Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

A communication framework for distributed access control in microkernel-based systems

Mohammad Hamad, Johannes Schlatow, Vassilis Prevelakis, Rolf Ernst

Controlling Concurrent Change

http://ccc-project.org/

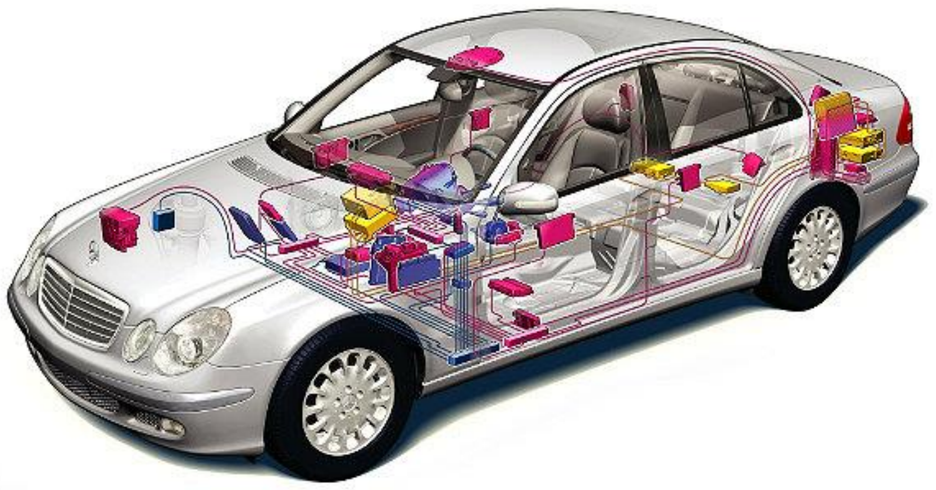
DFG Deutsche Forschungsgemeinschaft

tubs.CITY
Center for Informatics and Information Technology

Overview

Computers on wheels

- 70-100 ECU
- More than 10 millions LOC



Within each ECU

- Multi-levels of security and criticality
  - Different vendors
  - Different security perspective
- Protecting applications from each others
  - Toyota “EnForm” system !

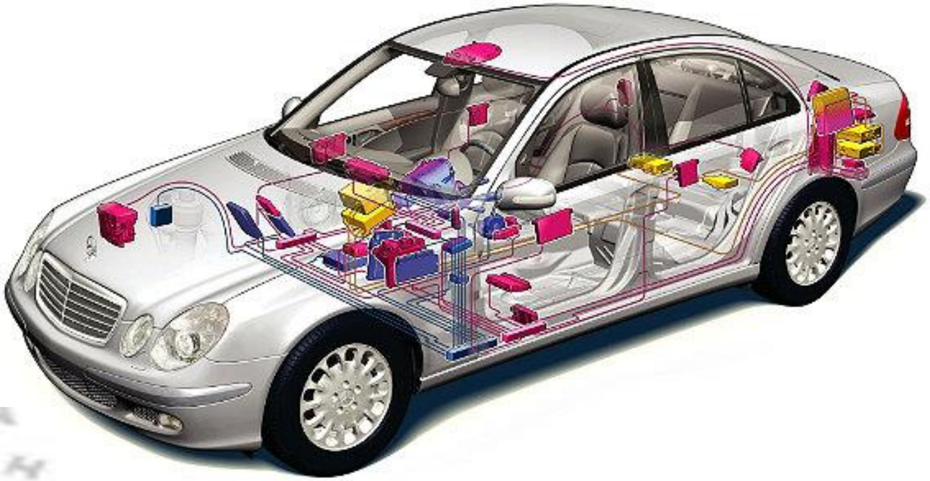
Microkernel is the first step

- Minimum privileged code and TCB
- Inter-process communication (IPC)

Overview

Network on Wheels

- ECUs connected using many bus systems
  - CAN, IP based for on-board
  - Wireless
- Uncontrolled interaction may cause a vulnerabilities
  - Entertainment system attack !



Contribution

Providing **distributed access control framework**

- Controlling **"who should talk to whom"**
- Providing **security services** (i.e. Integrity, mutual authentication, and confidentiality)
- **Distributed** policy enforcement points

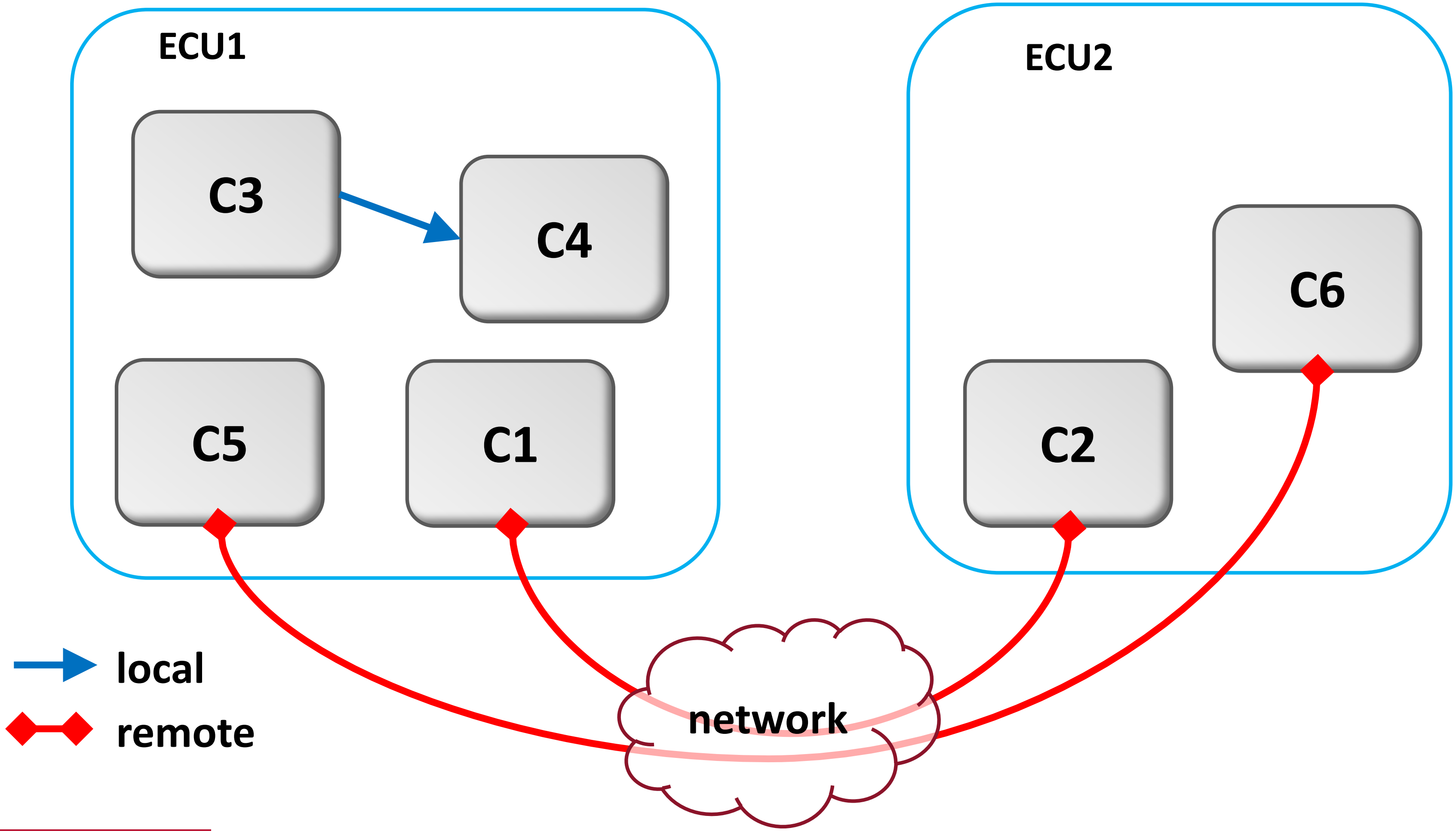
Outline

- **Communication scenario**
- **Framework architecture**
- **Design and implementation**
- **Evaluation**
- **Conclusion**

Outline

- **Communication scenario**
- Framework architecture
- Design and implementation
- Evaluation
- Conclusion

Communication Scenario



(Security) objectives

Objective	Local	Remote
Fine-grained access control	✓ ?	?
Integrity, mutual authentication, and confidentiality	✓ ?	?
Legacy application support	?	
Composability and migratability	?	
Minimum (application-specific) TCB	?	

Local communication

- IPC and Capability-based access control enforced by policy engine
- Capability used to identify the application (authentication)

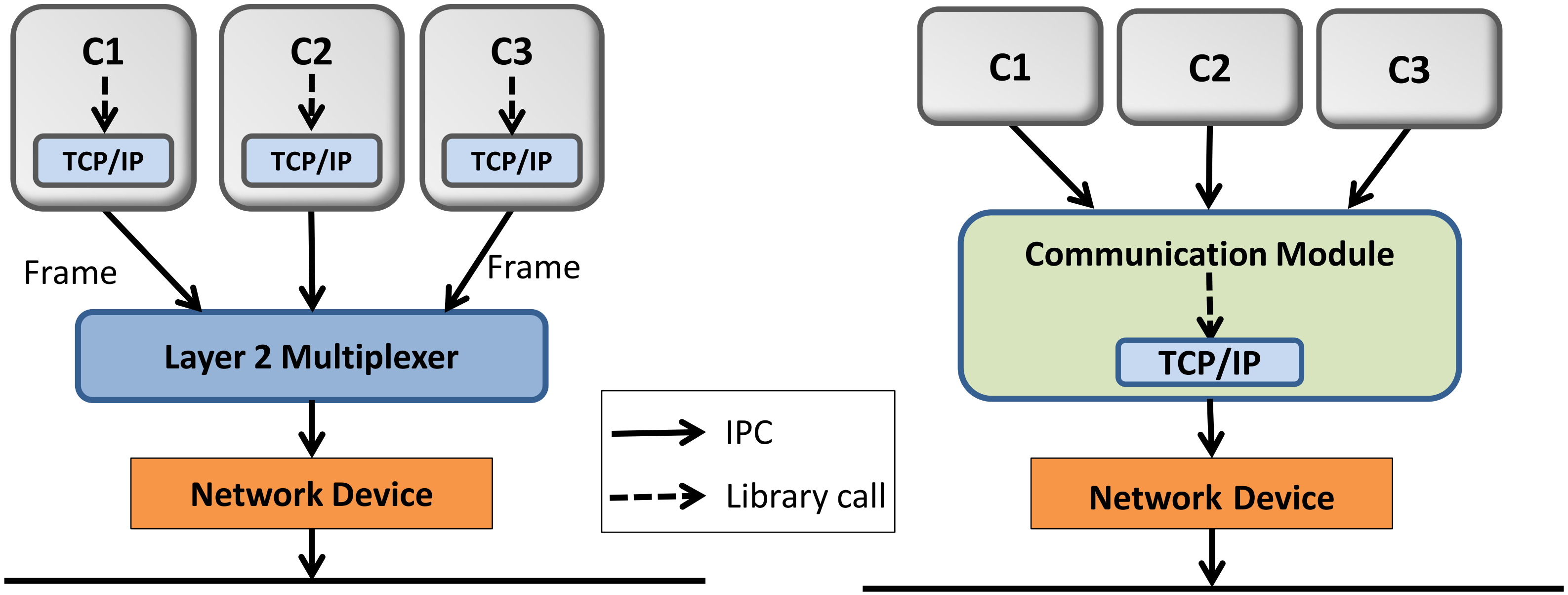
From IPC towards networked communication

- Controlling the direct access to the communication module
- Component needs capability and appropriate policy to use network

Outline

- Communication scenario
- **Framework architecture**
- Design and implementation
- Evaluation
- Conclusion

Framework Architecture



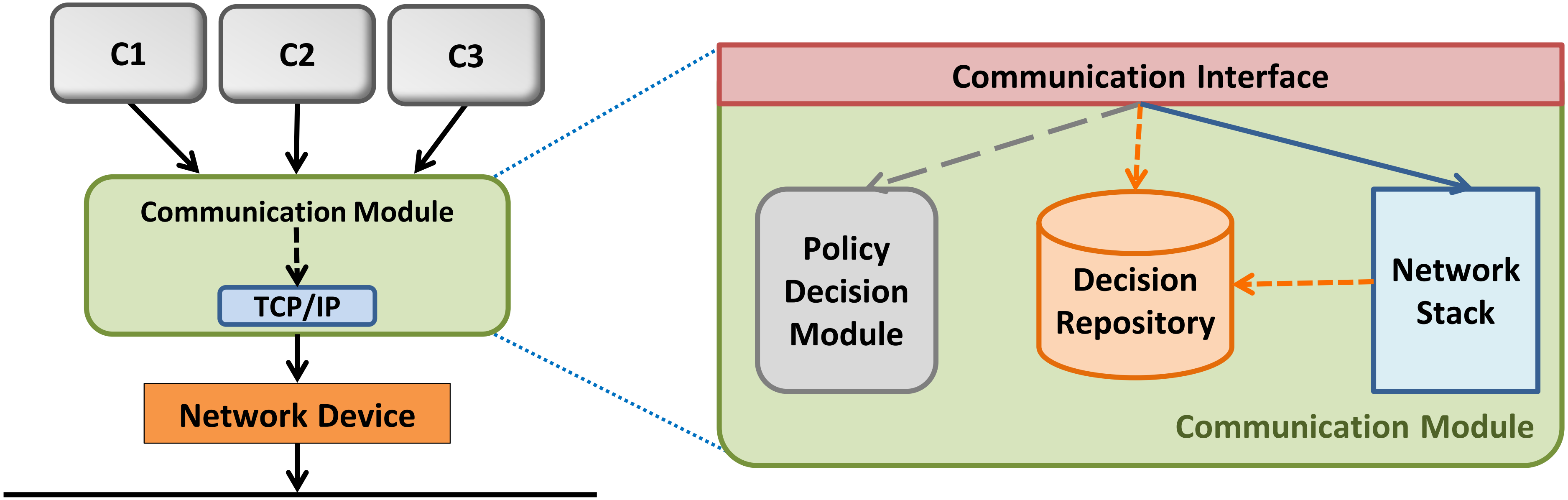
- User-level networking
- Dedicated stack per application
- Threats
  - Spoofing
  - DoS attack
- Layer 2 security

- For each ECU
  - Single Communication Module
  - Shared by all applications
  - Local firewall enforces part of the communication policy
- Distributed Firewall
- Layer 3 security

Outline

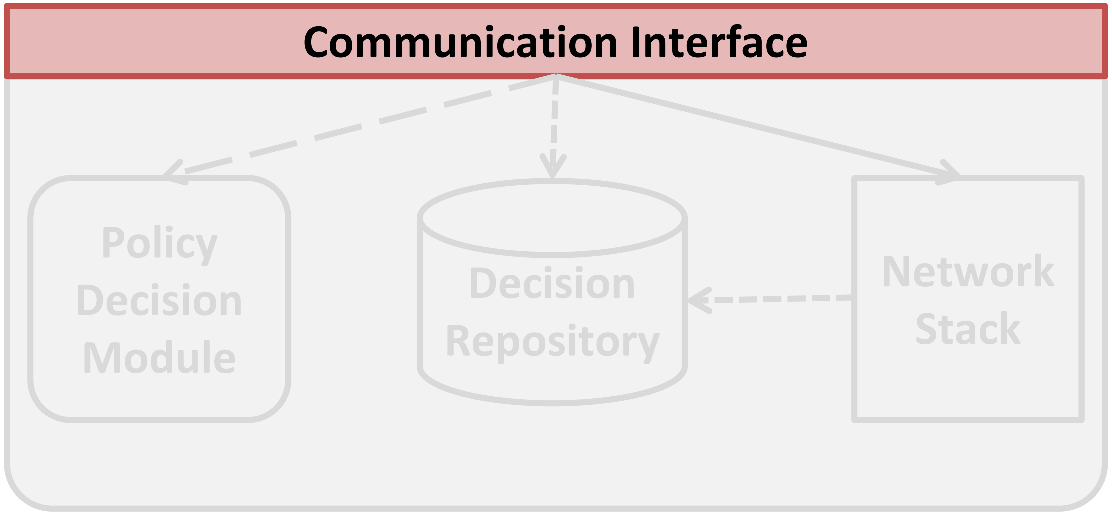
- Communication scenario
- Framework architecture
- **Design and implementation**
- Evaluation
- Conclusion

Design and implementation



- Communication Interface
- Policy Decision Module
- Network Stack
- Decision Repository

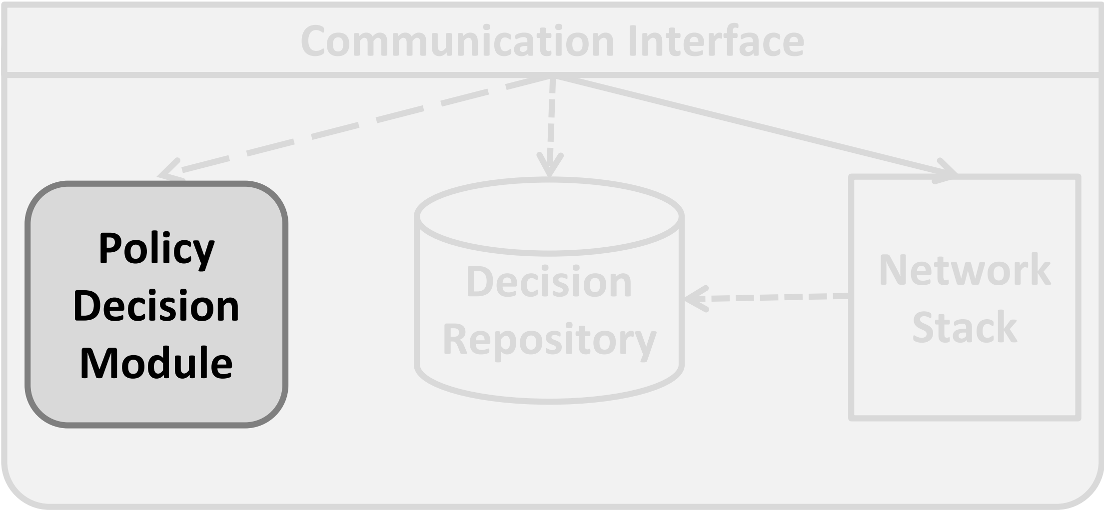
Design and implementation



Communication Interface

- Implementing the socket API calls as IPC calls
- Managing the shared memory between each application and Communication Module
- Checking the validity of the parameters
- Enforcing the Policy Decision Module results

Design and implementation



Policy Decision Module

- Monitoring the requests based on credentials and connection properties.
- KeyNote Trust management system
- KeyNote Policy definition language.
  - Application independent.
  - Delegation.

Design and implementation

Authorizer: Integrator_public_key

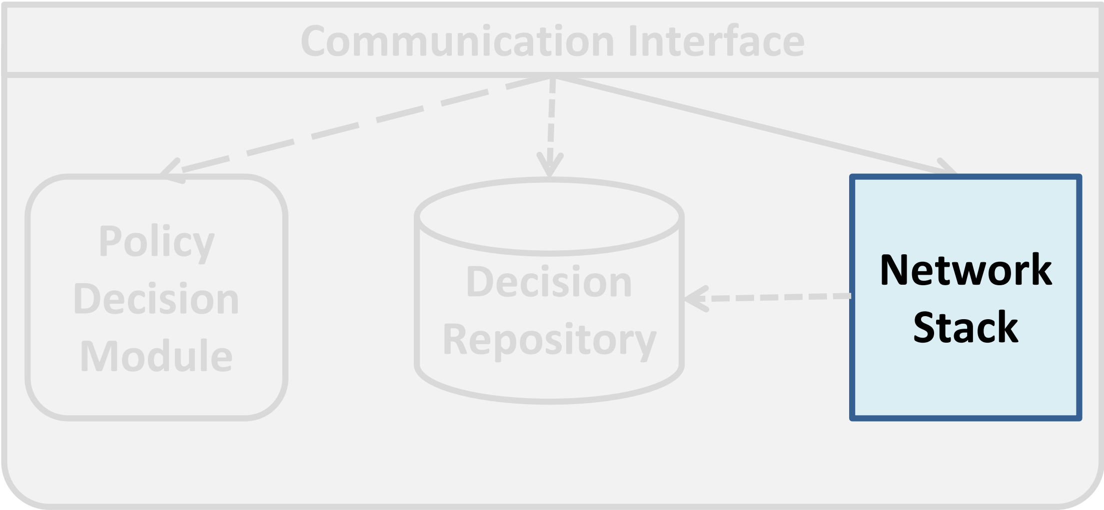
delegate

Licensees: Platform_public_key

Conditions: (Vendor_id ==“ACME_INSTRUMENTS“ && Src_device_name == “headlight_control“
&& Dst_device_name == “ambient_light_sensor“ && Src_device_type == CONTROL_PLATFORM
&& Dst_device_type == LIGHT_SENSOR && Security_level >= SL_INTEGRITY) -> “ALLOW“

Signature: Integrator signature

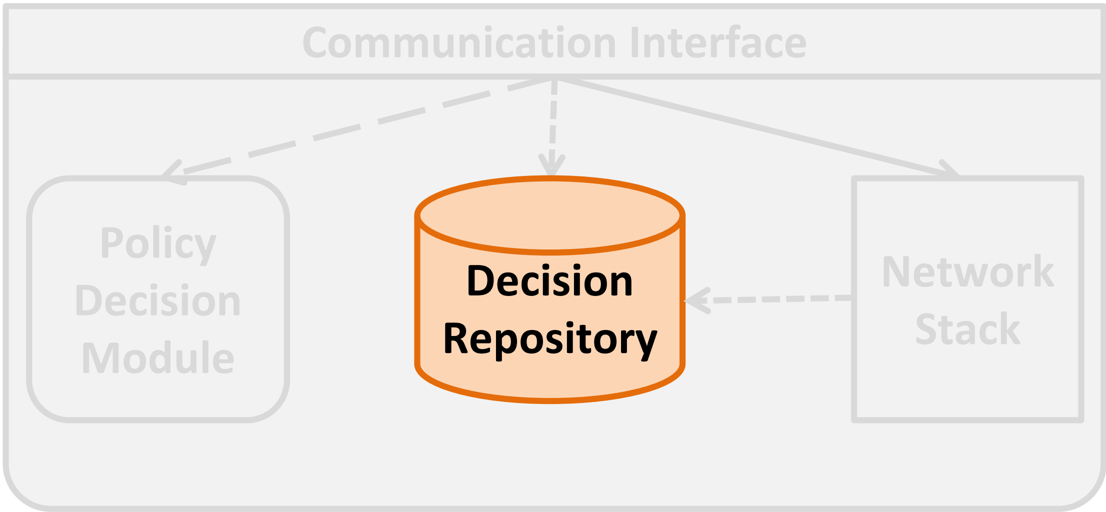
Design and implementation



Network Stack

- LwIP stack.
- Embedded IPsec.
  - Mutual authentication, integrity and confidentiality.
- Rate limiting, queuing priority.

Design and implementation



Decision Repository

- Storing the decision rules (i.e. source IP, destination IP, security level, etc.)
- Improve the efficiency

(Security) objectives

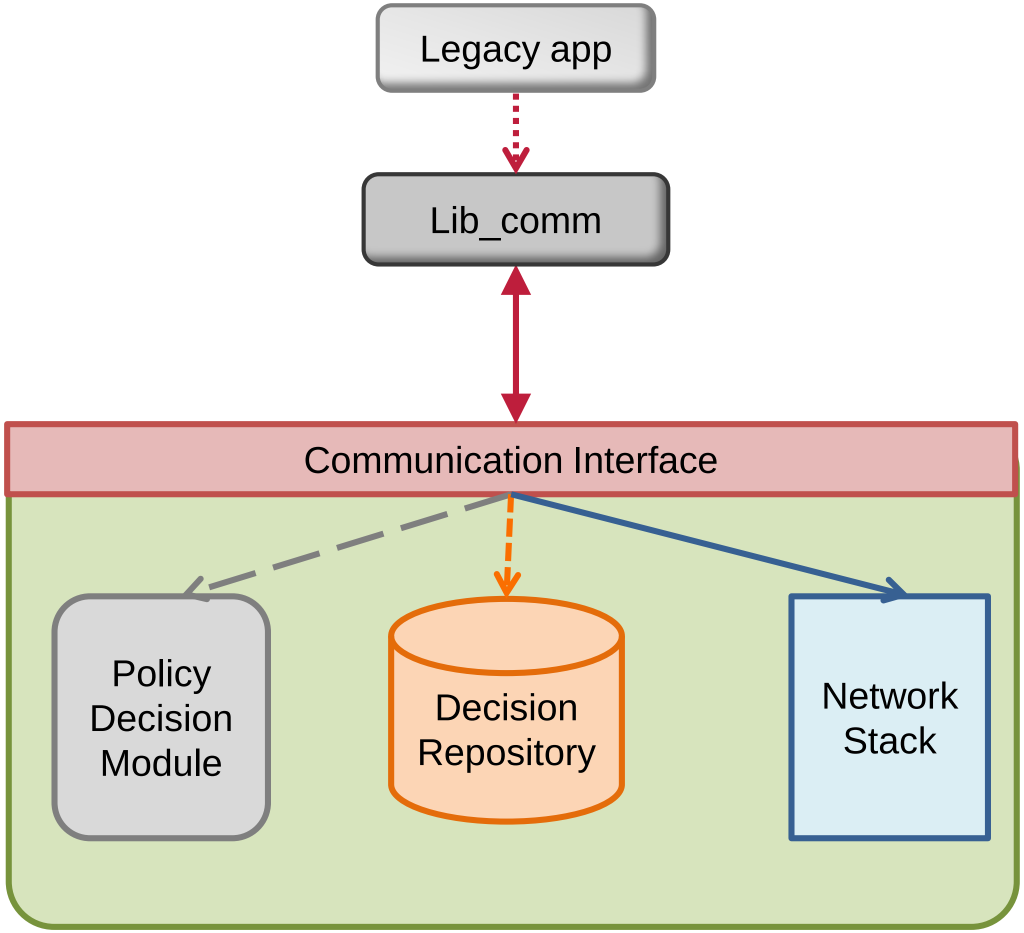
Objective	Local	Remote
Fine-grained access control	✓	✓?
Integrity, mutual authentication, and confidentiality	✓	✓?
Legacy application support	?	
Composability and migratability	?	
Minimum (application-specific) TCB	?	

Example

```
int  s = accept (...);
```

```
int accept (...){
  return  comm_accept(...);
}
```

```
int comm_accept (...) {
  Rule  *rule_r = search_rule (...);
  if (rule_r->empty()) {
    if (verify(...) ) {
      add_rule(...) ;
      return  lwip_connect(...) ;
    }    else   return -1  ;
  }
  If (rule_r->authorized())
     return  lwip_connect(...) ;
  else   return -1  ;
}
```

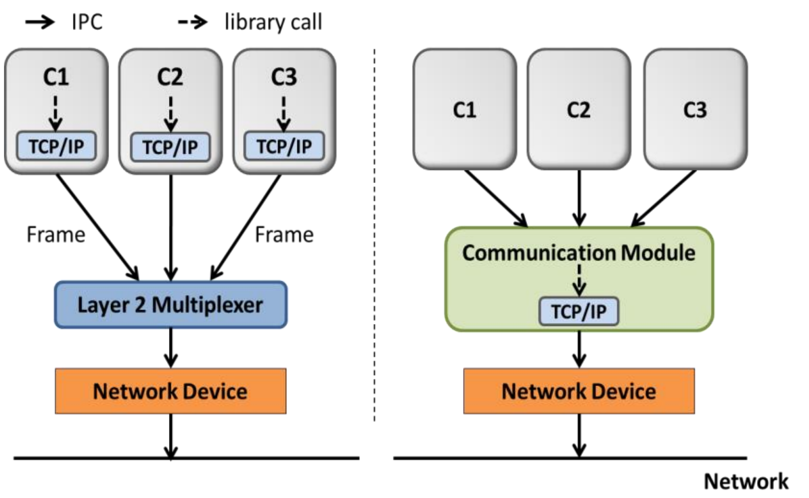


Outline

- Communication scenario
- Framework architecture
- Design and implementation
- **Evaluation**
- Conclusion

Evaluation

- implemented with Genode OS
- Compared to existing Genode OS
  - bridge with proxy-ARP



Source Lines Of Code (SLOC)

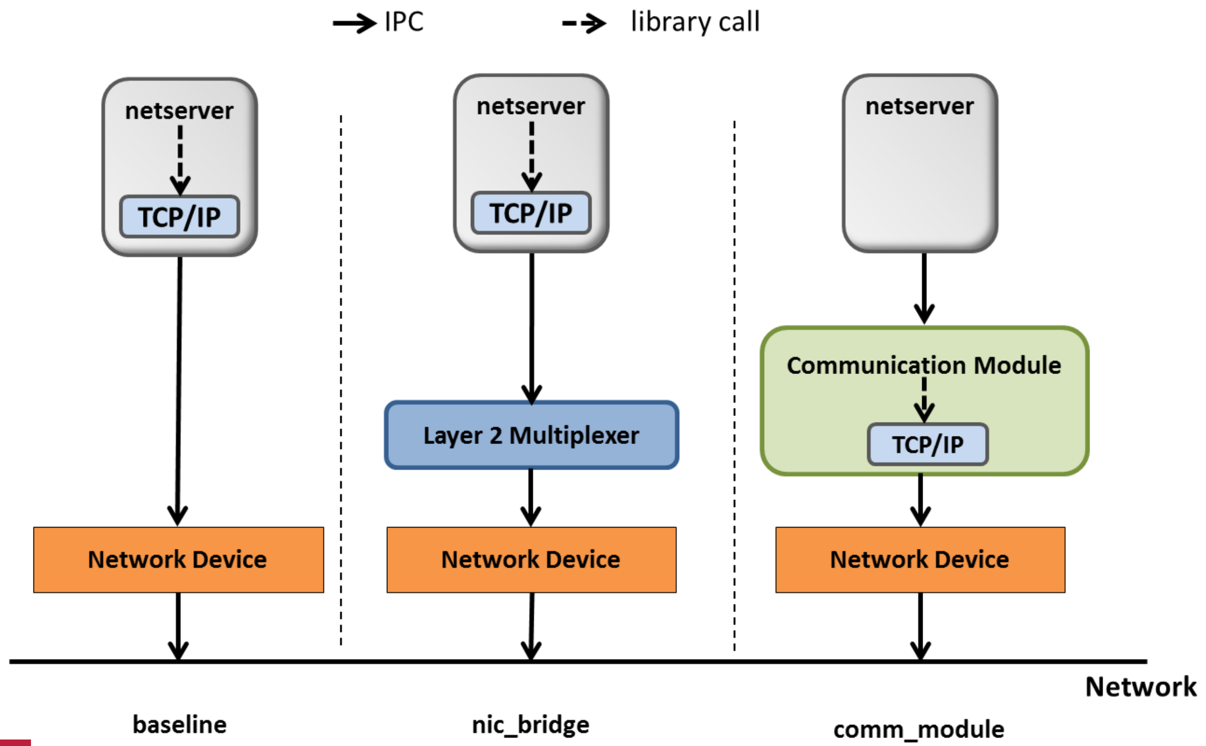
- **Save 750 SLOC**

Module Part	SLOC
Comunication interface	500
Policy Decision module Interface	300
IPsec extension of the Network Stack	2000
Decision Repository	600

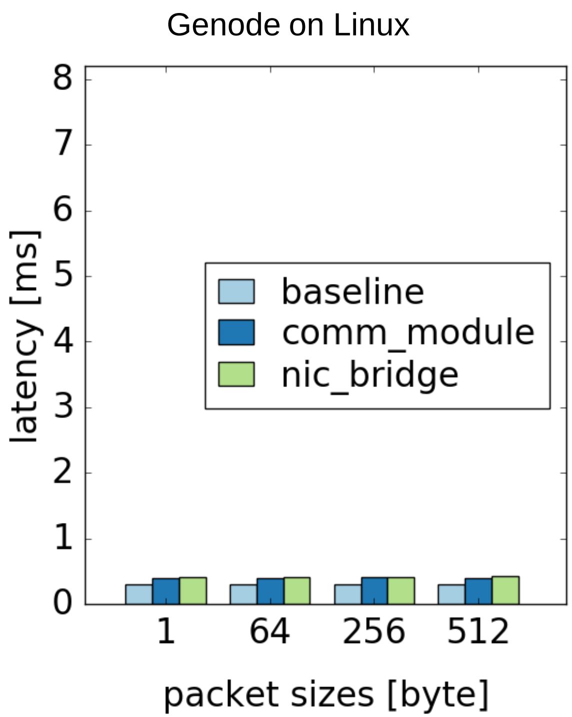
Evaluation

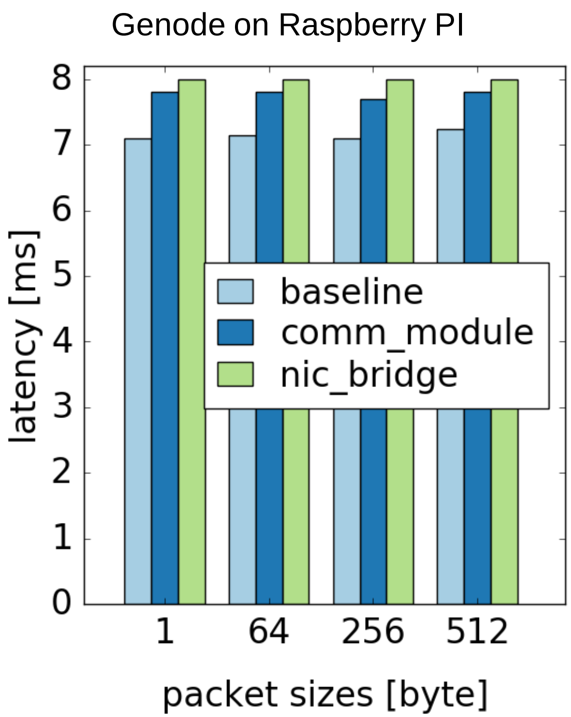
Latency

- Netperf tool
- Genode and Netperf runs on the same Linux machine
- Genode runs on Raspberry Pi, and Netperf runs on remote Linux machine



Evaluation





Our Module does not add extra overhead.

Outline

- Communication scenario
- Framework architecture
- Design and implementation
- Evaluation
- **Conclusion**

Conclusion

Objective	Local	Remote
Fine-grained access control	Capability- based	Communication module
Integrity, mutual authentication, and confidentiality	Capability- based	IPsce
Legacy application support	Lib-comm	
Composability and migratability	Proxy	
Minimum (application-specific) TCB	?	

Performance could be better

- Handle the copy operations.
- Cashing the credentials.