

# Security in Distributed Systems

## Part two

Architectures, Software Security Frameworks and  
Infrastructure

Prof. Walter Kriha  
Computer Science and Media  
Stuttgart Media University

# Overview

- Security Architectures and Infrastructure
- Software Security
- Anonymity and distributed Trust

# Security Infrastructure

- End-to-end security and secure delegation
- Backend Security
- Firewalls and Demilitarized zones
- Single-Sign-On
- Identity Repositories
- Federated Security
- DMZ
- Reverse Proxy

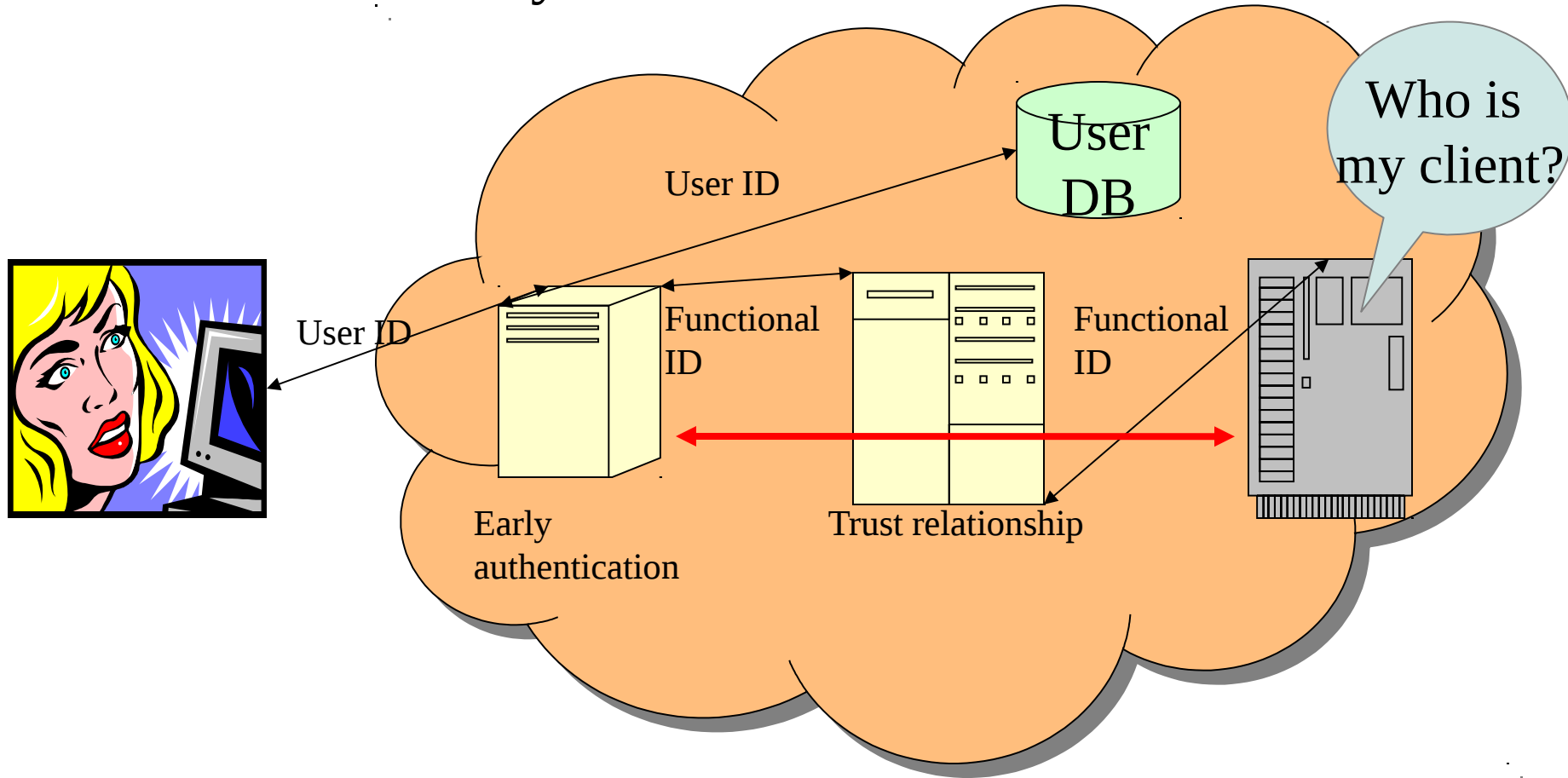
# End-To-End Security

## Ingredients for a secure Infrastructure

- **Identity and Access Management (User provisioning, self-service, central authority)**
- **Single-Sign-On Service (with federation and delegation)**
- **Traced Delegation with forward looking trust across infrastructure**
- **Propagation of rights**
- **Descriptive specification of access rules**
- **Multi-level security?**

**The security design patterns needed are: Trusted Third Party, Token, Signatures and mutual Authentication.**

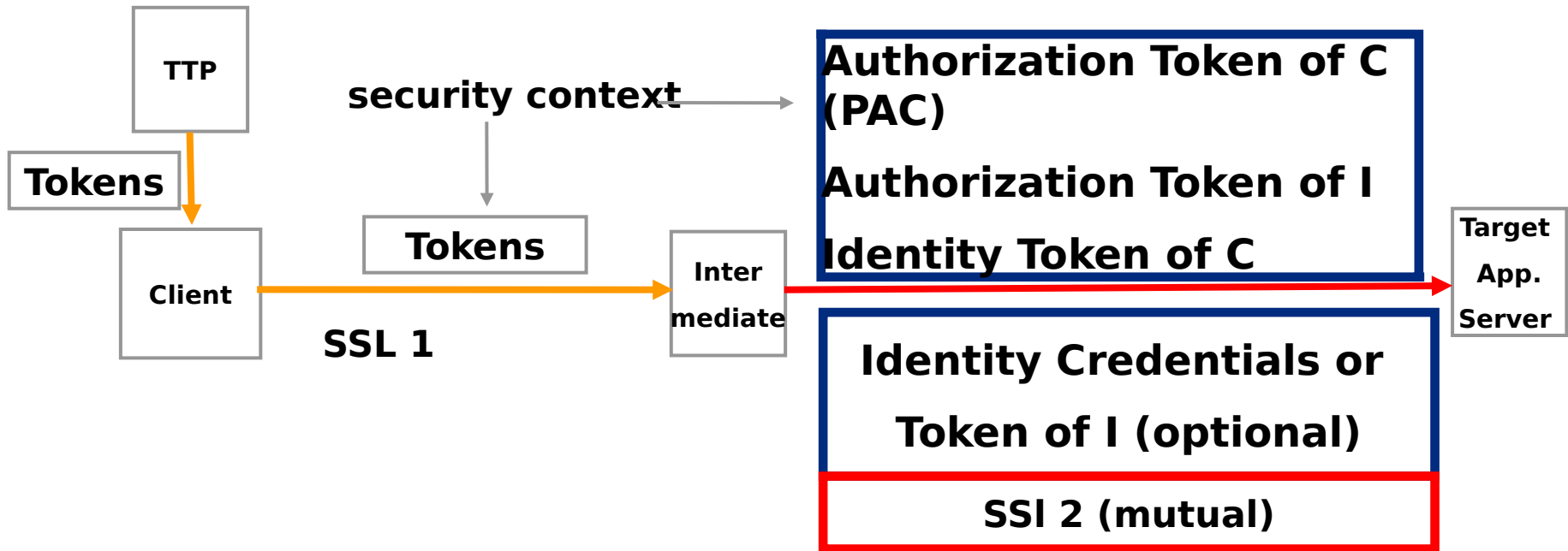
# Backend-Security: Where to authenticate/authorize?



The advantage of early authentication is that un-authenticated calls do not get deep into your network. The disadvantage is that now a trust relationship is established between the front end machines at the edges of your network and the backend processing and databases. This topology does not easily support organizational changes as well. An alternative would be secure delegation.

# Secure Delegation

## CORBA CSIV2 Mechanism



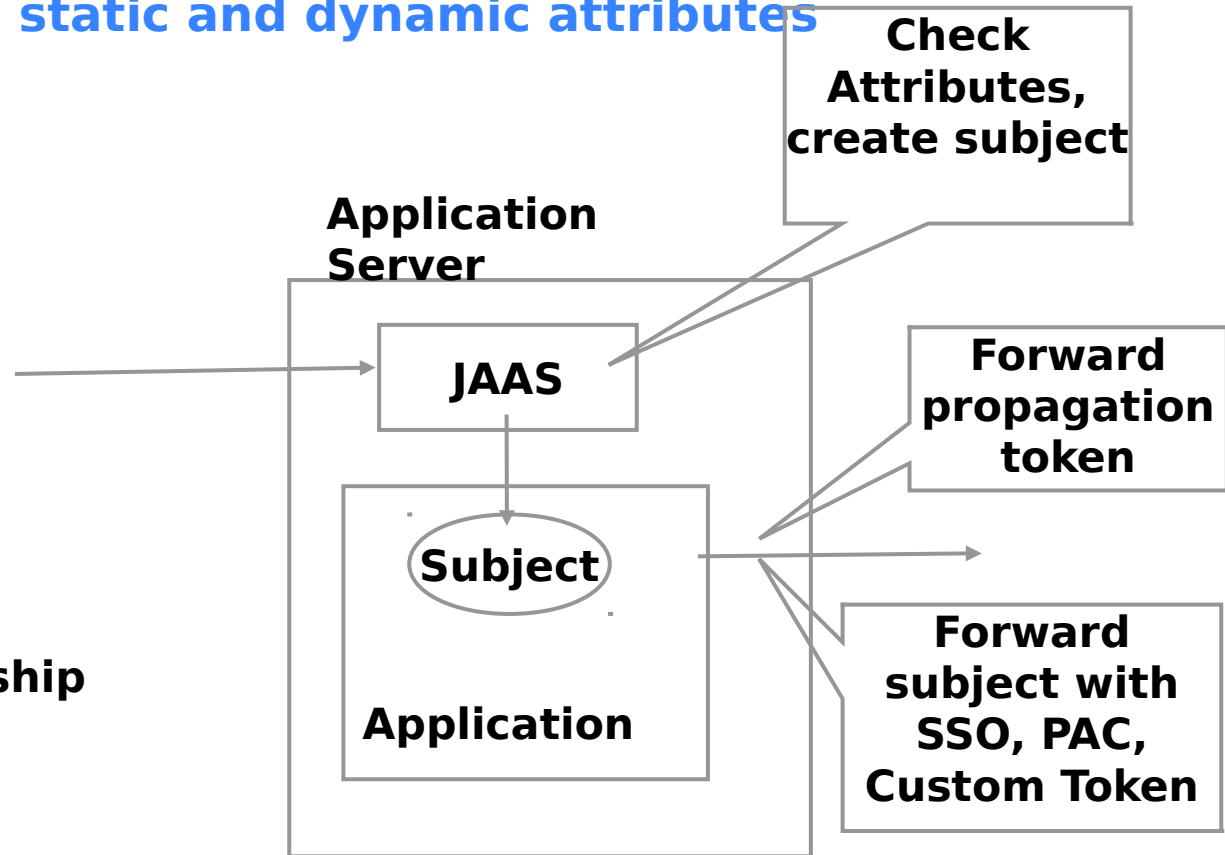
In a fully implemented delegation system no initial authentication credentials flow from clients to targets. Intermediates and targets authenticate each other and then transmit signed authorizations and identity tokens. SSL channels and private keys are needed but the trick is in the authority attributes conveyed to target systems (semantics of delegation).

# Privilege Attribute Certificate

## Power of Attorney, static and dynamic attributes

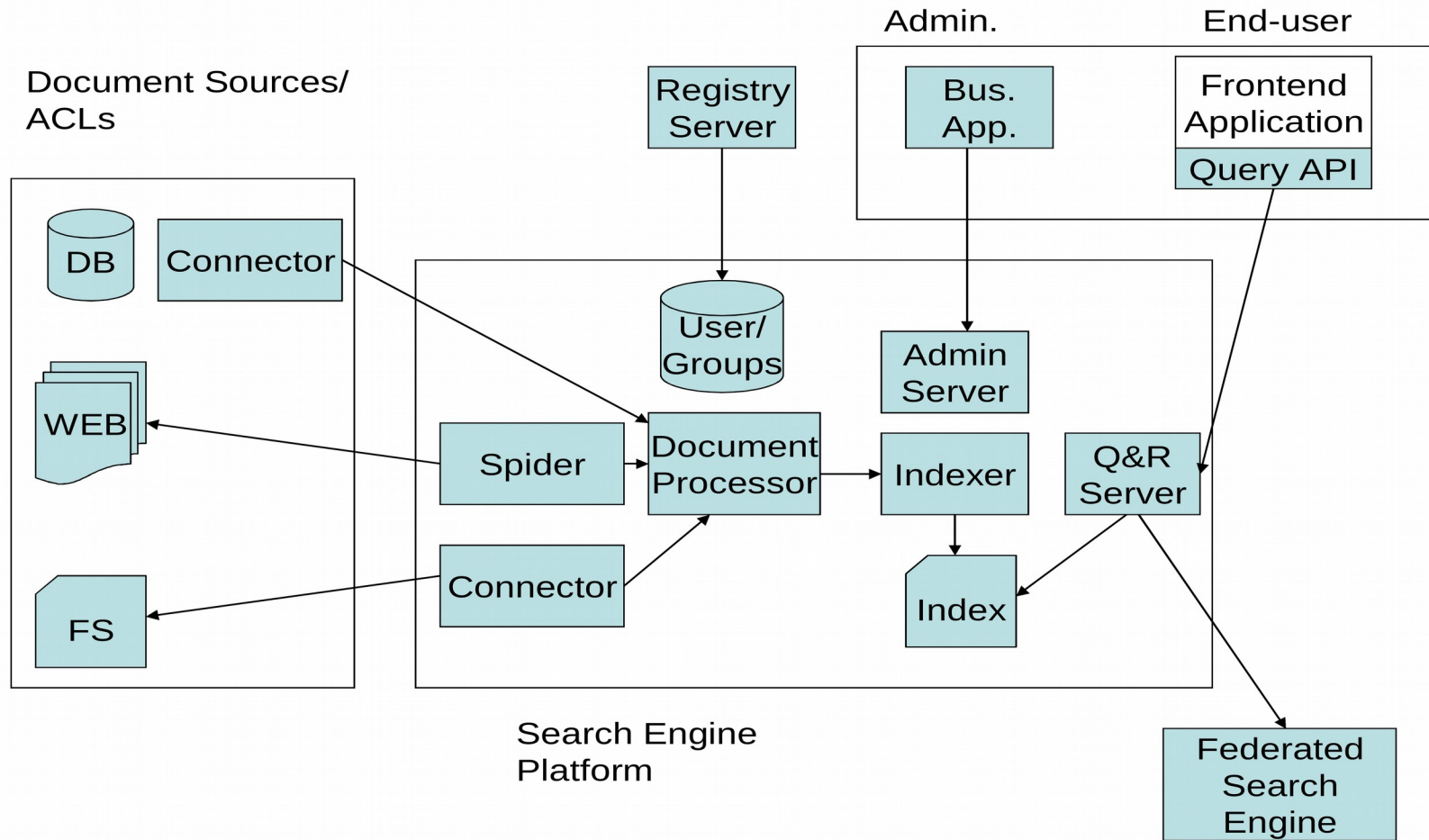
### PAC

- Login place and time
- Authentication quality
- Groups/Roles
- Power of attorney for intermediates
- Restrictions
- Business unit membership
- Credentials
- etc.



**Step-Up authentication e.g. uses the authentication quality for access decisions. But what if some software components do not check for this? WAS 6 has 4 token types: SSO, PAC, Custom, Propagation (on thread)**

# Backend-Security and Secure Delegation Example





# End-To-End Security cont'd

## Things to remember

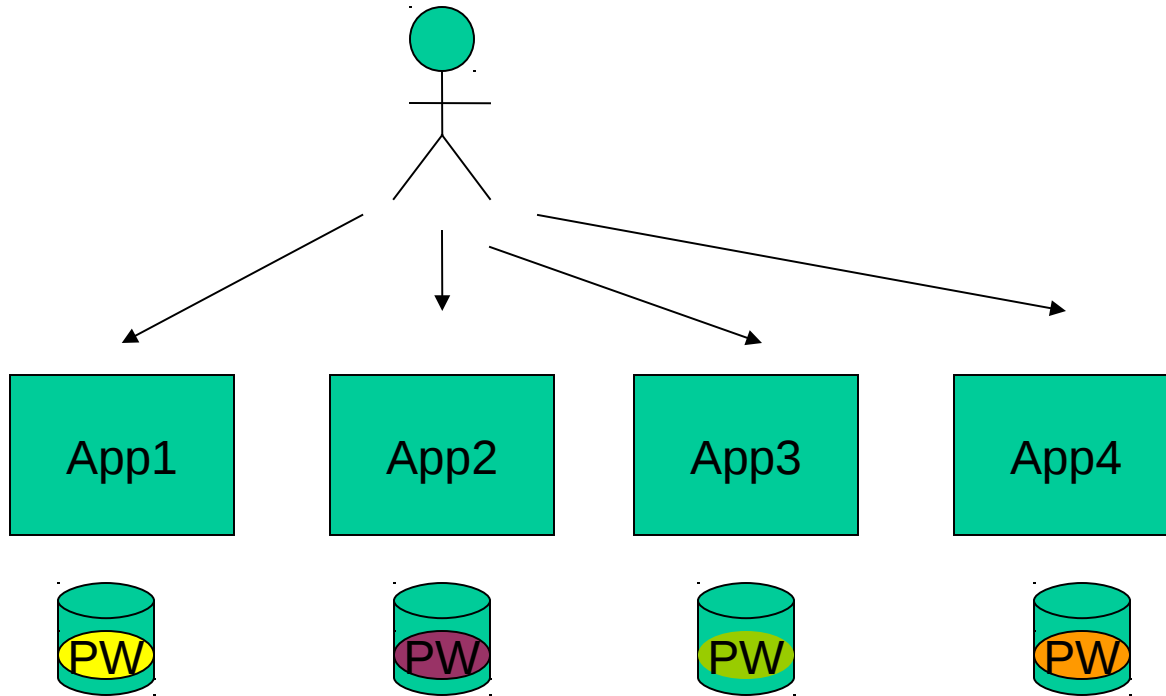
- **When requests flow across systems, passwords and symmetric keys have little value. Only signed statements (called certificates, tokens etc.) count when the receiver trusts the signer**
- **Complex business arrangements need a rich language to express the relations and constraints. Keys and channels are NOT enough. We need the Secure Association Markup Language (SAML)**
- **The concepts of delegation and impersonation are wrong in most books and specifications. Compare them with social contracts (e.g. power of attorney, shared PIN) to get it right.**
- **Cross-domain federation is still hard. It requires lots of mappings.**

# SSO Scenarios

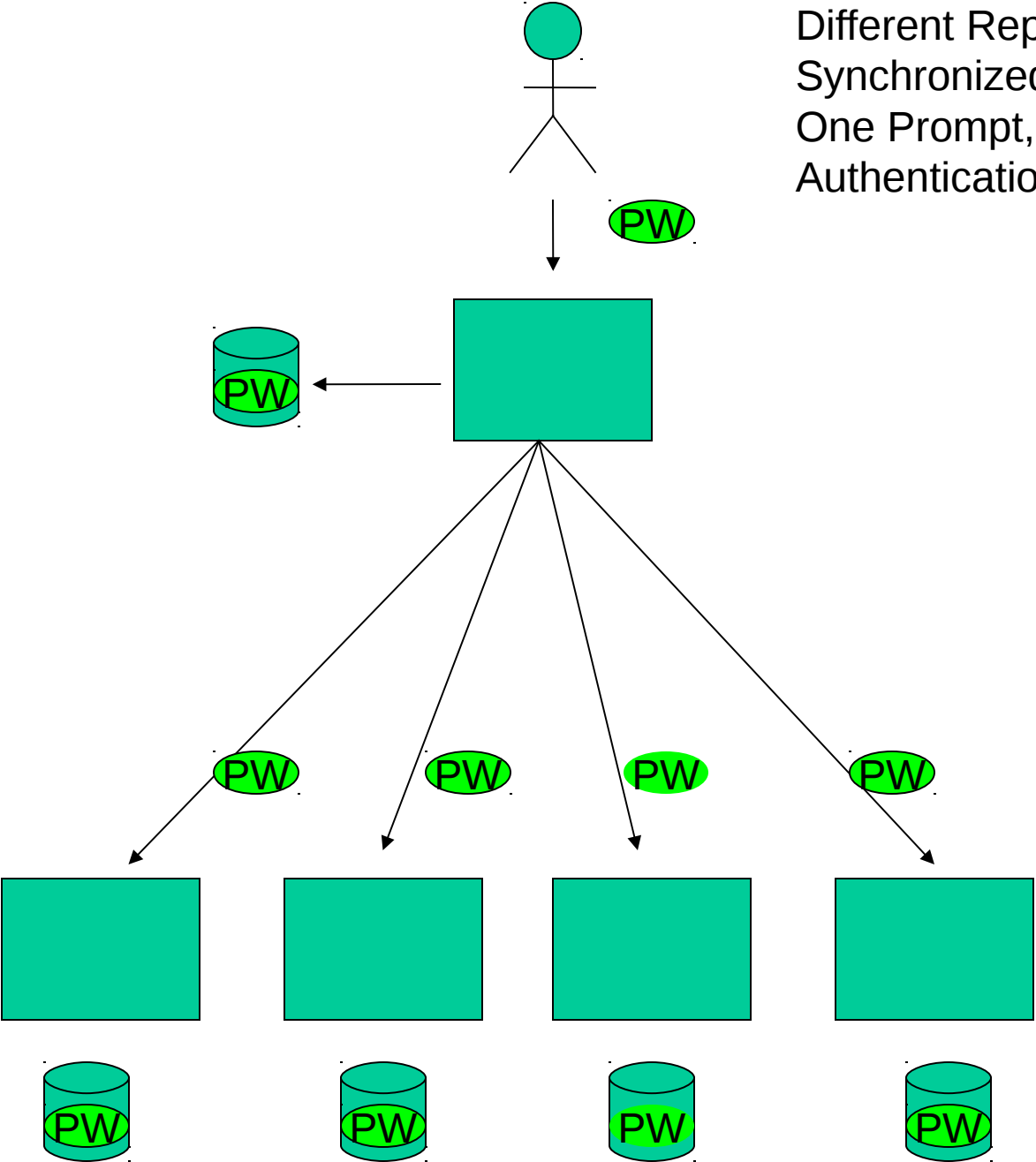
Which SSO do you mean???

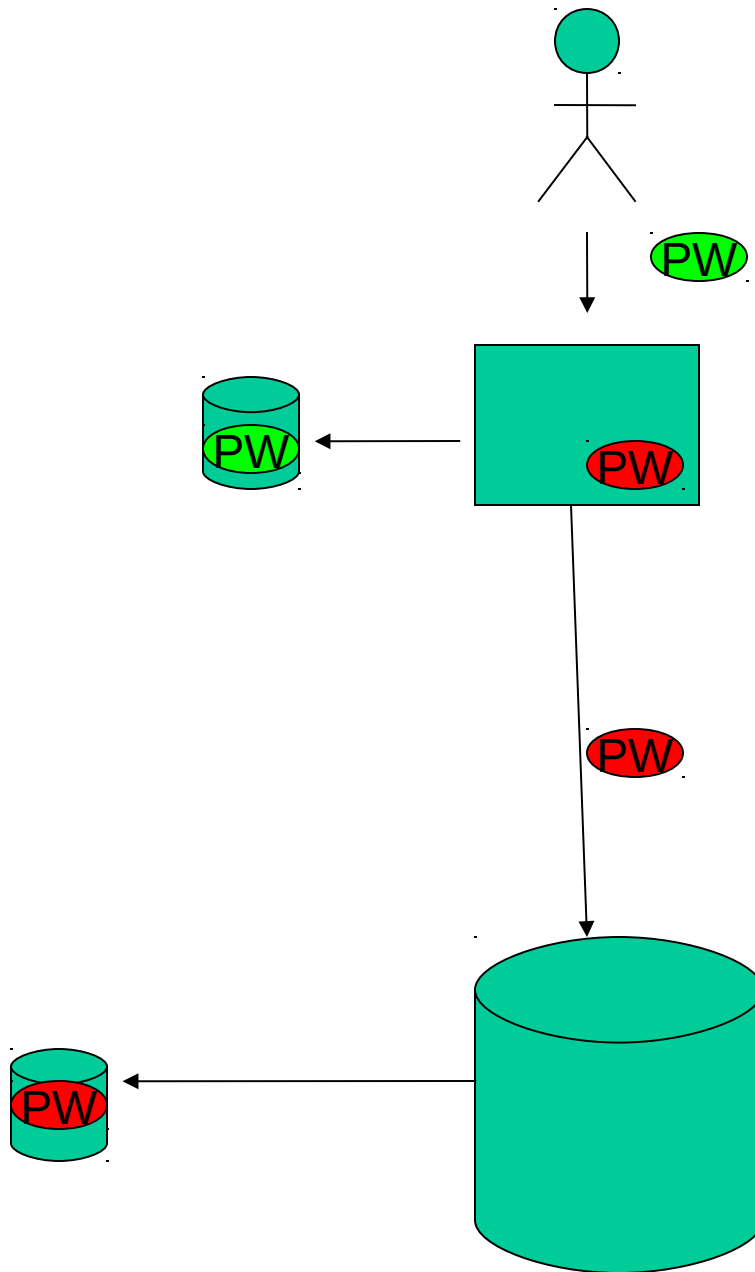
No SSO

Different Repositories,  
Different Passwords  
Many Prompts



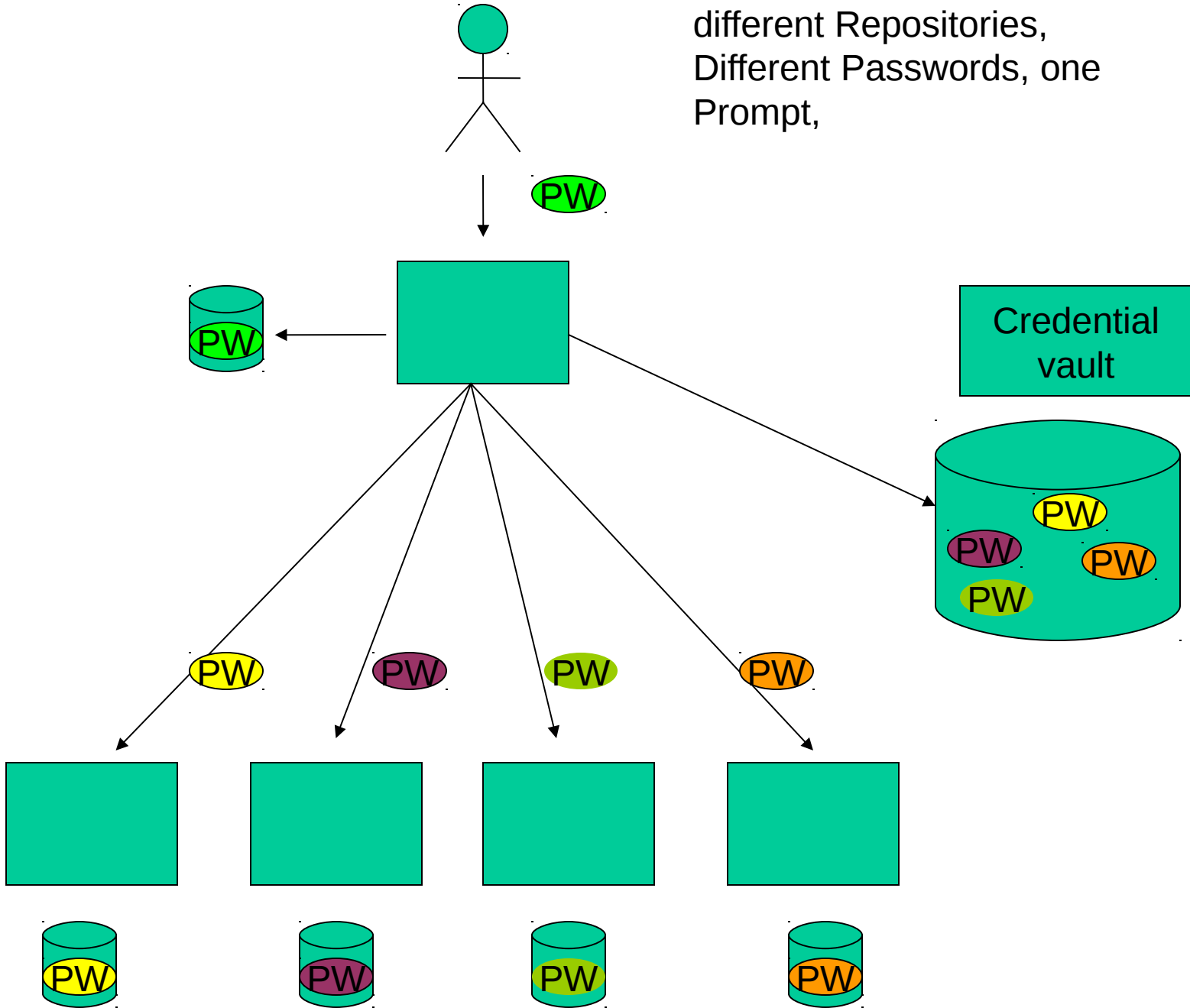
Different Repositories, User  
Synchronized Passwords  
One Prompt, Delegation of  
Authentication Credentials



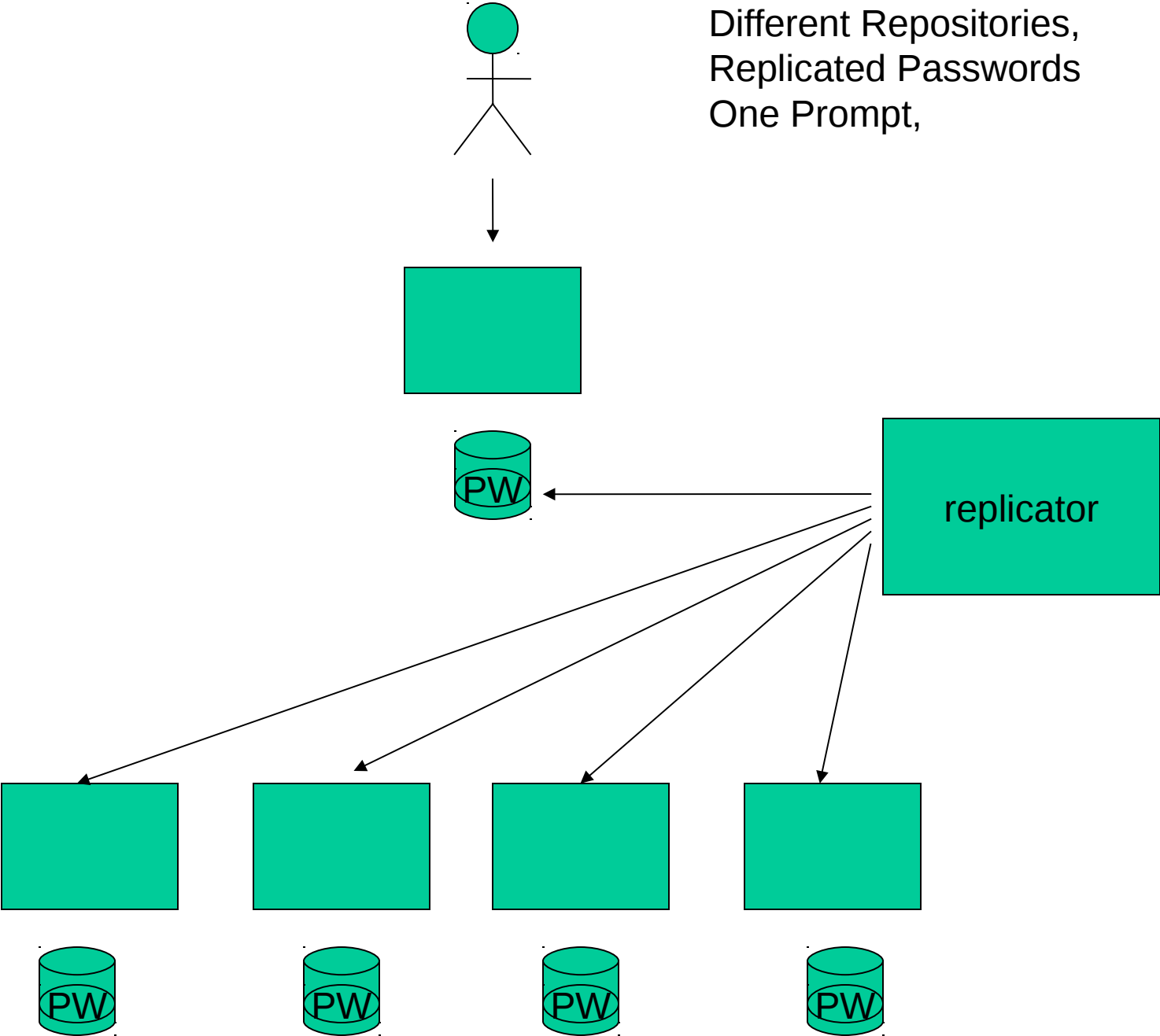


Different Repositories,  
One User Password  
One Prompt,  
Use of Functional User ID  
with fixed password

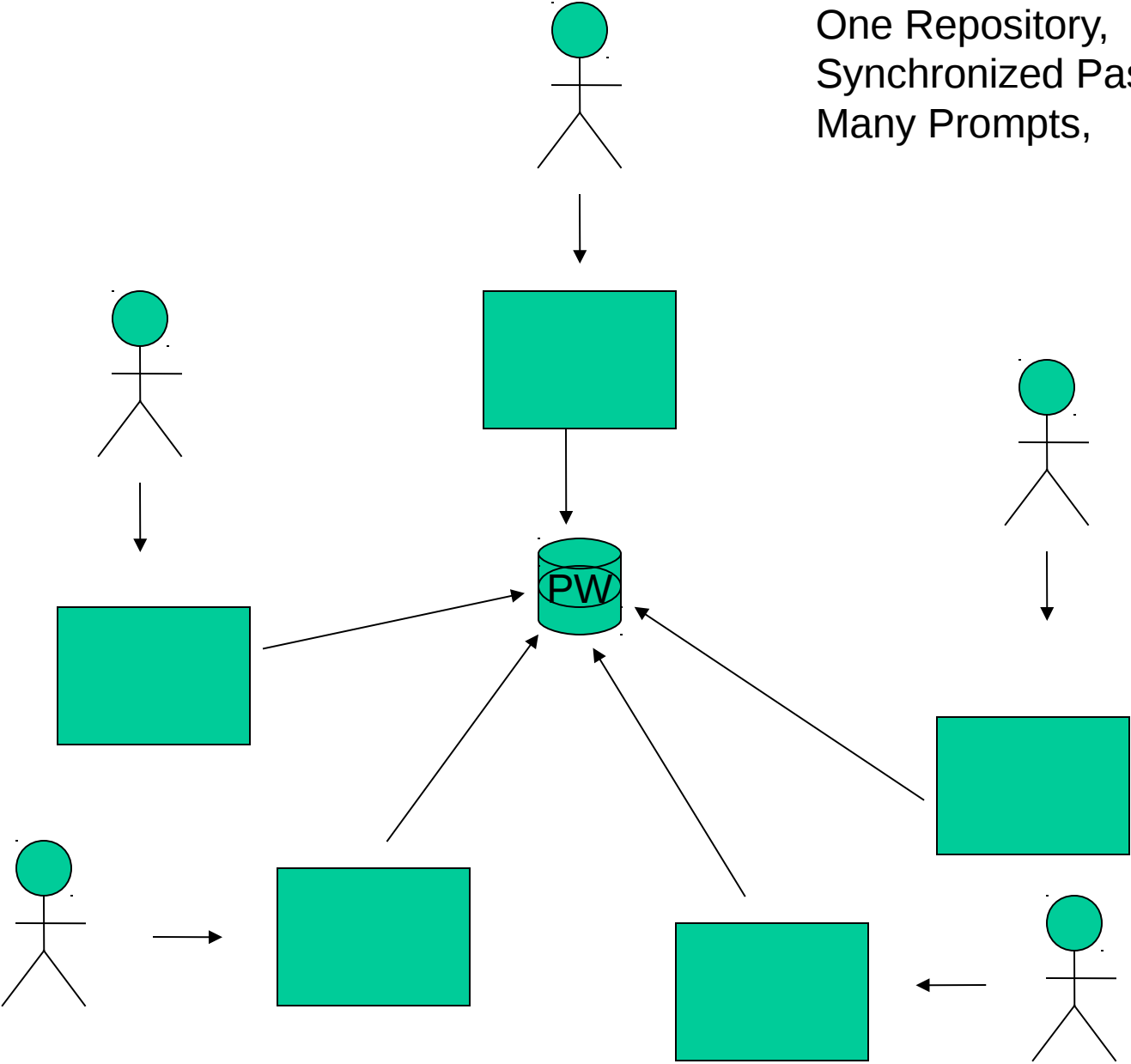
different Repositories,  
Different Passwords, one  
Prompt,



Different Repositories,  
Replicated Passwords  
One Prompt,

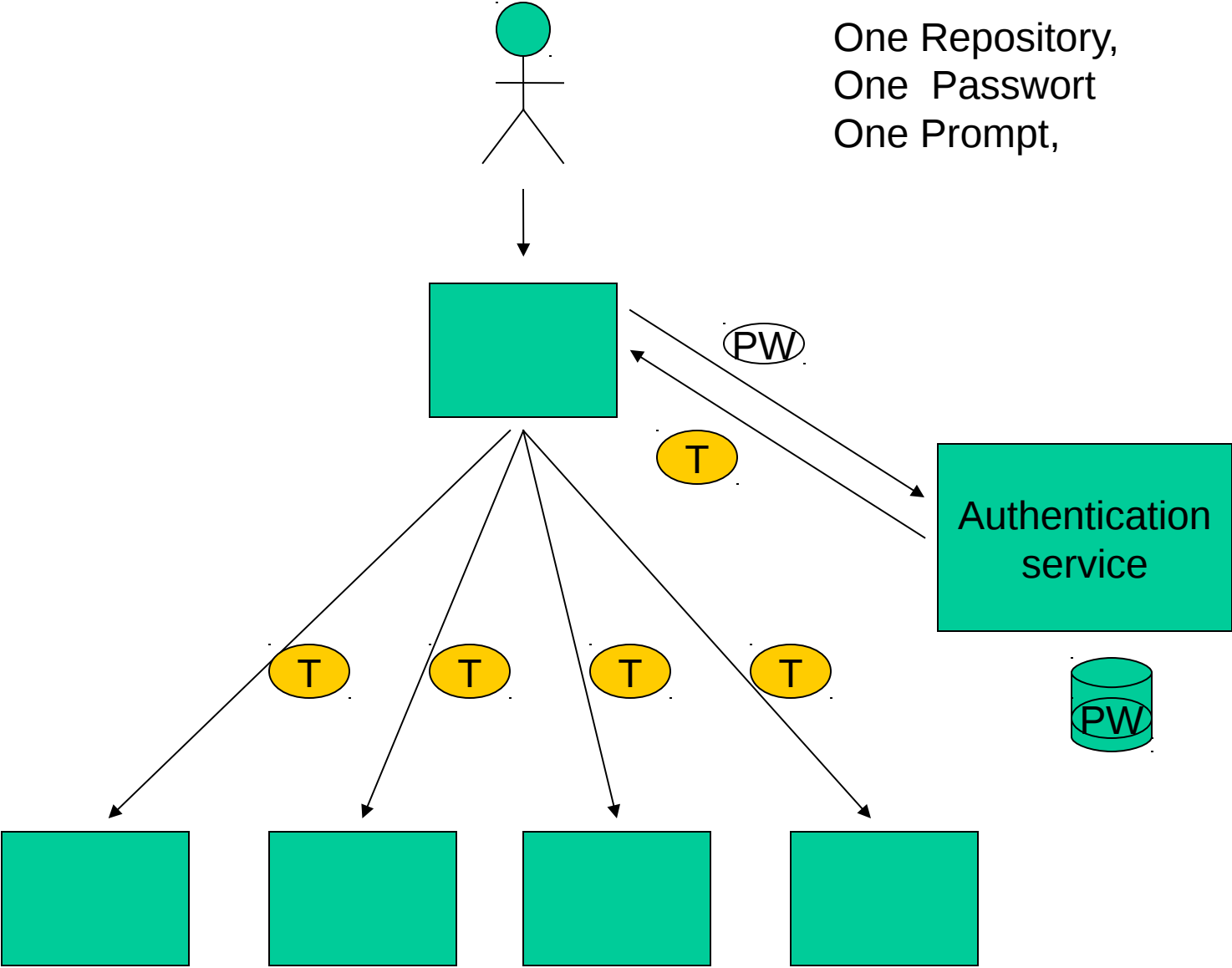



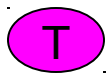

One Repository,  
Synchronized Password  
Many Prompts,



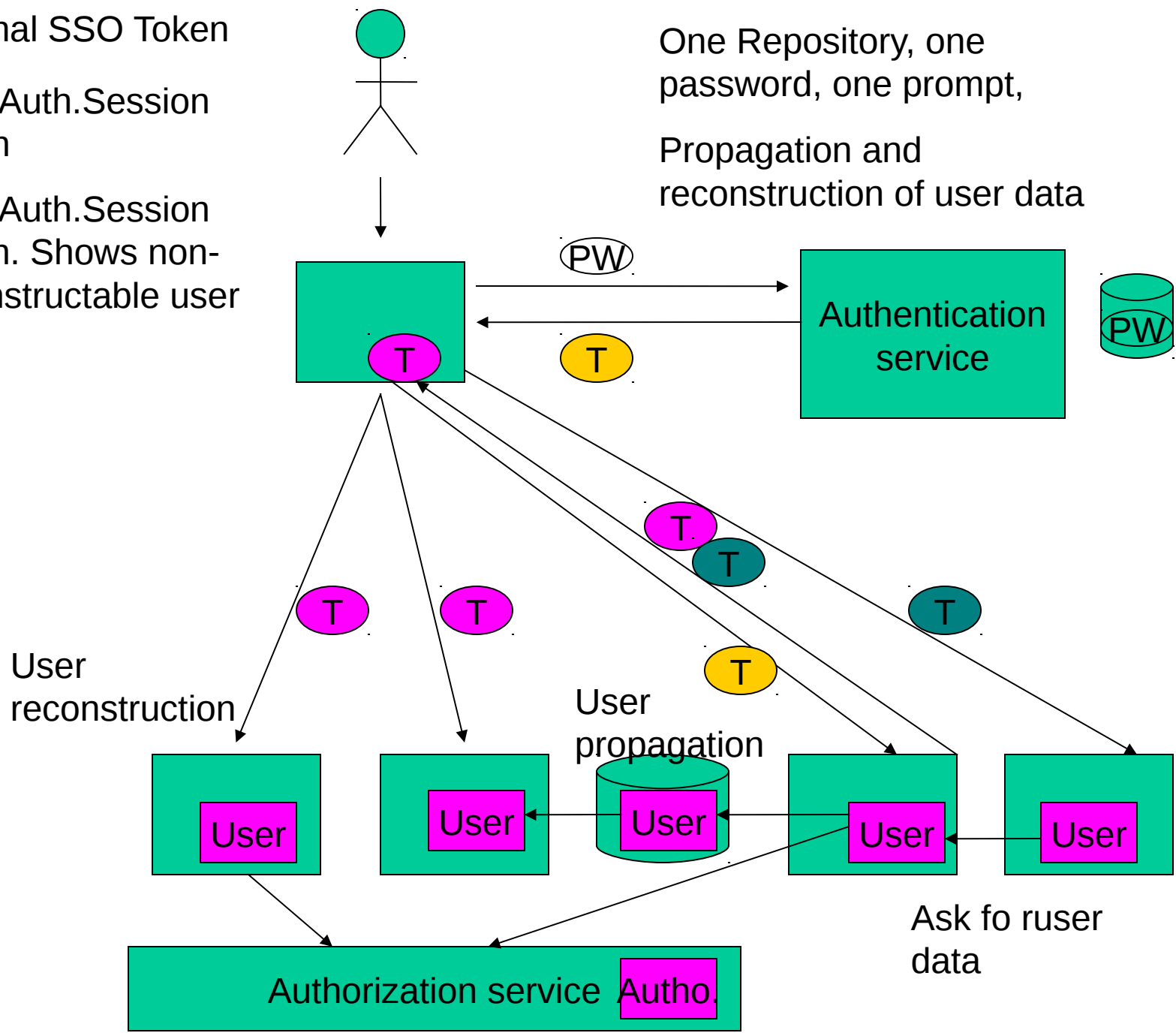


One Repository,  
One Password  
One Prompt,

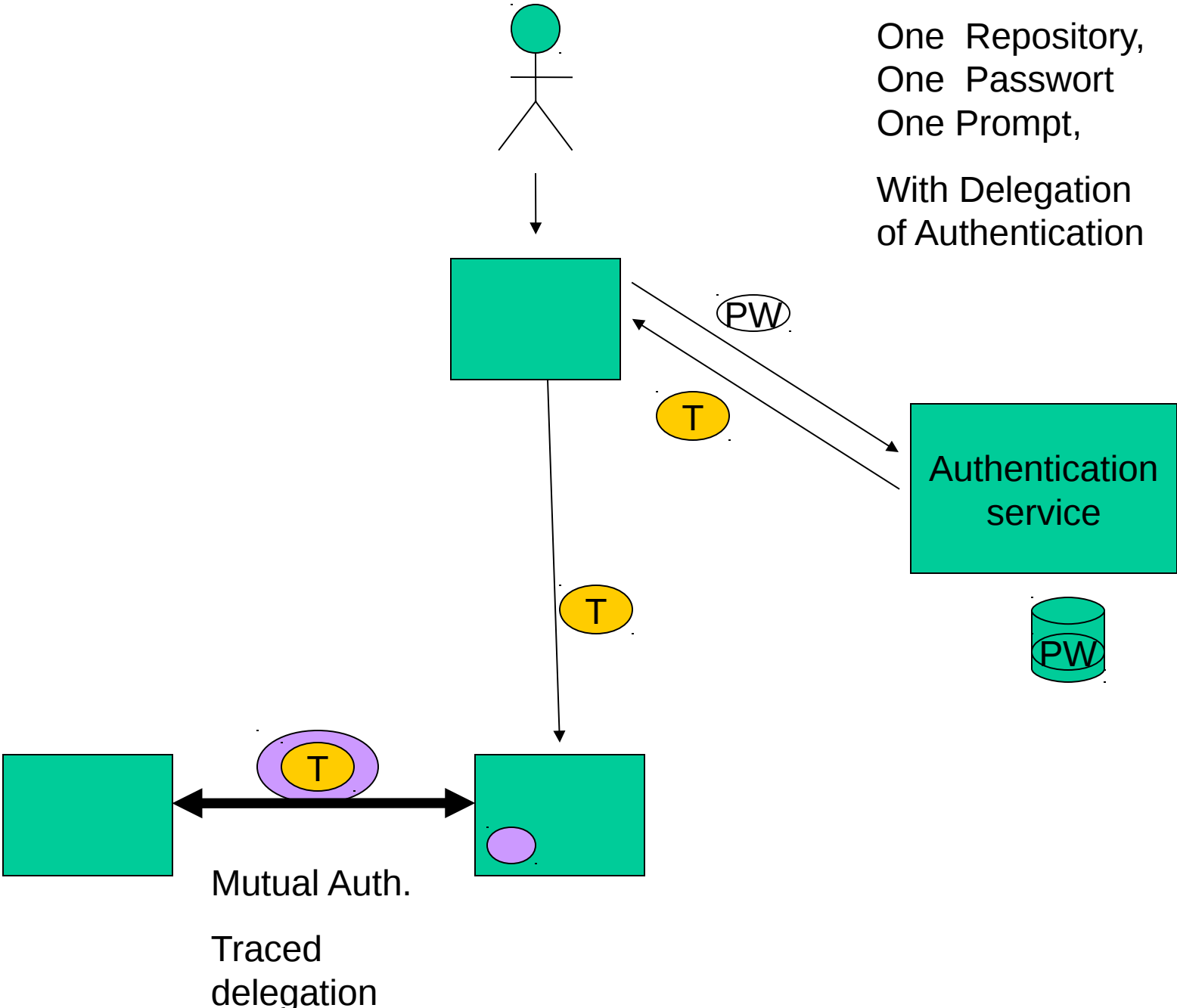


-  Original SSO Token
-  User Auth. Session Token
-  User Auth. Session Token. Shows non-reconstructable user data

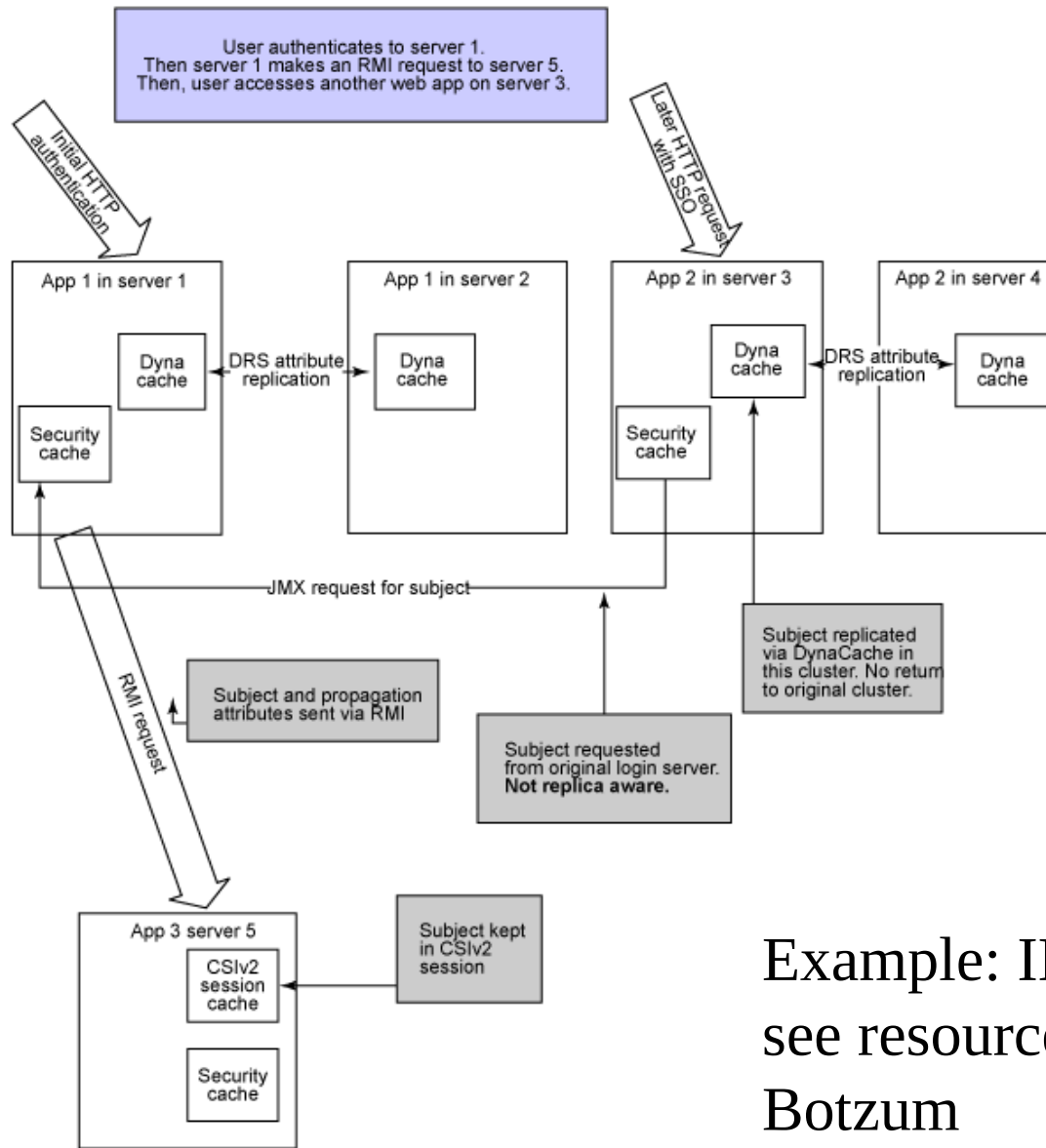
One Repository, one password, one prompt, Propagation and reconstruction of user data



One Repository,  
One Password  
One Prompt,  
With Delegation  
of Authentication



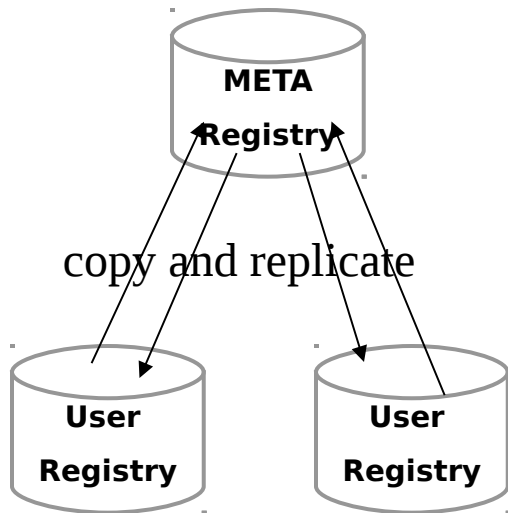
# Identity Propagation and Interfaces



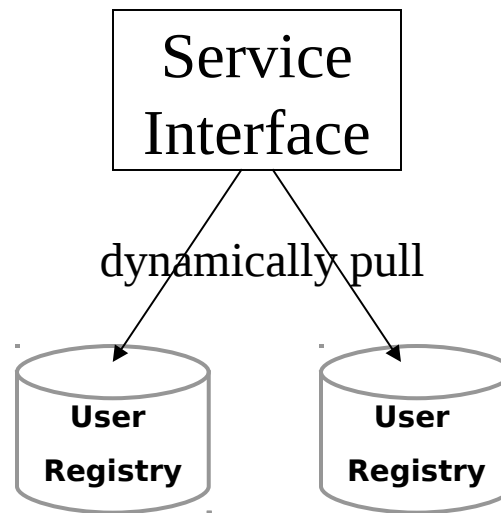
Example: IBM WAS,  
see resources:Keys  
Botzum

# Identity Repositories (Directories)

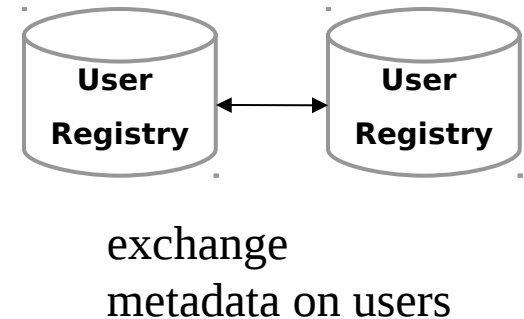
## Meta Directory



## Virtual Directory

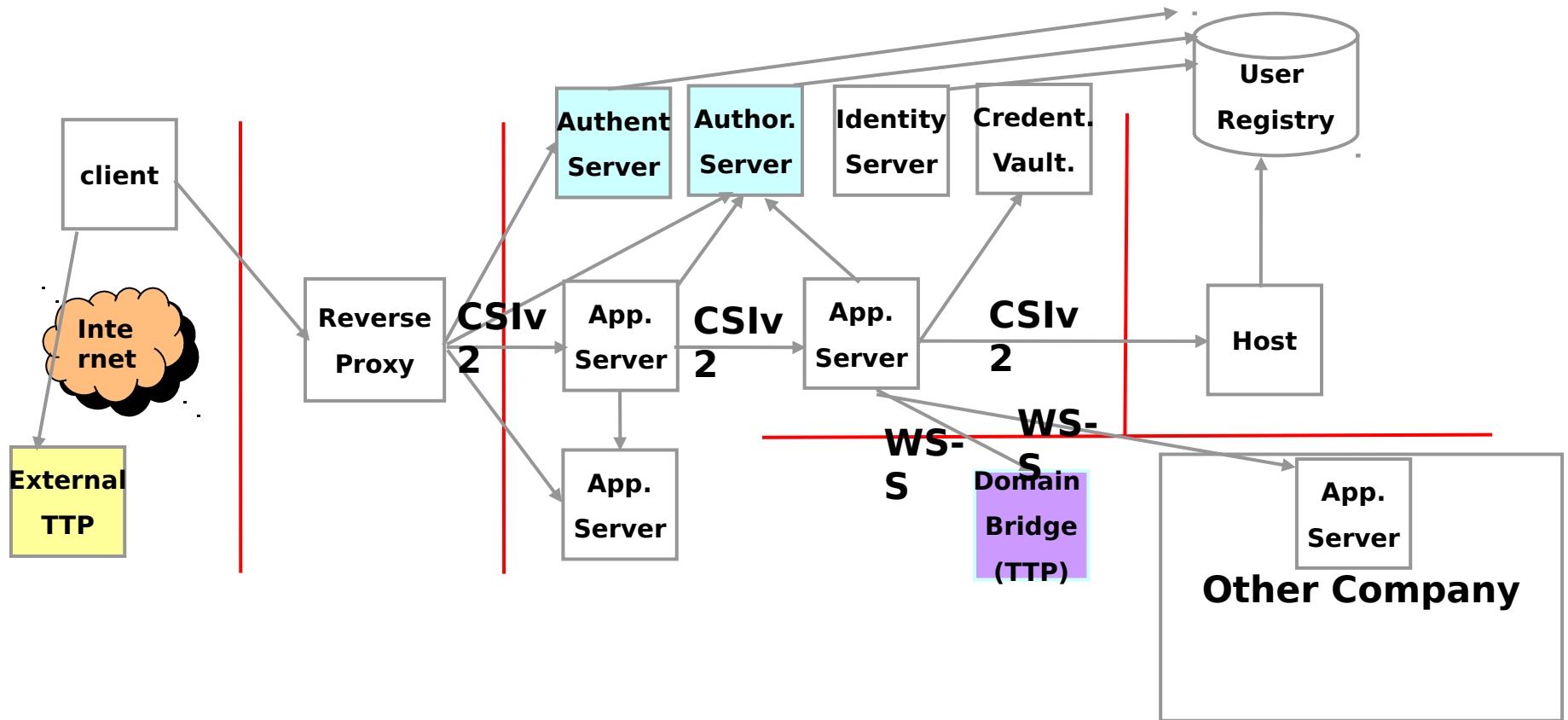


## Federated Identities



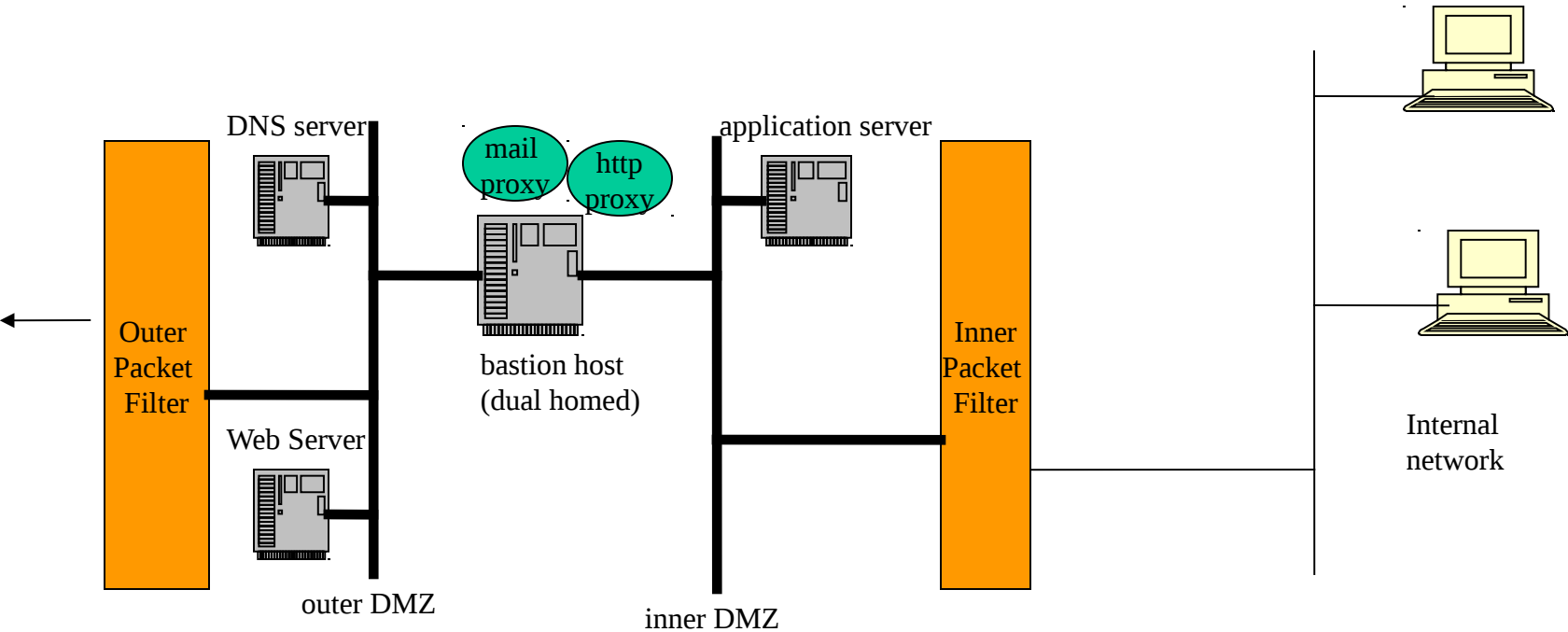
**Meta directories and virtual directories are riddled with typical distribution problems: consistency problems (who updates?), performance problems (how long to extract values?), reliability etc. But directories also need common naming standards and owners hat to give up control. See Windley, Digital Identity Management). Federation leaves information in place but requires contracts**

# Federated Security: Trusted Third Parties



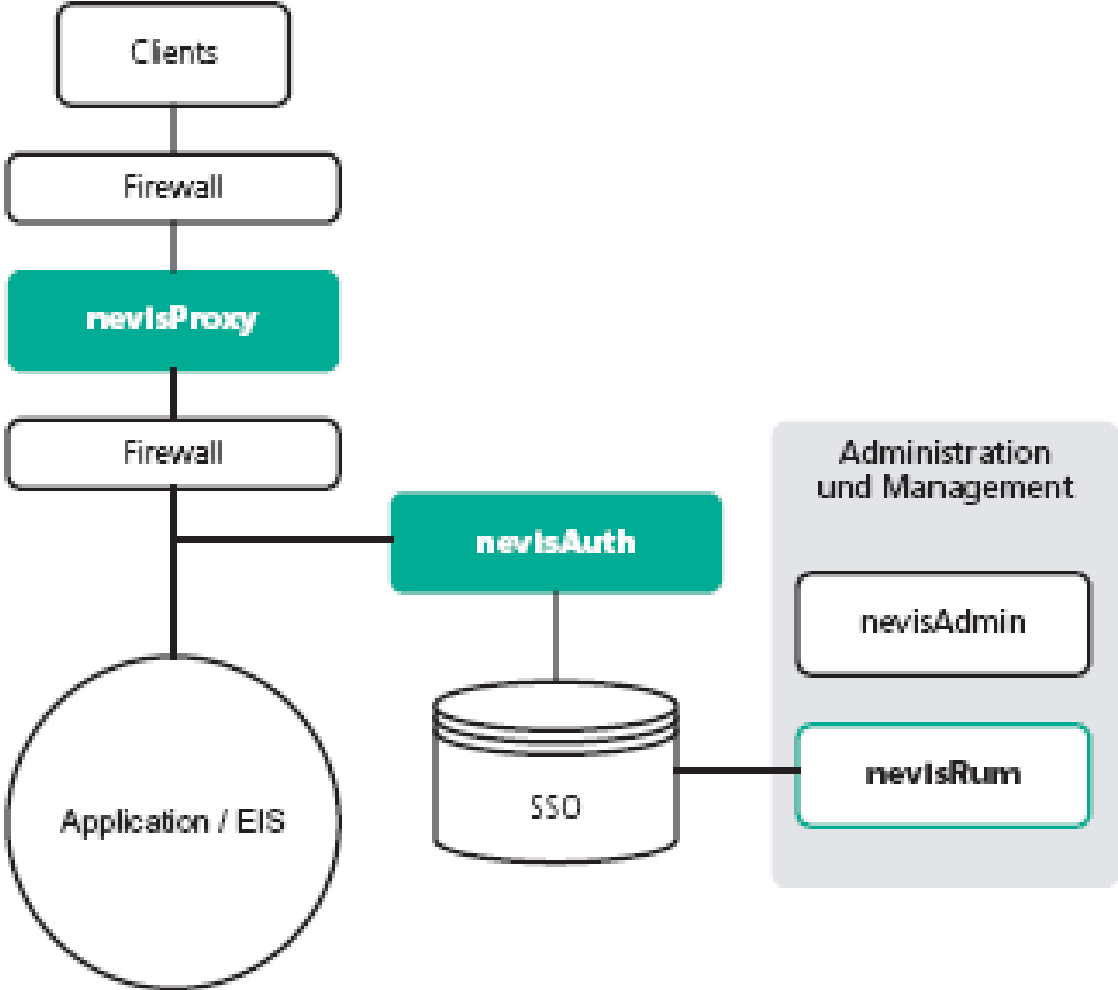
**Trusted Third Parties generate signed statements (tokens, certificates) which allow things, proof things etc. TTPs are useful to create federated domains as well. Theoretically the only place where client would produce her login credentials would be the first external TTP.**

# Demilitarized Zone(s) Architecture



Server that do not need a connection to the internal network run in the outer DMZ, still protected by one packet filter. More critical servers (e.g. application servers that need DB connections to the internal net) run in the inner DMZ. Protected by a dual homed bastion host that cleanly separates internal from external net.

# Reverse Proxy Architecture





# Security Frameworks

- Java Security Architecture
- Externalized authentication JAAS
- Subject and Principals, Impersonation
- Externalized access control, JAAS/EJB/JACL
- Logging and auditing framework
- PERMIS X.509 access control framework

# Java Language Access Control Architecture

Code based

User Based

Identity

Authority

Authority

Identity

Code location  
Or signature

Permissions

Permissions

Principals

Policy

Policy

Protection  
Domain

Caller  
Stack

Login  
Modules

Check  
Permission

Access  
Control

Subject with  
Principals

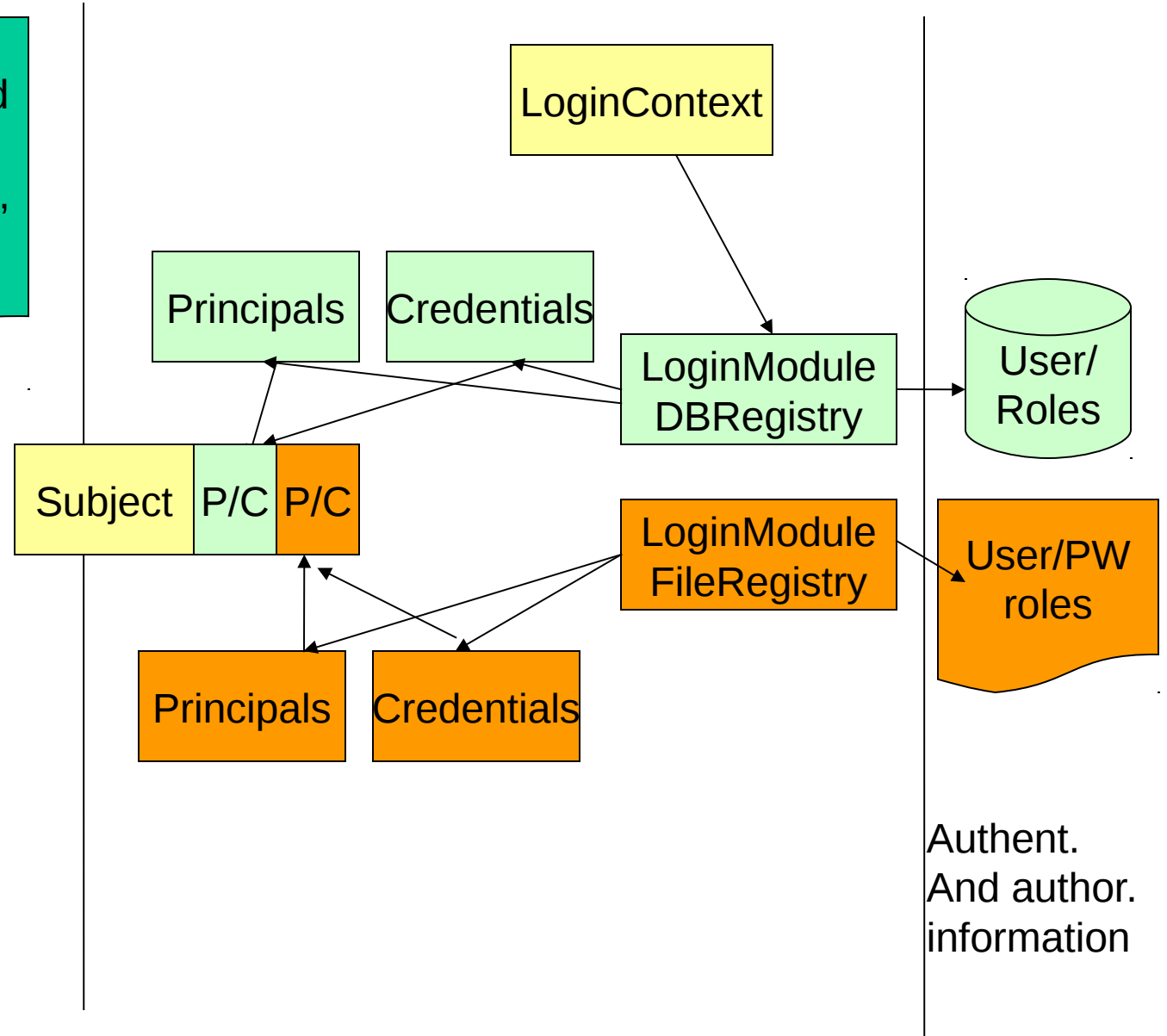
# JAAS Framework

```

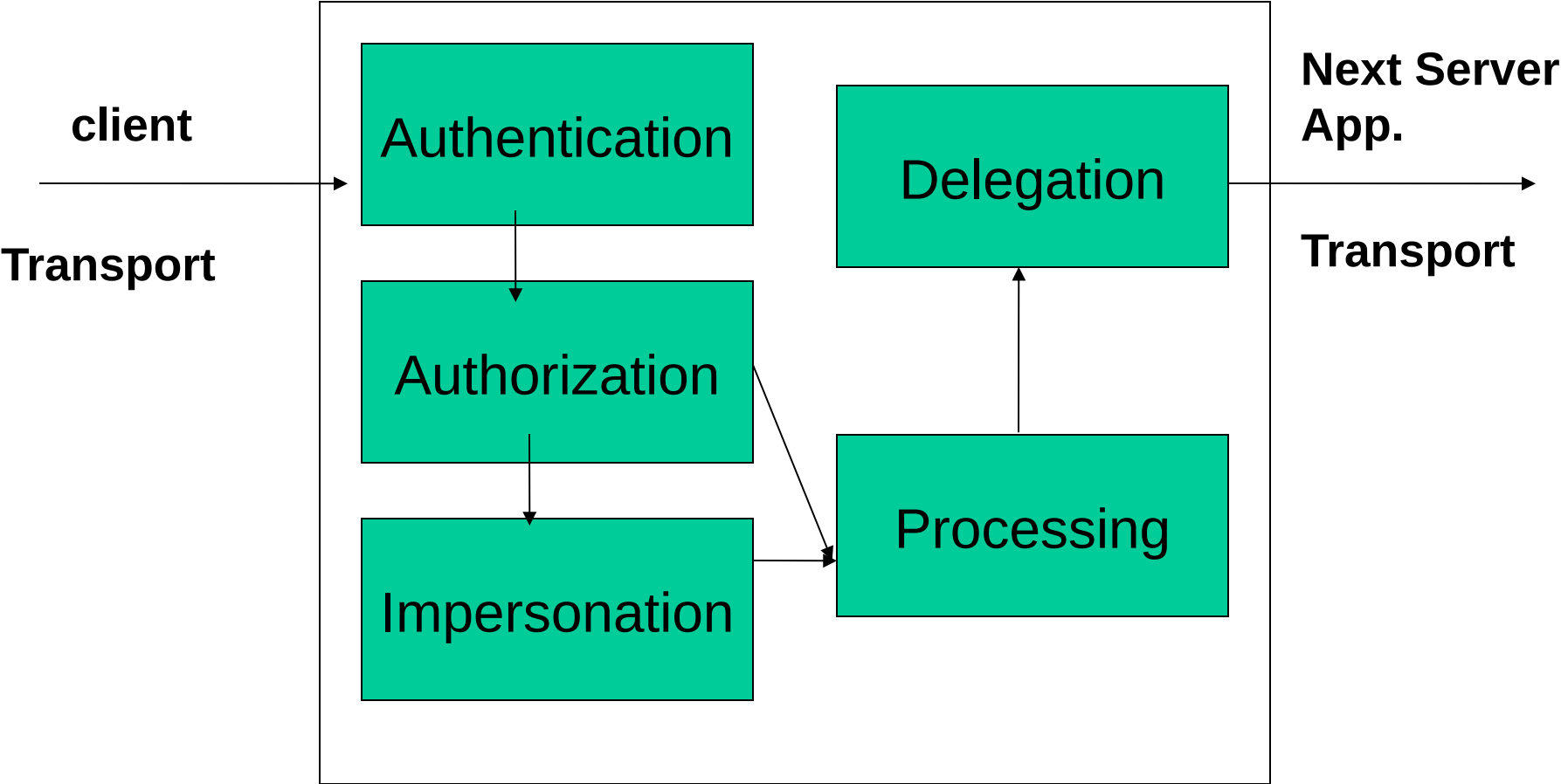
AppLogin {
FileRegistry, required
„pathToFile“
DBRegistry, required
„DBname
}

```

Login Config File  
with several  
LoginModules



# Server Application



```
Subject.doAs(subject, new PrivilegedAction()
```

```
{
```

```
    Public Object run()
```

```
    { // Subject is associated with the current thread
```

```
        java.security.AccessController.doPrivileged( new PrivilegedAction()
```

```
            { // NOTE: no Subject in interface!!
```

```
                public Object run()
```

```
                { // Subject cut off from the current thread
```

```
                    return null;
```

```
                }
```

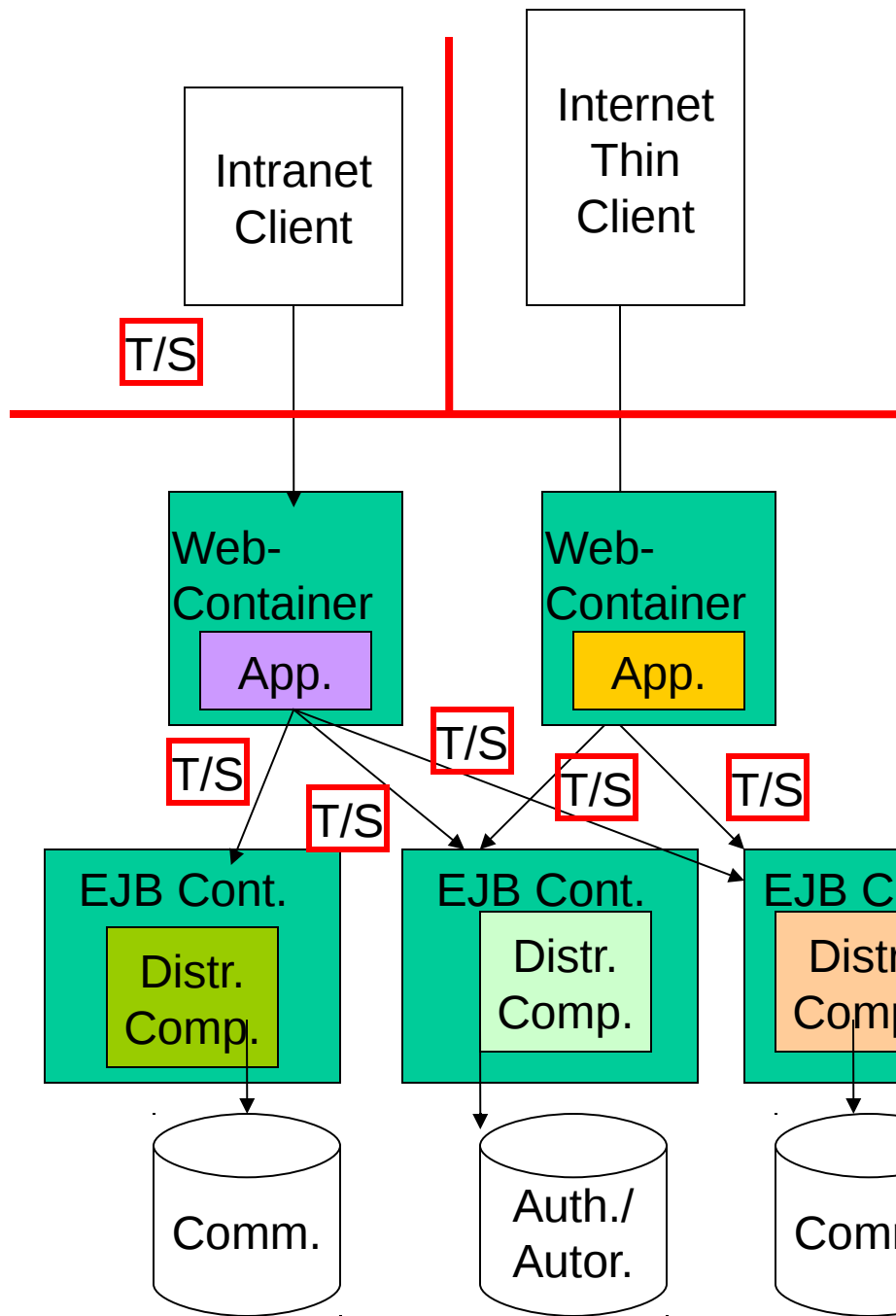
```
            });
```

```
        // Subject again associated with the current thread
```

```
        return null;
```

```
    }
```

```
});
```

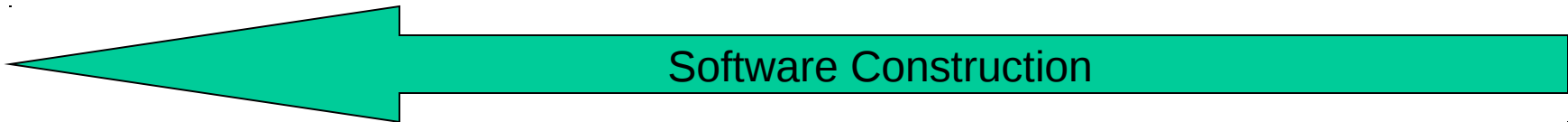
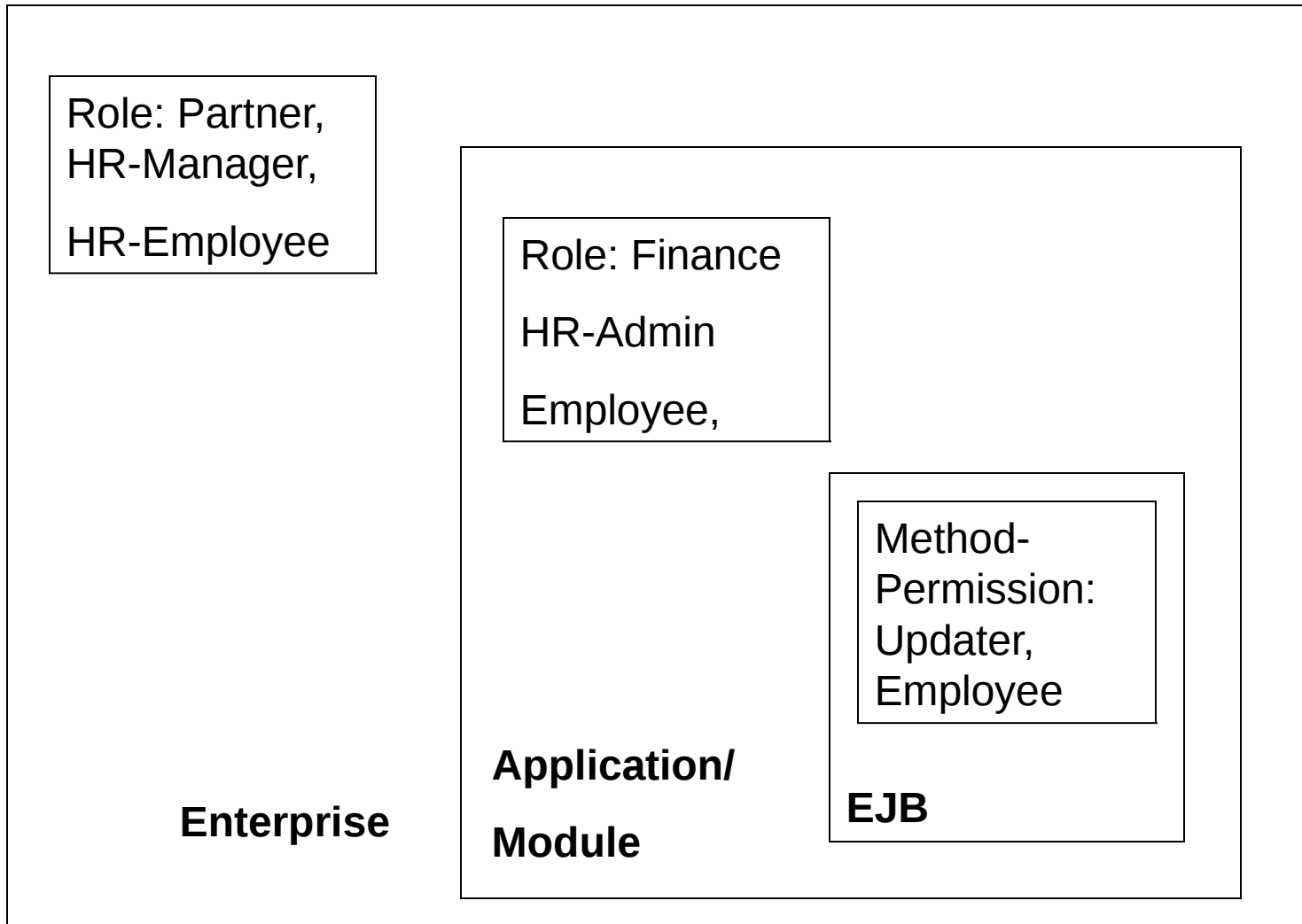


Multi-tier applications with thin client interfaces. Exposed to the Internet

J2EE type distributed components (EJBs)

**T/S** Transaction and Security Context

# EJB Role Definitions: Enterprise- Application – Module - Bean



```
@RolesAllowed("payroll") // needed??
@RolesAllowed("employee")
@DeclareRoles("payroll")
```

```
@Stateless public class PayrollBean implements Payroll {
    @Resource SessionContext ctx;
    public void updateEmployeeInfo(EmplInfo info) {
        oldInfo = ... read from database;
```

```
    // The salary field can be changed only by callers with role "payroll"
    if (info.salary != oldInfo.salary && !ctx.isCallerInRole("payroll")) {
        throw new SecurityException(...); }
```

```
    if (info.salary != oldInfo.salary && ctx.isCallerInRole("payroll")) {
        callerName = ctx.getCallerPrincipal().getName;
```

```
        if (callerName.equals(oldInfo.name()) //
            throw new SecurityException(..); payroll employee == employee!!!
```

```
    }
```

```
    ...
```

```
    }
```

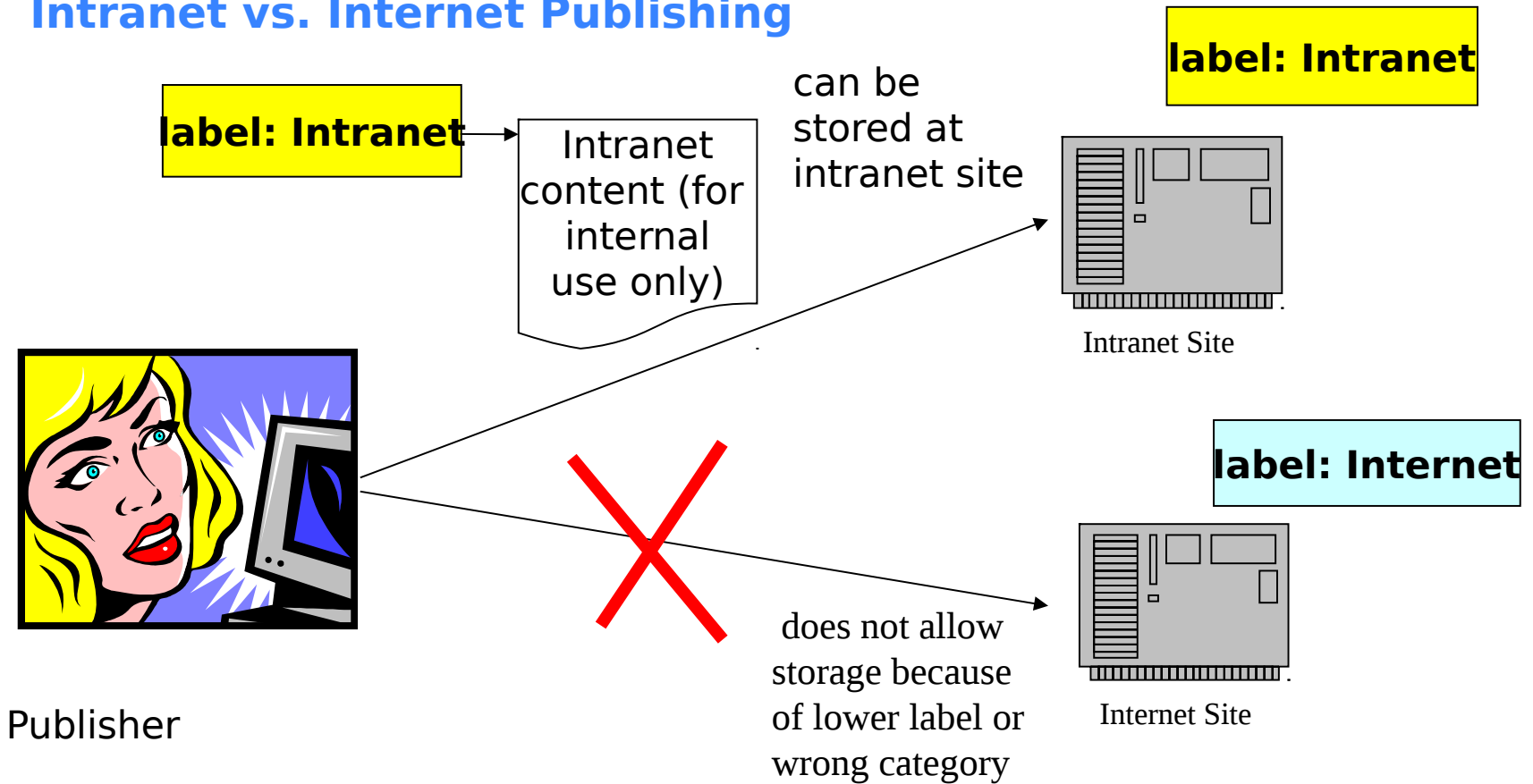
```
    ...
```

```
    }
```



# Multi-Level Security

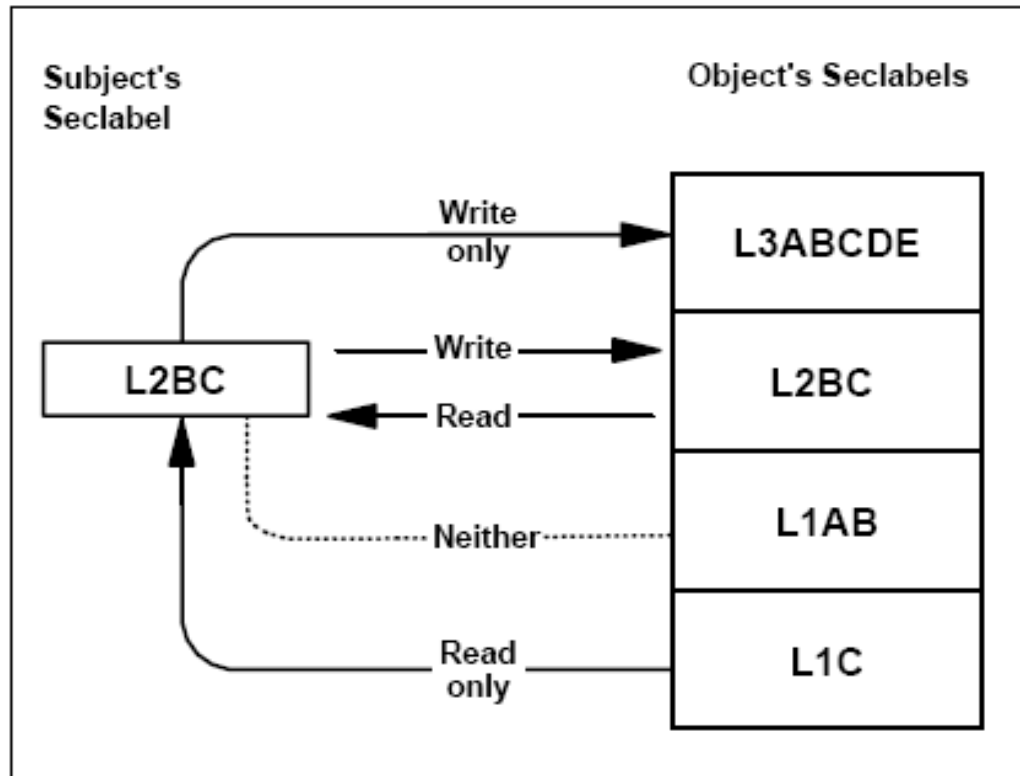
## Intranet vs. Internet Publishing



**Security of the information is now based on the information itself. The TCB now has the job to protect the label and the content across all processing steps (import/export/manipulation)**

# Multi-Level Security cont'd

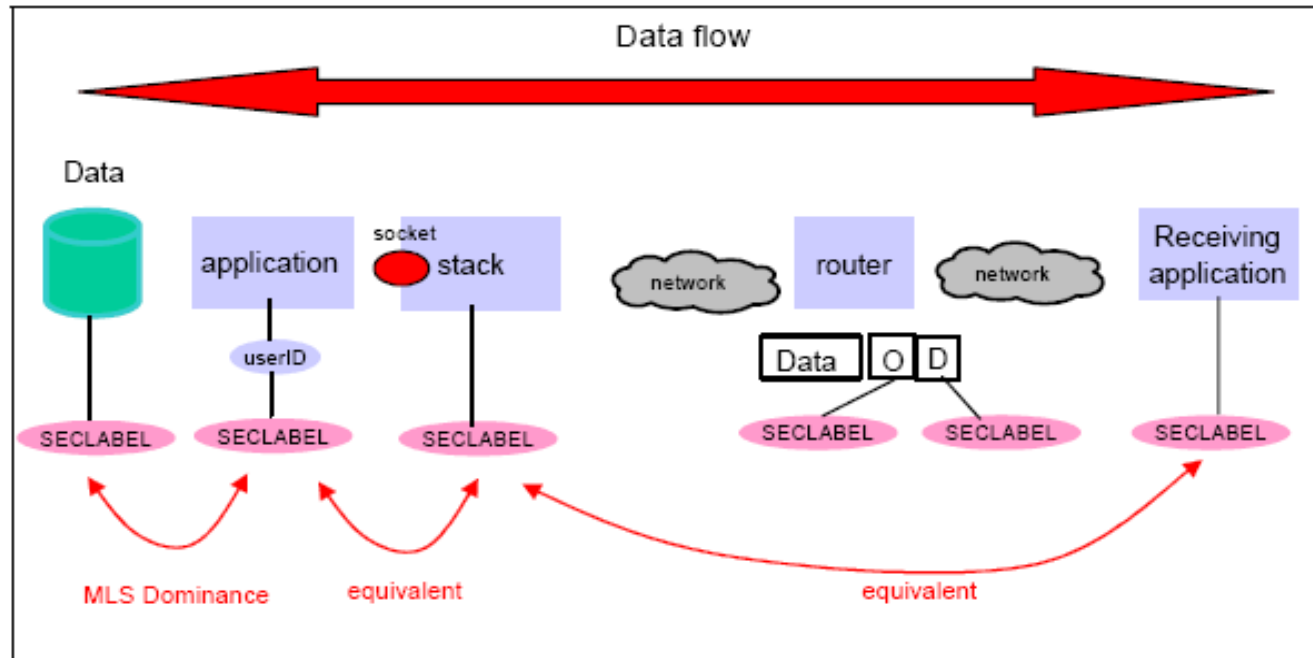
## Rules (\*properties)



**(C.Rayns). Please note that not every level needs categories. No write-down, no read-up allowed.**

# Multi-Level Security cont'd

## Labels are transitive



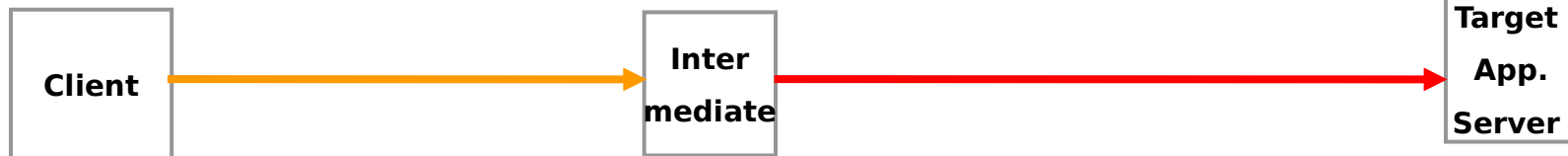
from C.Rayns. Obviously the TCB does not enclose the other system and requires a trust relation between both systems.

# Trust and Anonymity in Distributed Systems

- Secure Delegation without global authorities
- The effects of total distribution on security
- How to guarantee anonymity in distributed systems
- Micro-payments and vandalism
- Hash-cash
- Fight centralized security hubs (e.g. MS Hailstorm)
- Reputation building

# Proxy Certificates

## Secure delegation without global authorities



End Cert.

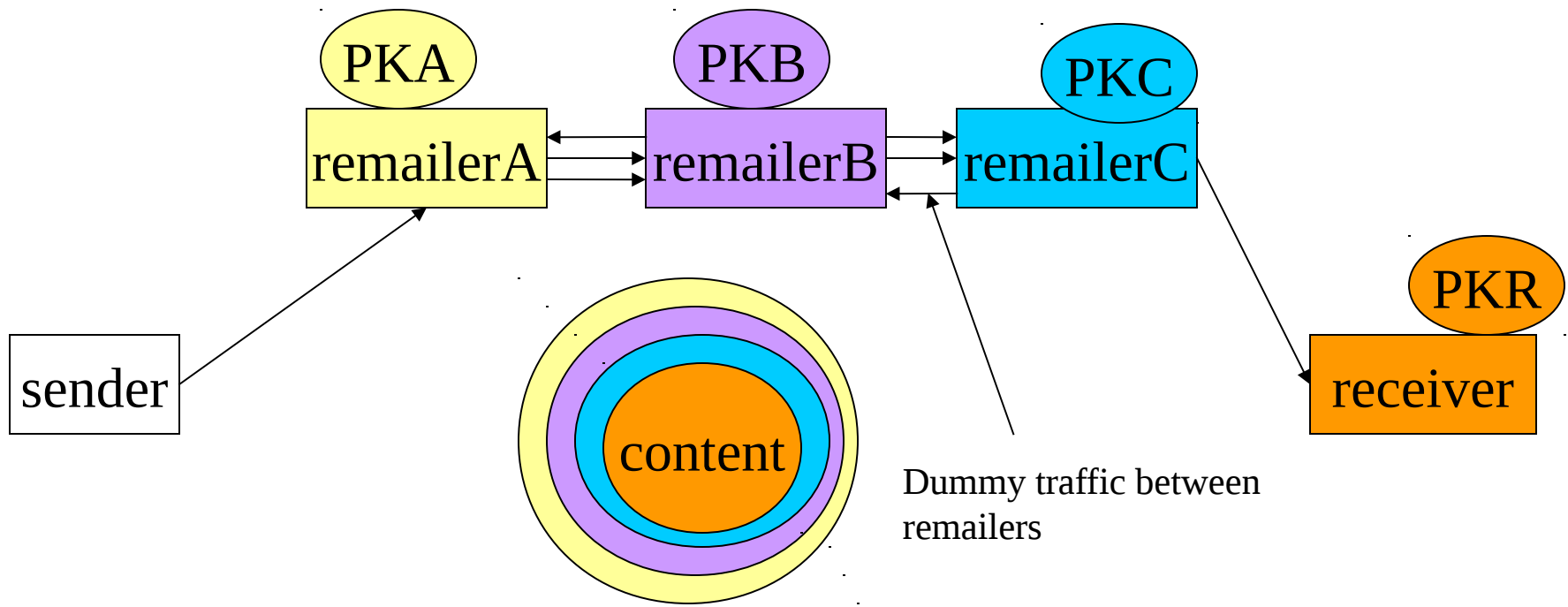
1. Intermediate creates temporary PKI key pair and sends public key to Client
2. Client creates Certificate with public key of Intermediate, puts in its own identity credential, the power of attorney, lifetime and validity constraints and signs it with its own end certificate. Then it sends the PC to Intermediate.
3. The Intermediate needs to call Target to finish the job for Client. It presents the PC to Target. Target challenges Intermediate and Intermediate uses the private key of the PC to prove ownership.
4. Target accepts Intermediate as acting for Client.

**A PC is a selfsigned power-of-attorney statement for the owner of a key. With SAML assertions the delegation can be restricted and specialized. This is used in Grid Computing to limit the use of authentication credentials.**

# The civil rights oriented security mantra

- Anonymity (Protect the anonymity of sender, transmitter and receiver to avoid e.g. political pressures)
- Guarantee availability of information to avoid political censorship
- Guarantee end-to-end privacy of messages without backdoors for governments
- Protect individual data from being combined to user “profiles” (e.g. double-click approach or user tracking through mobile phones, highway ticket control etc.)
- Fight centralized security hubs (e.g. MS Hailstorm)
- Explain the risks behind security technology (when Chaos Computer Club destroyed common believes into the safety of teletext based online banking)

# Preserving anonymity: Onion routing



A sender creates an „onion“ like mail: the most inner layer is the message encrypted with the receivers public key. The next layer contains the receivers address, encrypted with remailer C’s public key. C will unpack and decrypt this layer and then know where to forward the message. And so on. Note the dummy traffic between remailers to conceal the fact that sender has sent a mail. Otherwise somebody watching traffic could follow the mail from sender through remailers. Other systems use proxies etc. to achieve anonymity for both sender and receiver

# Distributed Trust: building a reputation

- Use of a reputation server
- Checking if principals keep their promises by informing peers
- Micro-payment systems
- score systems, e.g. e-bays ratings (have been abused!)
- Newsgroup discussions on products

How do you establish trust with people and services you don't know yet? How do you bootstrap a reputation?



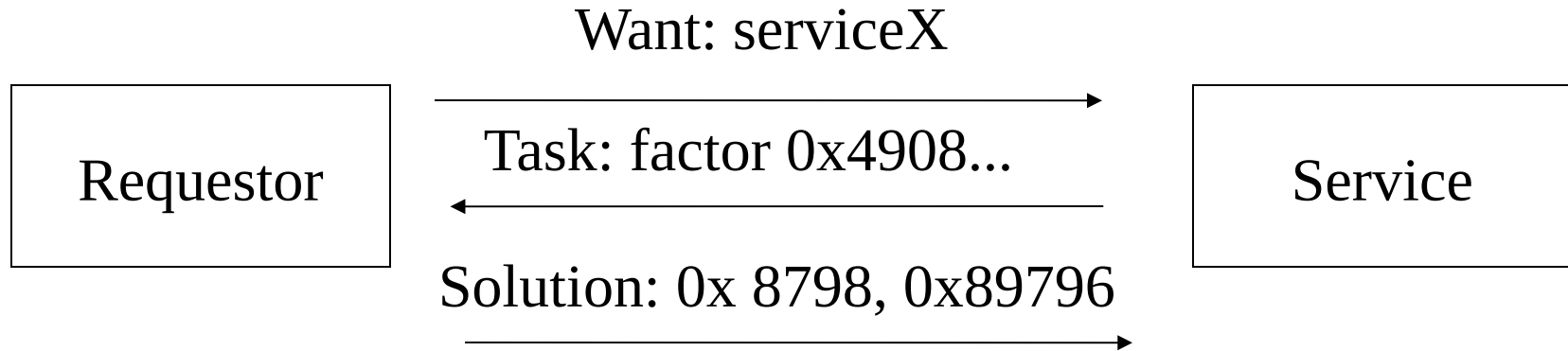
# Fighting Vandalism: Paying a Digital Price

## Micro-payment systems:

- interactive factoring of numbers
- non-interactive use of hash algorithms to create a stamp

Micro-payment systems work by forcing requesters to perform certain expensive computations before getting access to a service. When authentication based access control is not possible (e.g. p2p systems) these technologies can still prevent DOS attacks. They can be used to protect mailboxes against spam, to evaluate CPU performance or to protect WIKIs from vandalism. See David Mertz, Charming Python (resources)

# Interactive Challenge



The service generates new challenges for new requestors and can limit the lifetime of solutions. This technology requires direct communication between service and requestors.

# Non-Interactive Challenge (e.g.hashcash)

stamp

1:collision bits:date:resource:ext.salt:suffix

structure:

1:20:11/20/2004:foo@bar.com::1:<to be provided>

required hash:

000007832417587402470710274981.....

calculate hash from  
stamp with suffix and  
check for collision bits

Requestor

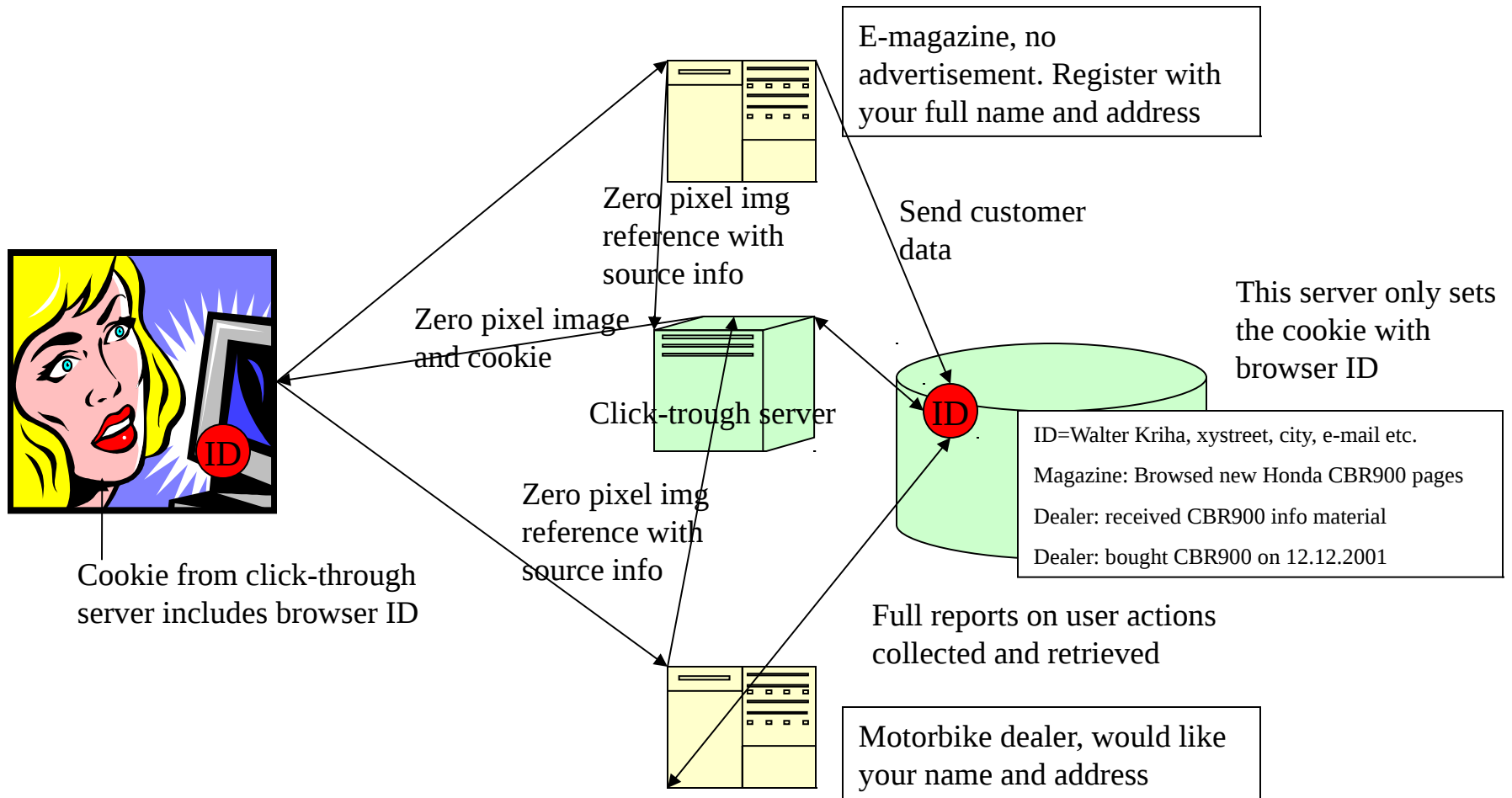
send stamp with suffix to service

Mail Service  
foo@bar.com

remember suffixes

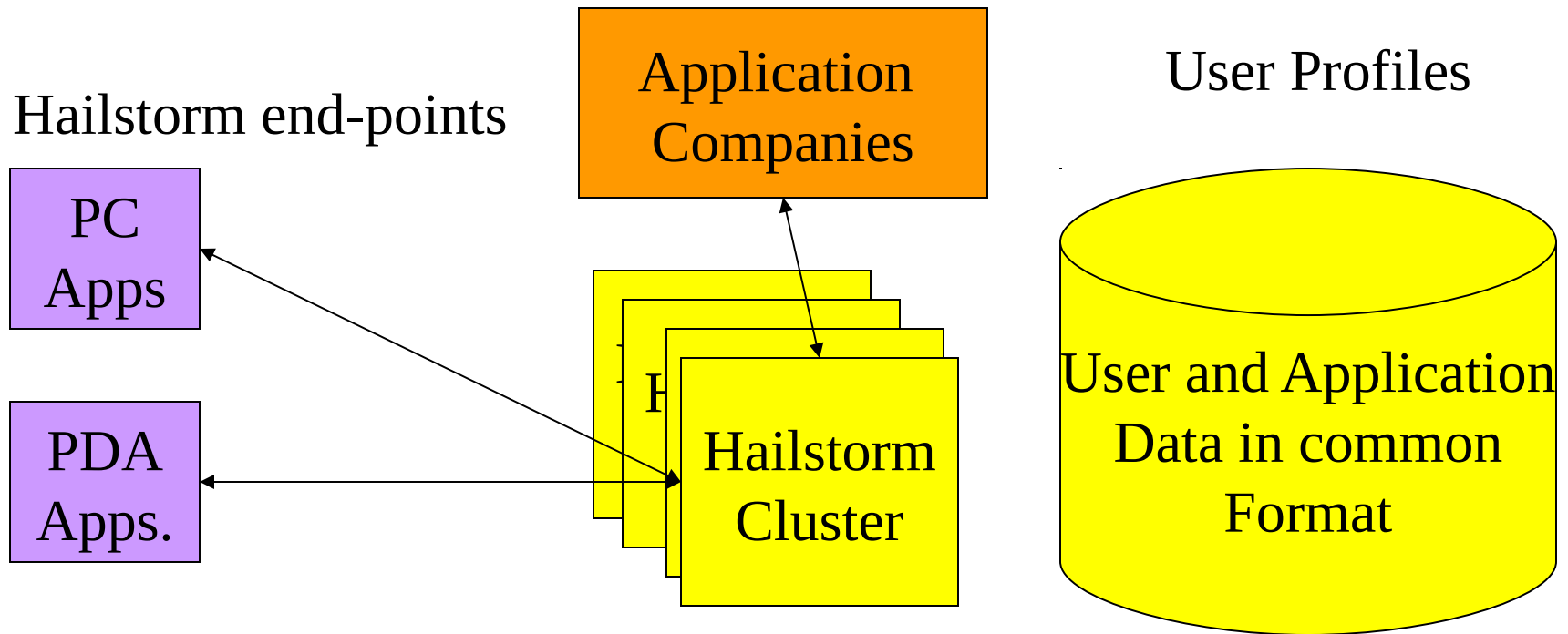
The challenge consists of calculating a suffix value that – concatenated with date, resource etc. produces a hash value that has X leading zero bits (collisions) at the beginning. In this case we have 20 leading zero bits which takes a couple of seconds on a modern processor. X can be adjusted for longer challenges.(see also [www.hashcash.org](http://www.hashcash.org))

# User profiling through cookies



The cookie serves as the common link between three sites. The dealer and the magazine pages include image references (zero pixel) to the click-through server (e.g. double-click) with page information. The click-through server sets a cookie with browser ID and stores all information about requests from this browser in a DB. Business clients pay for information from this DB. (see <http://philip.greenspun.com/wtr/dead-trees/53007.html>)

# Hailstorm, the Ueber-Profile



Users and Application producers license the access to hailstorm where user specific application data are kept. The user can say which application gets access to what data („opt-in“, granularity problem) and needs to sign-in only once per session. Applications can use user preferences from other applications to target actions. The old mainframe wisdom: who controls the customer data controls the company. Security of hailstorm supposedly based on Kerberos (with PKI?)

# Terms Used

- **Token:** a signed security statement
- **PAC: Privilege Attribute Certificate,** a token with authority attributes
- **TCB: Trusted Computing Base,** the infrastructure you must trust
- **POLA: Principle Of Least Authority,** never give more rights than are needed for an operation
- **CSIV2: Corba Common Interoperable Security Spec.** Defining delegation mechanisms and attribute structures
- **SAML: Security Assertion Markup Language.** XML Schema to express security related statements (name, authentication, authority)
- **XACML: XML Schema to express policies and rules.** Includes access infrastructure architecture (PDP, PEP, PIP)
- **RFID: Micro-Tags** embedded in materials which can contain identification information. Sent when reader applies field.
- **WS-Security:** XML Schemata to express (and convey) security information
- **BSI: Bundesamt für Sicherheit in der Informationstechnik** ([www.bsi.de](http://www.bsi.de))
- **Signed Transaction:** A request signed by a client
- **Federated Identities:** several authorities exist which certify identity
- **OSGI: Open Services Gateway Initiative.** Java based framework for embedded control which allows dynamic addition and removal of services via so called bundles.
- **Virtual Organization:** ad hoc organization formed from members of different real organizations. Own authentication and authorization domain which is mapped to real organizations.
- **Ambient Authority:** Every right that has NOT been given to a process explicitly via a capability.
- **Discretionary Access Control: Opposite: Mandatory.** In practice it means only that during every access control operation ALL rights of an owner of a resource are applied. Code or resource itself cannot restrict operations.
- **Message/token based security: Object based security.** Signatures allow author verification, not who really sent it. Replay attacks are possible.

# Resources (0)

1. **C. Ryan, Multi-level Security** ([www.redbooks.ibm.com](http://www.redbooks.ibm.com))
2. **CORBA security service specification v1.8** ([www.omg.org](http://www.omg.org))
3. **Identity and Access Management Solutions, sg24-6692-00** ([www.redbook.ibm.com](http://www.redbook.ibm.com))
4. **SAML 2.0 Specification** ([www.oasis-open.org](http://www.oasis-open.org))
5. **Proxy Certificates and digital purse. Grid Computing Initiative,** ([www.globus.org](http://www.globus.org))
6. **Hendrik Mieves, Sichere Verbindung von Middleware-Infrastrukturen (Corba CSIv2 - Webservices SAML bridges) Thesis, Technical University of Damstadt.**
7. **Principle of Least Authority: papers on** [www.erights.org](http://www.erights.org) and
8. <http://www.microsoft.com/billgates/speeches/2005/02-15RSA05.asp>  
**Bill Gates at RSA2005**
9. **Listeners Considered Harmful: The “Whiteboard” Pattern. Technical Whitepaper,** [www.osgi.org](http://www.osgi.org)
10. **Marc Stiegler, a capability based browser** ([www.combex.com](http://www.combex.com))

# Resources (2)

- Van Steen/Tanenbaum, Chapter 8
- Studie “Gesicherte Verbindung von Computernetzen mit Hilfe einer Firewall”, Andreas Bonnard, Christian Wolff, Siemens AG (für Bundesamt für Sicherheit in der Informationstechnik BSI)
- Internet Cryptography, Richard E. Smith, [www.visi.com/crypto](http://www.visi.com/crypto)
- WWW Security FAQ, [www.w3.org/Faq](http://www.w3.org/Faq) (with short bibliography)
- Cryptography FAQ, [www.faqs.org/cryptography-faq](http://www.faqs.org/cryptography-faq)
- RISKS, Forum on Risks to the Public in Computers and Related Systems <http://catless.ncl.ac.uk/Risks> (real life stories on the social and political consequences of security flaws)
- Eric Rescorla, SSL and TLS, Designing secure systems. Very good about SSL and its use to secure other protocols or applications. Shows that http and SSL don't fit perfectly.



# Resources (2)

- The EU commissions report on the US/UK Spy project Echelon – How the US and UK do industrial espionage against Europe.
- Simson/Garfinkel, Database Nation
- Bruce Schneier, Applied Cryptography (the bible of cryptography).
- Bruce Schneier, Secrets and Lies (Shows the dirty reality of bad products, wrong policies etc. A must read)
- [http://www.infosecuritymag.com/articles/april00/columns\\_cryptorhythms.shtml](http://www.infosecuritymag.com/articles/april00/columns_cryptorhythms.shtml) (Schneier on security as a process, good article)
- Diffie et.al., Privacy on the line (explains why encryption is a civil right that organizations like the NSA try to subvert) (Yes, THAT Diffie from Diffie-Hellman Key exchange)
- [www.cert.org](http://www.cert.org) , your most important source for information on new security breaches etc. Register for the newsletter! Also an excellent source on security technology
- Frederick Thomas Martin, Top Secret Intranet – How US Intelligence built Intelink – The worlds largest most secure network. Good to read.

# Resources (3)

- Improving the Security of Your Site by Breaking Into it:  
<http://www.fish.com/~zen/satan/admin-guide-to-cracking.html>  
A good introduction into cracking systems. Fun reading too.
- The strange tale of the denial of service attacks against grc.com, Steve Gibson, 2001. The power of distributed DOS attacks. ([www.grc.com](http://www.grc.com))  
Really funny to read!
- <http://www.snort.org/docs/idspaper/> on intrusion detection and denial of service ([www.snort.org](http://www.snort.org) home of snort IDS plus good docs)
- <http://csrc.nist.gov/publications/nistpubs/800-10/node1.html> (firewalls and screened subnets etc., oldie but goodie)
- Andy Oram, Peer-to-Peer, (Good overview of security policies and techniques for p2p. Very different to the typical company-internal views. Focus on privacy, anonymity, censorship avoidance etc.

# Resources (4)

- <http://csrc.nist.gov/publications/nistpubs/> good pubs on PKI etc.
- <http://www.ipprimer.com/nat.cfm> NAT primer, good tcp guide by Daryl Banttari
- [http://www.cisco.com/warp/public/3/it/present/webbit/webbit\\_vpn.pdf](http://www.cisco.com/warp/public/3/it/present/webbit/webbit_vpn.pdf)  
Good intro to VPN and layer 2 tunneling, ipsec, etc. SLIDES
- [http://www.securecomputing.com/pdf/wp\\_vpn.pdf](http://www.securecomputing.com/pdf/wp_vpn.pdf) vpn overview
- <http://www.cloudconnector.com/education.html> good edu materials on VPN, ipsec, attacks, script kiddies etc.
- Linux FreeS/Wan Overview: IPSEC in Linux, (easy)  
[www.freeswan.org/freeswan\\_trees/freeswan-1.1/doc/overview.html](http://www.freeswan.org/freeswan_trees/freeswan-1.1/doc/overview.html)
- <http://www.ietf.org/rfc/rfc2401.txt> IPSEC rfc
- Know your enemy, the tools and methodologies of the Script Kiddie,  
<http://project.honeynet.org/papers/enemy/enemy.html>
- RFC 2547bis: BGP/MPLS VPN Fundamentals. Juniper Networks,  
[www.juniper.net](http://www.juniper.net). Explains provider provisioned VPNs using border gateway protocol etc.

# Resources (5)

- [www.hashcash.org](http://www.hashcash.org) portal for hashcash applications
- David Mertz, Charming Python: Beat spam using hashcash (on [www.ibm.com/developerworks](http://www.ibm.com/developerworks) ,11/2004). Excellent short description of hashcash and its possible applications.
- BM WebSphere Developer Technical Journal: Advanced authentication in WebSphere Application Server *Managing user authenticity and privileges in a distributed application server environment* Keys Botzum et.al, 17 Aug 2005. Excellent article explaining initial login and identity propagation problems in web application servers
- ERCIM Magazine on Security and Trust Management.. Gives an overview of current research (policy languages, logic for access control etc.) [http://www.ercim.org/publication/Ercim\\_News/enw63/EN63.pdf](http://www.ercim.org/publication/Ercim_News/enw63/EN63.pdf) includes: Extensions to the PERMIS X.509 Privilege Management Infrastructure by *David Chadwick*
- P. Windley, Digital Identity Management (oreilly). Perfect book on management and high-level technical information. Includes practical experiences from building a large DIM for Utah.