IT-Security in Finance and Beyond

Principles and Examples from Banking

Prof. Walter Kriha,

Computer Science and Media Faculty

Stuttgart Media University

May 2013

walter@kriha.de

Agenda

1. It-Security in Finance
2. Damage Reducing Systems (not networks)

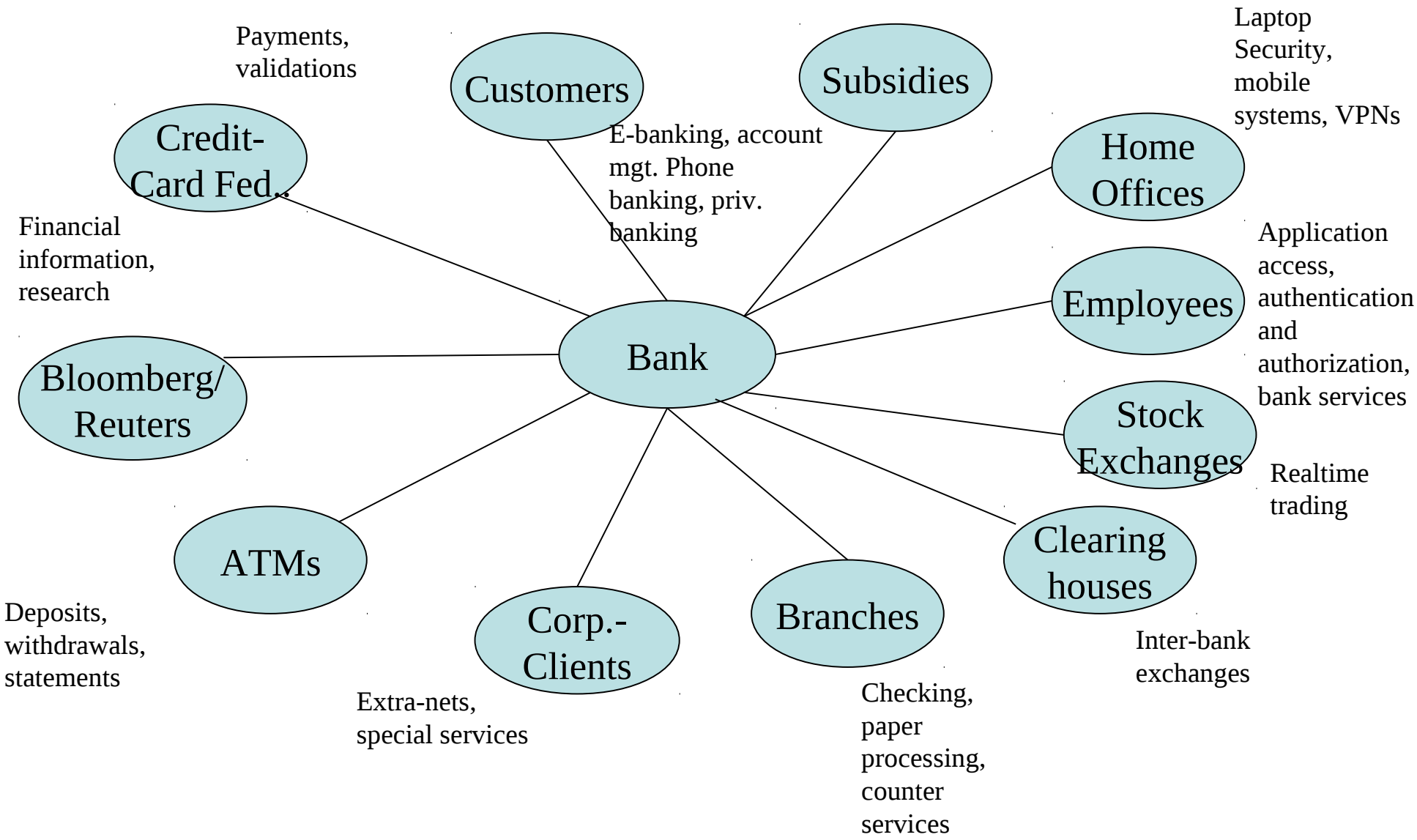
Agenda

1. It-Security in Finance
   a) Objectives
   b) Framework
   c) Software
   d) Infrastructure
   e) Client Facing Security

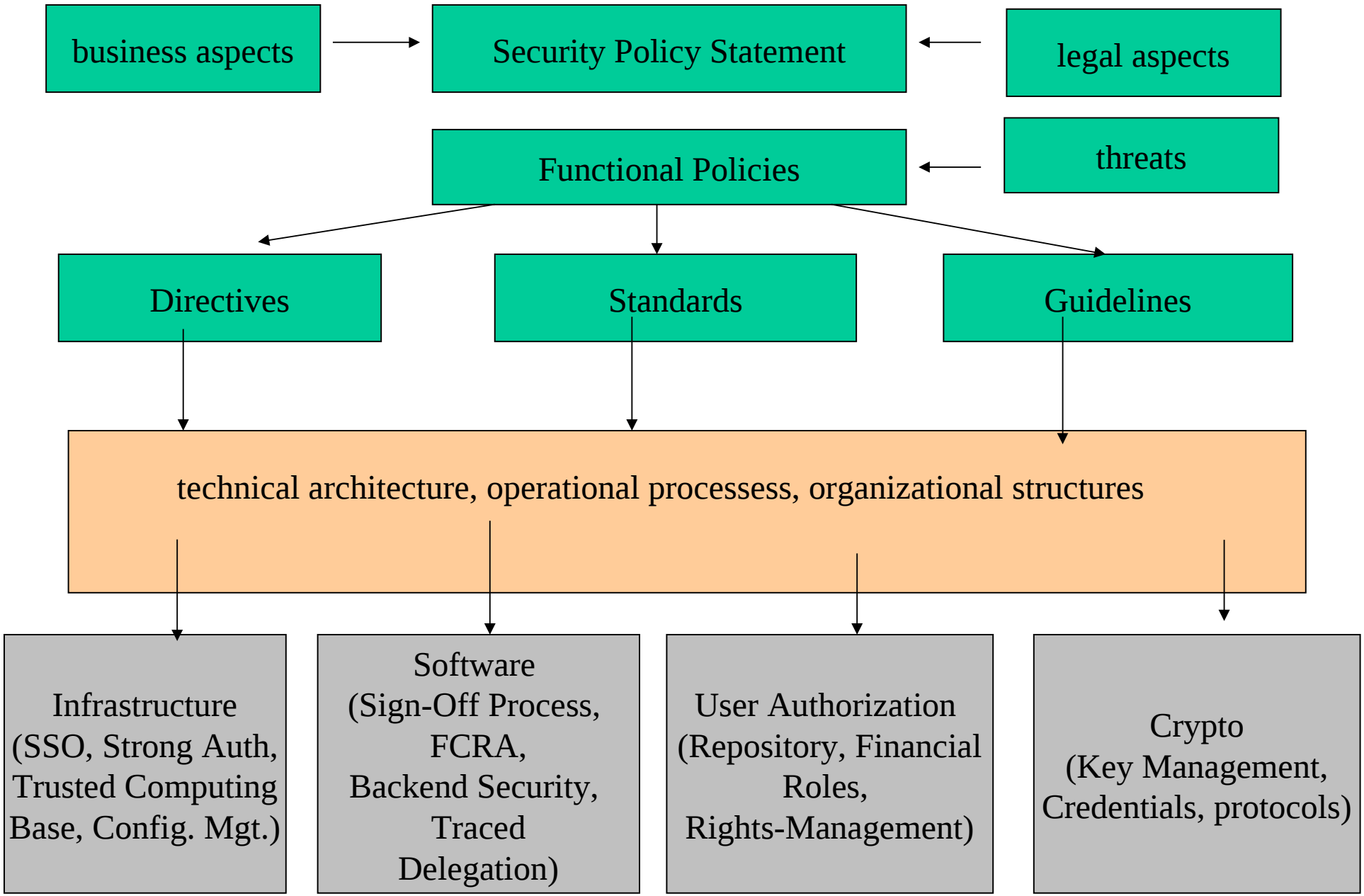
Objectives in Financial IT-Security

-Protect (digital) assets from outsiders AND insiders
- Protect private information of customers form outsiders AND insiders
- Comply with legal regulations and state law
- Ensure the availability of banking processes
- Provide audit trails for all business processes

Objectives: System Context Diagram



Framework


business aspects
Security Policy Statement
legal aspects
Functional Policies
threats
Directives
Standards
Guidelines
technical architecture, operational processess, organizational structures
Infrastructure (SSO, Strong Auth, Trusted Computing Base, Config. Mgt.)
Software (Sign-Off Process, FCRA, Backend Security, Traced Delegation)
User Authorization (Repository, Financial Roles, Rights-Management)
Crypto (Key Management, Credentials, protocols)

Framework: First Level Functional Policies

Mandatory Asset Ownership → drives user authorization process

Mandatory Data Classification → regulates data access, transmission and storage with respect to users, integrity and confidentiality

Mandatory Access Control → regulates authentication and authorization

„Need-to-know, Need-to-do“ → limits authority to required ones

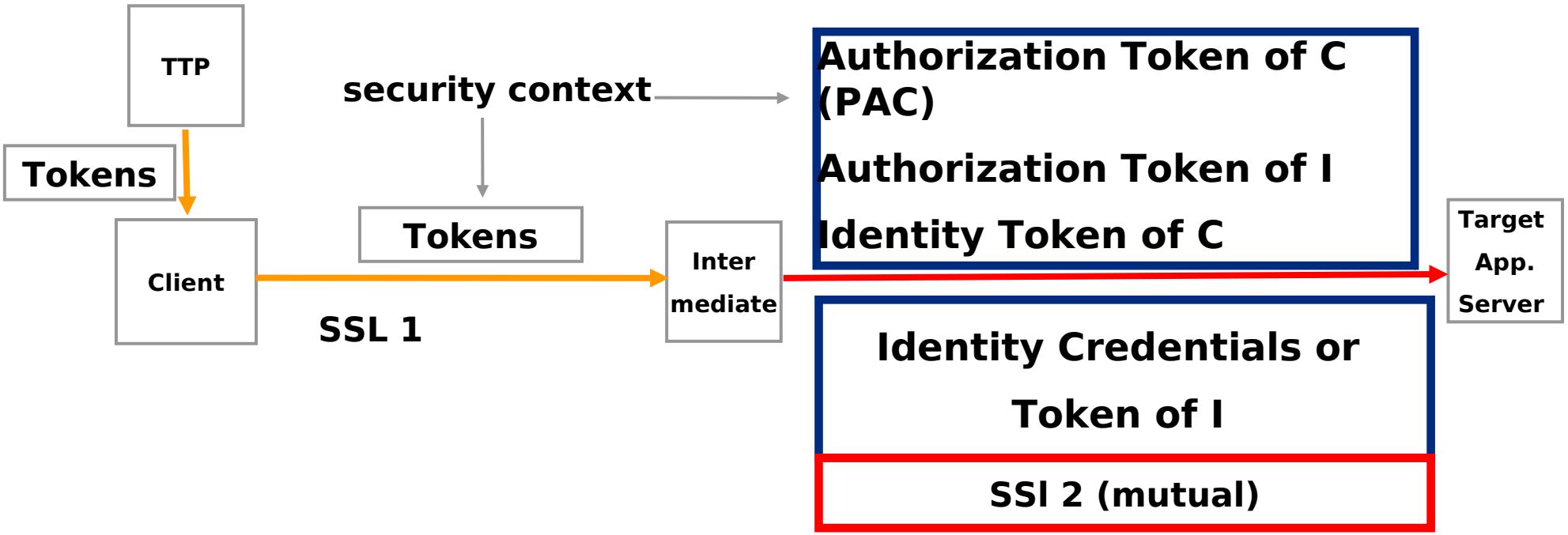
Risk Management and Incident Management → calculates and manages risks, prepares for disasters (e.g. duplicate data centers)

Software

- Security Sign-offs at various levels of development
- The importance of backend security
- Traced delegation vs. perimeter security
- Penetration Testing
- No authentication or custom authorization in applications

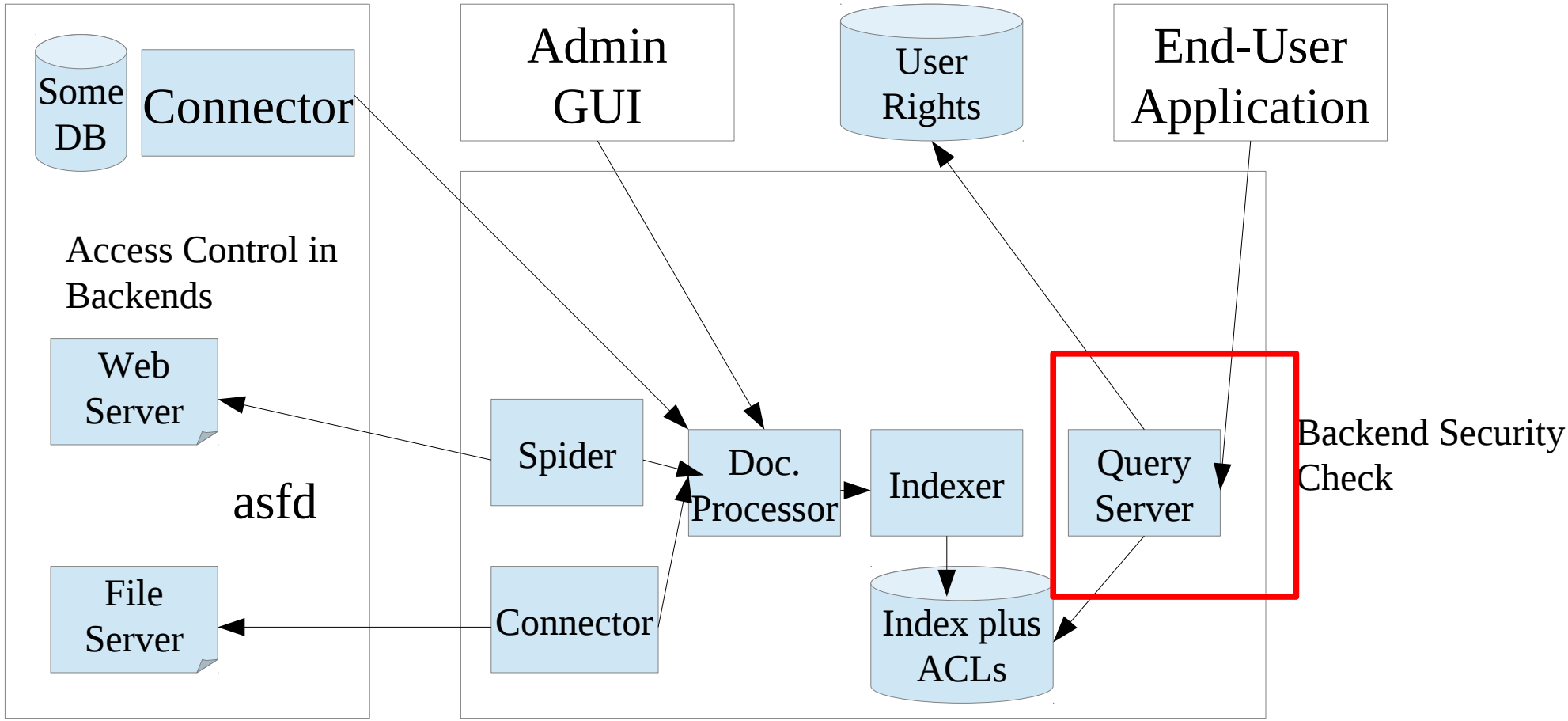
Software: Secure Delegation Concept

CORBA CSIv2 Mechanism



Every system involved authenticates itself against other tiers and flows client tokens. No secrets are shared. Defined routes prevent token abuse. Later tiers can verify original requestor and route.

Software: Backend Security in Enterprise Search

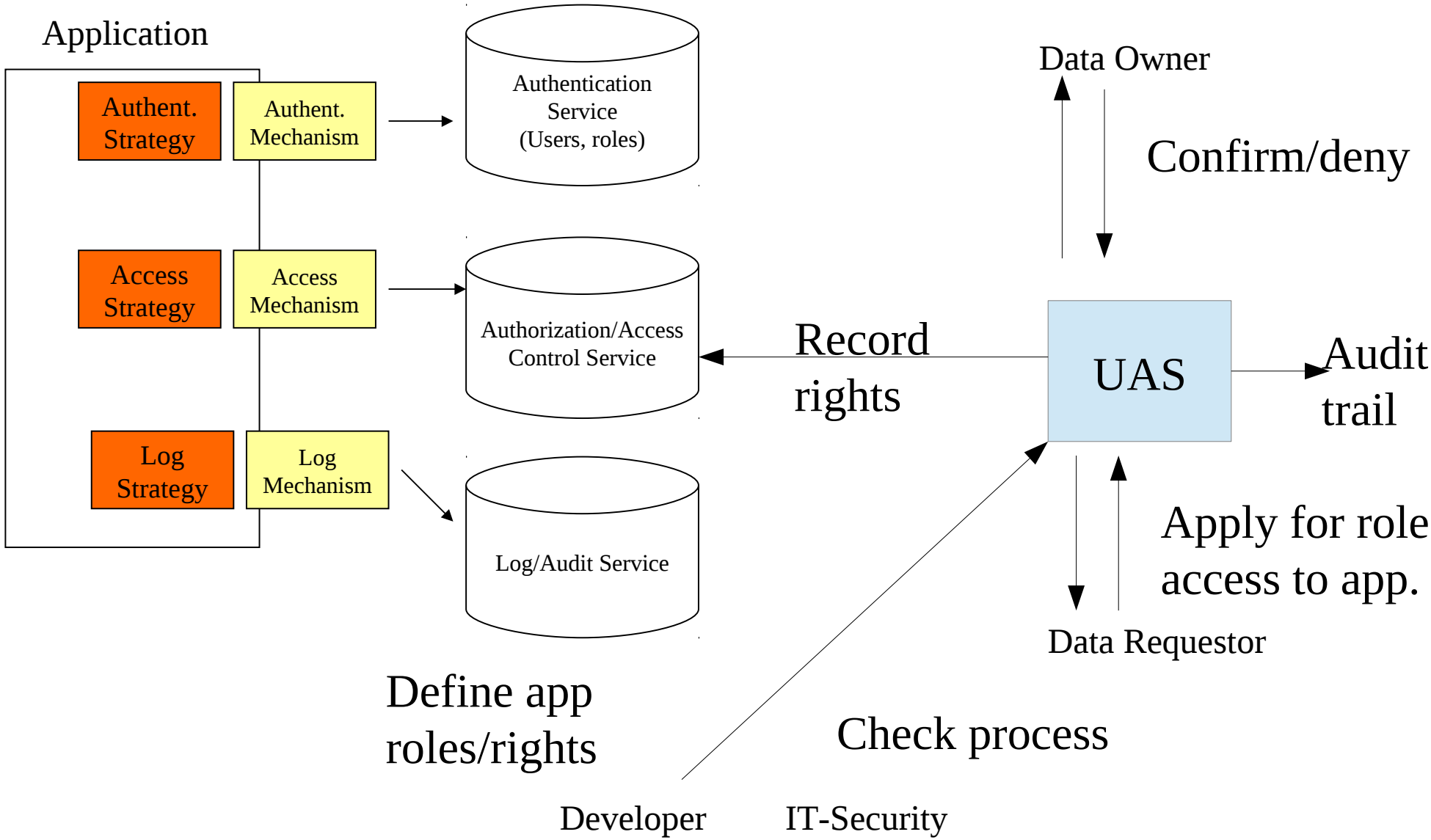


Administrative errors will not allow unauthorized access to search data because user rights are checked immediately against ACLs in index.

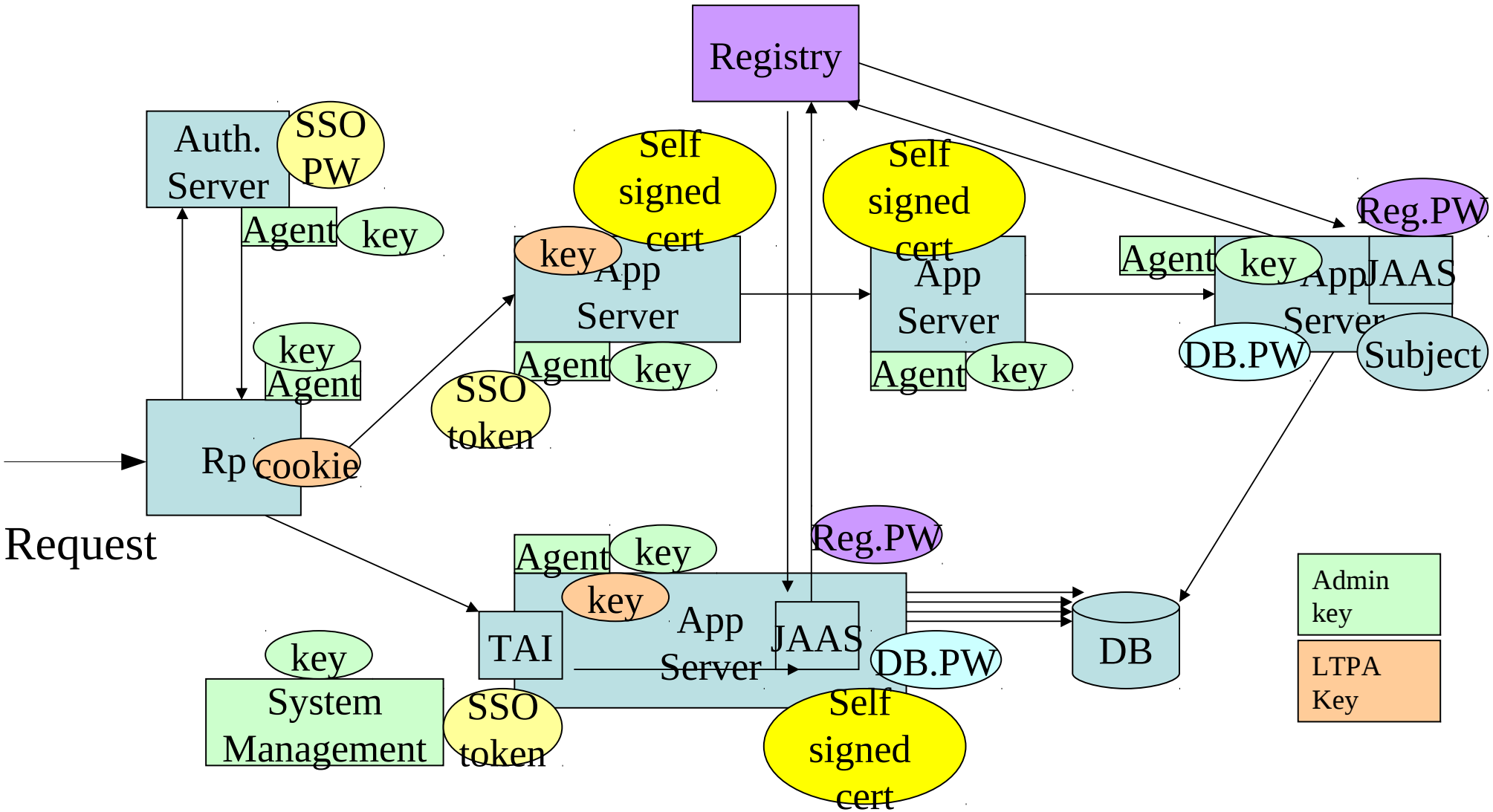
Infrastructure: Principles

- Strong authentication with smart cards
- Continuous system management with repository and automatic removal
- No developer access to production
- Traced software configuration and installation
- No interactive access to production systems
- No secret credentials over the wire
- Single Sign On
- Central User-Authorization System
- Core security services and data on IBM mainframes
- Mandatory protocol changes in firewall zones

Infrastructure: User-Authorization System

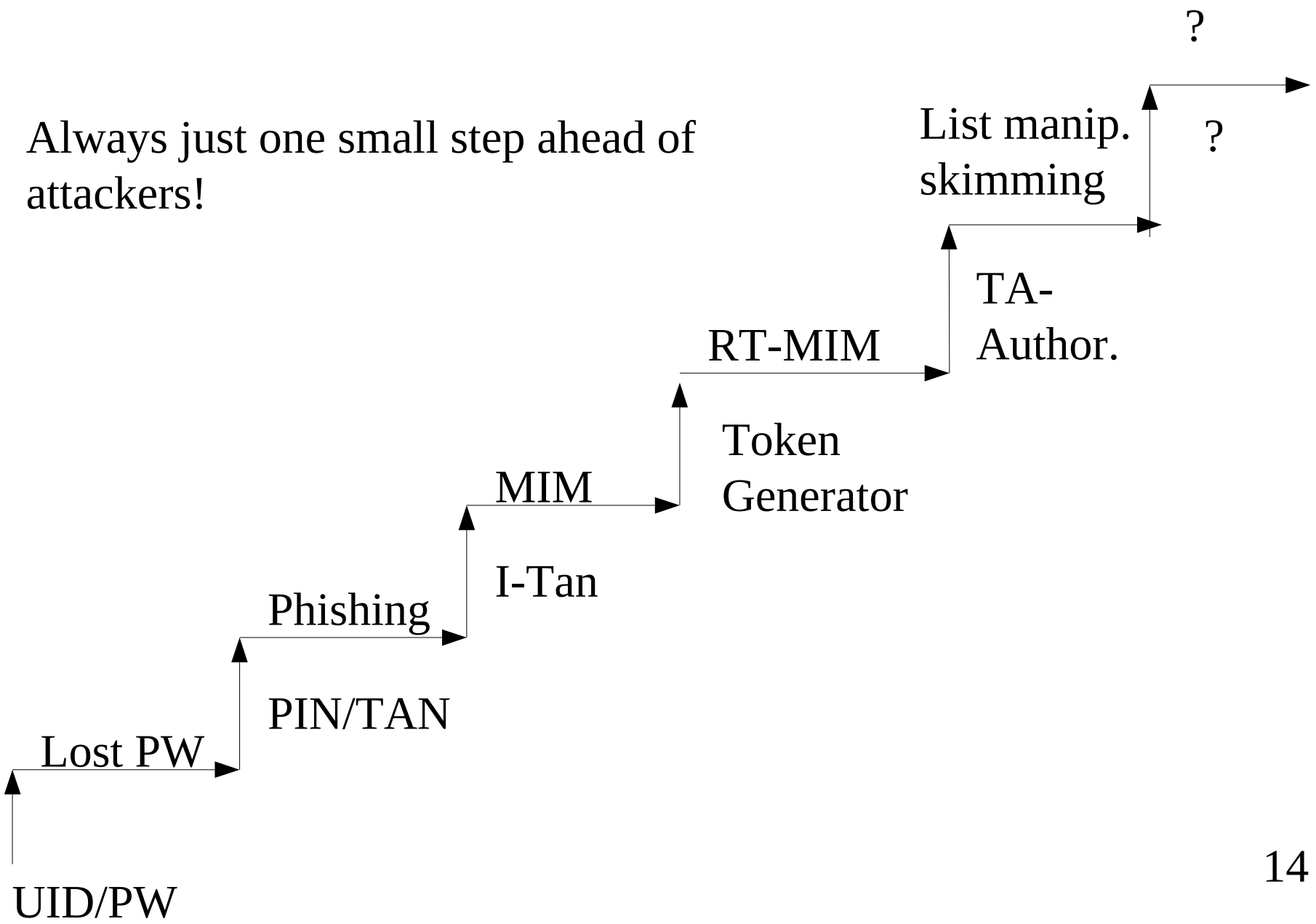


Infrastructure: Credential Management



Client-Facing Security: Authentication Race

Always just one small step ahead of attackers!



Does All This Really Help?

Yes, it does. Within limitations and at a high price.

- Input validation is still a problem (XSS, XSRF, inject, overflow)
- Channel based authentication within infrastructure is very static
- Systems in permanent danger of zero-day attacks
- What about Advanced Persistent Threats (Stuxnet and Co.)?
- What about information leaks?

Agenda

2. Damage Reducing Systems
   a) Fragility
   b) Root Causes
   c) Solutions

Fragility: Advanced Persistent Threats

- **You can no longer benefit from your level of security being a little bit higher than others. Your absolute level counts! (Schneier)**
- Attacks on single targets, SCADA systems
- Unheard of resources, skills, money used by attackers
- Attacks built over many years
- Special equipment only available to states used
- **Zero-Day-Attacks and Certificates easily bought**
- Insiders used to plant attack code
- Attack code no virus but an application or parts of it.
- Extreme measures to hide attack
- Some security bugs not fixed for years!
- Antivirus companies tell us they cannot detect ATPs

ATPs are a result of the so called „Cyber War“. States trying to attack other states or companies. New organizations are built, specialists trained on electronic warefare. How many of those will one day become corrupt?

Fragility : Buffer-Overflow

Our „aaaaaaaa..“ input from the keyboard is now the address where the next instruction should be read by the CPU. Now we know how to point the CPU to code we placed on the stack

```
Exception: STATUS_ACCESS_VIOLATION at eip=61616161
eax=00000012 ebx=00000004 ecx=610E3038 edx=00000000 esi=004010AE
edi=610E21A0
ebp=61616161 esp=0022EF08
program=D:\kriha\security\bufferoverflow\over.exe, pid 720, thread main
cs=001B ds=0023 es=0023 fs=003B gs=0000 ss=0023
Stack trace:
Frame     Function  Args
  90087 [main] over 720 handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
 104452 [main] over 720 handle_exceptions: Error while dumping state
(probably corrupted stack)
```

A program crash is a way into the system! But the real quality problem is much deeper: Stick a finger in some code and figure out what you can do from there. **What functions can you reach from any point in code? Who‘s failure is that?**

Fragility: Why is the following dangerous?

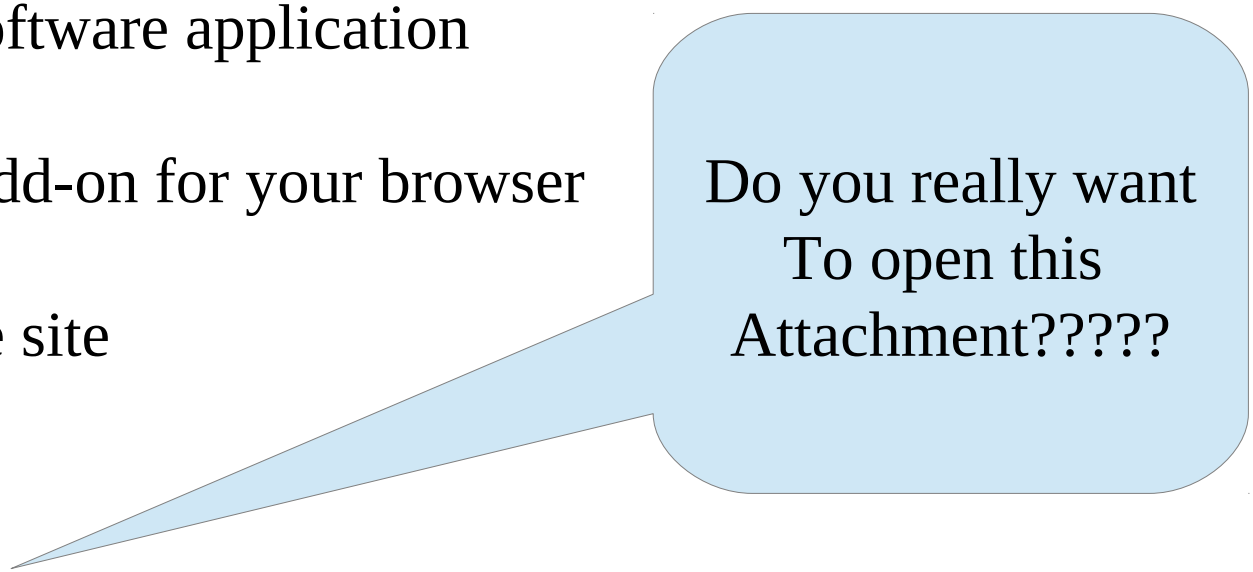
Get /file/path/to/something/../../../some_critical_resource

Opening some mail attachment of any format

Installing some software application

Selecting a new add-on for your browser

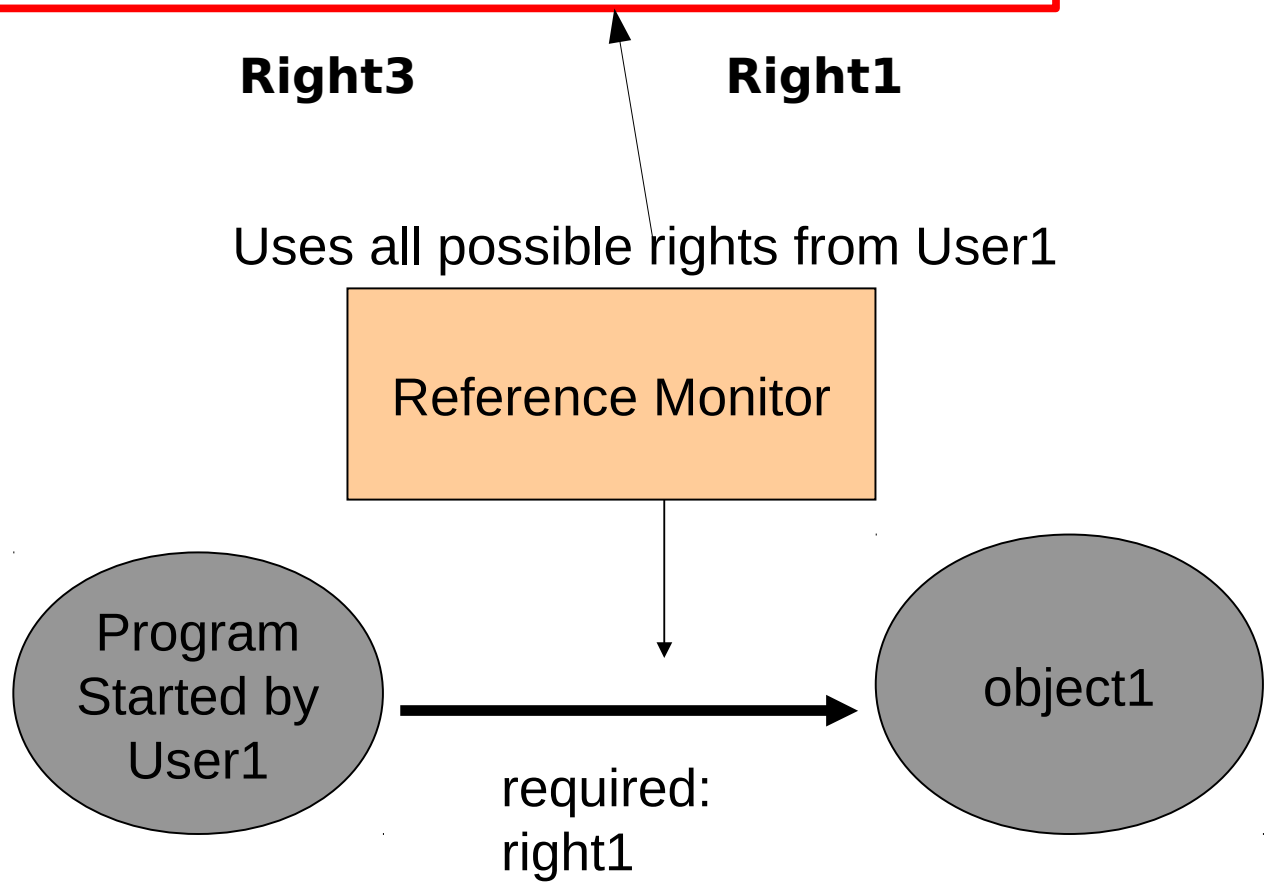
Browsing to some site



Root Causes: Ambient Authority Everywhere!

Access Control Matrix:

	Object1	Object2
Static Rights		
User1	**Right1**	**Right2**
User2	**Right3**	**Right1**



Root Causes: Designation vs. Authority

Open (char* filename, int mode)

// application needs to transform the symbolic filename into a ressource

Open (Filedescriptor fd)

// application receives an open resource without the need to perform any rights-related operations

An API like this forces the transfer of all authority from the user to the application because it is unclear what file will be opened at runtime. This is even more dangerous, if the application is privileged. Wrong arguments checking can lead to privilege elevation. The second API does NOT require ambient authority!

Root Causes: Software Architecture

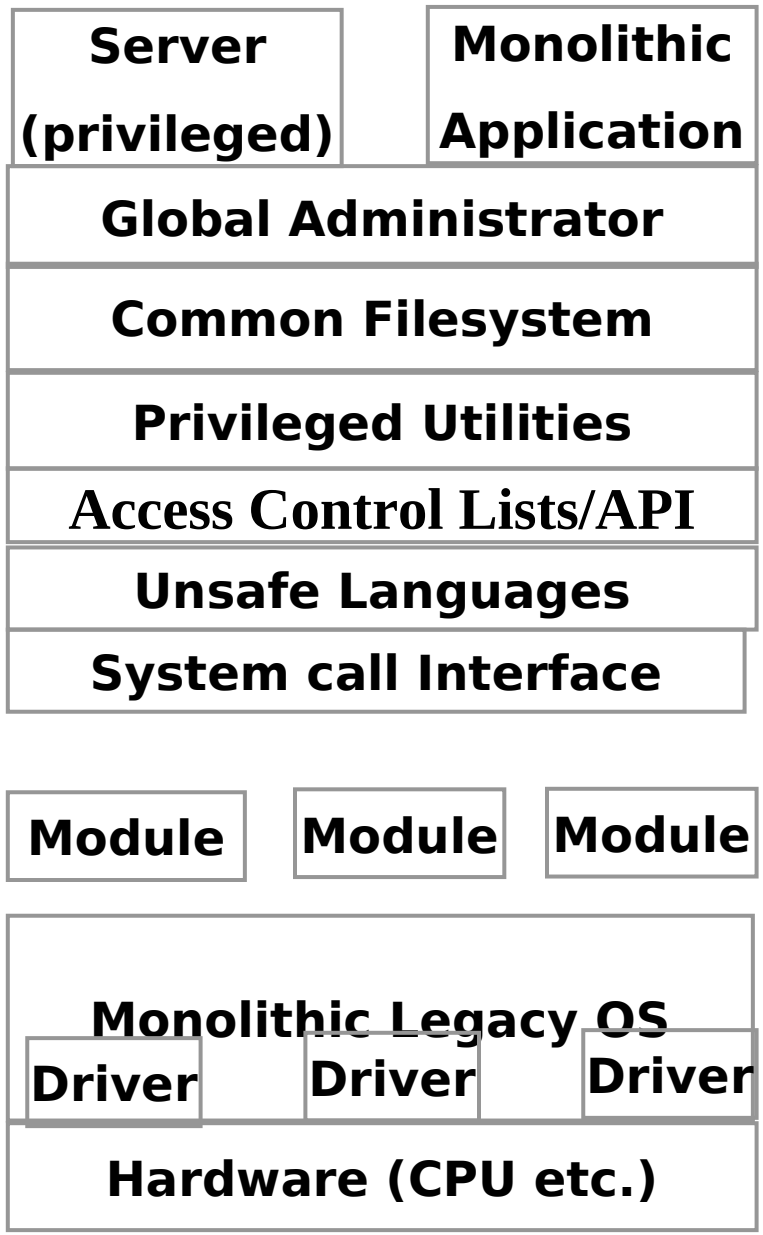
navigable filesystem with ambient authority

Tons of unsafe but privileged scripts and utilities (setUid)

>300 complex system calls always available

Unsafe extensions

>100.000 drivers for windows



Lots of unverified system libraries with memory leaks etc.

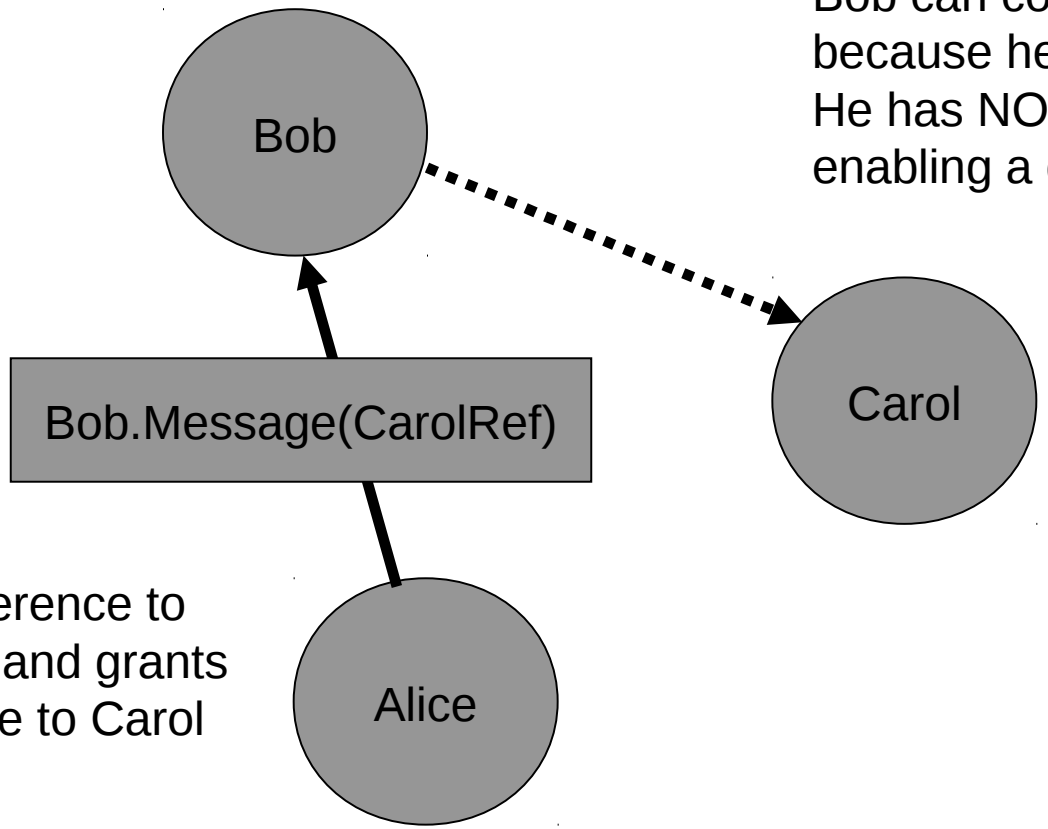
Incomplete quota administration (liveness problems)

Unsafe languages (memory safety)

Covered channels (cache, bios, CPU)

Huge TCB, 2 modes only

Solutions: Object Capabilities

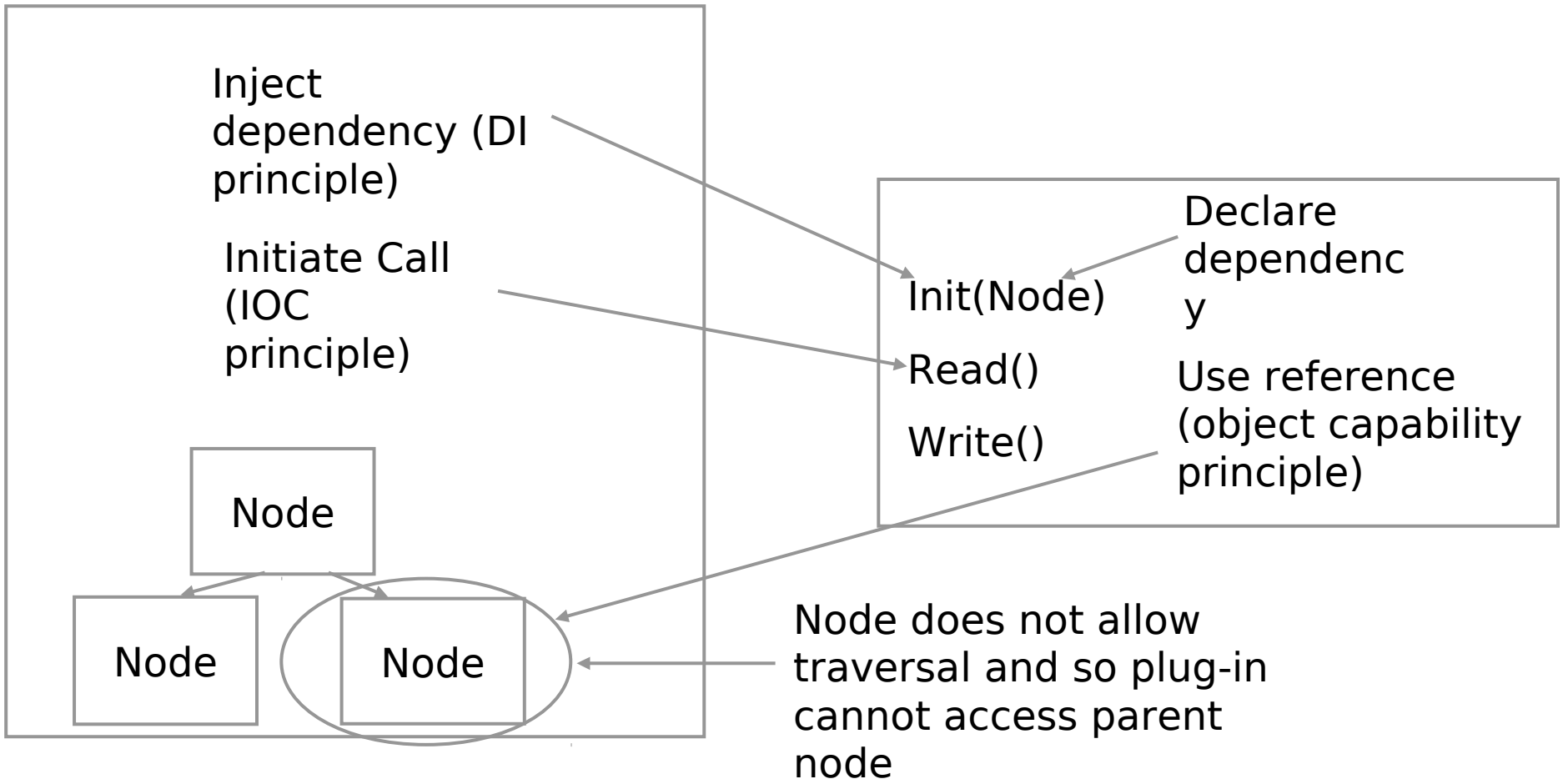


Bob can communicate with Carol because he got a reference to her. He has NO ambient authority enabling a call to Carol

Alice has a reference to Bob and Carol and grants Bob a reference to Carol

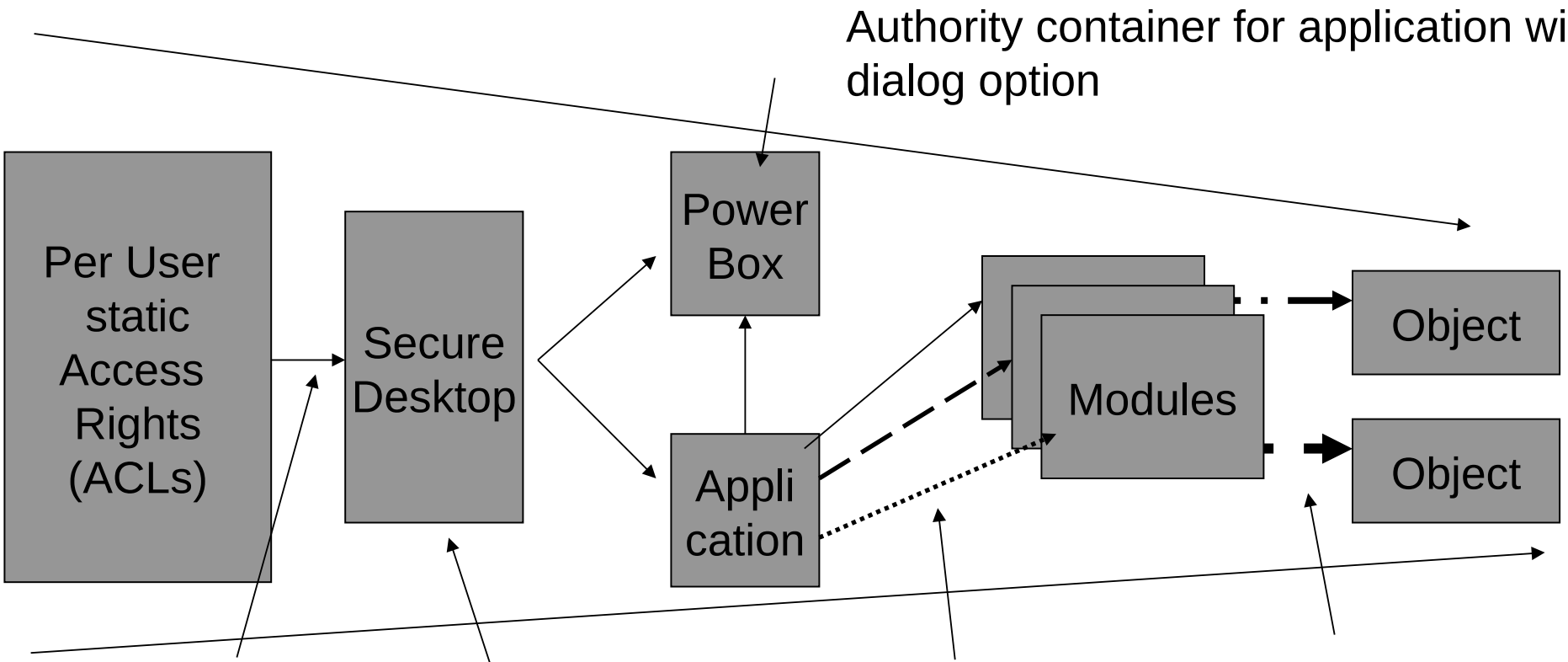
Object Capabilities reduce authority in a system: no access without a reference. And references combine access right and access method (designation and authority). They are a superior way to CONSTRAIN effects and are easier to analyze than external permissions. The diagram is called „Granovetter-Diagram“ after the well known sociologist Granovetter).

Solution: Safe Extensions by Inversion of Control



How do we make extensions safe? How do we achieve complicated business requirements like multi-tenant abilities? The answer is in Inversion-Of-Control architectures combined with strict control over references (no global crap for „flexibility“ reasons…) which effectively virtualizes the plug-in runtime environment

Solutions: Authority Reduction Architecture



Designation of object and action by User

(trusted path + authority by designation)

Transformation of names to capabilities and creation of powerbox per application

Granular distribution of authority (capabilities) to program modules

Granular delegation of authority to single objects

We need to narrow authority down from the global rights matrix (ACLs or Access Control Matrix) of a users rights to the minimum authority necessary to exectute a function. Test: try to find how many rights you REALLY need to copy a file!

Resources

1. **Secure languages and Systems: www.erights.org**
2. **Robust Composition: Mark Miller Thesis, 2006 http://www.erights.org/talks/thesis/index.html**
3. **Darpa Browser Architecture (www.combex.com)**
4. **Authority Reduction, Theoretical Foundations and Decidability: www.combex.com (powerbox Concept, secure desktop etc.**
5. **Safety Analysis: Fred Spiessens, Peter Van Roy, A Practical Formal Model for Safety Analysis in Capability Based Systems**
6. **Kriha/Schmitz, Sichere Systeme, www.kriha.de**
7. **Cap-talk mailing list**