State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray, Claas Lorenz, Alexander Müssig, Steffen Gebert, Thomas Zinner, Phuoc Tran-Gia

comnet.informatik.uni-wuerzburg.de
Motivation

- **Traditional deployment**: Single hardware device separates networks
Motivation

- **Traditional deployment**: Single hardware device separates networks
- **Resilience**: Secondary instance as hot standby
Motivation

- **Traditional deployment**: Single hardware device separates networks
- **Resilience**: Secondary instance as hot standby
- **Softwarization**: Firewall software running in virtual machine
Motivation

- **Traditional deployment**: Single hardware device separates networks
- **Resilience**: Secondary instance as hot standby
- **Softwarization**: Firewall software running in virtual machine
- **NFV**: Multiple active VNF instances scaling horizontally
Motivation

- First approaches described by the research community

- Demands:
  - Horizontal scalability: Scale in and scale out
  - Resilience: Failover mechanisms

- Open questions:
  - Detailed design specifications
  - Suitable synchronization mechanisms
  - Performance evaluations

→ No existing firewall VNF meets these requirements
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Firewalling

Packet Filter
OSI-Layer 3 (L4)
- `*: * → 132.187.15.12:80`
- `*: * → *:* (implicit)`

Stateful Firewall
OSI-Layer 4
- `*: * → 132.187.15.12:80`
- `*: * → 132.187.15.12:80`

Application Layer Firewall
OSI-Layer 7
- `*: *, HTTP GET /index.html`
- `*: *, HTTP GET /evil.html`

State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Example: Traffic Flow

State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Example: Fail over

State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Example: Scale Out

State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Implementation

- **Erlang** is a functional programming language by Ericsson:
  - Provides high availability
  - Specialized for multithreading

- Prototypical implementation:
  - Stateful firewall: Every state is logged and packets are inspected
  - Cluster size expands dynamically

- Parameter configuration:
  - Synchronization level
  - Data access
  - Synchronization strategy
State synchronization for fast failover of stateful firewall VNF

TCP states to synchronize

Stateful packet filtering

Nicholas Gray
Synchronization Levels

- Propagating levels for TCP states:
  - **NONE**: No changes are propagated
  - **ESTABLISHED**: Only essential state changes *Established* and *Closed* are propagated
  - **FULL**: All changes are propagated to the network
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Database Write

**Clean** transaction context

- Lock database
- Write to database
- Unlock database

+ Maintains data consistency:
  - Only one process can update a record
- Low performance:
  - Requires locking and unlocking the database

**Dirty** transaction context

- Write to database

+ High performance:
  - Directly update the record
- No data consistency guarantee:
  - Ignore side effects of concurrent access
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
**Confirmation Strategies**

**Synchronized confirmation strategy**

- + Maintains data consistency: Throughout the entire cluster
- - Low performance: Wait for all nodes to confirm write

**Asynchronized confirmation strategy**

- + High performance: Concurrent write and packet forwarding
- - No data consistency: A successful write cannot be ensured
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Test bed setup:
- One Monitoring node
- One active firewall node
- One backup firewall node

Scenario:
- Downloading *index.html* (1 Byte) from web server
- Different load levels of 25 and 100 concurrent connections
- 10 runs with 10,000 downloads each

Parameter configuration:
- Synchronization level
- Data access
- Synchronization strategy

Objective: TCP connection setup times
Database Access Strategies

- Load level 25:
  Minor difference up to 2ms

- Load level 100:
  Significant difference up to 15ms

- Increased impact for higher concurrency level

- Dirty context significantly faster than transaction

- Contrary to all expectations, synchronous transactions faster than asynchronous transactions
### Synchronization Level

<table>
<thead>
<tr>
<th>Load Level 25</th>
<th>Load Level 100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Dirty context: More synchronization leads to higher connection setup times
- Transaction context: *ESTABLISHED* faster than *NONE*  
  → More balanced database tables
- Synchronization levels show higher impact at higher concurrency levels
- Increased connection setup times for *FULL* synchronization

#### Database Access

**Sync. Level**  
- None  
- Established  
- Full

---

State Synchronization for Fast Failover of Stateful Firewall VNF  
*Nicholas Gray*
Cluster Sizes

- 3 nodes on physical KVM server
- Additional nodes connected via OpenStack

Master

VNF

VNF

VNF

Physical Server

OpenStack Cloud

State Synchronization for Fast Failover of Stateful Firewall VNF

Nicholas Gray
Cluster Sizes

- 3 nodes on physical KVM server
- Additional nodes connected via OpenStack

**Dirty context:**
- Larger cluster
  → Slower connection setup

**Transaction context:**
- Load level 25:
  - Cluster size 3 with highest setup times
- Load level 100:
  - Larger cluster
  → Slower connection setup

---

State Synchronization for Fast Failover of Stateful Firewall VNF

*Nicholas Gray*
Conclusion and Outlook

- Concept of a stateful firewall VNF
  - Horizontal scalability
  - Failover

- Prototypical implementation of a stateful firewall VNF
  - Different database access strategies
  - Varying synchronization levels

- Test bed setup
  - Multiple VMs running firewall VNF
  - Connection to OpenStack cloud to increase cluster size

- Investigation of TCP connection setup times w.r.t. consistency and performance
  - Synchronizing all states leads to 19-26% slower connection setup times
  - 20% faster connection setup times when focusing on performance
  - Cluster sizes of 6 and 9 adds delay of 7% and 10% in comparison to a size of 3

- Future work: Alternative data stores