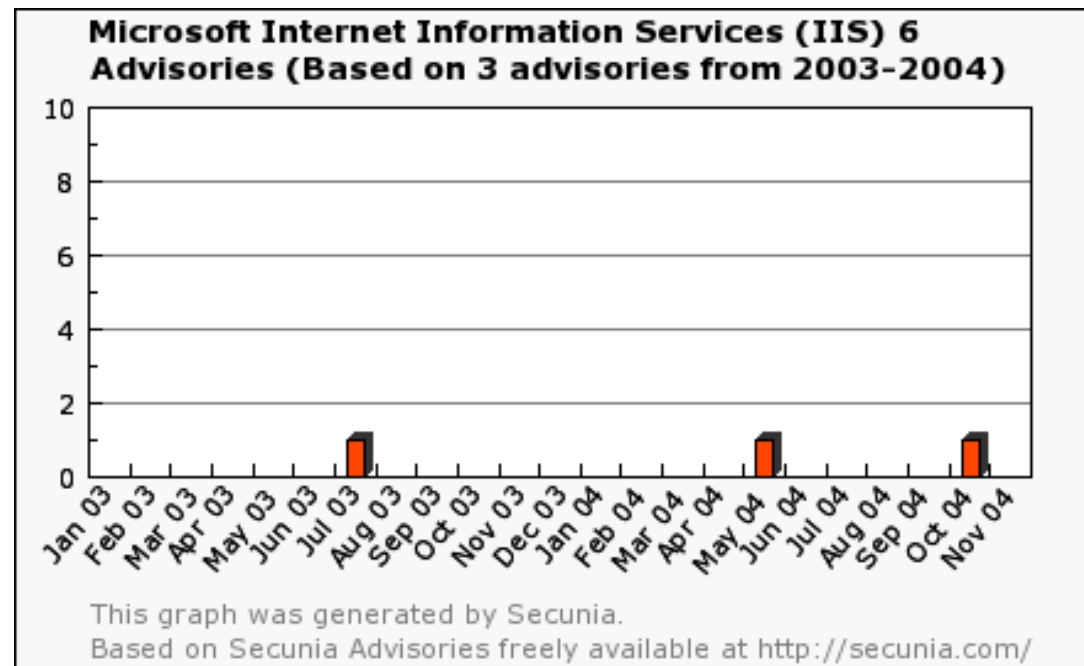
The Focus is Shifting

- The last ten years were devoted to System/Network Security
- We learned some really important lessons
- "Classic" Penetration-Test Approach
 - Buffer Overflows
 - Patch-Management
 - Attack Surface
 - Firewalls
 - Routers
 - VPNs



The Focus is Shifting

- System Services become much more secure these days



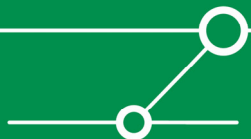
Now it's Time to Focus on Application Security!



- Applications usually deal with the most precious asset of a company
 - Information

- Applications (should) ensure
 - Who can access information
 - Which information can be accessed
 - Confidentiality and integrity of information

- Many, many other requirements for applications
 - Performance, scalability, ease of use.....



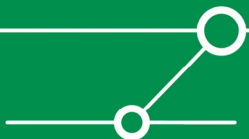
Security is a Feature



- Security is a crosscutting feature
 - Should be designed as such

- Impossible to bolt on security at the end of a project
 - Requires constant attention and iteration

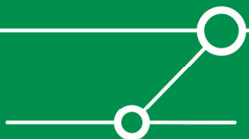
- Threat Modelling must become part of the development process
 - Are we secure?



Threat Modelling

■ STRIDE

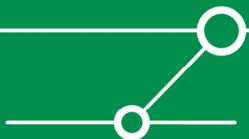
- **S**poofing Identity
- **T**ampering with Data
- **R**epudiation
- **I**nformation Disclosure
- **D**enial of Service
- **E**levation of Privilege



Spoofing Identity

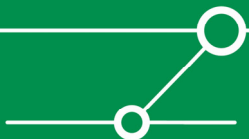
- Attacker pretends to be someone he is not
 - There are two flavors of this attack
- Spoofing client identity
 - Access a server and pretend to be a legitimate user
 - Gain access to sensitive data
- Spoofing server identity
 - Pretend to be a legitimate server to unsuspecting clients
 - Collect sensitive data from clients

- Mitigation: Strong Authentication (e.g. using Cryptography)



Tampering with Data

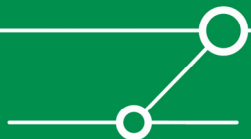
- Attackers often gain advantage by tampering with data
 - Tampering with persistent data
 - Modify password data to gain access to other user accounts
 - Change prices for products they want to buy online
 - Modify audit logs to cover their tracks
 - Tampering with network packets
-
- Mitigation: Hash Codes, Digital Signatures, Encryption



Repudiation

- Attacker denies an action, and victim cannot prove otherwise
 - An attacker might:
 - ◆ Claim he didn't delete a file
 - ◆ Claim he didn't make a purchase or return
 - ◆ Claim he didn't receive goods/services

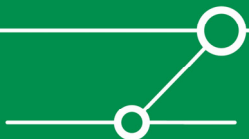
- Mitigation: Audit Logs, Receipts, Digital Signatures, Timestamps



Information Disclosure

- Attacker sees data he shouldn't be seeing
 - Local files
 - Data traveling between computers
 - Retrieving information from databases without authorization
 - Even error messages are considered information

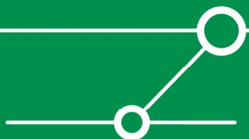
- Mitigation: Strong Authentication, Access Control, Encryption, Obscurity



Denial of Service (DoS)

- Attacker causes your application to become unavailable
 - Usually associated with services provided over a network
 - ◆ SYN flood attacks
 - ◆ Distributed Denial of Service Attacks (DDoS)
 - ◆ Race conditions
 - ◆ All designed to consume precious Bandwidth/Memory!

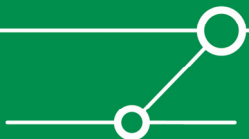
- Mitigation: Increase Availability, Reliability, and be a Good Neighbour



Elevation of Privilege

- Attacker finds a way to gain more privileges on the system
 - The ultimate goal is to gain Administrative Privileges
 - Most common exploit is (has been) the Buffer Overflow
 - Directory Traversal, Canonicalization Errors are the EOPs of today

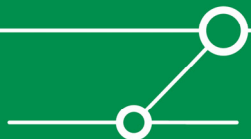
- Mitigation: Robust Code, Least Privilege, Input Validation



The Three Components of a Secure System

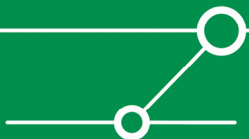


- Just as with physical security, we need all three
 - Protection
 - Detection
 - Reaction
- There are no unbreakable systems
- Detection and reaction have to be built into systems
 - Protection then becomes a way to slow down the attacker
 - Once detected, an attack can be halted by a sysadmin
 - Problems can be diagnosed and patched more quickly



A Process for Developing Secure Apps

- Security must be an integral part of the design process
 - Assets and security goals have to be verbalized a priori
 - Threats have to be determined by a formal Threat Analysis
 - Threats must be prioritized
 - ◆ Risk = (potential damage) x (likelihood of success)
 - Mitigation techniques have to be implemented
 - Detection and Reaction Mechanism have to be designed and tested
 - Security Strategy has to be revisited with each iteration!



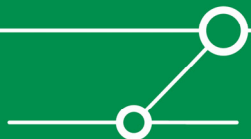
Other Principals to Live By



- Cryptography doesn't ensure security
 - How are the keys generated/transmitted/secured

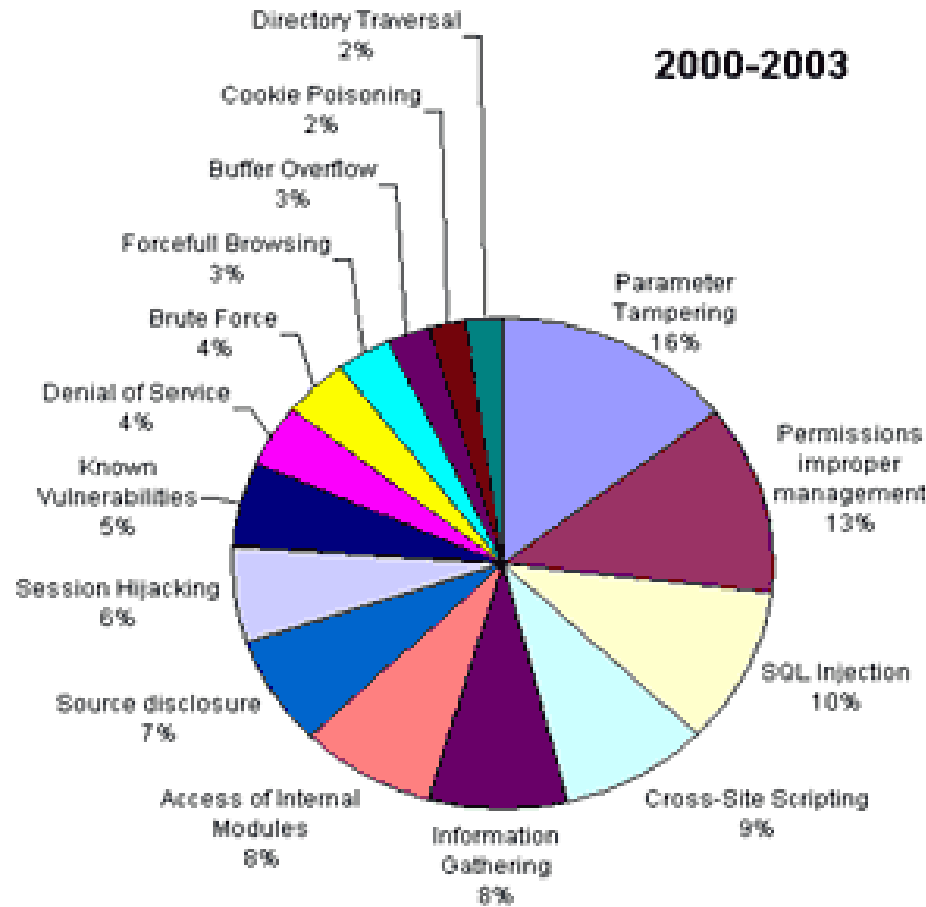
- Developers have to pay more attention to failure modes
 - At least as much time as implementing functionality

- Principle of Least Privilege
- Defense in depth



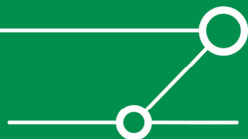
Statistics

2000-2003



| Attack Class | Hits |
|-------------------------------------|--------|
| Cross-Site scripting | 79.31% |
| SQL Injection | 61.72% |
| Parameter Tampering | 61.38% |
| Permissions improper management | 52.76% |
| Information Gathering | 36.90% |
| Cookie Poisoning | 36.21% |
| Access of Internal Modules | 34.83% |
| Known Vulnerabilities (Databases) | 32.76% |
| Known Vulnerabilities (Web Servers) | 27.4% |
| Buffer Overflow | 20.34% |

Source: Imperva Security (http://www.imperva.com/application_defense_center/papers/how_safe_is_it.html)

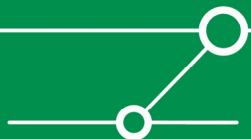


Future Consequences

- Besides being "uncool" developing insecure applications
 - Customers may lose Security Certifications (e.g. BS 7799) if using insecure apps
 - DPA (Data Protection Act)
 - CMA (Computer Misuse Act)

- ... and the legal situation is changing
 - Not yet very common in Europe, but
 - Application lawsuits will happen in 2005

 - e.g. PetCo



Future Consequences

SECURITY

Petco Settles FTC Security Charges

Nov. 18, 2004

Pet-products retailer agrees to settle charges that flaws in its Web site violated security and privacy promises made to customers.

By George V. Hulme

EMAIL THIS ARTICLE 

PRINT THIS ARTICLE 

DISCUSS THIS ARTICLE 

WRITE TO AN EDITOR 

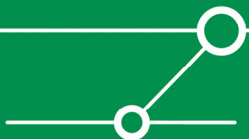
More Stories on:

[Security](#)

The flaws in question, which allow for a common type of intrusion known as a SQL injection attack, could permit hackers to access customer records, including credit-card numbers, the FTC said. The FTC also alleged that Petco didn't protect sensitive customer information it stored with adequate encryption. "As a result, a hacker was able to penetrate the Petco Web site and access credit-card numbers stored in unencrypted clear text," the FTC said in a statement.

This is the fifth time the FTC has successfully challenged deceptive claims made by businesses regarding their efforts to protect customer information. The previous cases included Eli Lilly, Guess, Microsoft, and Tower Records. Each case centered on promises the companies made in their privacy policies.

Source: <http://www.informationweek.com/story/showArticle.jhtml?articleID=53700517>)

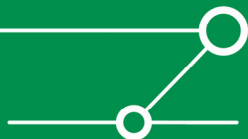


The Importance of Tool Support

- Security is hard to get right

- Black Box testing tools have been around for long (and are important)
 - @Stake WebProxy
 - SPI Dynamics WebInspect
 - Application Security Inc. AppDetective
 - Free Tools

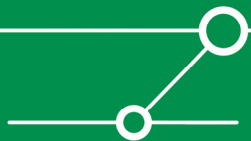
- The Application Toolkits get better, too
 - e.g. .NET 2.0 & Visual Studio Team System



The Importance of Tool Support



- The White-Box approach is getting more vital
- Assist developers while they are writing code
- Make it easier to concentrate on fault situations
- Integrated in the development cycle



Conclusion



- System and Network Security has been widely adopted
- Applications have undergone (nearly) no testing in the past

- Threat Modelling has to be done first
- Security is a Feature – should have the same priority in the Dev Lifecycle

- We need better education
- We need better tools

- We need a process

