Network Verification

Lecture 10, Computer Networks (198:552)
Traditional IP network

Management plane

- Configure routers
- Ex: OSPF link weights
- Ex: BGP local prefs

Control plane

- Track the topology
- Exchange messages
- Compute forwarding rules

Data plane

- Fwd packets using computed forwarding rules
Software-Defined Network (SDN)

Centralized control plane

Data plane

Data plane

Data plane

Data plane
Why verify networks?

- High-profile outages
  - Caused by human errors more than 50% of the time 😊
- “Complex systems break in complex ways”
- Interactions between protocols
- Interactions between different administrative domains
- Networks change all the time
- Security is increasingly important
- Intellectually interesting
  - Computer-Aided Design (CAD) for networks [George Varghese]
Verification:
A problem statement
for all M, does N satisfy P?

Decision Procedure: An algorithm that answers yes/no

Can ask the question under a network change model: static, incremental, or dynamic

Sequence of messages:
- Packets
- Routing protocol
- Link failures

Network representation:
- Data plane
- Control plane

Property of interest:
- Loop freedom
- Blackholes
- Reachability
- Equivalence
Example: Verifying firewall rules

- Assume packets just have 2 bits; there are only 2 ports
- Firewall config: 10 -> fwd(2); *1 -> fwd(1). All others dropped
- Boolean representation of the network:
  - N: \((d_1 \& \neg d_0) \lor ((d_1 \lor \neg d_1) \& d_0)\)
- Property: only the packets from 00 are dropped
  - P: \((\neg d_1 \& \neg d_0)\)
- Messages (M): all combinations of boolean variables \(d_0, d_1\)
- Verification question: for all \(d_0, d_1\), is formula \(N \lor P\) valid? i.e.,
  - Is \(((d_1 \& \neg d_0) \lor ((d_1 \lor \neg d_1) \& d_0)) \lor (\neg d_1 \& \neg d_0)\) a tautology?
- Decision procedure: SAT solver!
Typical considerations for verification

- Size of network representations
  - $O(\# \text{ rules})? \# \text{ packets? Some product of these things?}

- Speed of decision procedure, e.g., SAT solving
  - Typically NP-hard or worse in the worst case
  - Verification: leveraging average-case complexity

- Coverage of possible network events
  - Does property hold under firewall rule changes? New protocol messages? Link failures?

- Strength of properties and counter-examples
  - Does P hold for all packets? Are we looking for one counterexample, or the whole set of violating packets?
Verification, testing, synthesis, eq checks

- Verification: for all M, does N satisfy P?
- Testing: For the given M, does N satisfy P?
- Synthesis: Given P, can you produce an N that satisfies it
  - For all M?
  - For a given set of M?
- Let N1 be another network representation
- Equivalence checking: For all M, do N and N1 behave in the same way?, i.e.,
  - Either both satisfy P or both violate it
Properties to verify

• Reachability, isolation, loop freedom
• Equivalence between data plane rules
  • Replicated configurations (for availability or performance)
  • Reduce to simpler configurations
• Waypoint properties
  • e.g., does traffic always go through a monitoring node?
  • Ordering constraints on processing: e.g., DPI must follow ACLs
• Temporal properties, e.g.:
  • After first message from a source, don’t broadcast traffic destined to it
• Performance properties: e.g., arrival distributions & congestion
10,000 ft overview of the literature

• Data plane verification
  • Static: header space analysis
  • Incremental: Veriflow
  • Dynamic: NICE

• Control and data plane verification
  • Static: p4v
  • Incremental: Batfish
  • Dynamic: Minesweeper
Scaling challenges

• Too many messages and events
  • Packet headers
  • Link failures
  • Protocol messages
• Orderings between events matters!
• Too many network rules
• Too large a network
Discussion of Header Space Analysis

- Compact boolean representation + composition operations
- Why is an inverse always well-defined?
- Linear fragmentation assumption
- Representation as difference of two HSAs
- Generic loops and infinite loops
- Per-port loop detection vs. stopping at any port: pros & cons?
- What else could you run on the propagation tree?
Discussion of VeriFlow

• Trie-like representation of packet headers
• Forwarding equivalence classes: help scale!
• Implicit assumption that many FECs aren’t affected at once
• What computations could you do over the forwarding graph?
• How do you check for blackholes using VeriFlow?
• Could you extend the trie for performance verification?
• Are there bad wildcard rules that make the “affected FEC” set grow really large with a rule insertion (e.g., exponentially)?
• What changes are required for packet modification?