Hochverfügbarkeit mit **Reliable Server Pooling**

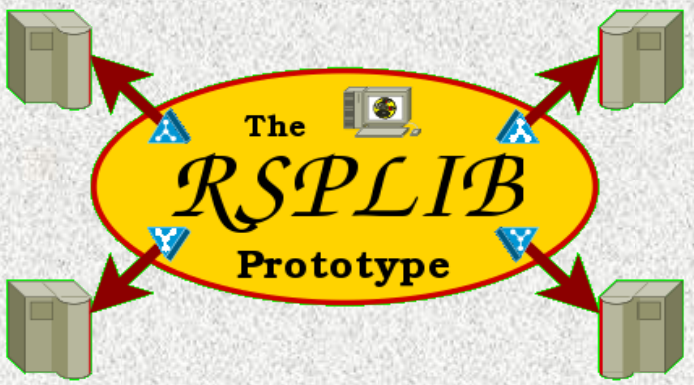
UNIVERSITÄT DUISBURG ESSEN

Dr. Thomas Dreibholz
Institut für Experimentelle Mathematik
Universität Duisburg-Essen
dreibh@iem.uni-due.de
http://www.exp-math.uni-essen.de/~dreibh

Inhalt

UNIVERSITÄT DUISBURG ESSEN

- Motivation
- Was ist Reliable Server Pooling?
  - Demo-Vorführung
  - Architektur und Terminologie
  - Protokoll-Stack
  - Eigenschaften
- Unsere Implementierung *RSPLIB*
  - Designziele
  - Installation und Test
  - Aufbau der Komponenten
  - Nutzung RSerPool-APIs
  - Scripting Service
- Unsere Aktivitäten zu Reliable Server Pooling

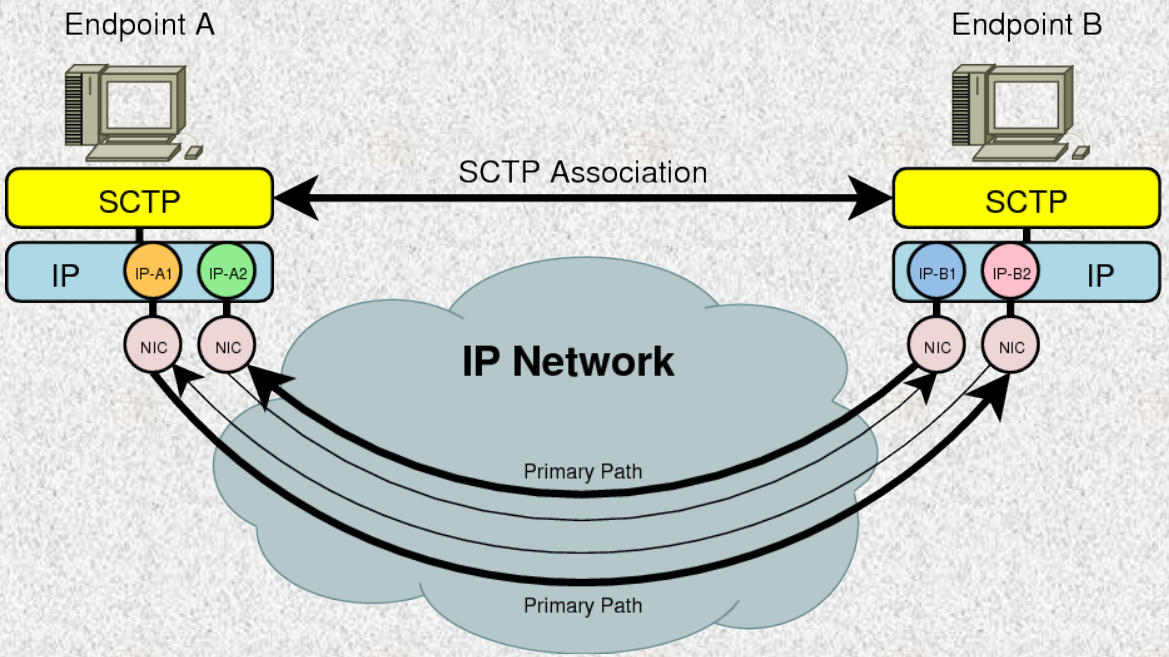


Thomas Dreibholz's Reliable Server Pooling Page
http://tdrwww.exp-math.uni-essen.de/dreibholz/rserpool/

Motivation (I)

UNIVERSITÄT DUISBURG ESSEN

- Ursprüngliche Motivation:
  - Telefonsignalisierung (SS7-Protokoll) über IP
  - Hohe Anforderungen an Verfügbarkeit
- Das Stream Control Transmission Protocol (SCTP) [RFC 2960]
  - „TCP Next Generation“
  - **Multi-Homing**
  - Multi-Streaming
  - Message-Framing
  - Schutz vor DoS-Angriffen
    - 4-Wege-Handshake
    - Verification Tag
  - Add-IP: dynamische Adreßrekonfiguration



- SCTP schützt vor einer Vielzahl von Netzproblemen, aber ...
- ... nicht vor einem **Serverausfall**

⇒ Konzept für **Server-Redundanz** ist **notwendig**

Motivation (II)

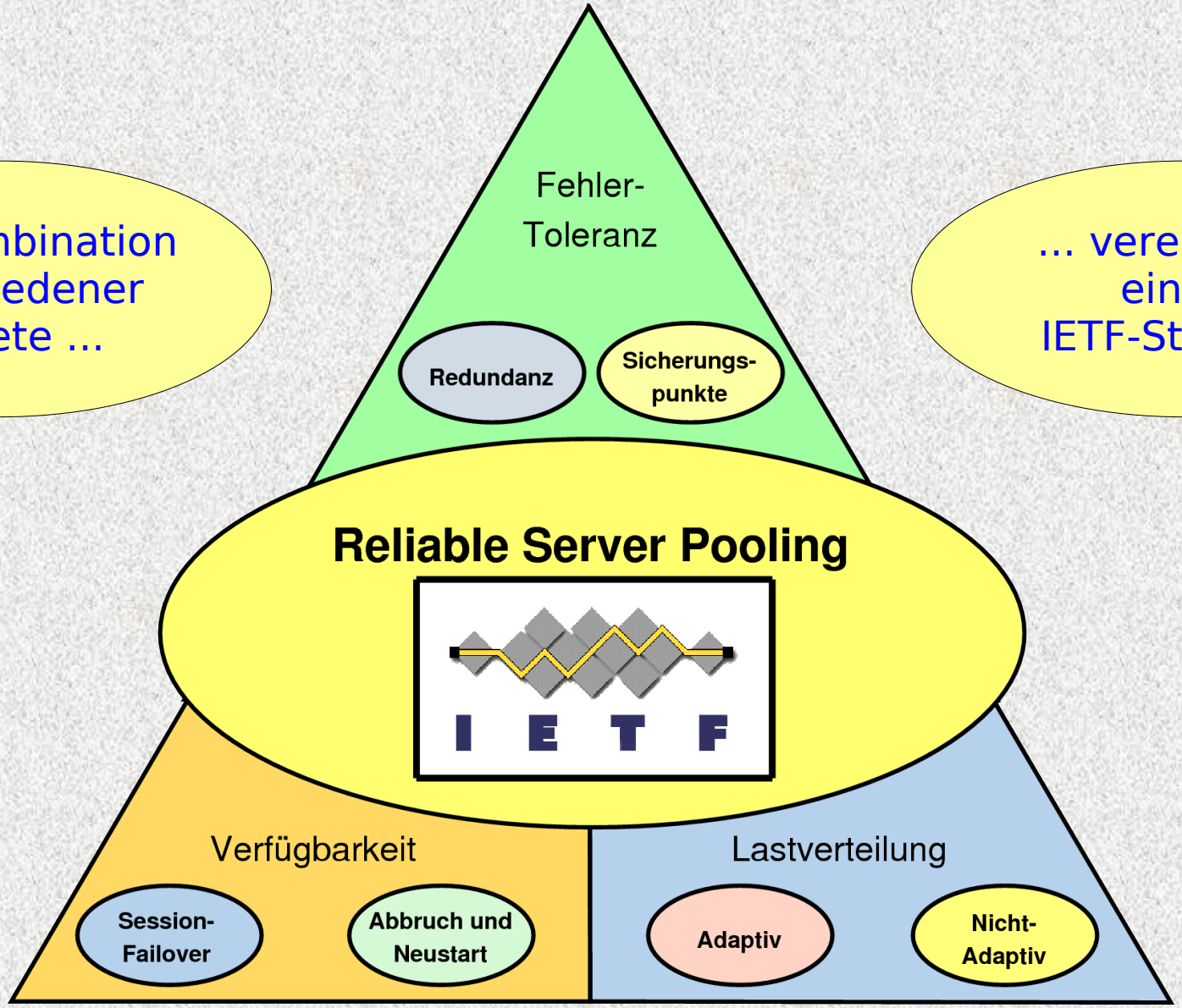
UNIVERSITÄT DUISBURG ESSEN

- Reliable Server Pooling (RSerPool)
  - Standardisierung in der IETF RSerPool WG
  - RSerPool ist ein **Framework** zur **Pool- und Sitzungsverwaltung**
- Anforderungen an RSerPool:
  - **Lightweight** (z.B. auch auf Embedded Devices nutzbar)
  - **Echtzeit** (schneller Failover)
  - **Skalierbar** (z.B. große (Firmen-)Netzwerke, nicht auf gesamtes Internet)
  - **Erweiterbar** (z.B. durch neue Serverauswahlregeln)
  - **Einfach** (automatische Konfiguration)
- Anwendungsgebiete für RSerPool:
  - Telefonsignalisierung (SS7) über IP, Voice over IP (VoIP)
  - **Distributed Computing**
  - Mobility-Management in Verbindung mit SCTP und Add-IP
  - IP Flow Information Export (IPFIX)
  - **Load Balancing**
  - Battlefield Networks, e-Commerce, m-Commerce, ...

Verwandte Themengebiete

UNIVERSITÄT DUISBURG ESSEN

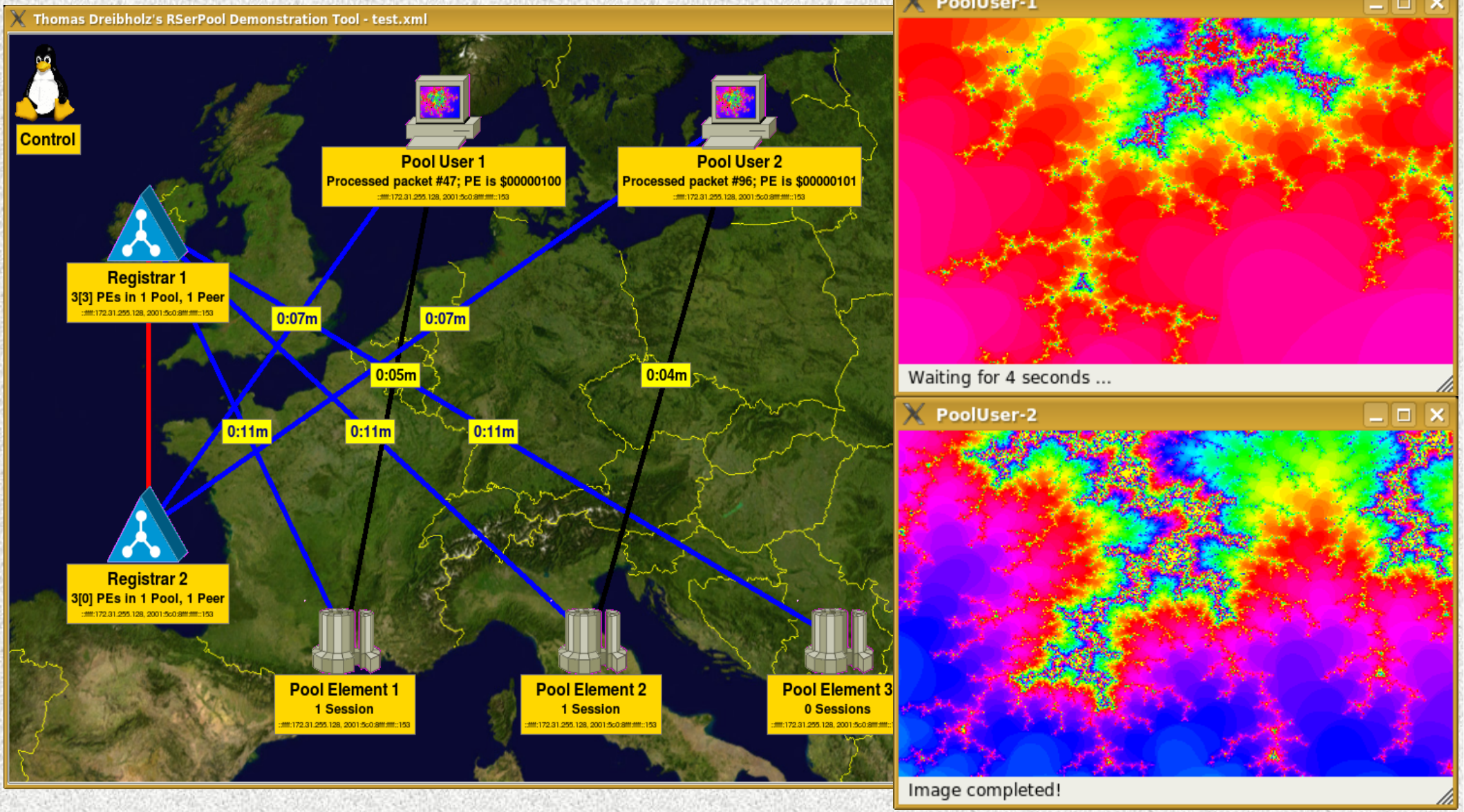
Eine Kombination verschiedener Gebiete ...



... vereinigt zu einem IETF-Standard

RSerPool Demo!

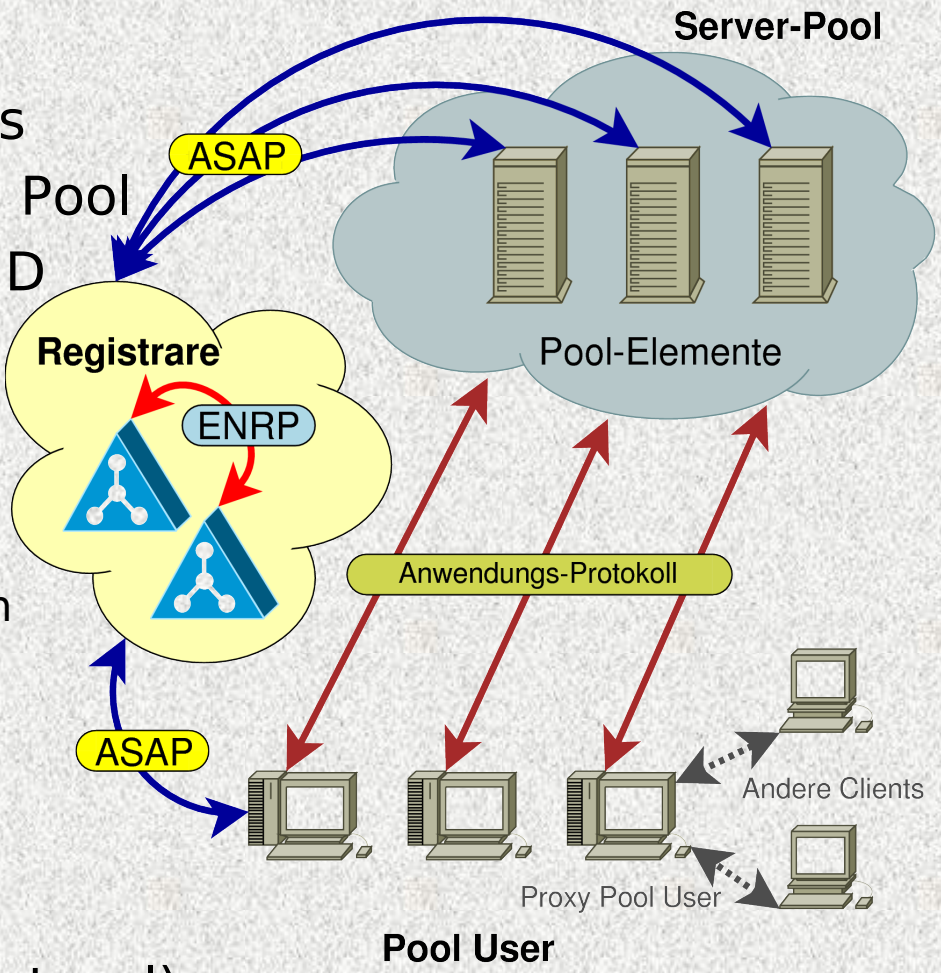
UNIVERSITÄT DUISBURG ESSEN



Reliable Server Pooling (RSerPool)

UNIVERSITÄT DUISBURG ESSEN

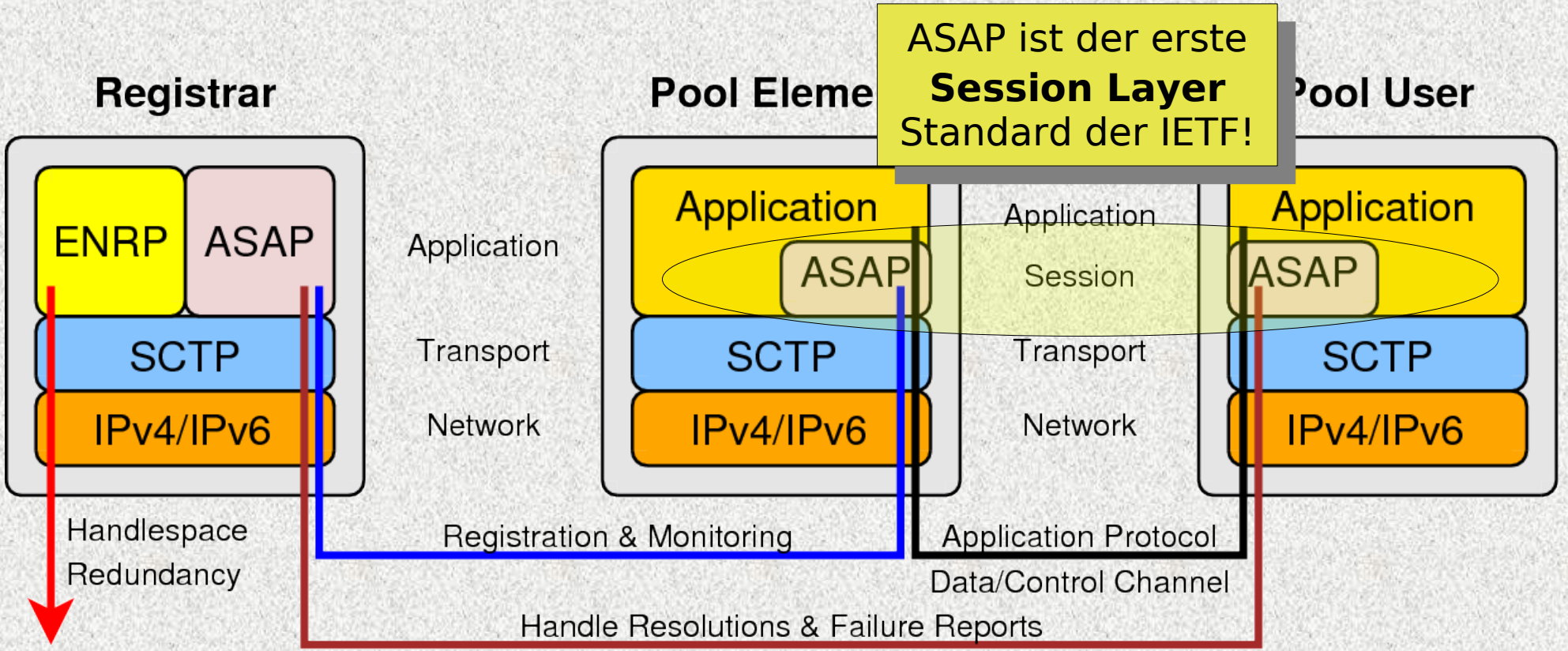
- Terminologie:
  - **Pool Element (PE)**: Server
  - Pool: Menge von PEs
  - PE ID: ID eines PE im Pool
  - Pool Handle: Eindeut. Pool-ID
  - Handlespace: Pool-Menge
  - Pool **Registrar (PR)**
  - **Pool User (PU)**: Client
  - Unterstützung für bestehende Anwendungen
    - Proxy Pool User (PPU)
    - Proxy Pool Element (PPE)



- Protokolle:
  - **ASAP** (Aggregate Server Access Protocol)
  - **ENRP** (Endpoint Handlespace Redundancy Protocol)

Der RSerPool-Protokollstack

UNIVERSITÄT DUISBURG ESSEN



Protokolle bei Reliable Server Pooling

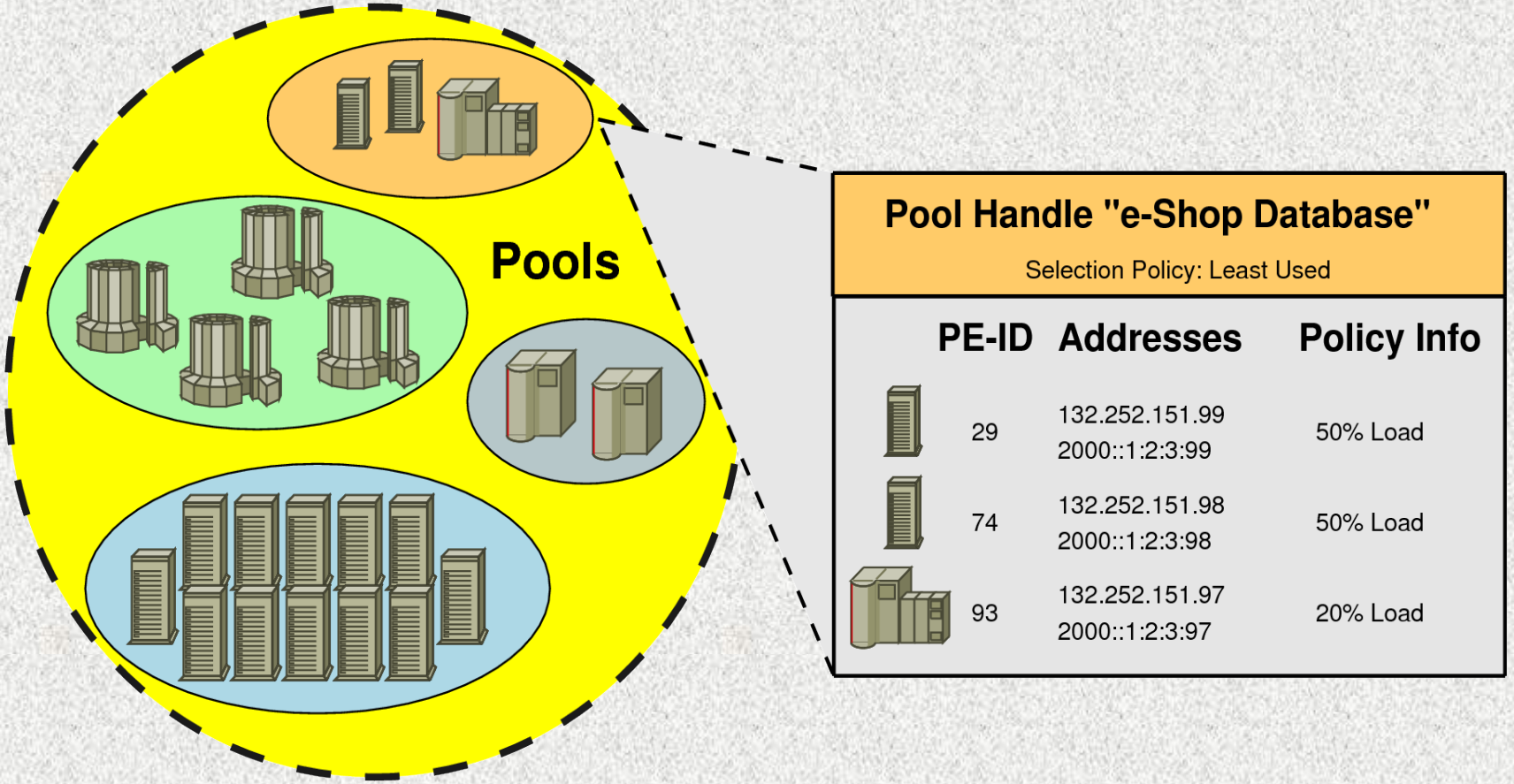
ASAP (Aggregate Server Access Protocol)

ENRP (Endpoint Handlespace Redundancy Protocol)

Der Aufbau des Handlespaces

UNIVERSITÄT DUISBURG ESSEN

- Menge nicht-leerer Pools
- “flach”, d.h. keine Hierarchie für PHs
  - sehr effizient implementierbar (siehe [FGCN2007-HsMgt][Contel2005])

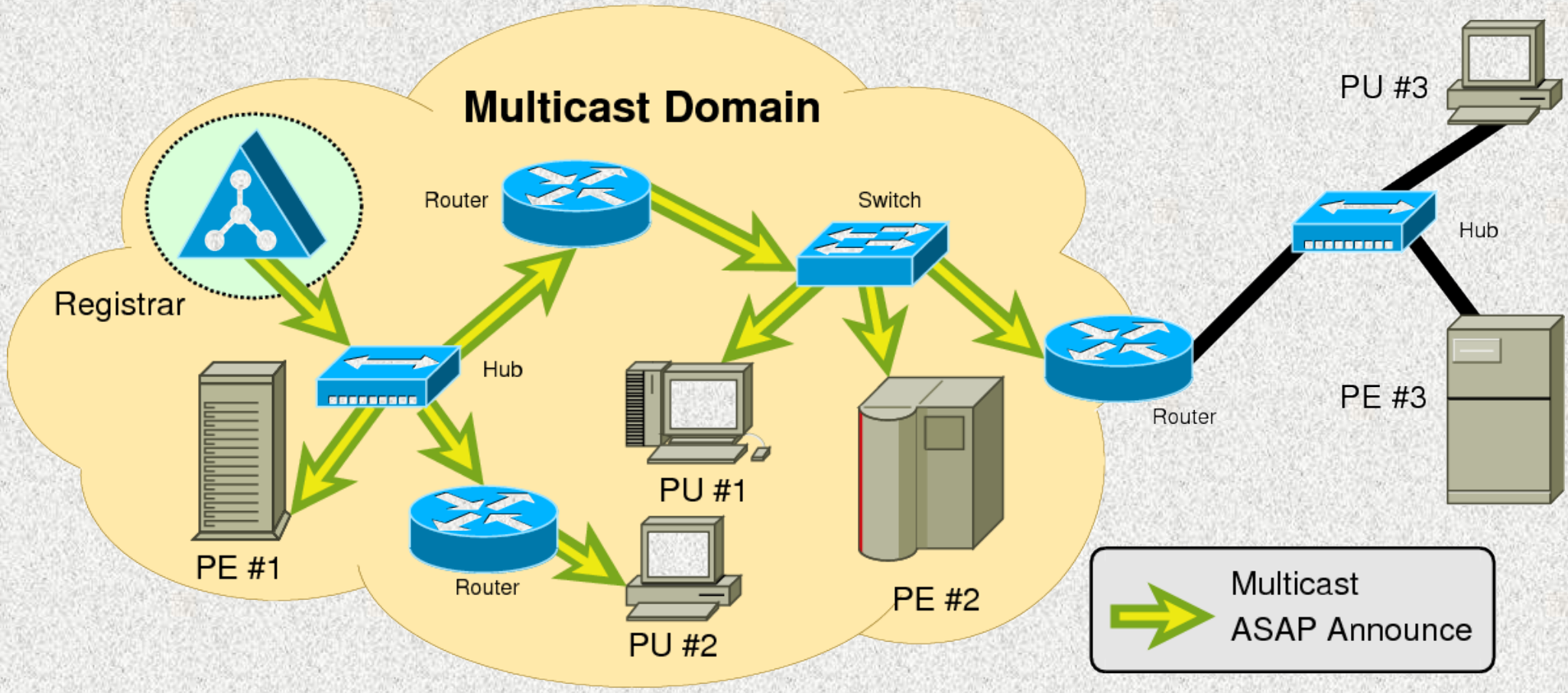


Automatische Konfiguration mittels Registrar-Announces

UNIVERSITÄT DUISBURG ESSEN

- Transport der Announces

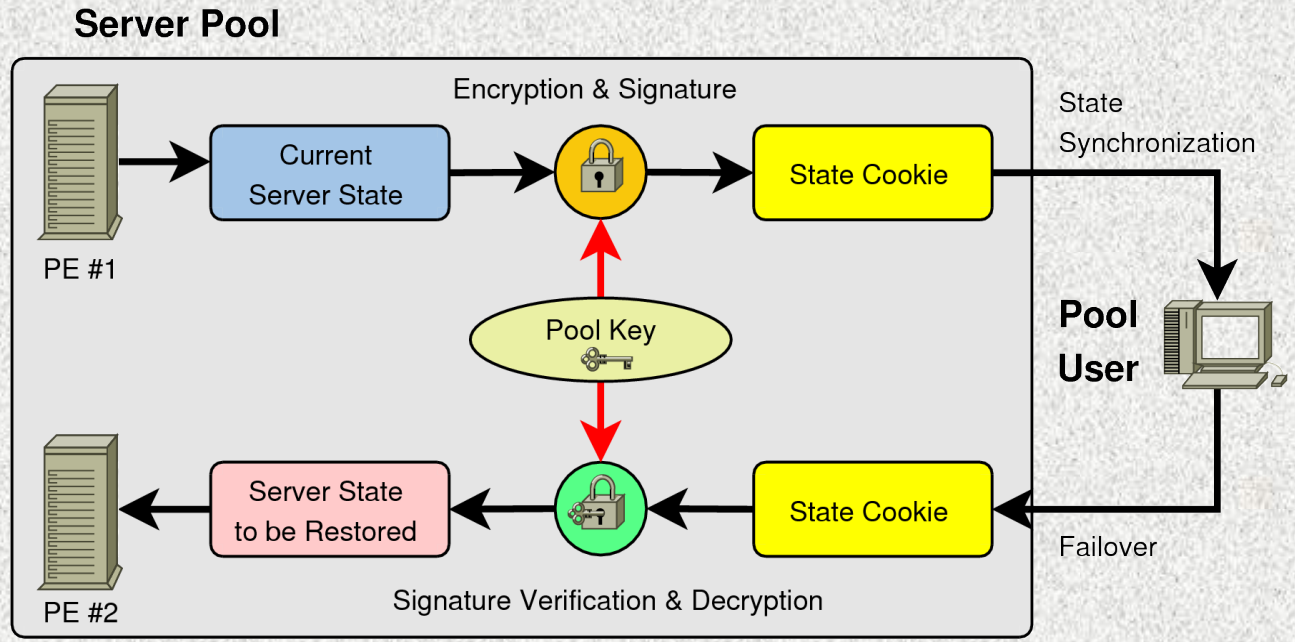
  als UDP-Pakete via IP-Multicast

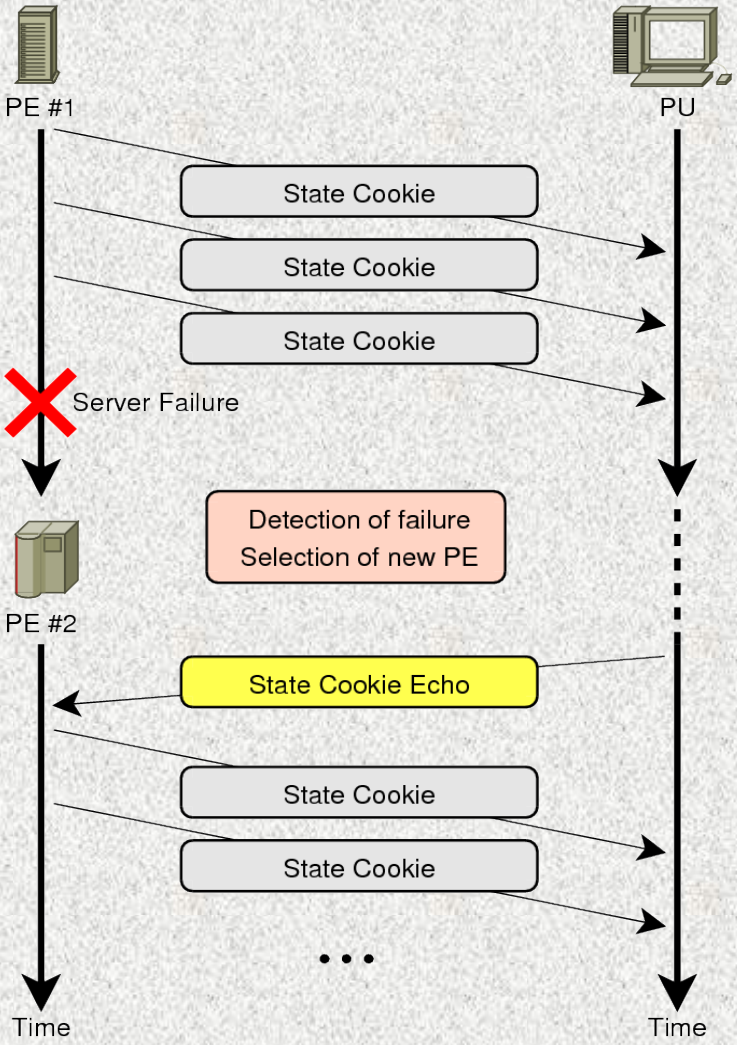


Failover mit Client-Based State Sharing

UNIVERSITÄT DUISBURG ESSEN

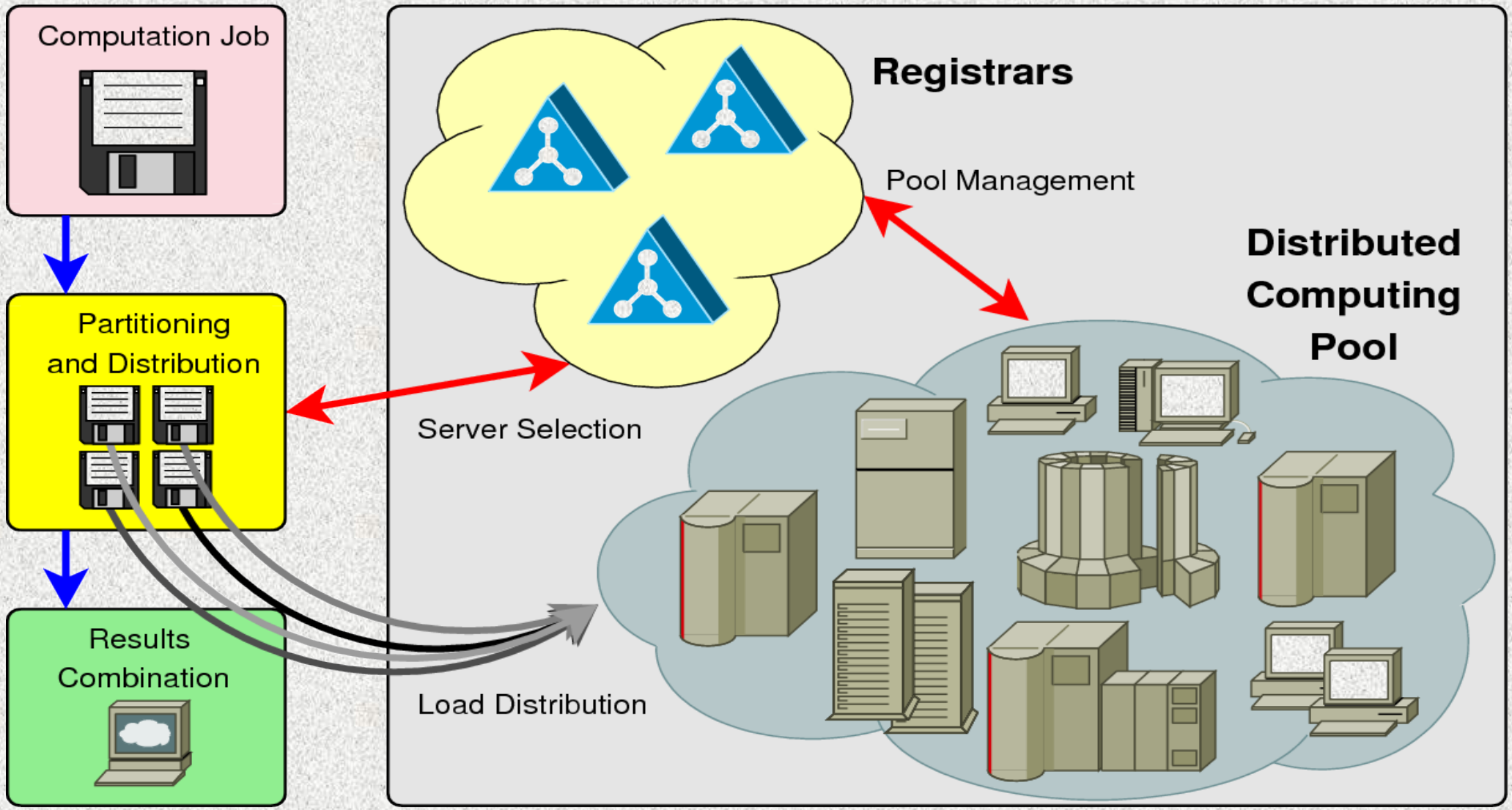
- Zum Failover notwendig:
  Neues PE muß den Sitzungszustand des alten PEs wieder herstellen
- Einfache Lösung für viele Anwendungen:
  Benutzung von State Cookies [LCN2002]
  Ist jetzt Teil des ASAP-Protokolls





Anwendungsszenario für RSerPool: Real-Time Distributed Computing

UNIVERSITÄT DUISBURG ESSEN

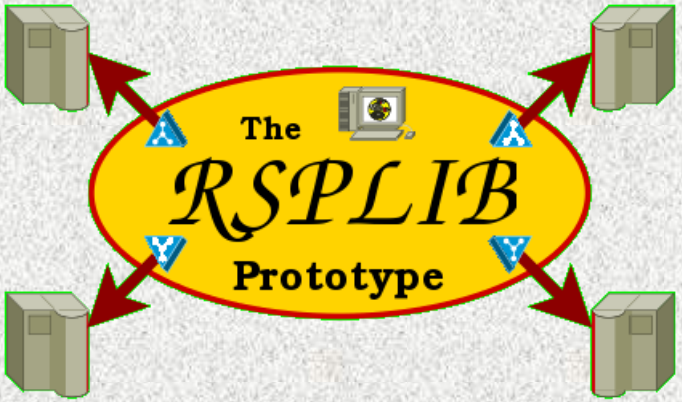


Beschrieben in [draft-dreibholz-rserpool-distcomp-03.txt]

Der *RSPLIB*-Prototyp

UNIVERSITÄT DUISBURG ESSEN

- Designentscheidungen
  - Open Source, GPLv3-Lizenz
  - Plattformunabhängigkeit
    - Systeme: Linux, FreeBSD, MacOS X, Solaris
    - CPUs: x86, x86_64, PPC, MIPS
  - Implementiert in ANSI-C
- Grundlegende Bestandteile
  - *RSPLIB* Library für PUs und PEs
    - ASAP-Protokoll (PU/PE-Seite)
  - Registrar
    - ASAP-Protokoll (PR-Seite)
    - ENRP-Protokoll
  - Demo-System, weitere Beispiele



Entwickelt in Kooperation mit Siemens AG, München
mit Unterstützung durch BMBF und DFG

Thomas Dreibholz's Reliable Server Pooling Page
http://tdrwww.iem.uni-due.de/dreibholz/rserpool/

Installation

UNIVERSITÄT DUISBURG ESSEN

- Download des Source-Archivs:
  - http://tdrwww.iem.uni-due.de/dreibholz/rserpool/
- Abhängigkeiten:
  - *lksctp*-Paket (Library für Zugriff auf Kernel-SCTP)
    (Alternative: unsere eigene Userland-SCTP-Implementierung *sctplib/socketapi*)
  - *Qt3*-Entwicklerdateien (für Fraktal-Demo, optional)
  - *BZip2*-Entwicklerdateien (für CalcApp-Testapplikation)
  - Debian/Ubuntu:
    ```
    sudo apt-get install libsctp-dev libbz2-dev libqt3-mt-dev
    ```
- Entpacken und Übersetzen
  - `tar xzf rsplib-<Version>.tar.gz`
  - `cd rsplib-<Version>`
  - `./configure --enable-kernel-sctp --enable-qt`
  - `make`
  - `sudo modprobe sctp` (ggf. Kernel-Modul für SCTP laden)

Ein kleiner Test

UNIVERSITÄT DUISBURG ESSEN

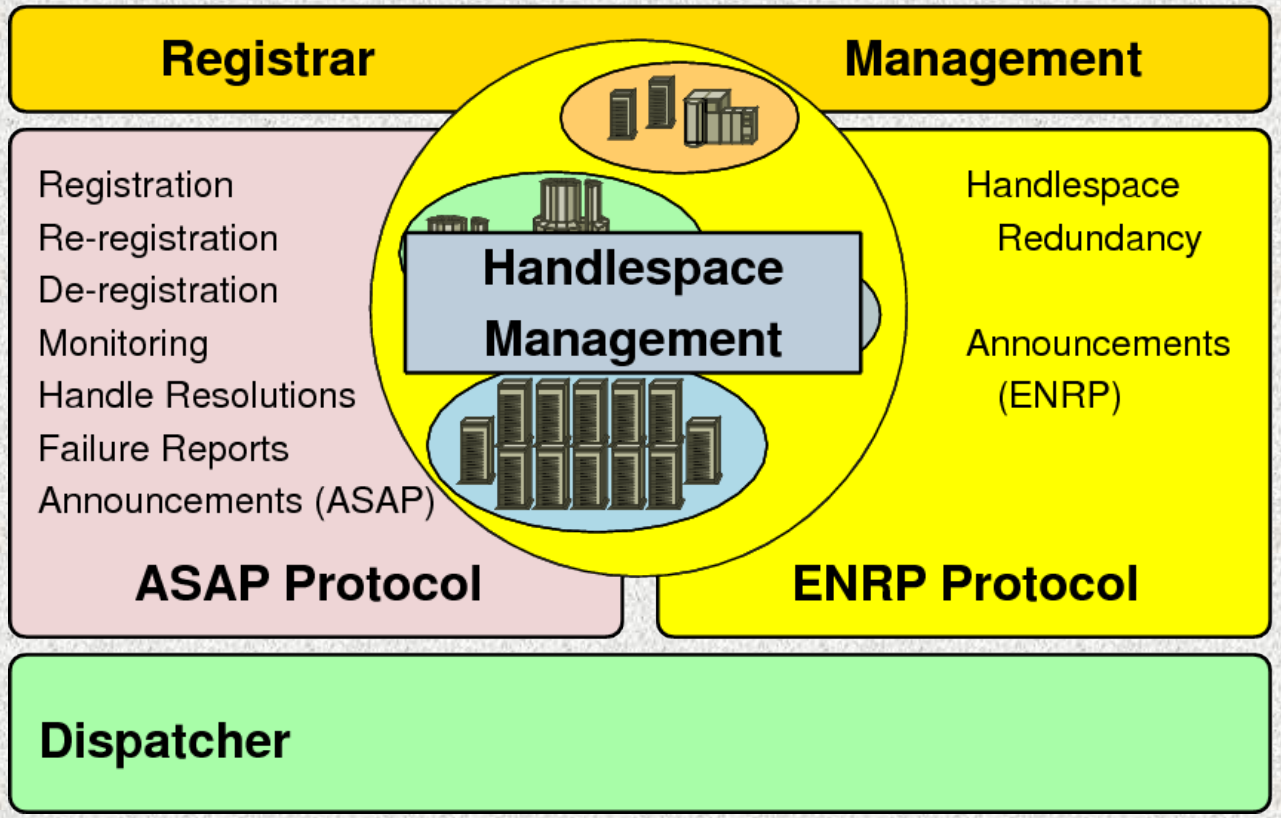
- Registrar starten
  - `registrar`
  - Bei Loopback-Betrieb (d.h. auf einem Rechner ohne Netzanbindung):
    - Host benötigt mindestens private IP-Adresse (z.B. 192.168.x.y)
    - Interface muß Multicast-Flag gesetzt haben
    - `sudo ifconfig dummy0 192.168.100.200 netmask 255.255.255.0 up multicast`
- Fraktal-PEs starten:
  - `server -fractal`
- Fraktal-PUs starten:
  - `fractalpooluser`

Weitere Informationen gibt es im
RSPLIB Handbook
unter
http://tdrwww.iem.uni-due.de/dreibholz/rserpool/

Der Aufbau des Registrars

UNIVERSITÄT DUISBURG ESSEN

- Dispatcher:
  - Plattformspezifische Funktionen:
    - Timer
    - Sockets
    - Threads
- Protokolle:
  - ASAP
    - PR↔PE
    - PR↔PU
  - ENRP (PR↔PR)
- Registrar-Verwaltung:
  - Zugriffskontrolle
  - Adreßverifikation und -filterung

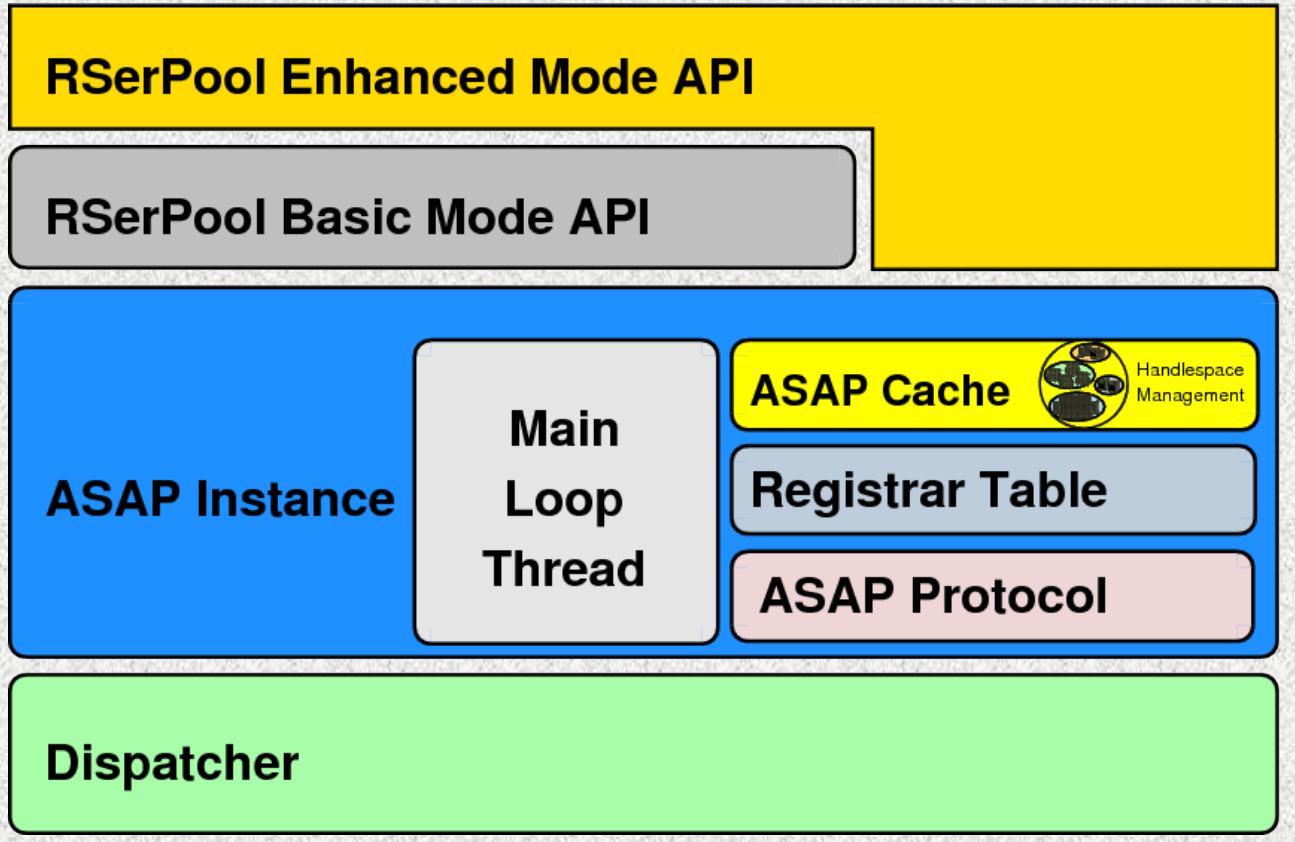


- Handlespace-Management (siehe [FGCN2007-HsMgt][Contel2005])

Der Aufbau der PU/PE-Library

UNIVERSITÄT DUISBURG ESSEN

- Dispatcher
- ASAP-Instance:
  - ASAP-Protokoll
    - PE↔PR
    - PU↔PR
    - PU↔PE
  - ASAP-Thread
    - Anfrage-Pipelining
  - Liste von Registraren
    - aus Announces
    - statische Konfig.
  - Cache für PE-Auswahl
- RSerPool-APIs:
  - Basic Mode
  - Enhanced Mode



Die beiden RSerPool-APIs

UNIVERSITÄT DUISBURG ESSEN

- Basic Mode API
  - Nur die Grundfunktionen (Registrierung, Deregistrierung, Handle Resolution)
  - PU $\leftrightarrow$ PE-Kommunikation **übernimmt Anwendung** komplett selbst!

- Enhanced Mode API
  - Kompletter **Session-Layer**
  - Für PEs:
    - Registrierungsverwaltung
    - Verwaltung eingehender Sitzungen (Sessions)
    - Client-basiertes State Sharing
  - Für PUs:

    Sessions mit Pools, inklusive
    - Auswahl eines PEs
    - Aufbau, Überwachung und **Verwaltung** einer **Transportverbindung**
    - **Failover**-Unterstützung
    - Cookie-Speicherung und Failover mittels Cookie

Das API der *RSPLIB*-Library: Enhanced Mode für PUs

UNIVERSITÄT DUISBURG ESSEN

- API analog zu TCP-Sockets
  - Ablauf bei TCP-Sockets: *socket*() -> *connect*() -> ... -> *close*()
  - Jetzt: Session (RSerPool-Socket) statt einzelner Verbindung!

```
/* Create session */
session = rsp_socket(0, SOCK_STREAM, IPPROTO_SCTP);
rsp_connect(session, "MyPool", ...);

/* Run application: file download */
rsp_send(session, "GET Linux-CD.iso HTTP/1.0\r\n\r\n");
while((length = rsp_recv(session, buffer, ...)) > 0) {
   doSomething(buffer, length, ...);
}

/* Close session */
rsp_close(session);
```

- Anmerkung:

  doSomething() erhält ggf. Wiederholungen – je nach Cookie-Intervall!

Das API der *RSPLIB*-Library: Enhanced Mode für PEs

UNIVERSITÄT DUISBURG ESSEN

- API analog zu TCP-Sockets
  - Ablauf bei TCP-Sockets: *socket*() -> *bind*() -> *listen*() -> *accept*()
  - Wieder: Sessions (d.h. RSerPool-Sockets) statt einzelner Verbindungen

```
void serviceThread(session)
{
  rsp_recv(session, command, ...);
  if(command is a cookie) {
    /* Got a cookie -> restore session state */
    Restore state;
    rsp_recv(session, command, ...);
  }
  do {
    /* Handle commands from pool user */
    Handle command;
    rsp_send_cookie(session, current state);
    rsp_recv(session, command, ...);
  } while(session is active);
  rsp_close(session);
}


int main(...)
{
```

```
  /* Create and register pool element */
  poolElement = rsp_socket(0,SOCK_STREAM,IPPROTO_SCTP);
  rsp_register(poolElement, "MyPool", ...);

  /* Handle incoming session requests */
  while(server is active) {
    /* Wait for events */
    rsp_poll(poolElement, ...);

    if(incoming session) {
      /* Accept new session */
      session = rsp_accept(poolElement, ...);
      Create service thread to handle session;
    }
  }

  /* Deregister pool element */
  rsp_deregister(poolElement);
  rsp_close(poolElement);
}
```

Der Scripting-Service: RSerPool in Shell-Skripten

UNIVERSITÄT DUISBURG ESSEN

- Weitere Beispielanwendung: **Scripting Service**
  - **Scripting PE**:
    - Erhält von PU Tar/GZip-Datei
    - Datei wird entpackt, darin enthaltenes Skript wird ausgeführt
    - Resultate werden Tar/GZip-gepackt und an PU zurückgeschickt
  - **Scripting PU**:
    - Erhält (vom Anwender) Tar/GZip-Datei mit Skript (und Eingabedaten)
    - Verteilt Tar/GZip-Datei an Scripting-PE im Pool
    - Bekommt Ergebnisdatei zurück
- Anwendungsbeispiel:
  - **Verteilung** von **Simulationsläufen**
  - Aufwand dafür: nur ca. 50 Zeilen *bash*-Code!
- Aktuelle Arbeiten am Scripting Service (in Studentenprojekten):
  - Sicherheit! Idee: Verwendung einer **Virtualisierungslösung** (z.B. Xen)
  - Failover-Handhabung: Application **Checkpointing**

Unsere RSerPool-Aktivitäten

UNIVERSITÄT DUISBURG ESSEN

- Forschung im Rahmen eines DFG-Projektes seit Oktober 2004
  - Simulationsmodell *RSPSIM*
  - Prototypimplementierung *RSPLIB*

Diplom-, Bachelor-, Master- oder Studien-Arbeit im RSerPool? Sprechen Sie mit uns!

von der Simulation in die Wirklichkeit

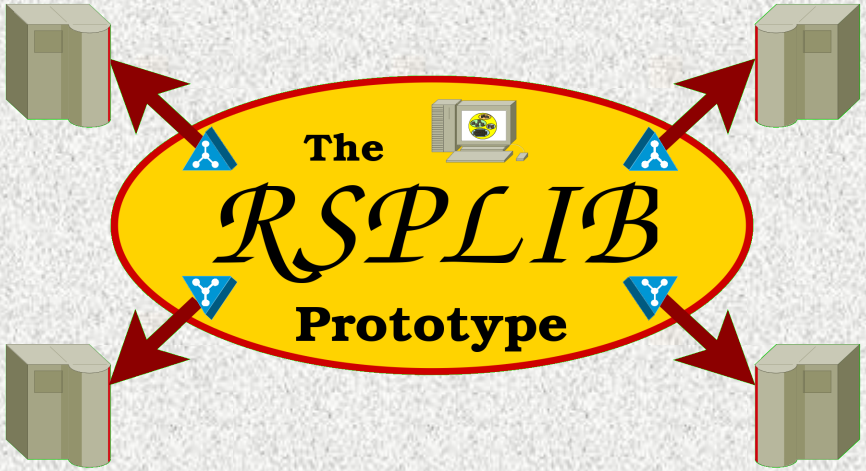
aus der Forschung in die Anwendung

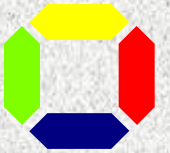
- Standardisierung in der IETF
  - Mitarbeit an 4 **Working-Group-Drafts** ...
    - draft-ietf-rserpool-overview-02.txt
    - draft-ietf-rserpool-policies-07.txt
    - draft-ietf-rserpool-mib-04.txt
    - draft-ietf-rserpool-api-00.txt
  - ... und mehreren **Individual Submissions**
  - IETF-Standardisierung läuft über „running code“ - den haben wir!
  - *RSPLIB* ist die weltweit erste vollständige Implementierung
    - **Open Source** (GPLv3-Lizenz)
    - **Referenz-Implementierung** der IETF RSerPool WG

Noch Fragen?

UNIVERSITÄT DUISBURG ESSEN

UNIVERSITÄT DUISBURG ESSEN





Fortsetzung folgt ...

Projekt-Homepage:

http://tdrwww.iem.uni-due.de/dreibholz/rserpool/

Thomas Dreibholz, dreibh@iem.uni-due.de