13th IFIP/IEEE International Workshop on
Distributed Systems: Operations & Management

October 21-23, 2002, Montreal, Canada

A Hot-Failover State Machine for Gateway Services and its Application to a Linux Firewall

Harald Roelle

MNM TEAM
MUNICH NETWORK MANAGEMENT TEAM

Department of Informatics, University of Munich
Email: roelle@informatik.uni-muenchen.de

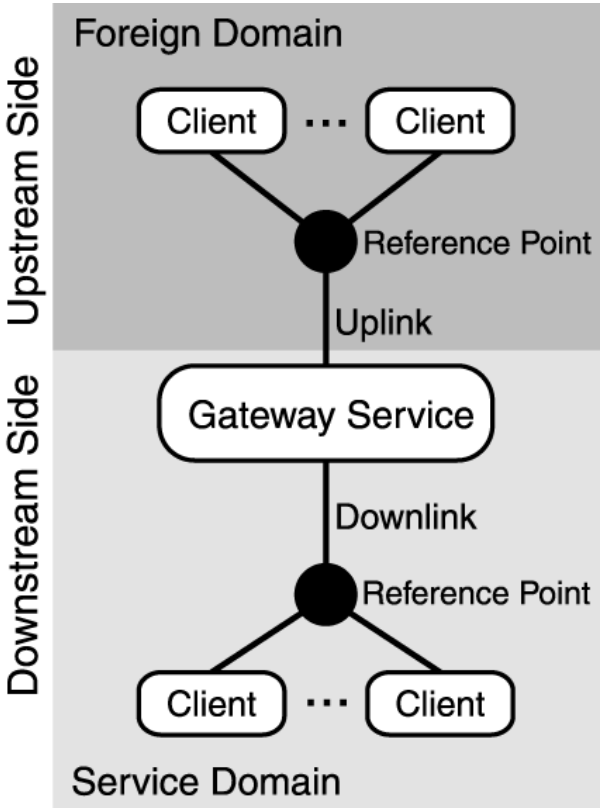
Motivation and Abstracted Scenario

- Lower budget environments: Gateways built upon off-the-shelf hard- and software

⇒ Rising number of components increase probability of faults

⇒ Adequate fault-tolerance solution necessary

- Gateway service
  - Resides in one domain
  - Linked to different domains
- Single point of failure by gateway service



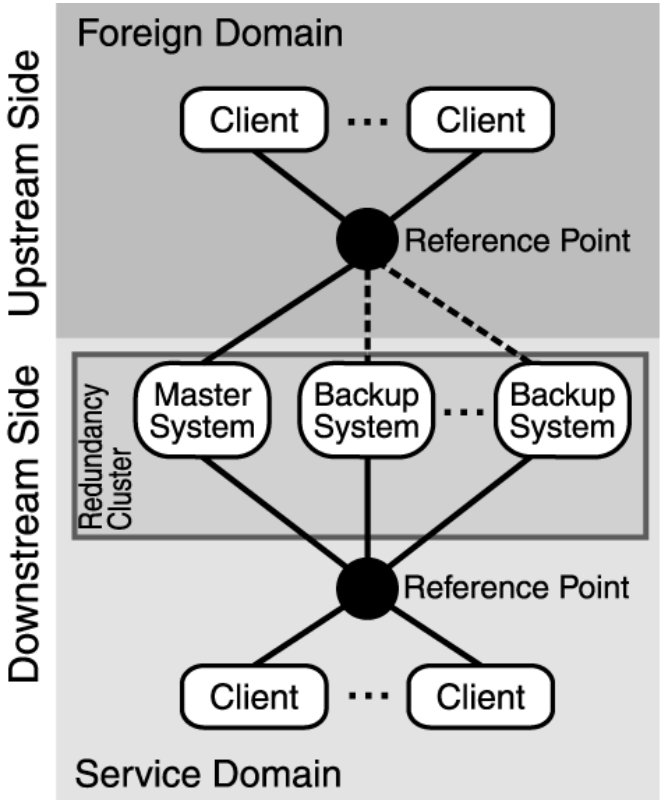
Motivation and Abstracted Scenario

- Lower budget environments: Gateways built upon off-the-shelf hard- and software

⇒ Rising number of components increase probability of faults

⇒ Adequate fault-tolerance solution necessary

- Redundancy cluster
  - System providing service: Master
  - Backup systems ready to take over service provisioning
- Problems to solve:
  - Detect failures
  - Hand-over service provisioning



Important Requirements for a Generic Solution

- Service monitoring from client's perspective
  → Both monitoring service and its accessibility necessary

- Separation of logic and actions
  → Keeps solution applicable for different concrete services

- Minimal active links to foreign domain
  → Security threats lowered by keeping upstream links down until needed

- Independence from specific services and communication technology and communication primitives
  → No changes in surrounding environment necessary

- No need for extra hardware
  → Flexibility of used systems and short setup times

Related Work

- Virtual Router Redundancy Protocol (VRRP, RFC2338)
  - – Assumes IEEE 802 / IP
  - – Only simple 3 state machine specification
  - + Good inspiration, also addresses management
- Hot Standby Router Protocol (HSRP, RFC2281)
  - – Requires dynamic routing protocol
  - + In-detailed state machine was valuable starting point
- IETF Working Group for Reliable Server Pooling
  - – Solution 1: Introduces new, special protocol
  - – Solution 2: Classical architecture with load balancers
  - + Helped identifying requirements
- High-Availability Linux (HA-Linux) Project
  - – Limited setup and monitoring
  - + Considers security explicitly
- Linux Virtual Server (LVS) Project: Implements VRRP
- Load Balancers: Don't solve the problem, but are subject itself

Solution by Generic State Machine

Main design principles:

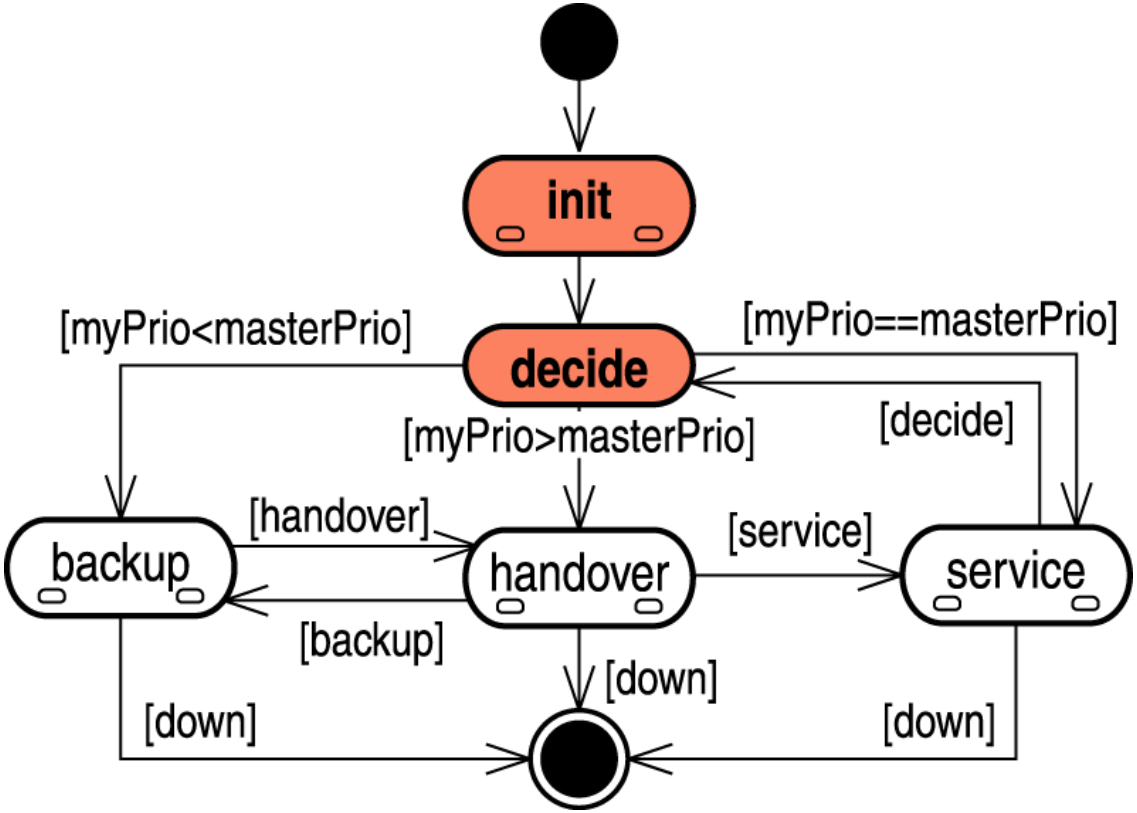
- Control both observation of service and communication links
- Communication on present links to control handover
- Arbitrary number of backup hosts
- Dynamically add/remove hosts from redundancy cluster

Main components:

- Host-local state machine
  - Specifies service and link monitoring
  - Coordinates handover of service functionality
- Messages
  - Inform other hosts of status changes
  - Trigger actions on remote hosts
- Status table
  - All hosts in cluster priorized by a total order
  - Maintained and distributed by current master
- Local alarm timers
  - Trigger local actions

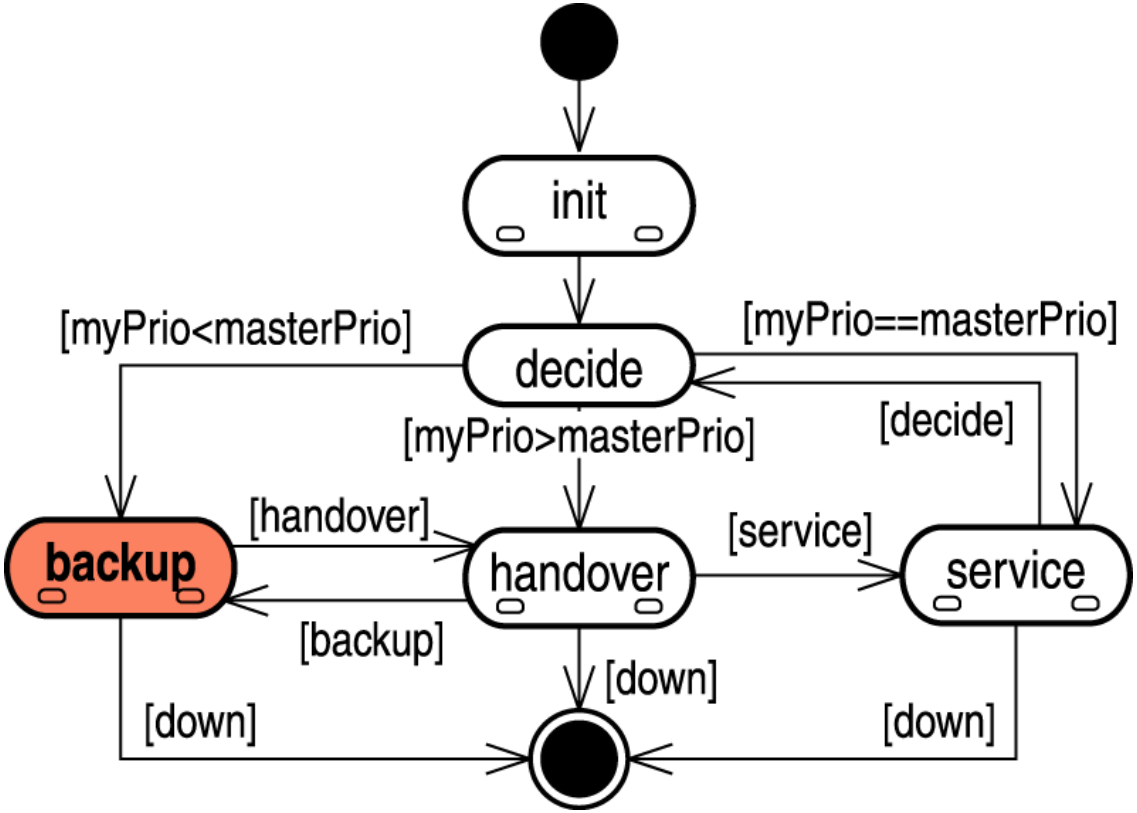
Main State Machine

- Init and Decide main states
  - Detects initial priority of host
  - Differentiates initial bootstrap and dynamic addition
  - Decide on initial role of host



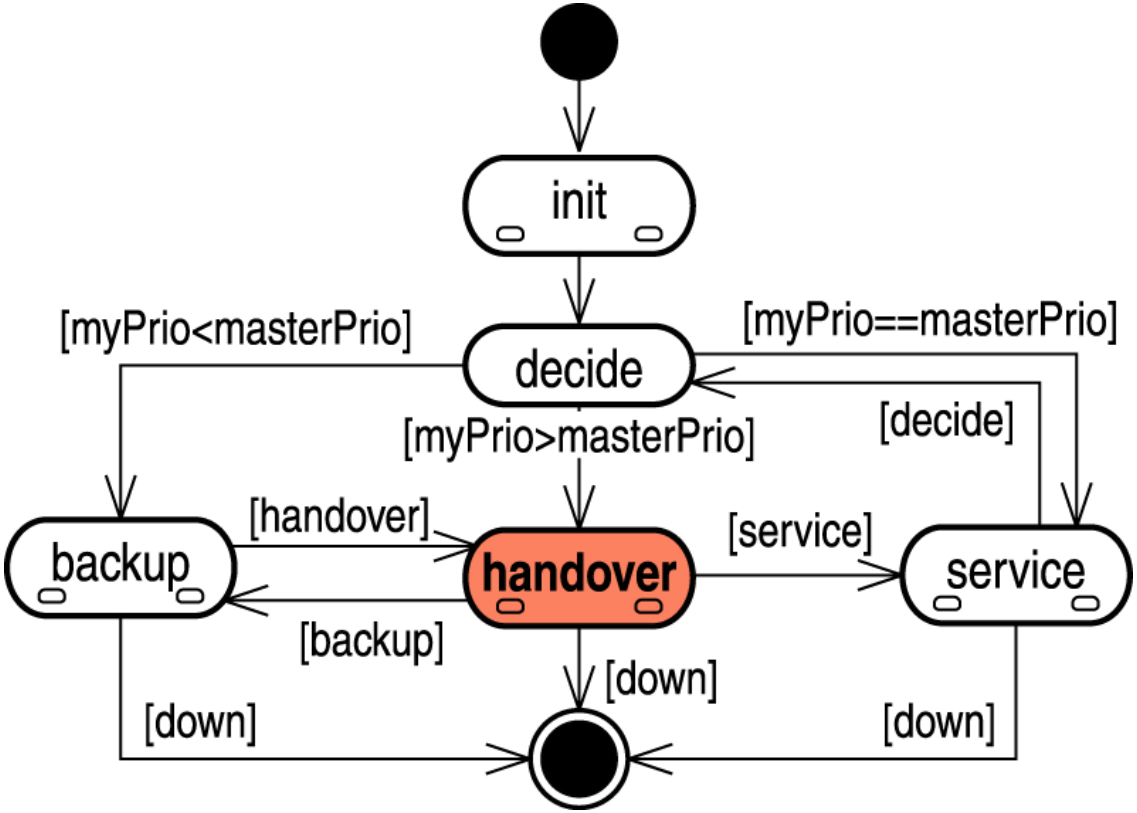
Main State Machine

- Backup main state
  - Performs active service monitoring from client's perspective
  - Triggers local monitoring on master host



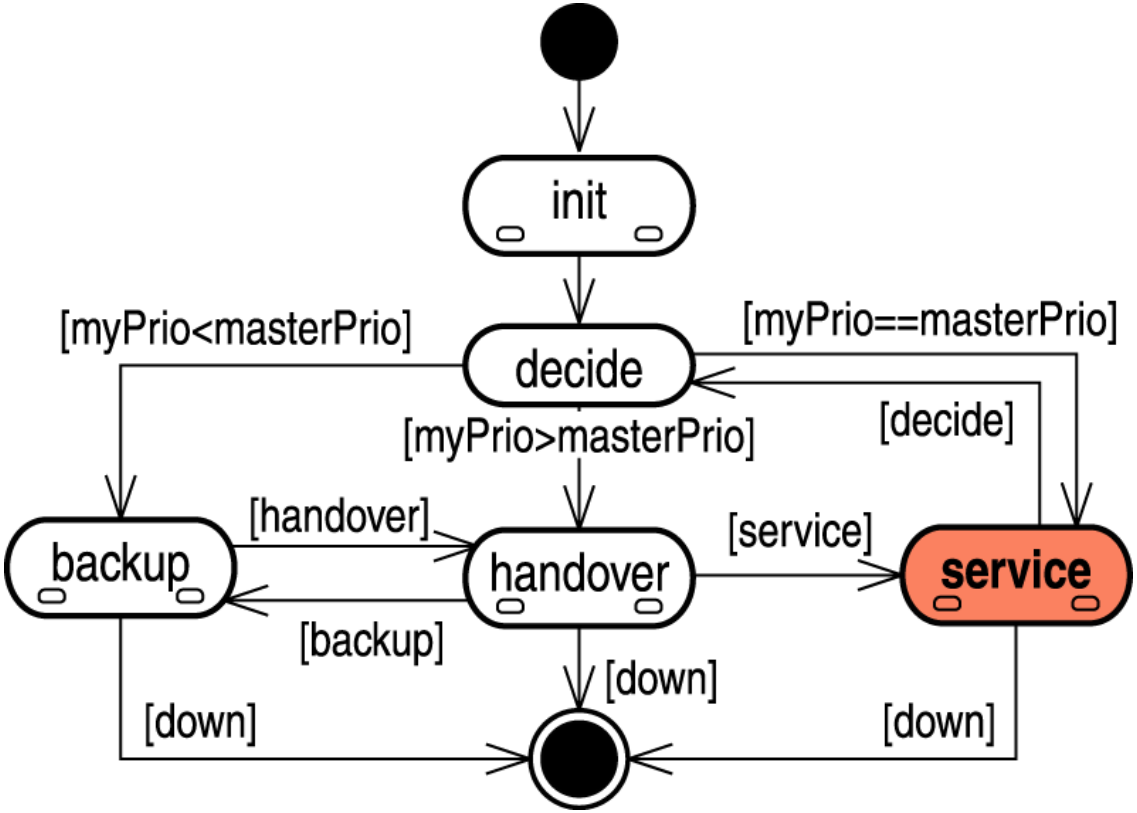
Main State Machine

- Handover main state
  - Initiates transfer of service provisioning
  - Distinguishes real service failures from failures in backup's links
  - Activates upstream link

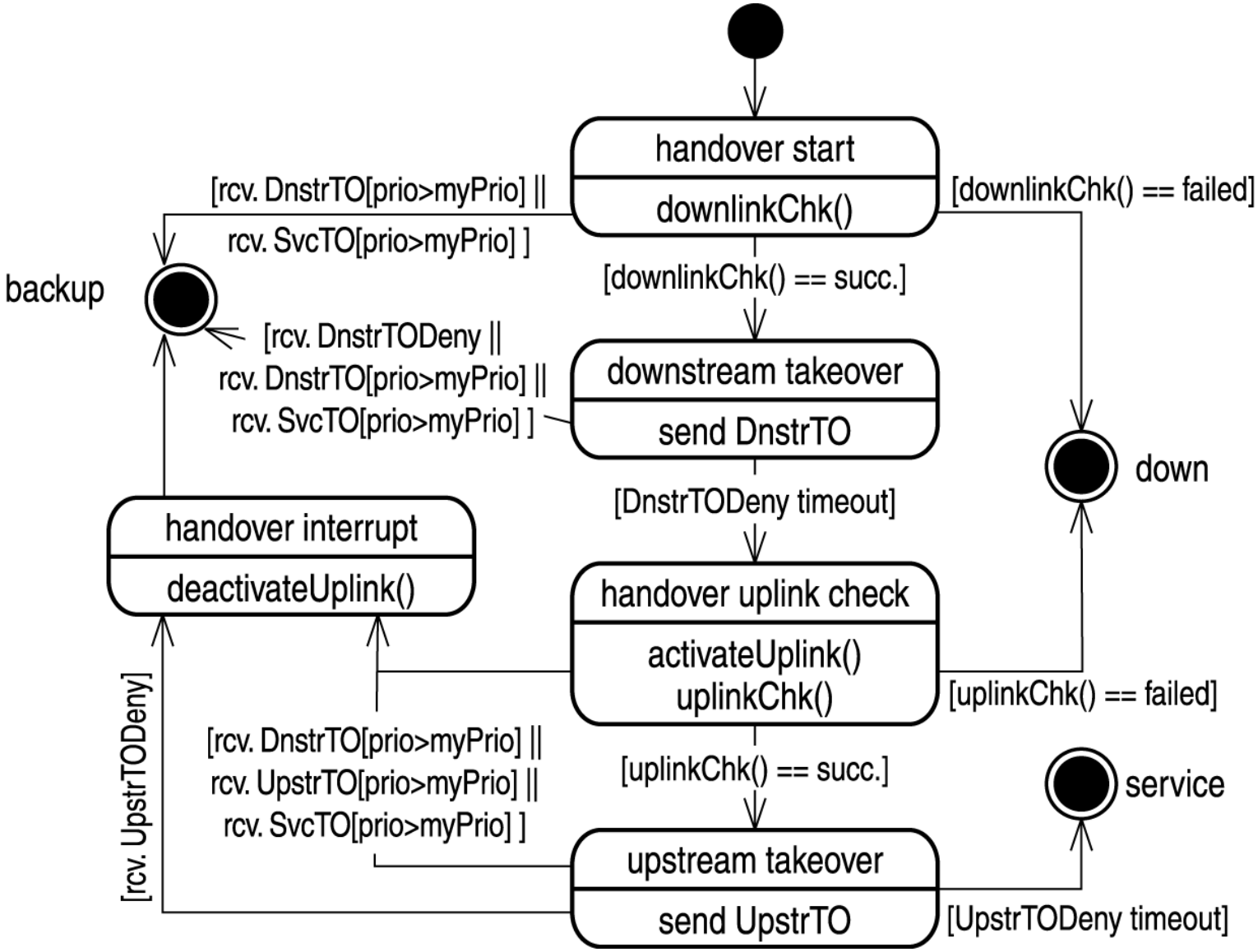


Main State Machine

- Service main state
  - Designates a host as the master
  - Maintains and distributes status table



Handover Main State



Customizable Procedures

- Realizes separation from logic and actions:
  State machine adoptable to concrete services without altering the handover logic
- Monitoring and Testing Procedures
  - Deliver boolean results
  - Must be positive definite
  - 2 procedures for up- and downlink checks
  - 2 procedures for local and remote service checks
- (De-) Activation Procedures
  - Only used to carry out actions, no return value
  - Success checking ensured by logic of state machine
  - 2 procedures for uplink (de-) activation
  - 2 procedures service (de-) activation
- Client Related Procedures
  - Announce changes on up- / downstream side
  - 4 procedures: takeover and release on either up- and downstream side

Prototype: Universal IP Service Daemon

- Implemented on Linux in C as user space daemon
- Assumes Layer 3 to be IP
- Roving IP addresses on Up/Downlink via "single link multihoming"
- Implemented procedures:
  - Client related: Announce address changes via broadcast pings
  - Monitoring: Downlink by broadcast ping, uplink by ping of next hop router
- Remaining procedures left for implementation as external program/script
- Scripts for packet-filtering Firewall on Ethernet:
  - Uplink (De-) Activation: (un-) loading card driver
  - Service (De-) Activation: iptables

Conclusion

- In-depth specification of generic handover logic
- Fulfills requirements for gateway services
- Lightweight solution without extra hardware
- Handover logic remains unchanged on application for specific service by customizable procedures delivering hooks for specific actions
- Directly implementable
- Status transfer not focused
- Examples of use:
  - Standalone solutions
  - Integration into services
  - Basis for further development, e.g. of VRRP

Current and Future Work

- Formal verification
  - In cooperation with Alexander Knapp and Stefan Merz (Research group of Prof. Martin Wirsing, LMU, http://www.pst.informatik.uni-muenchen.de/)
  - Using model checking tools
  - Logic verification: almost finished
  - Timing verification: t.b.d.

- Specify security mechanisms on level of the state machine
  - Authentication mechanisms

- Multi service redundancy
  - Coordinate multiple services by single state machine

- Active feedback of backup hosts
  - Backups influence ranking in priority table
  - Enables load balancing in case of failure