



Infrastructures w/o Infrastructures ?

Dieter Gollmann

Hamburg University of Technology

TUHH

Technische Universität Hamburg-Harburg

FORWARD, Nice, May 4th 2009

Preamble



- The Internet is a critical infrastructure.
 - Many applications rely on it, and their number is growing.
- The infrastructure of the Internet: DNS, Bind, BGP, ...
- This infrastructure is creaking in its joints.

- Should we fix it, or do without it?

DNS



- DNS: Directory of hosts on the Internet.
- Authoritative DNS servers manage sections of the name space.
- Provides a level of indirection: Binds (long lived) DNS names to (short lived) IP addresses.
- Weak, non-cryptographic authentication by challenge/response.
- Attacks:
 - DNS spoofing, DNS cache poisoning: impersonate authoritative DNS server; “old” problem; recent attack by Dan Kaminski.
 - MD5 certificate collisions
 - DNS rebinding: malicious authoritative name server creates binding to the victim’s IP address.
 - ???
- DNSSEC would not solve all problems.

SSL



- Creates secure channels between client and server.
- Server certificate names server by DNS host name.
- Phishing attacks; man-in-the middle attacks:
 - Does the user know the correct server?
 - Does the user recognize the correct server?
- Should the user be involved when verifying server certificates?
Can the user be involved when verifying server certificates?
- When DNS is broken, SSL cannot provide the intended service.

- SSL is fine as a security mechanism, but it is not clear that it provides a useful security service for applications.

Web application security



- Web applications: HTTP as major transport protocol, web browser as the main client-side component.
- Same origin policies: applets, cookies
- Origin = host name (DNS)
 - Why?
 - Because DNS already exists?
 - Because DNS is a universal infrastructure?
- If DNS is broken, web application security is broken.
- If origin cannot be authenticated, web application security is broken.

Web application attacks



- Cross-site scripting: Script executed on client with rights of a trusted server.
- Cross-site request forgery: Script executed on server with rights of a trusted client.
- JavaScript hijacking.
- One view of these attacks: Broken authentication as the root of the problems.
- More to come: Mashups, cross-domain policies

Challenges



- How to solve these authentication problems without postulating the existence of a universal security infrastructure, e.g. a global PKI?
- How to define the authentication problem that needs to be solved?
 - Recognition? “The same as last time.”
 - Know Thyself? Distinguish requests created locally from requests sent as proxy for someone else.
 - “Transitivity of trust”: Can we rely on authentication performed by some other parties?

Web application security ...



- ... from first principles.
- We need names for the applications.
 - Why DNS names? Can we do without?
- We need a rendezvous service to give us an address where the application is provided.
 - Why DNS/Bind?
 - Which security guarantees are required, if any?
Do we need authentication of the application address?
- Once we are in contact with the application, we might need authentication.
 - Of whom? A human rather than a bot? The same peer as last time? A named user?
 - How? At which protocol layer?

Conclusions – questions



- Do we need a universal infrastructure, such as DNS?
- Different rendezvous services for different domains?
 - AAA services at rendezvous service for private applications?
 - Different authentication mechanisms for different domains?
- Is it more secure doing without a universal security infrastructure?
- Can we succeed in bootstrapping new applications without building on existing universal infrastructures?
- Should the emphasis be on “no universal **security** infrastructure”?
- Separate the security of critical applications from network security?