# Web security

2007-11-21

Robert Malmgren
rom@romab.com
+46-708-330378
www.romab.com


**Robert Malmgren AB**
**Trust is good**
***control* is better**

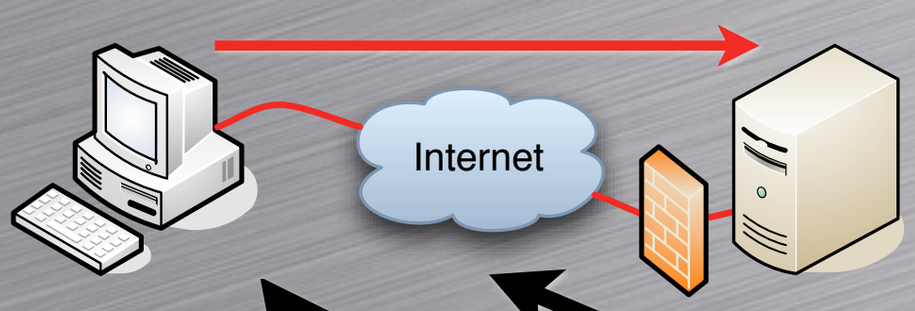
# Upplägg

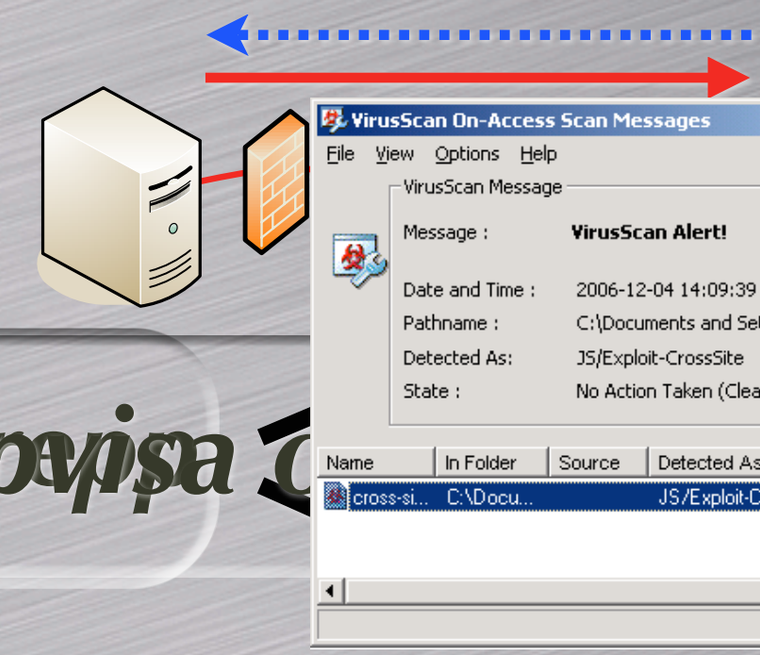
- Angreppsscenarios
- Hot och risker
- Säkerhetsinitiativ
- Något om nya teknologier
- Ofta funna dumheter
- Do's & Don'ts
- Säkerhetskultur & säkerhetsmognad
- Sammanfattning

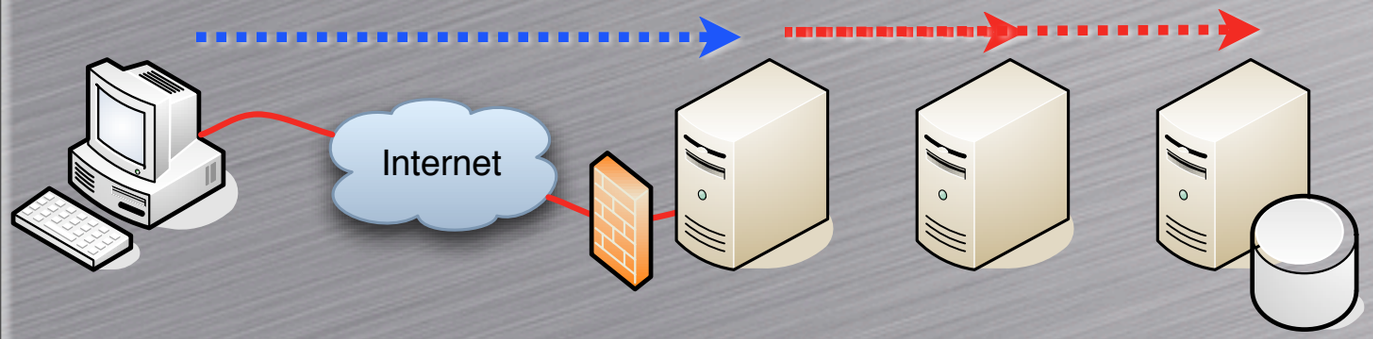
# Angreppsprinciper (1)

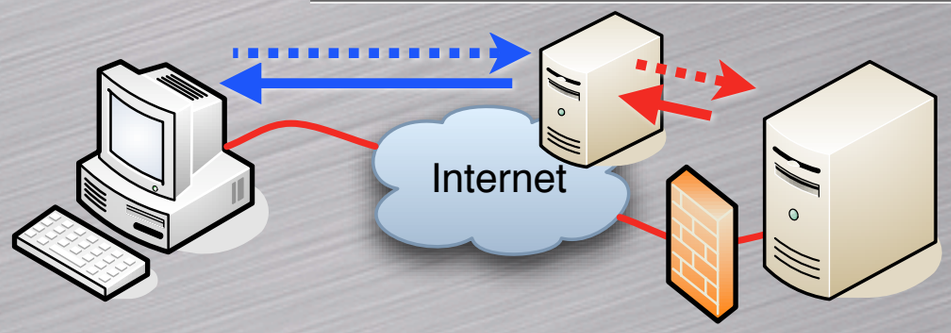
Klient/Server angrepp

Server/Klient angrepp



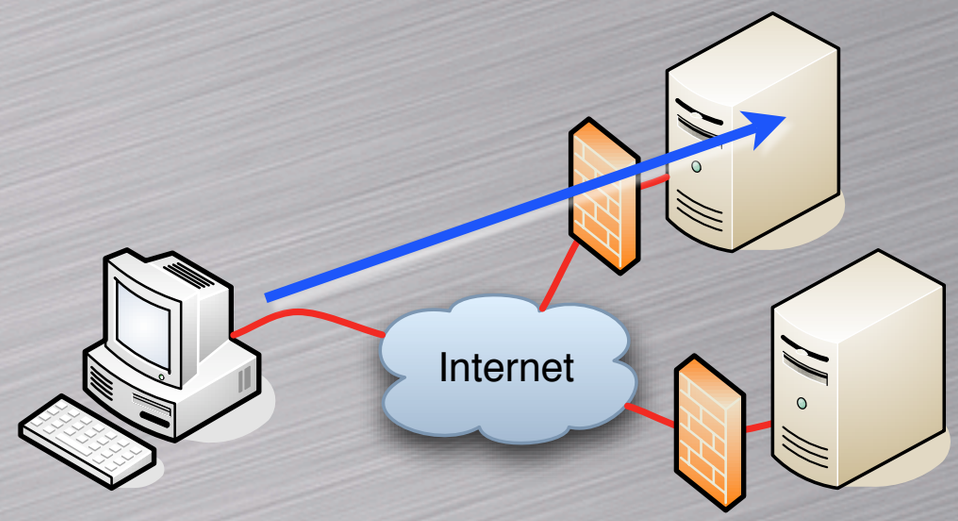



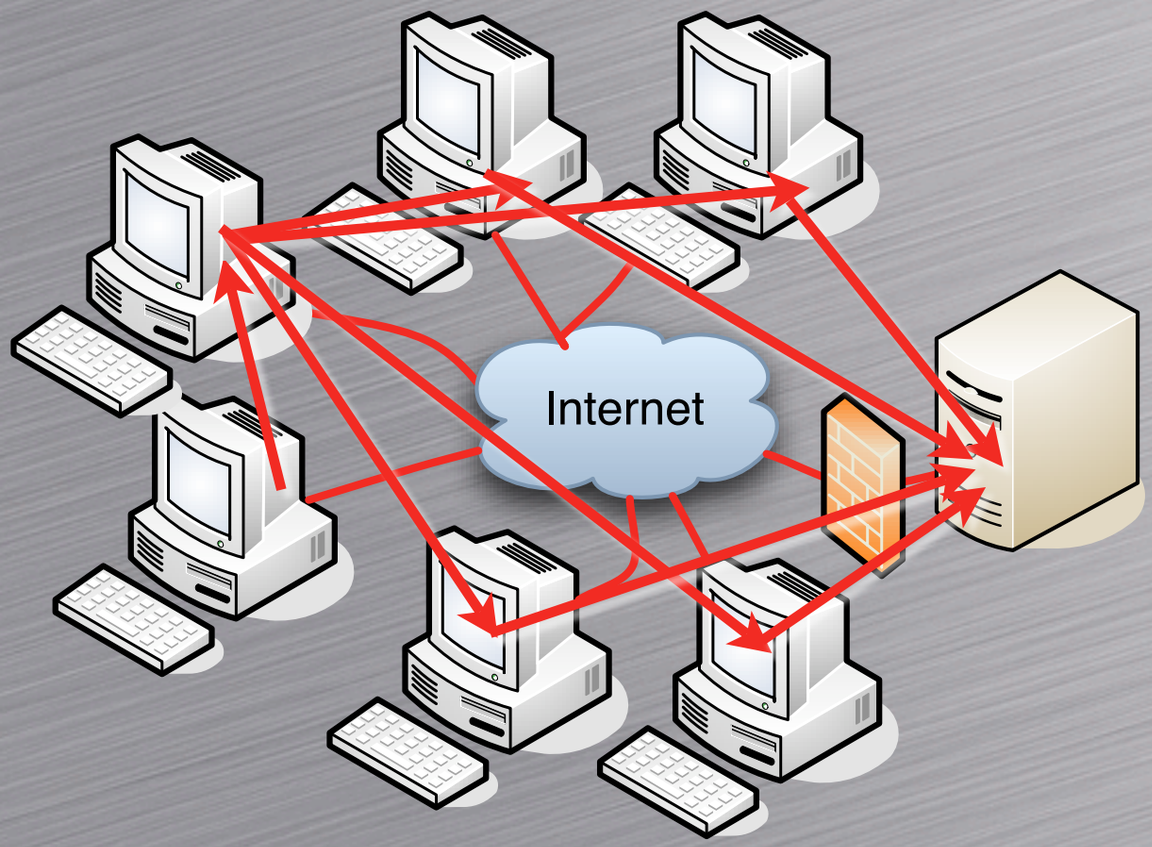



# Angreppsprinciper (2)

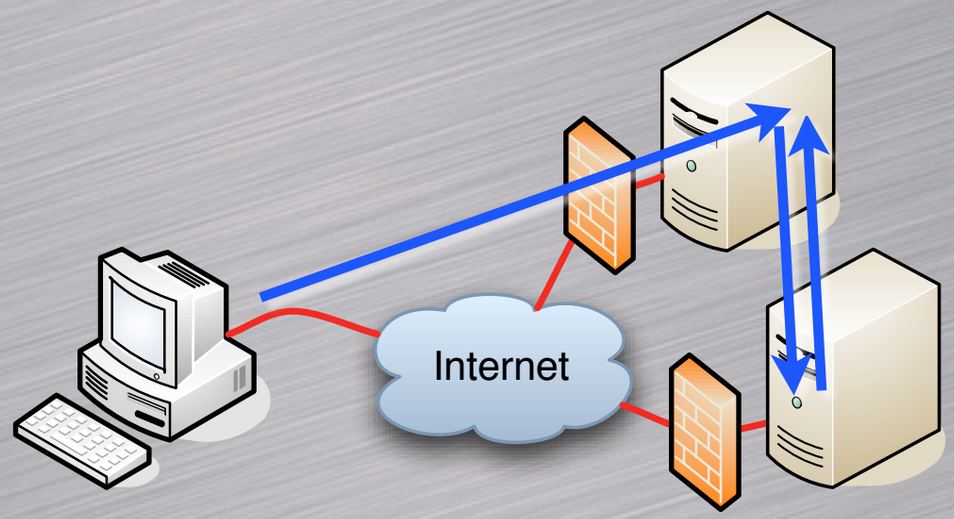
*Förtal/parodi/phising-site*



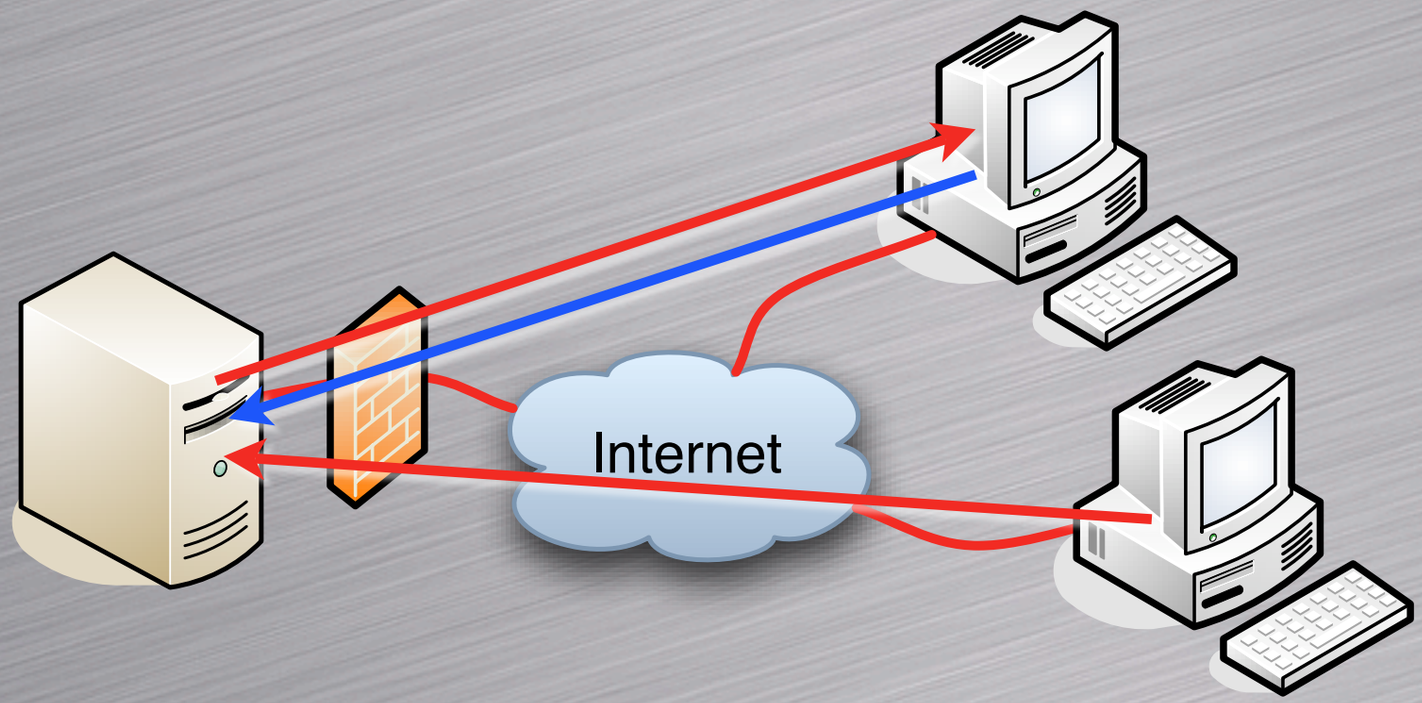

*Distribuerad DoS-attack*



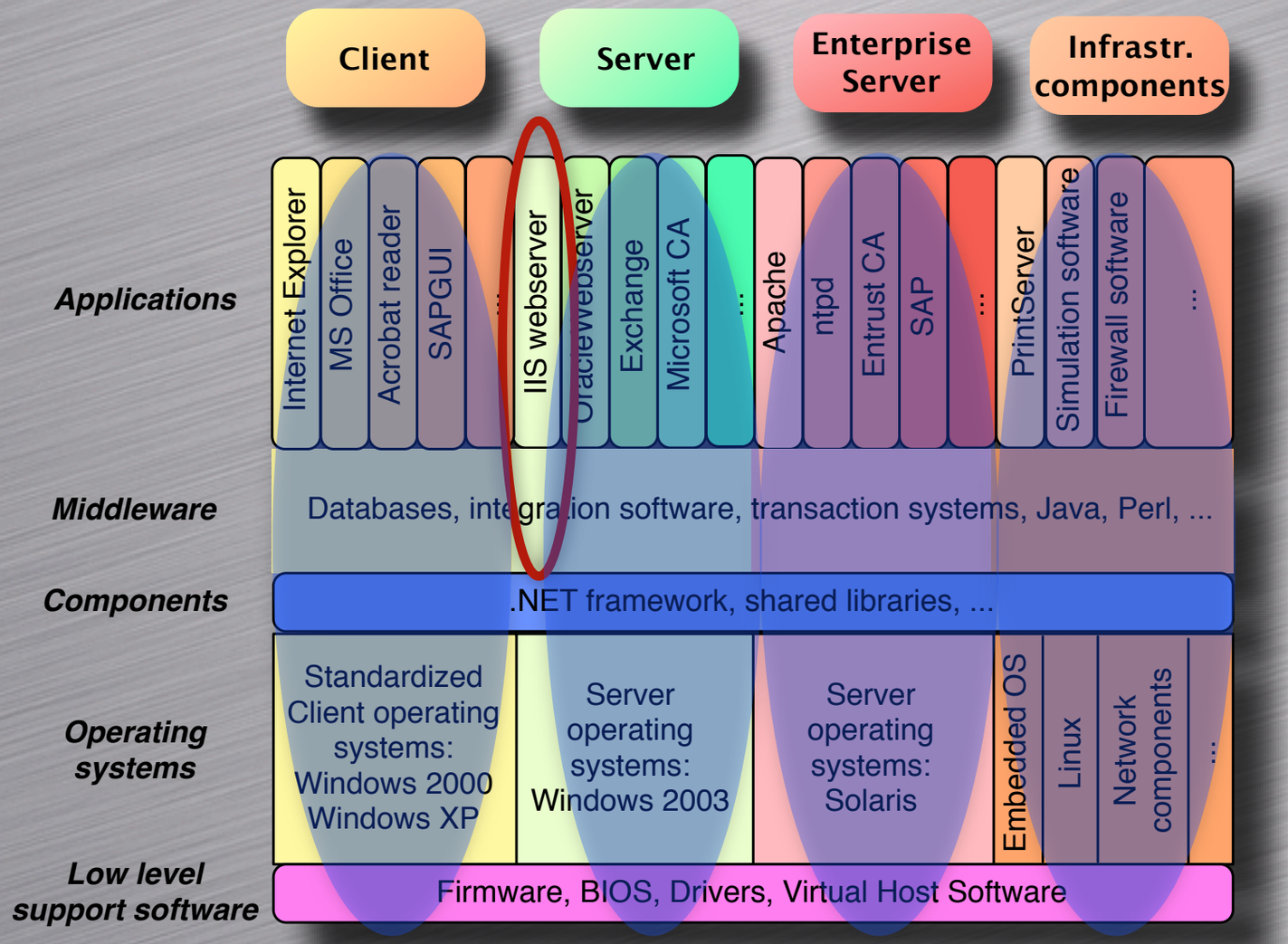

*Google hacking*



# Angreppsprinciper (3)



# Komplexitetsfaktorn



# Hot

## *Tekniska hot*

Information på webbsidan förvanskas
***defacement***, ***bedrägeri***, ***malwarespridning***

Värddatorn övertas och missbrukas av obehöriga

Omedvetna fel

Opublicerad information läcker ut *(vid fel tillfälle)*

Interna system innehållande känslig information blir tillgängliga för angrepp pgr för hård integration utan kontroll på risker och typ/nivå av exponering

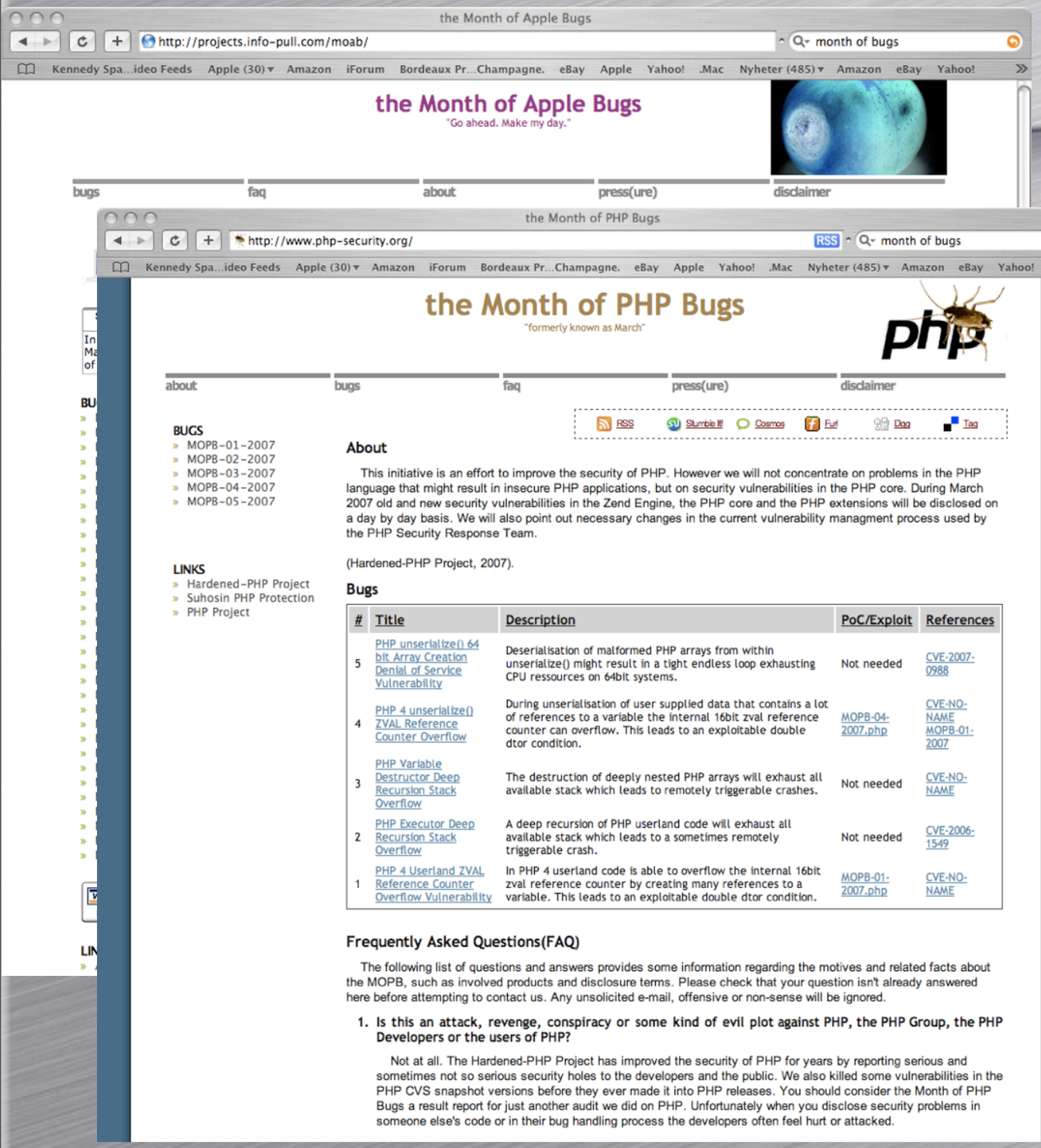
## *Legala eller andra hot*

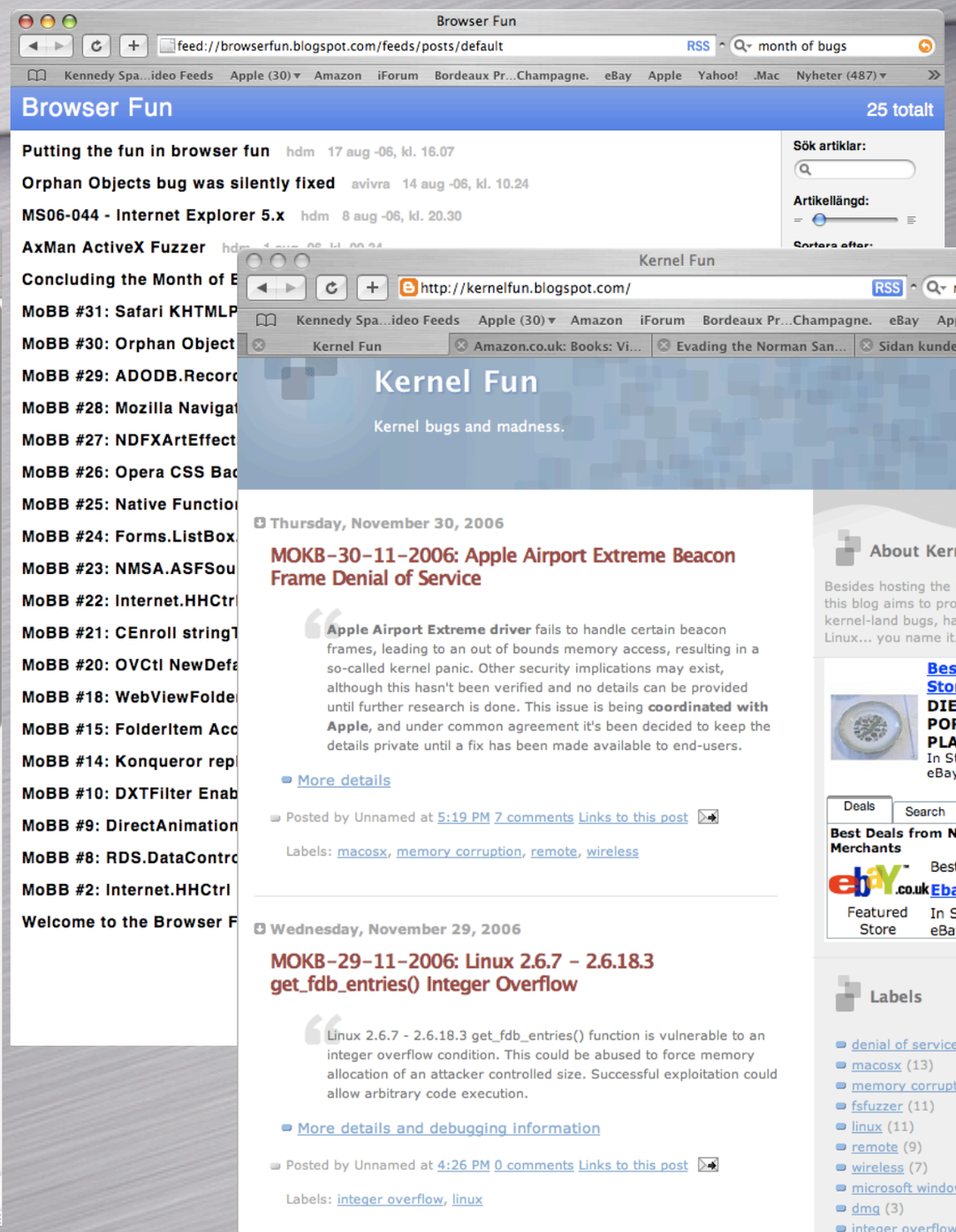
Juridiskt ansvar för hur webbplatsen nyttjas eller ***missbrukas***: chattar, kommentarsfält, gästbok

Medveten handling

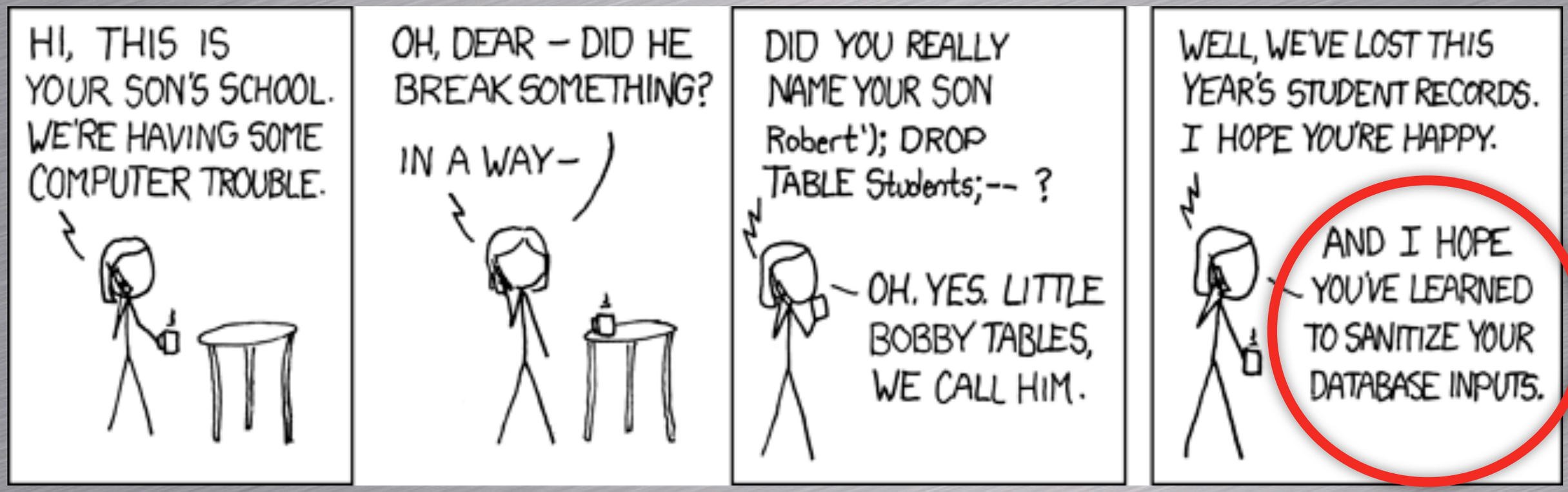
Spårdata och annan dynamisk inte sparad och går inte att återskapa

# Hot: Jakten på säkerhetshål fortsätter



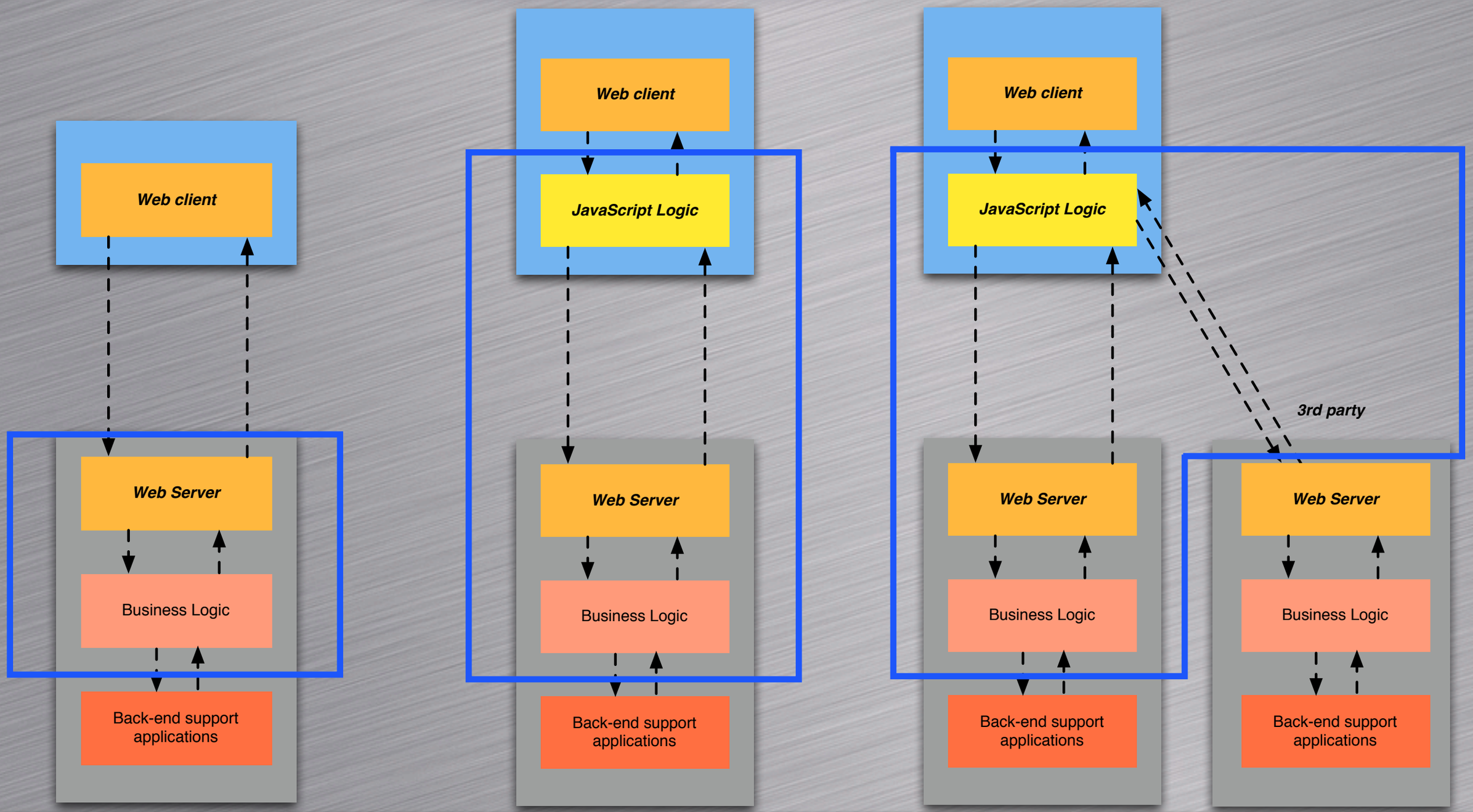



# Säkerhetsproblem i populärkulturen...




http://www.xkcd.com/327/

# Buzz word bingo: WebServices, Web 2.0



# Buzz word bingo: WebServices, Web 2.0

HTML & CSS
JavaScript
XHR-objekt
HTTP


- Web 2.0 öppnar upp för nya typer av angrepp mot, inte minst, klienten
- Brandväggar för WebServices är andra djur än klassiska IP-adressfiltrerande brandväggar
  - Svårt - Tvingar nät/brand-väggsadministratörerna att tolka affärslogiken
- Skapar en mängd nya säkerhetstjänster
- Många lösningar nyttjar inte de säkerhetskoncept som finns framtagna:
  - autentisering, signering, etc

# Initiativ för webbsäkerhet

## *Permanenta tekniska skydd*

Nätbaserad brandvägg

Hostbaserad brandvägg

Nätbaserad IDS

Hostbaserad IDS

Nätbaserad IPS

Hostbaserad IPS

Reverseproxy

Härdat OS på värddator

Penetrationstester

**Pärlor för svin?**

Investeringarna felaktiga alt. ofullständiga om vi ***inte förstår*** *tekniska förutsättningar*, *vad som skall skyddas* eller *skyddsvärdet på det som skall skyddas*

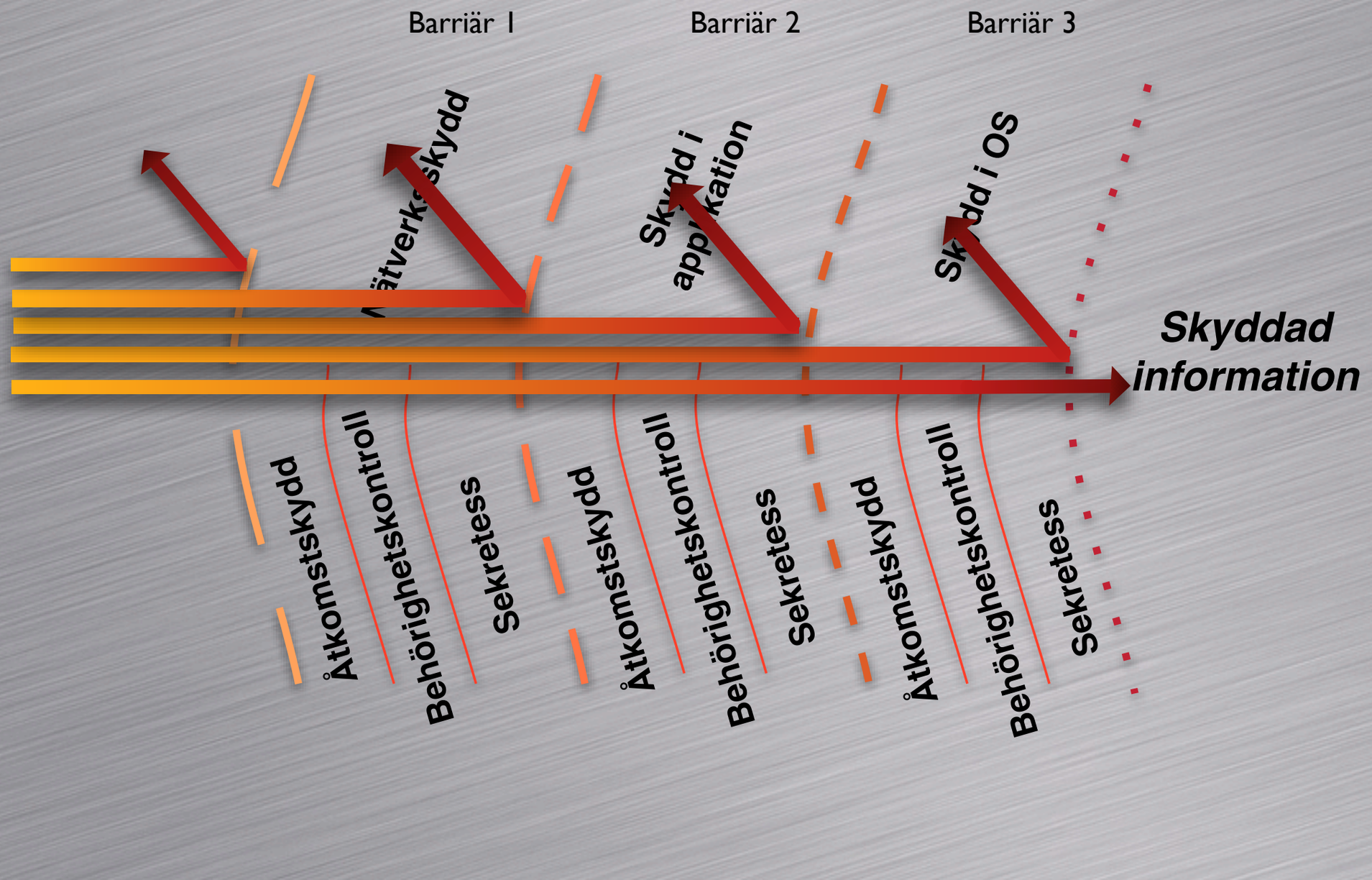
## *Permanenta organisatoriska skydd*

Ha genomtänkt informationsstruktur, test- och produktionssättningförfarande

Inför en ***BRA*** utveckling
metodik / kräv att l
kan påvisa att de ny dan

Microsoft Secure Development Lifecycle Model

# Försvar i djupled



# Seriöst dumma idéer

- Att lita på indata som kommer från något som en användare kan påverka
- Att använda samma dator till alltför många tjänster
- Att hyra en “hacker” för att testa säkerheten
- Att lägga ut information publikt på webben som egentligen skulle ligga på ett ***skyddat och begränsat åtkomligt extranet***
- Slita på E-legitimation för än det ena, än det andra - ena dagen privat, andra dagen på jobbet, tredje dagen för en förenings räkning
- PIN-kodsbaserade inloggningar, medför såväl enkla uttömmande sökningar som DoS-attacker
- Hårdkodade lösenord i program och skript
- Att naivt tro att alla är snälla som läser filen robots.txt

# Do's & Dont's

## Do's

- Förstå riskerna med införande av nya lösningar
- KISS-principen
- Förstå databasen som nyttjas - använd fullt ut de säkerhets-mekanismer som ingår
- Använd mekanismer som finns i programspråk för att undvika säkerhetsproblem:
  - taint, sandlådor, etc
- Försvar på djupet

## Don'ts

- Blanda inte utveckling, test och produktionsmiljöer
- Gör inte utveckling i produktionsmiljön
- Lägg inte ut information i tron att den ***inte syns*** bara för att den saknar länkar från webbsidor
- Undvik att *koda fast* dig i specifika versioner, tex av appservern
- Ge inte ut för mycket information i felmeddelanden

# Do's & Dont's

## Do's

- LOGGAR!!!

Loggar behöver vara *korrekta*, *kompletta* och ***användbara***

- Skydda den personliga integriteten

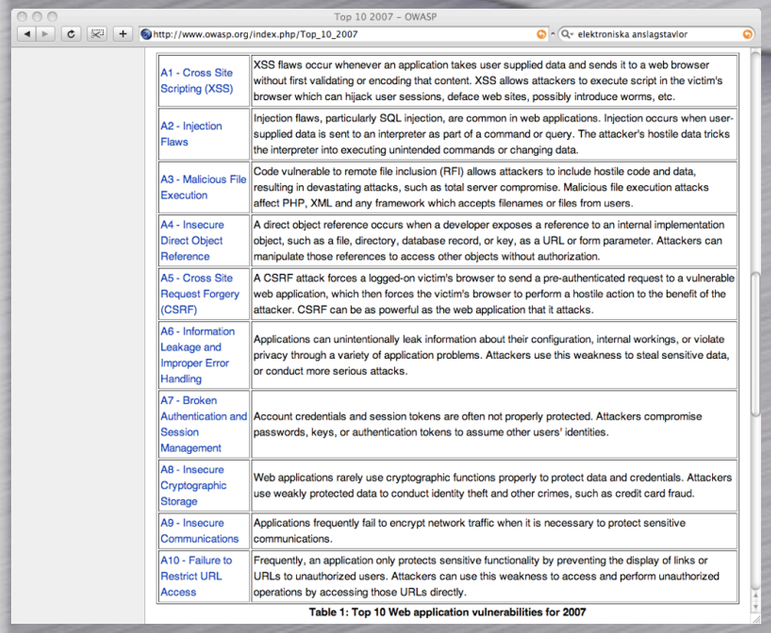
## Don'ts

- Lägg inte in säkerhetsfunktioner ***i enbart klientsidan***
- Använd inte Pop-up-rutor med text som ändå ingen läser....
- Okontrollerad indexering och sökning av de egna webbsidorna
- Som användare på webben - lämna inte integritetskänslig information

Kom ihåg: Internet ***glömmer inte***

# OWASP är din vän!

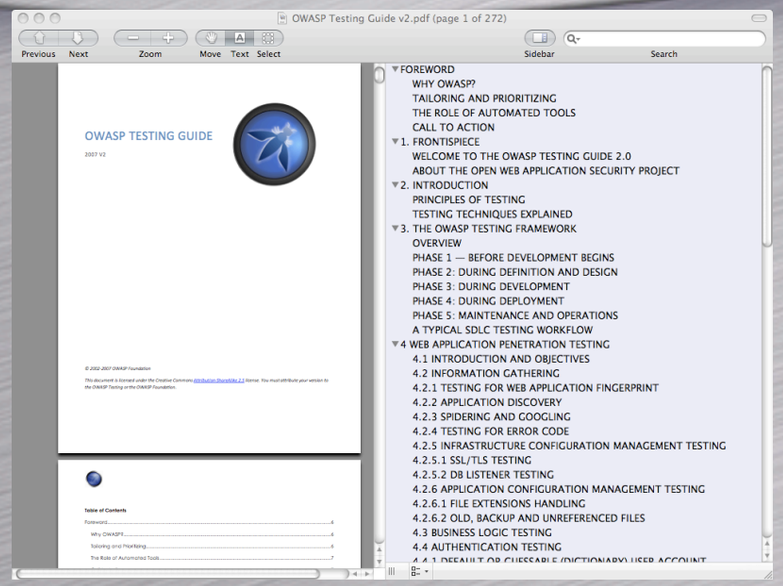
*Open Web Application Security Project*

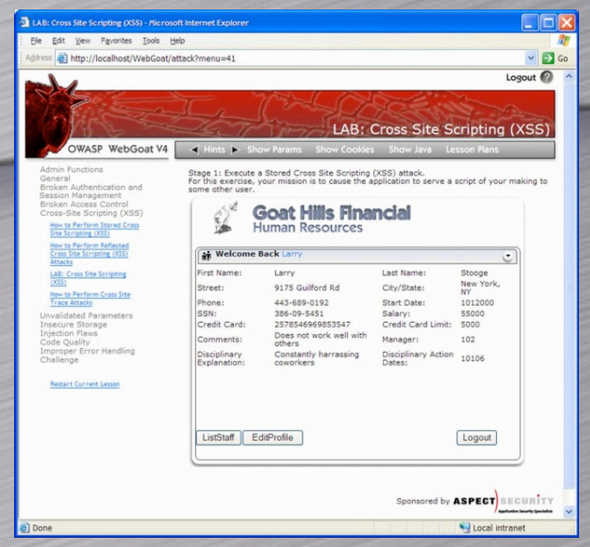

Top 10 2007 – OWASP

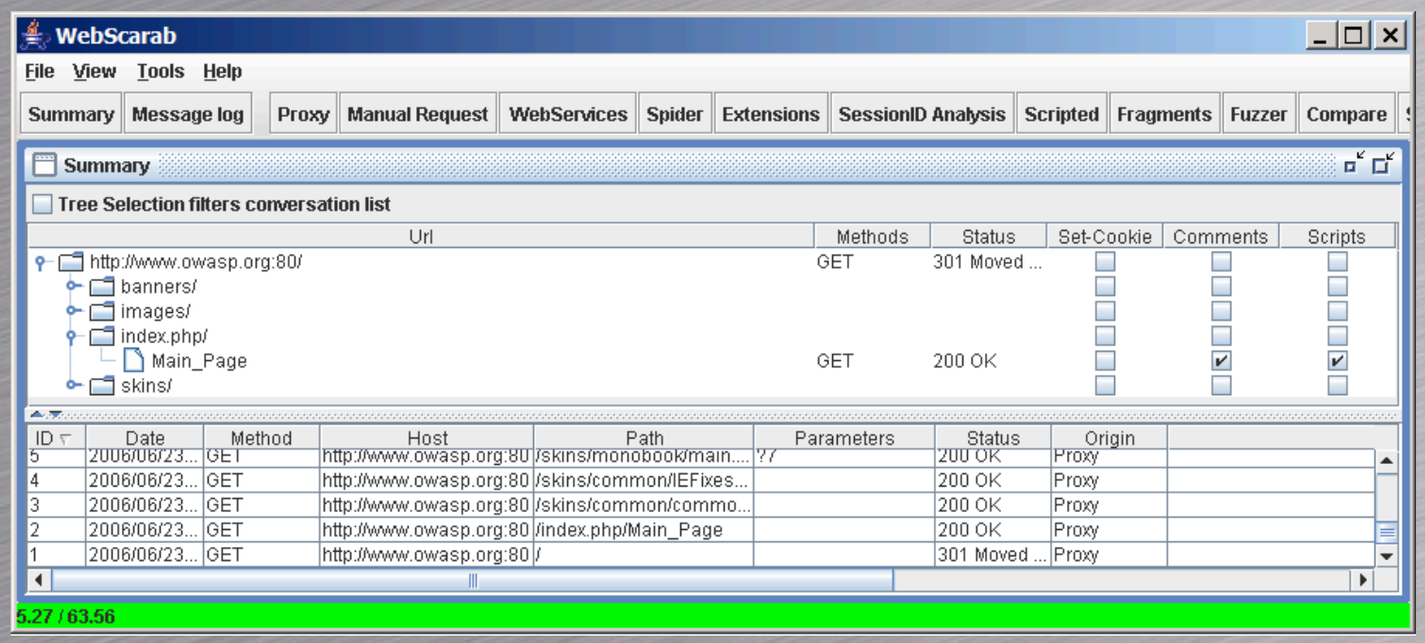
http://www.owasp.org/index.php/Top_10_2007

| | |
|---|---|
| A1 - Cross Site Scripting (XSS) | XSS flaws occur whenever an application takes user supplied data and sends it to a web browser without first validating or encoding that content. XSS allows attackers to execute script in the victim's browser which can hijack user sessions, deface web sites, possibly introduce worms, etc. |
| A2 - Injection Flaws | Injection flaws, particularly SQL injection, are common in web applications. Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. The attacker's hostile data tricks the interpreter into executing unintended commands or changing data. |
| A3 - Malicious File Execution | Code vulnerable to remote file inclusion (RFI) allows attackers to include hostile code and data, resulting in devastating attacks, such as total server compromise. Malicious file execution attacks affect PHP, XML and any framework which accepts filenames or files from users. |
| A4 - Insecure Direct Object Reference | A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. Attackers can manipulate those references to access other objects without authorization. |
| A5 - Cross Site Request Forgery (CSRF) | A CSRF attack forces a logged-on victim's browser to send a pre-authenticated request to a vulnerable web application, which then forces the victim's browser to perform a hostile action to the benefit of the attacker. CSRF can be as powerful as the web application that it attacks. |
| A6 - Information Leakage and Improper Error Handling | Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Attackers use this weakness to steal sensitive data, or conduct more serious attacks. |
| A7 - Broken Authentication and Session Management | Account credentials and session tokens are often not properly protected. Attackers compromise passwords, keys, or authentication tokens to assume other users' identities. |
| A8 - Insecure Cryptographic Storage | Web applications rarely use cryptographic functions properly to protect data and credentials. Attackers use weakly protected data to conduct identity theft and other crimes, such as credit card fraud. |
| A9 - Insecure Communications | Applications frequently fail to encrypt network traffic when it is necessary to protect sensitive communications. |
| A10 - Failure to Restrict URL Access | Frequently, an application only protects sensitive functionality by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly. |

Table 1: Top 10 Web application vulnerabilities for 2007

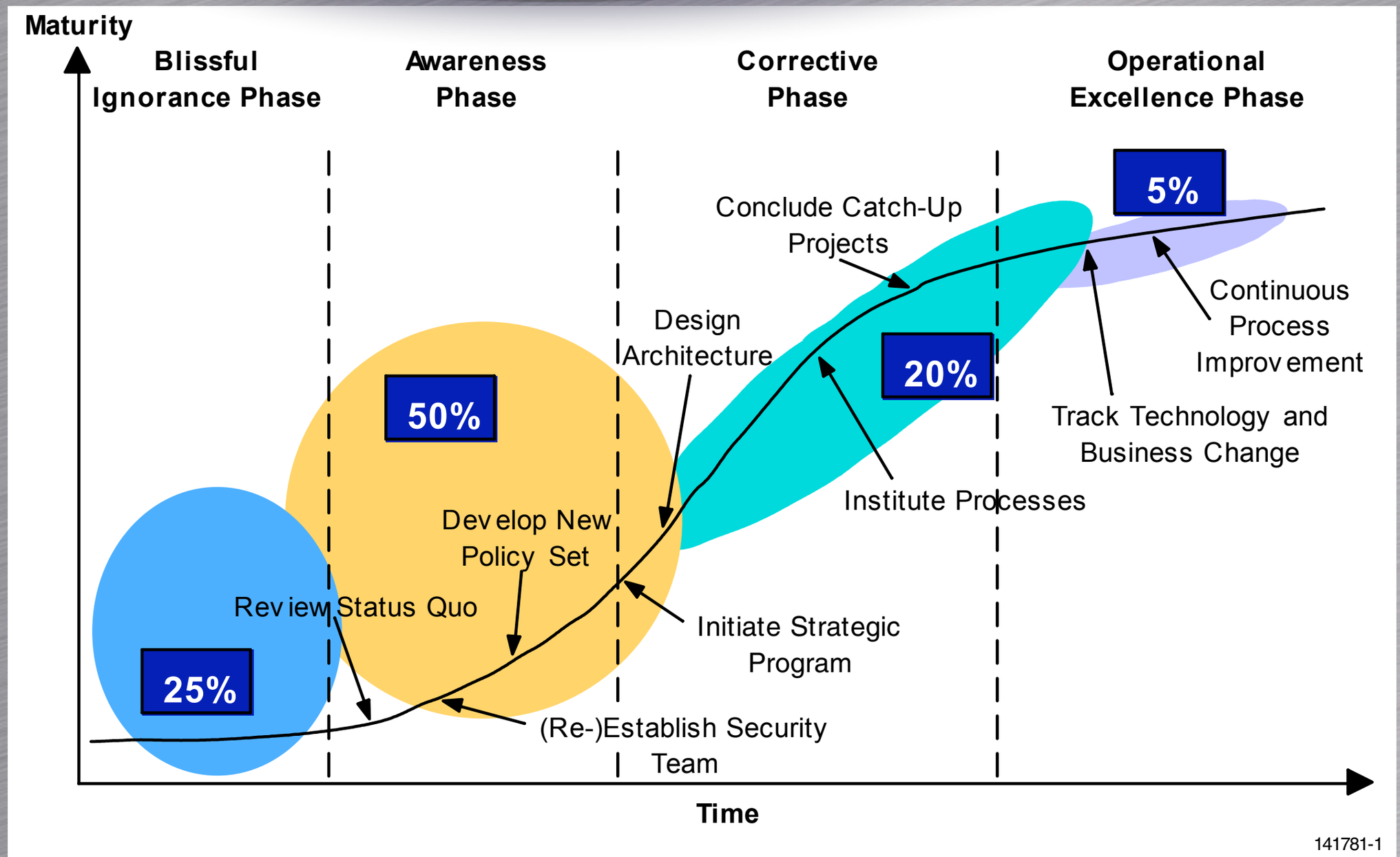



- BRA utvecklingsguider
- BRA testguider
- mängder med BRA testverktyg
- Exempel för utbildning
- BRA topplistor över angrepp
- **www.owasp.org**

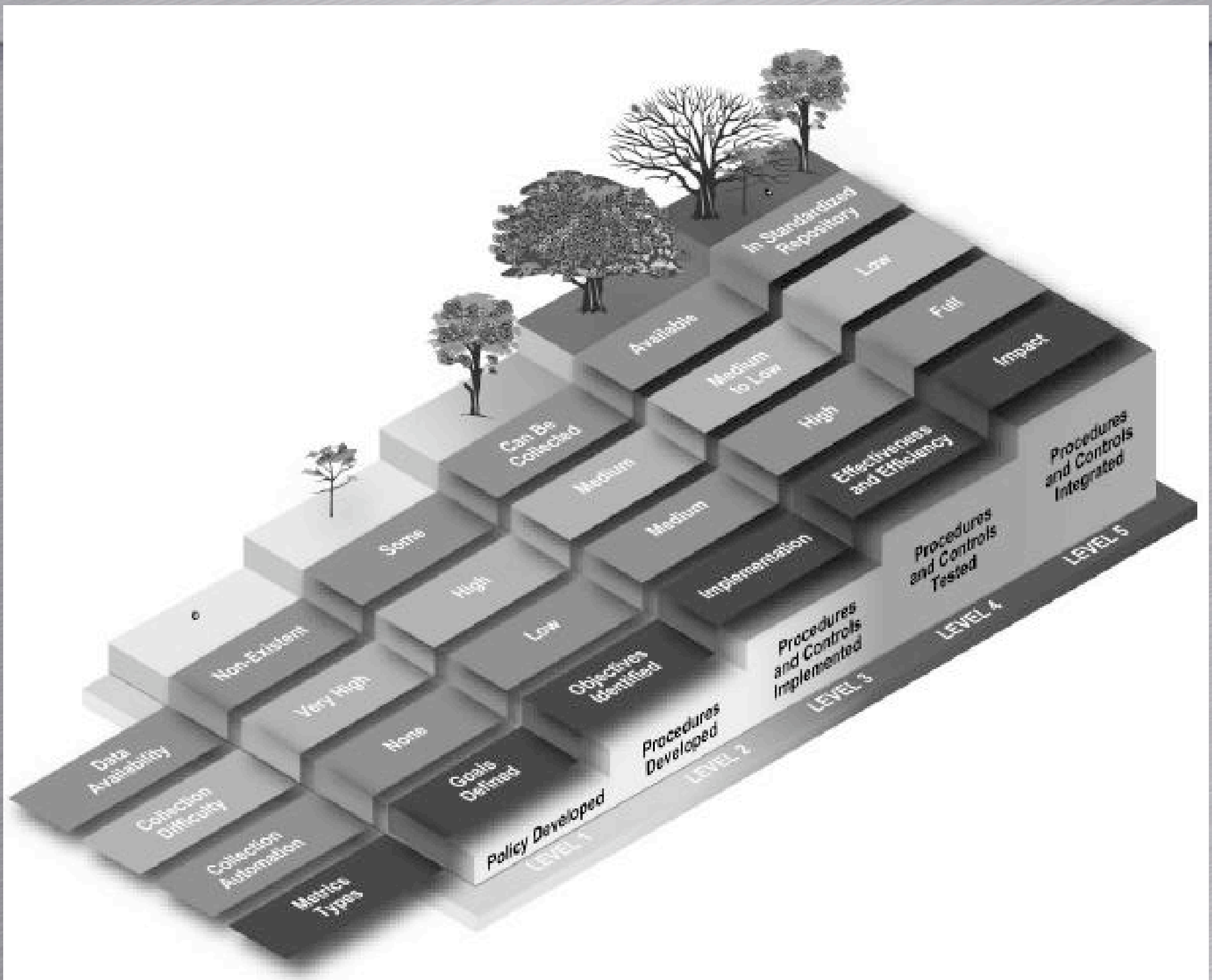
# Utvecklingskurvan för säkerhet



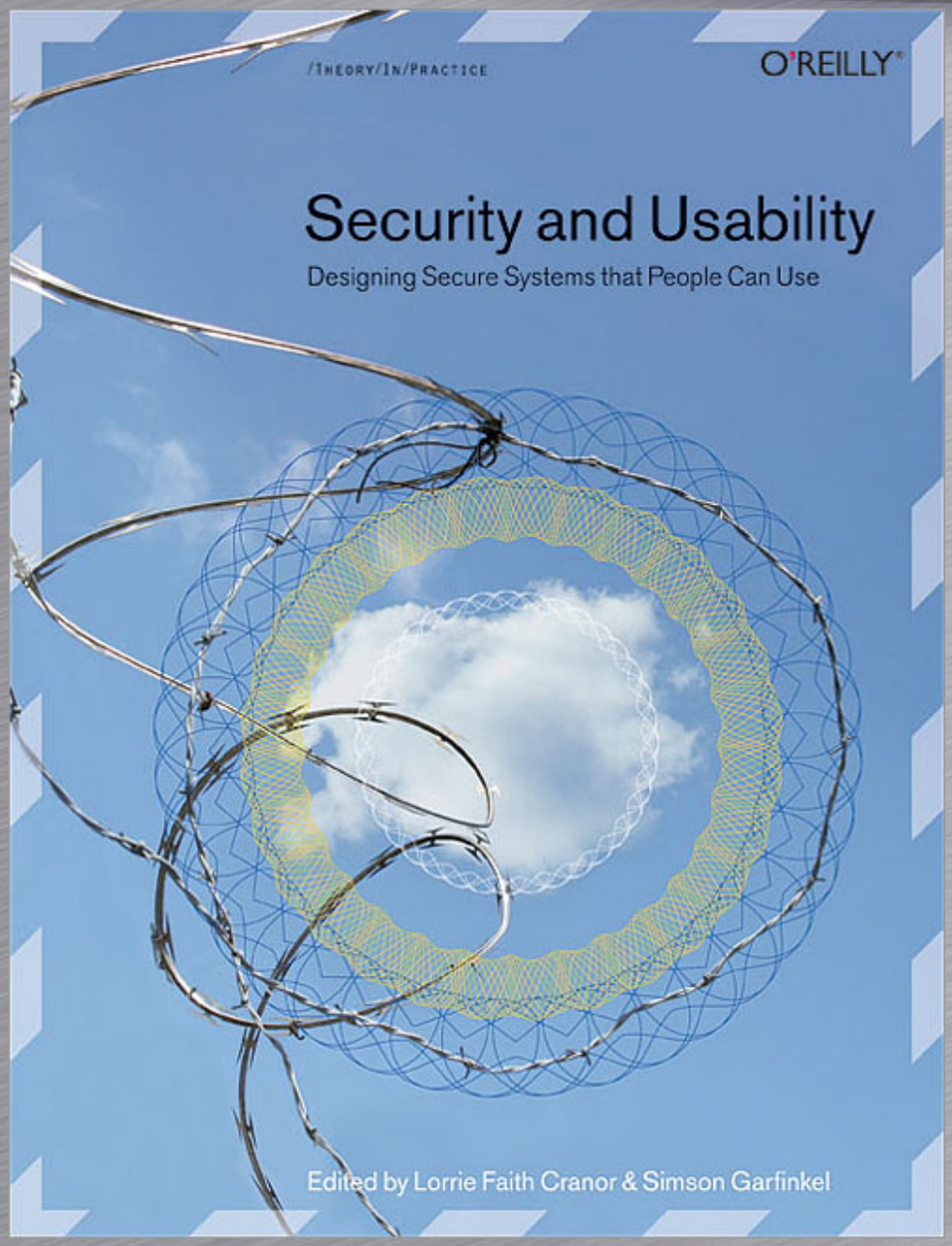

Note: The population distributions represent typical large G2000-type organizations.

**Source: Gartner (July 2006)**

# Utvecklingskurvan för säkerhet 2



# Apropå Gulans föredrag...




Källa http://www.oreilly.com/catalog/securityusability

**Psychological Acceptability Revisited**

**Design for Usability**

**Designing Systems That People Will Trust**

**Guidelines and Strategies for Secure Interaction Design**

**Fighting Phishing at the User Interface**

**Sanitization and Usability**

**Security Administration Tools and Practices**

**Privacy Issues and Human-Computer Interaction**

**A User-Centric Privacy Space Framework**

**Five Pitfalls in the Design for Privacy**

**Firefox and the Worry-Free Web**

**Users and Trust: A Microsoft Case Study**

# Apropå Gulans föredrag...

# Secure Interaction Design

Ka-Ping Yee
with Norm Hardy, Mark Miller, Chip Morningstar,
Kragen Sitaker, Marc Stiegler, Dean Tribble, and Miriam Walker

## Basic Concepts

### ACTOR-ABILITY MODEL

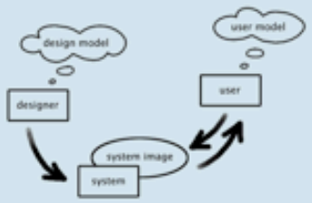
At any point in time, the user's model contains a set of **actors** in the system and a set of **potential actions** for each actor. For a system to be secure, the actual abilities of any actor must never come to exceed the bounds in the user model.

actors $A = \{A_0, A_1, \ldots, A_n\}$
perceived abilities $P = \{P_0, P_1, \ldots, P_n\}$
real abilities $R = \{R_0, R_1, \ldots, R_n\}$

$P_0 \subseteq R_0$
$P_i \supseteq R_i$ for $i > 0$

### SYSTEM IMAGE

The actors, actions, and objects in the user's mental model are derived from observing the **system image**, not from knowledge of its internal design.




### USERS AND USER AGENTS

The software system intended to serve and protect the interests of the user is the **user agent**. On a stand-alone PC, this is the operating system shell, through which the user interacts with an arena of entities such as files and programs. On a networked PC, a second level of user agent represents the user's interests in a larger arena of interacting computers.

## Fundamental Principles


### PATH OF LEAST RESISTANCE

The natural way to do any task should also be the secure way.

### APPROPRIATE BOUNDARIES

The interface should expose distinctions between objects and between actions along boundaries that matter to the user.

## Actor-Ability State

### VISIBILITY

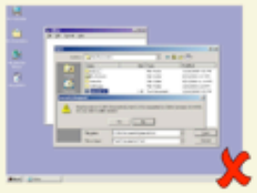
The interface should allow the user to easily review any active authority relationships that would affect security-relevant decisions.

### REVOCABILITY

The interface should allow the user to easily revoke authorities that the user has granted, wherever revocation is possible.

### EXPLICIT AUTHORITY

A user's authorities must only be provided to other actors as a result of an explicit action that is understood by the user to imply granting.



### EXPECTED ABILITY

The interface must not generate the impression that it is possible to do something that cannot actually be done.

## Input and Output

### TRUSTED PATH

The interface must provide an unspoofable and faithful communication channel between the user and any entity trusted to manipulate authorities on the user's behalf.

### IDENTIFIABILITY

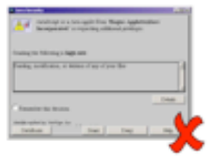
The interface should enforce that distinct objects and distinct actions have unspoofably identifiable and distinguishable representations.

### EXPRESSIVENESS

The interface should provide enough expressive power to (a) describe a safe security policy without undue difficulty and (b) allow users to express security policies in terms that fit their goals.

### CLARITY

The effect of any security-relevant action must be clearly apparent to the user before the action is taken.



Källa http://people.ischool.berkeley.edu/~ping/sid/sid72.tif

# Att vara förberedd...

It-krasch under terrorövning – IDG.se

http://www.studio.idg.se/17.108/2.1085/1.105300

krisberedskapsmyndigheten webb kras

Soekris on O...ng Diskless | Positive Ath... Quotations | Data Visuali... Approaches | nwsmp – tech+bus+world | newsmap | del.icio.us ...ular treemap

IDG.se — En sajt i IDG.se-nätverket — ONSDAG 21 november 2007

Ny Internetworld UTE NU! >>

ÖVERSIKT | PRENUMERERA | KUNDSERVICE | ANNONSERA | OM IDG

SÖK | BLI MEDLEM | LOGGA IN

NYHETER 360° | TIDNINGAR | TESTER | TIPS & GUIDER | JOBB & KARRIÄR | FRITID & NÖJE | VERKTYG & NYTTA | COMMUNITY | BUTIKEN | EVENTS

Nyheter efter intresseområde » IDG-TV | NÄTVERK & TELEKOM | PRYLAR | MOBIL IT | SERVER | EXPERTSVAR | VOIP | VIRTUALISERING | IDG-nyheter i RSS-format RSS

KRIS I KRISÖVNINGEN | 2007-04-25 10:57 - Computer Sweden:

## It-krasch under terrorövning

Av Daniel Goldberg | ComputerSweden

**Samhälle (Uppdaterad) Ett it-haveri har satt käppar i hjulen för krisberedskapsmyndighetens, KBMs, stora terrorövning som pågår i Stockholm. Redan tidigt på morgonen kraschade KBMs webbplats, vilket har skapat stora problem för de inblandade myndigheterna.**

ANNONS


**Samverkansövningen SAMÖ 2007** är en av de största i sitt slag någonsin i Sverige och iscensätter en terrorattack mot Stockholms spårtrafik.

**Men redan tidigt** på onsdagsmorgonen kraschade KBMs webbplats, troligen på grund av en kraftig trafikökning.

**FAKTA**

**Samverkansövning 2007**

Fler än 40 myndigheter och upp till 4000 personer deltar i SAMÖ 2007 som sägs vara den största övningen i sitt slag någonsin i Sverige.

Övningen iscensätter en terrorattack mot Stockholms spårtrafik med efterföljande spridning av radioaktiva ämnen. Krisberedskapsmyndighetens roll är att samordna och planera övningen. Syftet är att utvärdera och stärka Sveriges beredskap vid terrorattentat.

Bland de medverkande myndigheterna märks Försvaret, Smittskyddsinstitutet och Räddningsverket. Övningen pågår mellan den 24 och 26 april i Stockholm, men utspelas enbart dagtid.

**LÄS MER**

- Allt om haverier och systemfel på CS samlingssida

Skriv ut | Tipsa | Kommentera

Spara till del.icio.us | Dela på Facebook

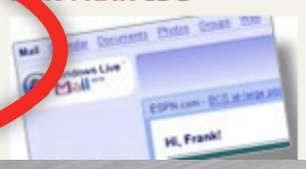
**Bloggkommentarer till artikeln** TWINGLY Twingly bloggsök

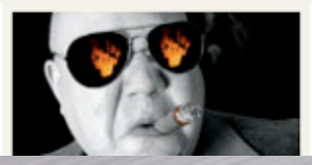
2 kommentarer, visar de 2 mest länkade. Visa alla.

**IT-systemet kraschade under berdskapsövning** ANMÄL
on 25 apr 2007 - iterationer

**Ironi 7** ANMÄL
lö 28 apr 2007 - Till yttermera visso

**ER FRÅN IDG**





**TIPSA OSS**

Mejla oss om du har hört talas om ett scoop eller har något annat som du vill att vi bevakar.

Vill du höra av dig av någon annan anledning hittar du kontaktuppgifter på denna sida.

ANNONS


**NYHETSBREV**

**Heta nyheter till din mejlbox**

Få vårt nyhetsbrev med de senaste it-nyheterna, direkt i din inkorg varje morgon! Nu också med dagens Dilbert.

**E-postadress**

Registrera | Avregistrera

ANNONS

# Slutsatser (1)

- Det finns ingen magisk silverkula
- "JavaScript is the new Shellcode"
- WebServices kan leda till exponering av programåtkomst
- Många lösningar skapar cirkelberoenden mellan komponenter, vilket omöjliggör patchning och uppgradering
- Sårbarheter finns överallt, inte minst i plattformen man nyttjar, tex *XML Core components*, PHP, Java, Ruby, WAS, IAS, Apache, IIS, etc
- Antagonistfaktorn och programmerade hot utelämnas eller misförstås i riskanalyser

# Slutsatser (2)

- Angrepp sker numera ofta från server till klient, efter det att en intern användare "bjudit in".
- Spårbarhet är en av de grundläggande säkerhetsfunktionerna. Spårbarhet (i form av loggar) är i praktiken ofta ***helt oanvändbara*** när de behövs som mest...
- Undvik att bli en tjänst som nyttjas för att skada tredje part: SMS-bombning, phishing,
- Tänk på det juridiska ansvaret som ändå finns: PuL, Lag om ansvar för elektroniska anslagstavlor, mm, mm
- ***Att ligga i teknikens framkant medför per automatik större risker: oprövad teknik, oupptäckta säkerhetshål, etc***

"Sir, now might be a good time to bring up our security issues. The guards are all paper cutouts, and the swords and spears are made out of rubber."