Philipps Universität Marburg

Shibboleth Single-Sign-On für Web-Anwendungen

HRZ Uni Marburg

Manuel Haim, Stand 10/2011

Shibboleth

Die Philipps-Universität Marburg

- gegründet 1527
- ca. 20.000 Studierende
- ca. 3.000 Mitarbeiter

Warum Shibboleth?

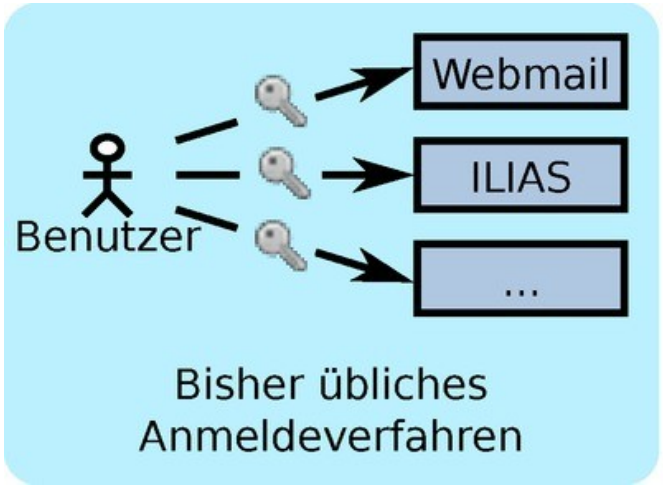
- **Authentifizierungsdienst:**
  zentrale Anmeldeseite für alle Web-Dienste (vgl. OpenID, Facebook, ...)
- **De-facto-Standard:**
  im Bibliotheksumfeld international verbreitet
- **Nutzerfreundlich:**
  installationsfreie Alternative zu VPN-Zugang
- **Integration bestehender Web-Anwendungen:**
  einmal anmelden → alle Dienste nutzen

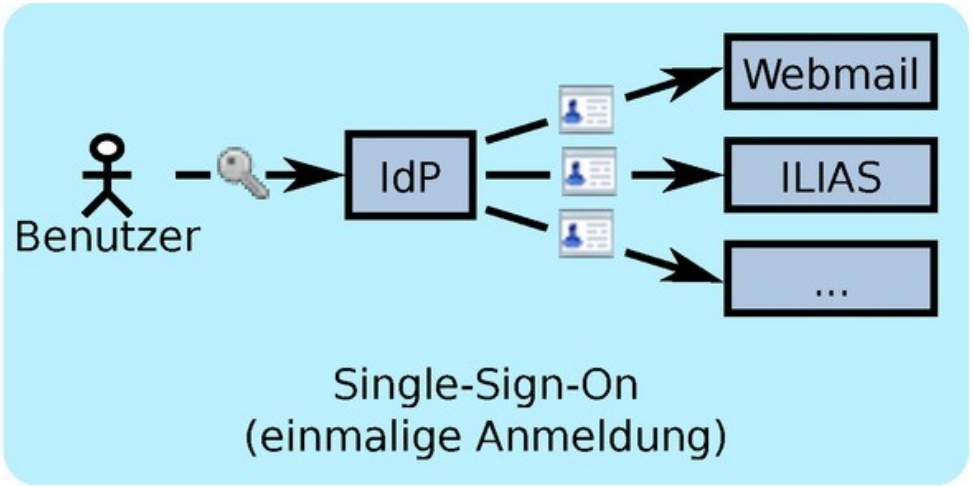
Problem:
DFN-AAI fordert:
Benutzerdaten max.
zwei Wochen alt!

Unser Nahziel:
Shibboleth für lokale
Web-Anwendungen

Was kann Shibboleth?

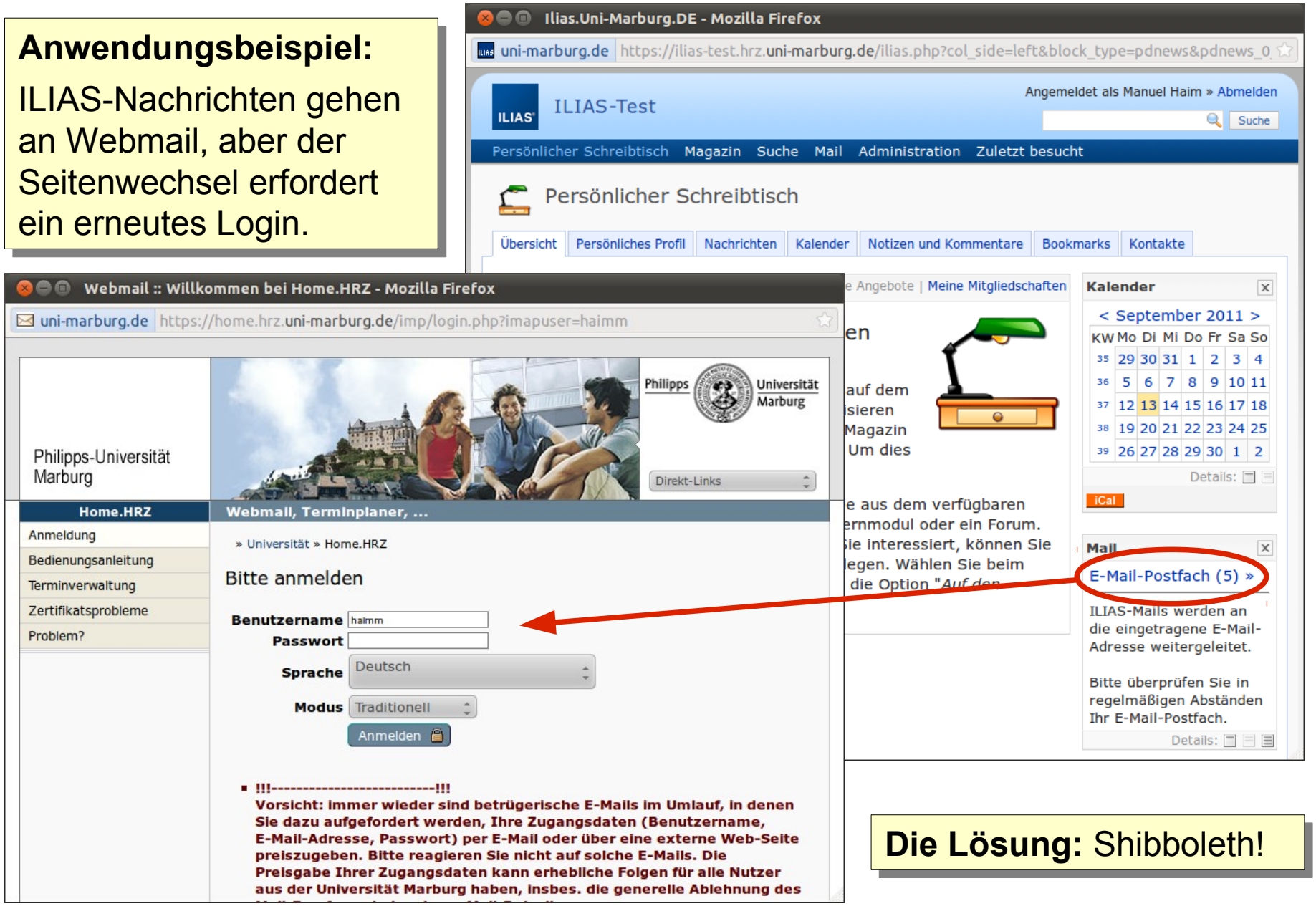
- nimmt einmalig Anmeldung entgegen und erhält sie aufrecht
- übermittelt als „Identity Provider“ eine anwendungsabhängige Liste von personenbezogenen Daten an die jeweilige Web-Anwendung





Anwendungsbeispiel:

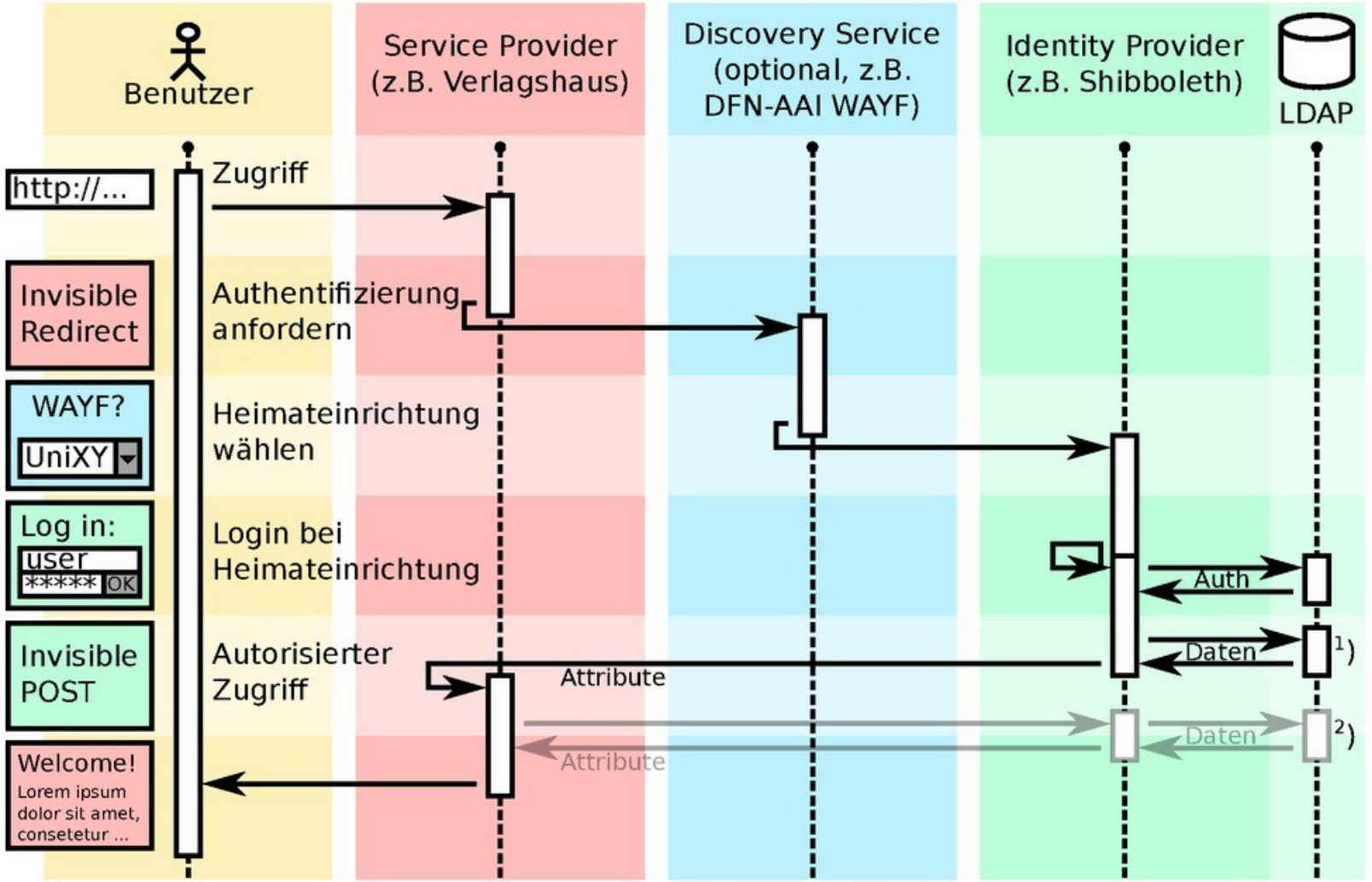
ILIAS-Nachrichten gehen an Webmail, aber der Seitenwechsel erfordert ein erneutes Login.



Die Lösung: Shibboleth!

Wie funktioniert Shibboleth?

M. Haim, 12/2010



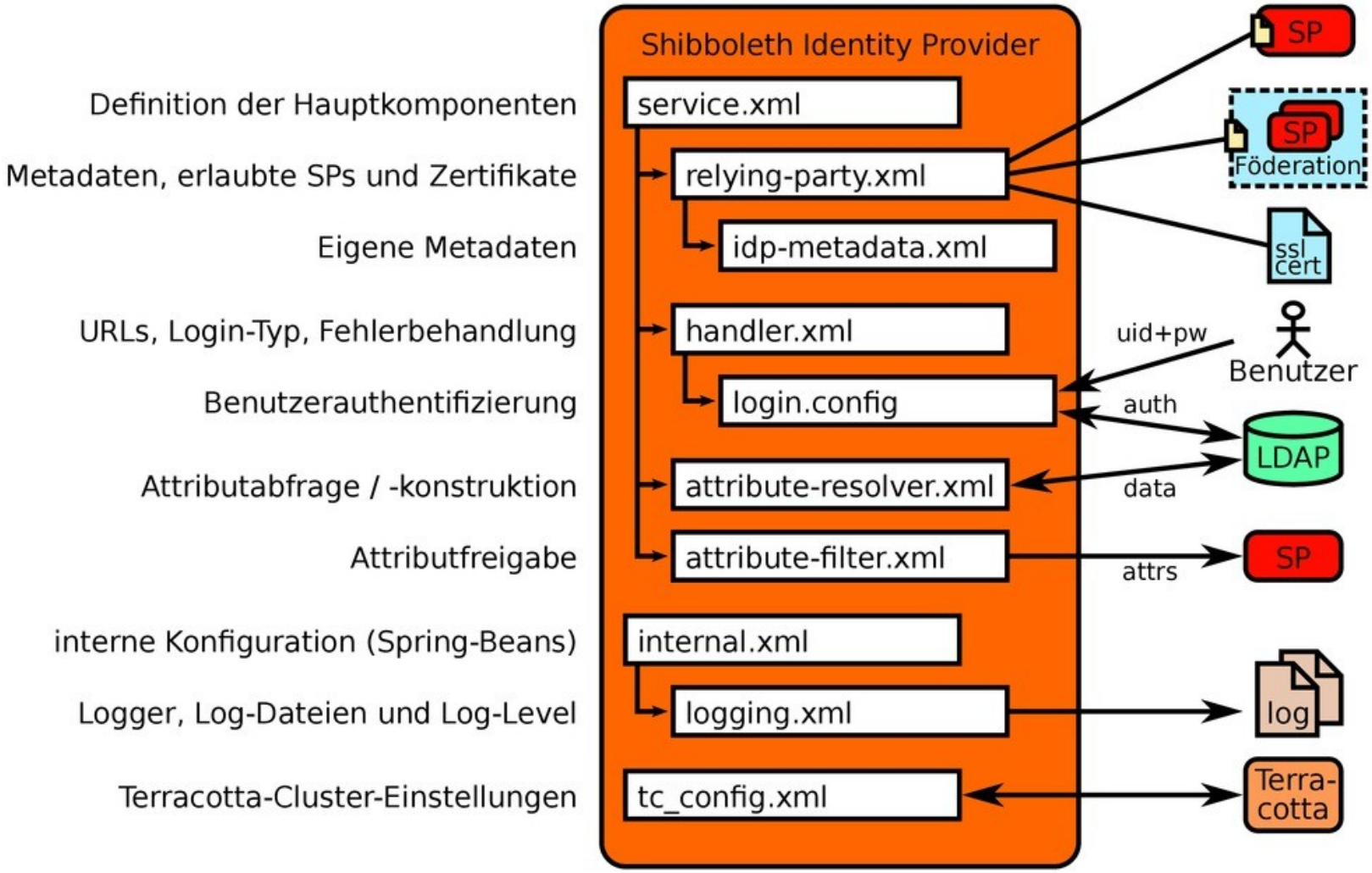
[1]) SAML2: Attribute werden XML-verschlüsselt & signiert mittels Benutzer-Client übertragen

[2]) SAML1: Attributanfrage erfolgt ohne XML-Verschlüsselung über verschlüsselten Rückkanal

Konfiguration

Konfigurationsdateien des Shibboleth-IdP 2.x

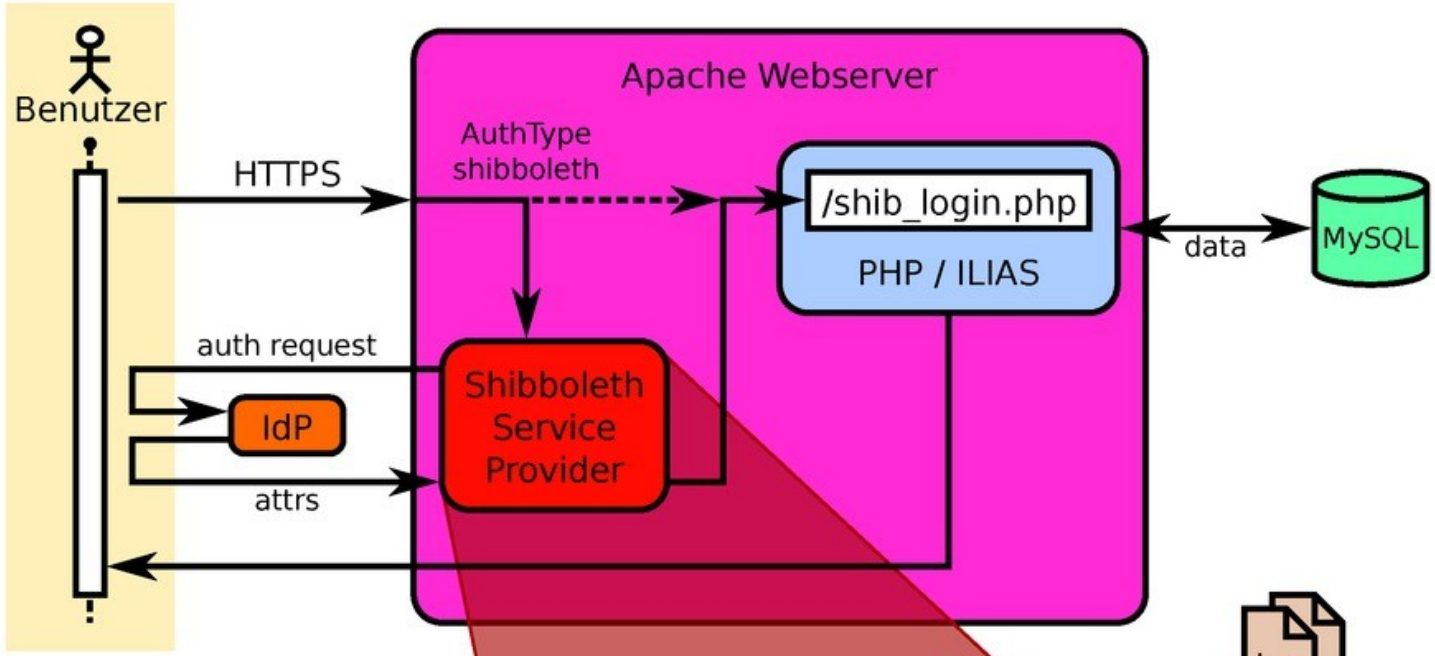
M. Haim, 12/2010



+ Webapplikation war/idp.war bzw. src/main/webapp/

Shibboleth-Login für ILIAS

M. Haim, 01/2011

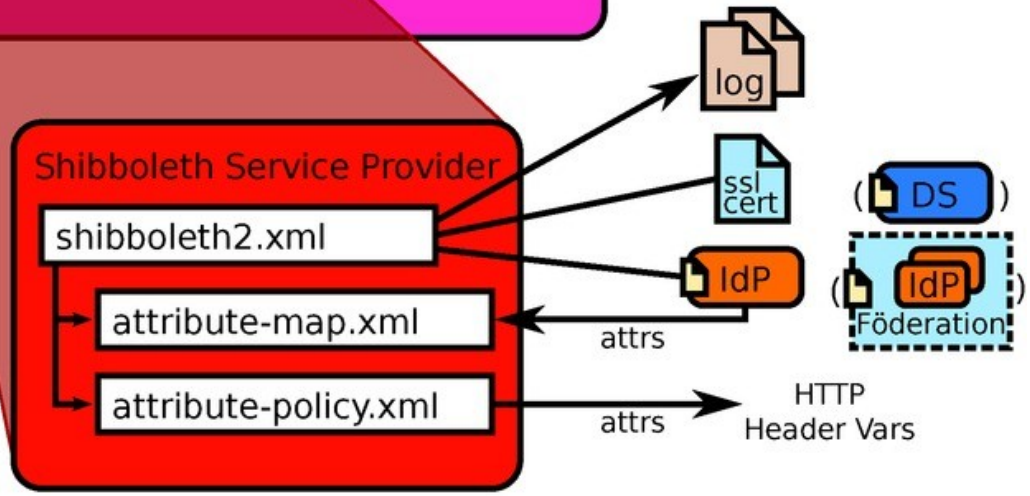


Shibboleth-SP 2.x

Metadaten, Logs, IdPs, Zertifikate

Attributzuweisung

Attributfreigabe



Ausbau zum hochverfügbaren Service

Lastverteilung + Ausfallsicherung = High Availability

- **Grundidee: Mehrere Server bilden einen Cluster**

  → Die Gesamtlast verteilt sich auf die Knoten (Lastverteilung)

  → Kein Dienstausfall bei Ausfall eines Knotens (Ausfallsicherung)

- **Knoten müssen Zustands-Informationen teilen**

  → Shibboleth-IdP speichert alle Zustandsdaten im StorageService

  → Problem: Java-Map (Objektänderungen ohne Zurückschreiben)

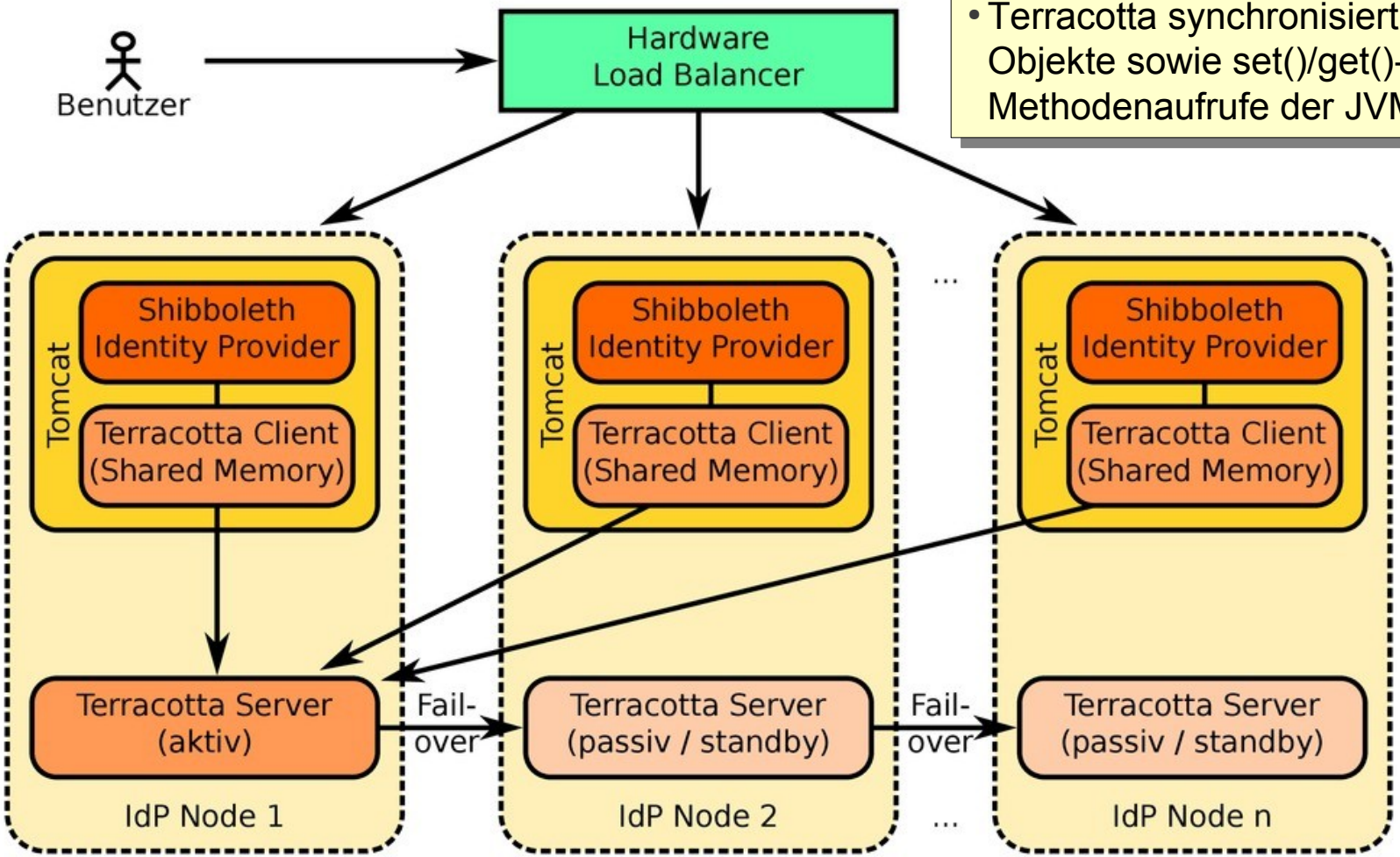
  → Standardlösung: Java-VMs per Terracotta abgleichen

Aufbau eines Shibboleth-Clusters

(gemäß Shibboleth-Wiki)

M. Haim, 12/2010

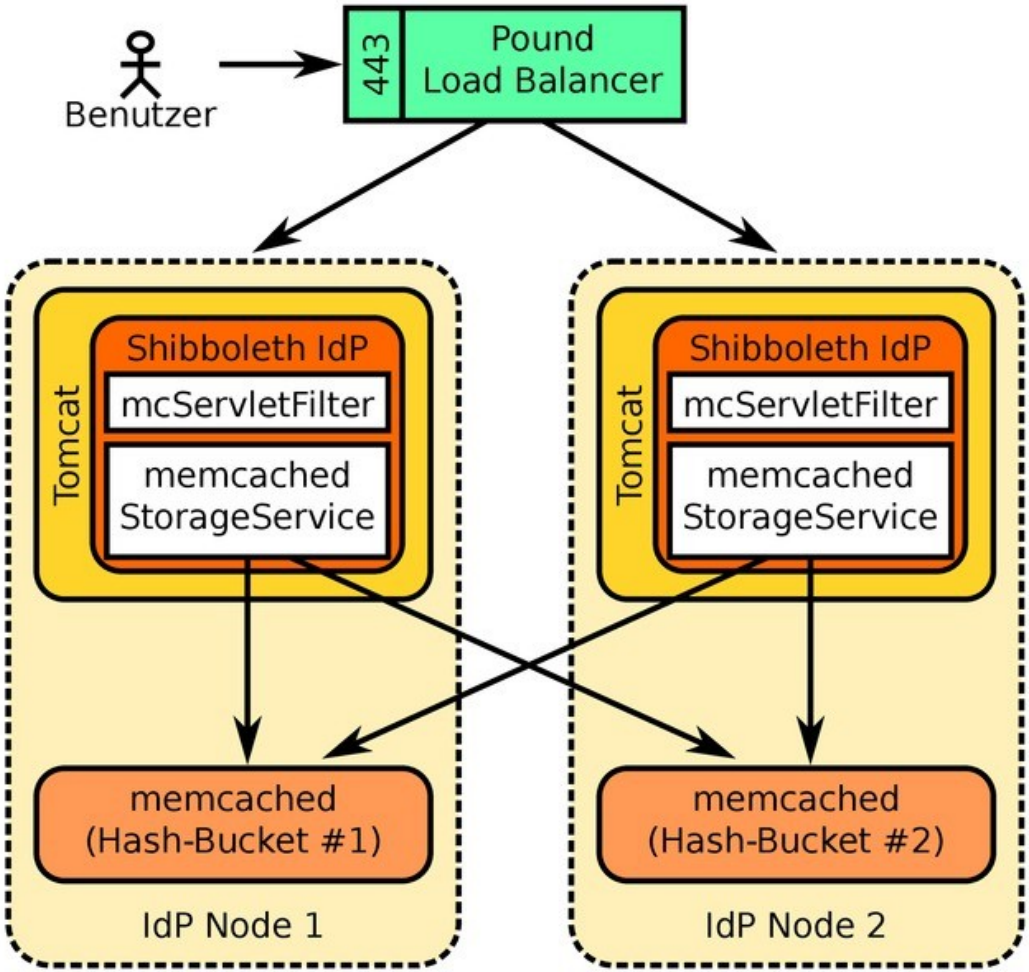
Funktionsweise:
- Terracotta synchronisiert alle Objekte sowie set()/get()-Methodenaufrufe der JVMs.



Alternativer Aufbau eines Shibboleth-Clusters

M. Haim, 09/2011

(mit IdP Memcached StorageService)



Funktionsweise:

- Pound mit „sticky session“.
- StorageService speichert alle Objekte immer in lokaler Map. (wg. Java-Objektreferenzen)
- StorageService wird bei get()- und set()-Operationen mit memcached abgeglichen. (BeanUtils.copyProperties())
- Ein ServletFilter wird nach jedem IdP-Servlet ausgeführt, um Änderungen am Session-Objekt explizit zurück in den StorageService zu schreiben.
- Daten werden per Hash-Funktion auf mehrere memcached-Instanzen verteilt.
- Bei Ausfall eines Servers ist schlimmstenfalls ein neues Login am IdP erforderlich.

Lasttest mit „The Grinder“ – Ergebnisse

Anzahl Knoten	Cluster-Methode	-Xmx	RAM	Speicherverbrauch der jeweiligen Cluster-Methode pro Knoten	TPS*
2	–	1536m	3GB	–	65
2	Memcached (disjunkt)	1536m	3GB	10 MB RAM permanent, 13 MB RAM pro 5.000 Logins**	63
2	Terracotta 3.5.1	1536m	4GB	800 MB RAM permanent, 300 MB HD pro 5.000 Logins	22

* Transaktionen pro Sekunde, hier: Anmeldungen pro Sekunde.
** Bei Verwendung von nur einem Memcached-Server: ca. 25 MB RAM pro 5.000 Logins.

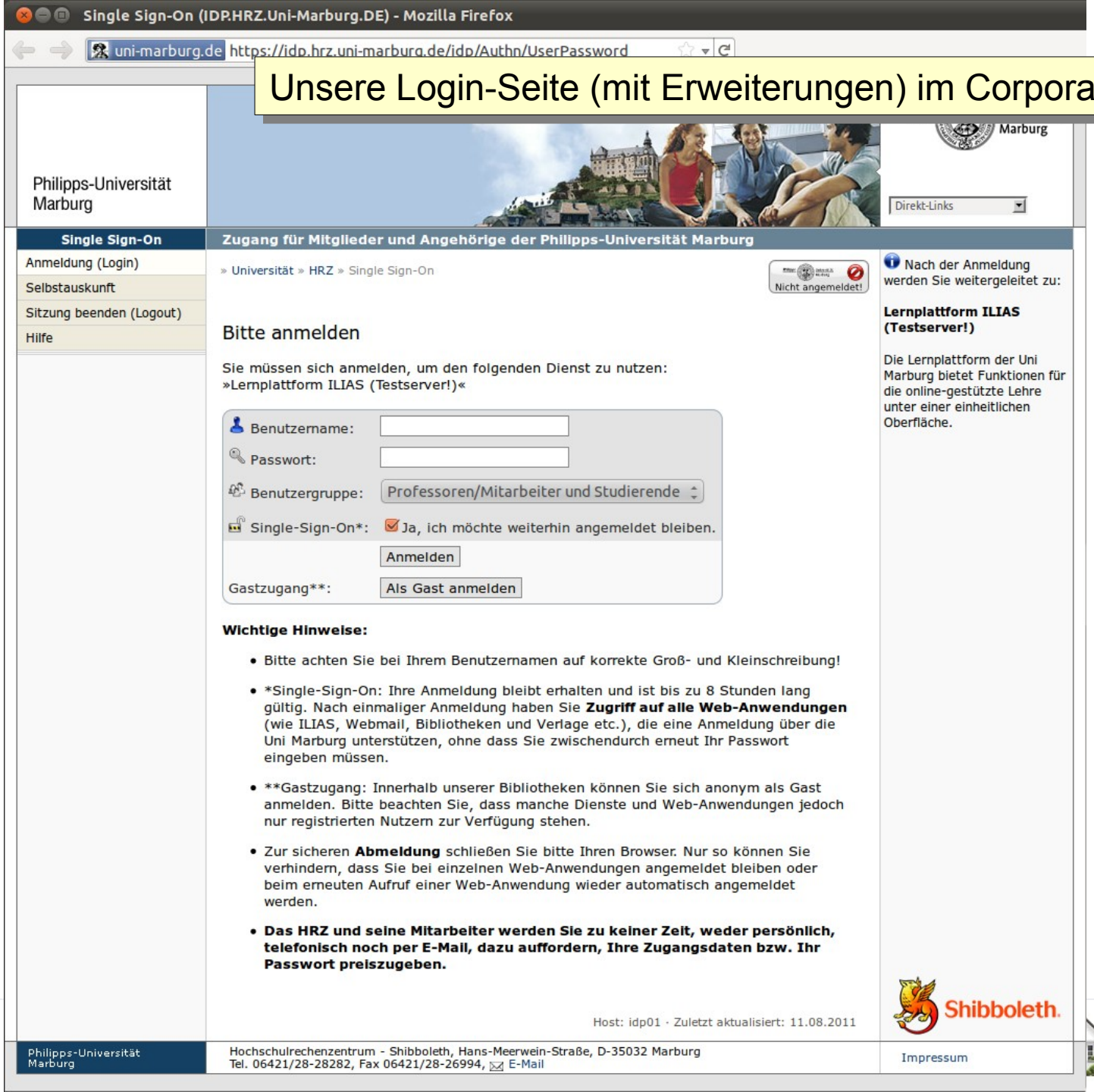
- Der Speicherverbrauch des „geteilten“ Speichers entwickelte sich in den Tests linear.
- Memcached arbeitete auch nach 150.000 Anmeldungen munter weiter.
- Terracotta lief im Modus „permanent-store“ (wie im Shibboleth-Wiki vorgegeben) und wurde vermutl. durch die Festplattennutzung ausgebremst. Nur 5.000 Logins getestet.

Tipps zur Optimierung der einzelnen Knoten

- Debian Linux: KVM, je 4 CPUs @ 2,5GHz, 3 GB RAM
  (RAM ist wichtig: Tomcat wird bei Nutzung der Swap-Partition sehr sehr langsam!)
- Tomcat 6: Java-Heap-Space erhöht (-Xmx 1536m)
- LDAP-Anfragen ohne :caseExactmatch:
- LDAP: Eigener dnResolver in login.config
  vt-ldap unterstützt noch kein LDAP Connection Pooling,
  vgl. http://code.google.com/p/vt-middleware/issues/detail?id=118
- ConnectionPool in attribute-resolver.xml
  (→ nicht zwingend nötig, macht Shibboleth bereits automatisch)
- Cluster-Lösung: memcached (-m 1024) statt Terracotta

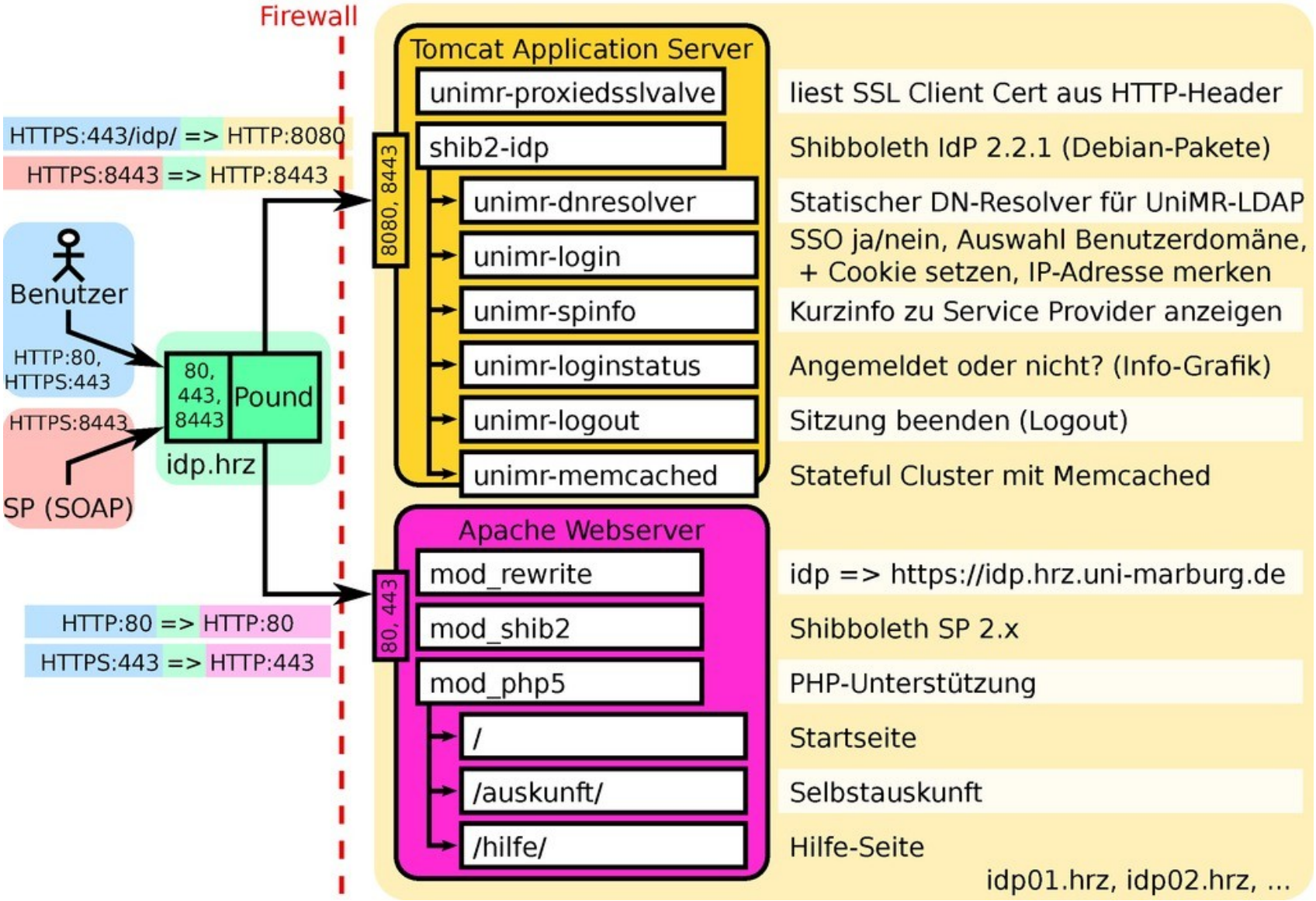
Ausbau von Shibboleth zum Single-Sign-On-Portal

Unsere Login-Seite (mit Erweiterungen) im Corporate Design



Shibboleth-Cluster idp.hrz.uni-marburg.de

M. Haim, 08/2011



Weiterführende Links

- Shibboleth:
  http://www.shibboleth.net,
  https://wiki.shibboleth.net/confluence/display/SHIB2/Home
- Föderation DFN-AAI:
  https://www.aai.dfn.de
- Shibboleth Single Sign-On an der Uni Marburg:
  https://idp.hrz.uni-marburg.de
- IdP Memcached StorageService (Eigenentwicklung):
  https://wiki.shibboleth.net/confluence/display/SHIB2/Memcached+StorageService

Danke für Ihre Aufmerksamkeit!

Noch Fragen?

→ Gern jetzt im Anschluss :-)

→ sonst per E-Mail: Manuel Haim, haim@hrz.uni-marburg.de

Quellennachweis Icons: „Crystal Project“ (GNU LGPL).