Formal specifications of Software-Defined Networks

Ing. Ondrej Lichtner
UIFS
Brno University of Technology, Czech Republic
ilichtner@fit.vutbr.cz
Advisor: Švéda Miroslav, prof. Ing., Csc.
Specialist: Ing. Ondřej Ryšavý, Ph.D.

BRNO UNIVERSITY OF TECHNOLOGY
FIT
FACULTY OF INFORMATION TECHNOLOGY

Presentation outline

- Software-Defined Networks
- Formal specifications of SDNs
- Future options
- Summary
- Questions

Current network

- Static
- Hierarchical
- Tiers of switches creating tree structures
- Centered around client-server computing

Changes in networking

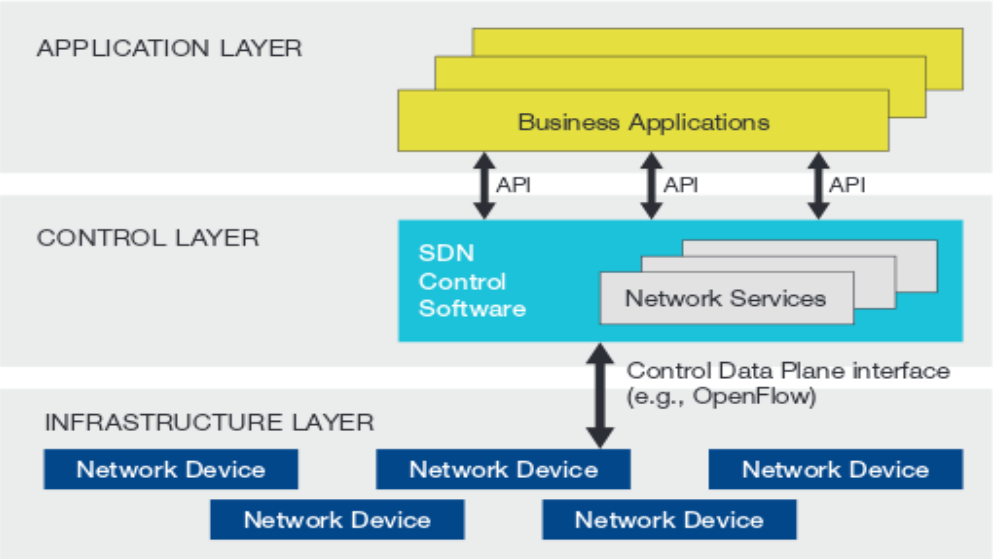
- Changes in traffic patterns
- Server virtualization
- Explosion of mobile devices and content
- Cloud services
- Big data

Limitations we are facing

- Complexity
- Scalability
- Vendor dependence

What is SDN?

Software Defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable.



Benefits of SDNs

- Centralized and dynamic
- Simplifies management
- Simplifies devices
- Removes vendor dependence

Why?

- New and unproven
- Rapid development
- This can mean a lot of problems
- Verification and analysis

Common tools

- Specification and Description Language (SDL)
- Algebra of Communicating Shared Resources (ACSR)

Specification and Description Language

- Object-oriented formal language
- Specification of complex, event-driven, real-time and interactive applications
- Theoretical model consists of a set of extended finite state machines that run in parallel
- Communication with discrete signals

Specification and Description Language

Components:

- Structure – system, block, process, procedure hierarchy
- Communication – signals with optional parameters and channels
- Behaviour – processes
- Data – abstract data types
- Inheritance – relations and specialization

Algebra of Communicating Shared Resources

Formal specification and manipulation of distributed systems with resources and real-time constraints.

Timed actions and instantaneous events.

Algebra of Communicating Shared Resources

Timed Actions

- *Res* – finite set of reusable resources, $r \in Res$
- *A* – resource consuming action
- p(A) – set of resources used by A
- $\pi_A(r)$: $Res \rightarrow R^{\geq 0}$
- $\pi_A(r) = 0$ when $r \notin p(A)$
- $A = \{(r_1, p_1), \ldots, (r_n, p_n)\}$
- Time domain is $R^+ \cup \{\infty\}$, we use u, v, w to denote time values
- A^u denotes the execution of A for the duration of u
- Synchronous composition – A|B, single action of A and B happening simultaneously
- Closure $[A]_U$ – increase used resources to include everything from U, with priority 0. Reservation of resources

Algebra of Communicating Shared Resources

Instantaneous events

- Synchronization mechanism
- Event *e* with label *l(e)* and priority *π(e)*
- Labels are drawn from $\Lambda = \mathcal{L} \cup \{\tau\}$, we assume a complement operation over $\mathcal{L}$, τ denotes internal actions
- Pair wise synchronization – composition operation over events
- *e*|*f* – defined when both have complementary labels

Algebra of Communicating Shared Resources

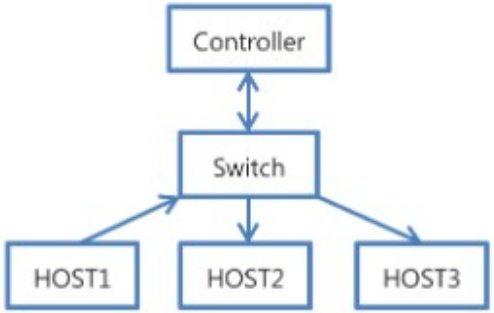
Computation model

- Behaviour of a process is given by a labelled transition system
- Subset of *Proc x Act x Proc*
- *P* executes action *α* and turns into process *P′*, if *(P, α, P′)* is in the labelled transition system
- Execution step written as $P \rightarrow^{\alpha} P'$
- Process syntax:
  $P ::= 0 \mid A^{u}{:}P \mid e.P \mid P{+}P \mid P \| P \mid P \Delta_{v} Q \mid P \dagger Q \mid [P]_{U} \mid P \backslash F \mid rec\ X.P \mid X$

Algebra of Communicating Shared Resources (example)

- Network nodes represented as ACSR processes
- Packet forwarding is specified as event sending and receiving
- Packet matching with rules are represented by synchronization between input and output events

Algebra of Communicating Shared Resources (example)




```
System = (Host1 || Host2 || Host3 || Switch || Controller) \ {events}
Switch = activeRule1 || activeRule2 || activeRule3 || Requester
Host1 = ('packet1, 1).('packet2, 1).('packet3, 1).Host1
Host2 = (packet1', 1).Host2 + (packet3', 1).Host2
Host3 = (packet1'', 1).Host3 + (packet2', 1).Host3
Requester = (packet1, 0).('requestRuleForPacket1, 99).Requester
            +(packet2, 0).('requestRuleForPacket2, 99).Requester
            +(packet3, 0).('requestRuleForPacket3, 99).Requester;
Controller = (requestRuleForPacket1, 99).('activatingRule1, 99).Controller
            + (requestRuleForPacket2, 99).('activatingRule2, 99).Controller
            + (requestRuleForPacket3, 99).('activatingRule3, 99).Controller
activeRule1 = (packet1, 4).('packet1, 999).activeRule1
inactiveRule1 = (activatingRule1, 99).activeRule1
activeRule2 = activeRule2' + activeRule2''
activeRule2' = (packet1, 3).('packet1'', 999).activeRule2
activeRule2'' = (packet2, 3).('packet2', 999)activeRule2
inactiveRule2 = (activatingRule2, 99).activeRule2
activeRule3 = (packet3, 6).('packet3', 999).activeRule3
inactiveRule3 = (activatingRule3, 99).ActiveRule3
```

Sender	Receiver	Types of packets
Host1	Switch	packet1, packet2, packet3
Switch	Host2	packet1', packet3'
Switch	Host3	packet1'', packet2'

Other approaches and Future

- Z notation – focused on each switch and controller
- Timed-Automata
- Formally Verifiable Networking – novel approach to unifying the design, specification, implementation and verification of networking protocols.

Future

- SDNs are still very new
- Usefulness is questionable
- Rise in popularity
- Formal verification for developing new standards

Summary

- Problems of current networks
- How SDNs try to solve them
- Tools for formal specification of SDNs
- ACSR Example
- Other approaches and Future

Sources

- Patrice Brémond-Grégoire, Insup Lee (1995). A Process Algebra of Communicating SharedResources with Dense Time and Priorities. Retrieved from http://repository.upenn.edu/cgi/viewcontent.cgi?article=1219&context=cis_reports
- Wang, A., Jia, L., Liu, C., Loo, B. T., Sokolsky, O., & Basu, P. (2009). Formally verifiable networking. *8th ACM Workshop on Hot Topics in Networks*. Retrieved from http://repository.upenn.edu/cis_papers/747/
- Kang, M., Park, J., Choi, J. Y., Nam, K. H., & Shin, M. K. (2012). Process algebraic specification of software defined networks. Proceedings - 2012 4th International Conference on Computational Intelligence, Communication Systems and Networks, CICSyN 2012 (pp. 359-363).
- Kang, M., Kang, E.-Y., Hwang, D.-Y., Kim, B.-J., Nam, K.-H., Shin, M.-K., & Choi, J.-Y. (2013). Formal Modeling and Verification of SDN-OpenFlow. 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation, 481-482. Ieee. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6569764
- Open Networking Foundation https://www.opennetworking.org/index.php