Contribution ID: **8** Type: **not specified**

A new paradigm for network simulations; model-checking meets event simulation

Tuesday, 11 September 2012 13:30 (12 minutes)

For large-scale system simulations, two main components need to be developed; the problem model, and the simulation model. For basic sciences, the models always reflect the physical phenomena, and the challenges arise from numerically implementing those models in computational environment, and then verifying the physical phenomena. In networking, or the Internet in particular, the modeling problem is still an open question. Finding the correct interaction model between network components is an open research question, especially with the increase of complexity and types of components. The simulation component of different models can then follow.

We propose using formal analysis to first model large networks (the model), then leverage temporal model-checking approaches to simulate the dynamic network behavior over time (the simulation). In doing so, we propose two stages for network modeling:
- Configuration Modeling: This covers policies (routing, security, QoS, …) as well as topology information and high level applications. Developing models across communication layers while taking care of topology will integrate both vertical and horizontal understanding of network operations.
- Network State Modeling: This covers dynamic behavior of physical network components, including link behavior (capacity, connectivity, quality, etc)

The Model

We have used model-checking for network configuration analysis for performing large-scale network verification.
We will leverage our experience to extend the models, add more network components, integrate real-time processing and enable large-scale network simulation. Also, a vital addition is distributed analysis of such models. This section provides a summary of our previous work on model-checking as a means for network analysis.

The problem of model checking a multi-faceted system (i.e., multilayer multi-device network) can be broken down into two main subproblems: 1) How to merge these heterogeneous systems into one monolithic framework, and 2) what is the system state upon which we can define transitions and build the model-checker?

When it comes to modeling a multi-component system, the problem of finding a middle ground, or a common representation becomes extremely important. Utilizing highly specialized data structures, or representation model will work for one layer but will break the other (or at least becomes highly inefficient). To address this problem, one can revert to one of the basic structures: sets/collections, Boolean expressions, formal grammar description, etc. Choosing a grammar description will change the focus of the work from model-checking the system to designing a more complete (and more complex) language. This will never be flexible or efficient enough for large scale analysis. Using basic sets while is very flexible, it is far from being scalable without using a symbolic representation. Boolean expressions comes as a plausible solution providing both: simple set-like operations, as well as having many very efficient practical implementations.

The first step in analysis/modeling the network will start by digesting all the information and policies of multiple device types and compile them into basic expressions. Every predicate a network device defines can be simply written as a Boolean expression. The problem now becomes one of defining the variables and labels upon which such expressions are built. It is important to mention that using such a generic representation enables the complete separation of device specifics and syntax from the actual analysis as long as the settings, policy and status got mapped into a Boolean expression.

The other part of the problem is modeling the system/network state. Let us start with a domain specific assumption: packets move through the network faster than the network configuration and layout can change. This assumption will lead us to define the system state from a packet-centric perspective rather than a network one. In other words, the state space defined for the model checker is composed of all possible packets (i.e., packet types, header values, etc) and their status as they travel across the network (i.e., current location of the packet, quality of service received so far, whether it is encrypted/tunneled or not, etc).

In our previous work, we compiled large numbers of devices with heterogeneous types into a single state machine. The states are defined as explained above, and the transitions drawn between them are defined by the topology, hardware capacity, and network/service policies. While the number of states is intractable, they are efficiently represented symbolically. Also, the valid transitions from a state to state are defined collectively via symbolic representation.

Fine tuning the model for performance was possible by exploiting efficient encoding of network data into the used Boolean variables as well as tweaking the order over which we build the expression trees using binary decision diagrams (BDD). We managed to concurrently model a few thousand devices of different types and crossing multiple layers, answer security and reachability queries, and add updates to the model in efficient and scalable manner.

The Simulation

The formal model (described above) can be used to answer queries on states reflecting packet transitions, or locations. Constraints can be defined for specific location, domain, or time modality. The query and its response reflect a snapshot of the network operations, whether temporally or spatially. We will take this static evaluation one step further, and evaluate continuously over time, while changing multiple constraints. The new constraints can be modeled to represent network dynamic conditions, configuration changes, ...
We look at the problem as integrating discrete-event simulation with model-checking, where events are steps in time where the model needs to be evaluated.

So, our network simulator will start with the configuration (topology, routing, security...), and move along state transitions given network constraints (flow values, link changes, ...). In simulation modeling, the network operation needs to be monitored and tracked over time, without restrictions. For this, only an initial state needs to be identified, and the simulation will track the model response at each time step.

When large network traces are available, those could be used in replay mode to trigger model tracking. Knowing end-to-end flow information from offline traces (CAIDA, ...), model constraints can be changed incrementally as new flow information becomes available from the traces. With those constraints, the model can answer the queries (as described above), and each time-based snapshot (query result) gives a snapshot of the network state which corresponds to the simulation outcome for this time step.

Configuration and network state models can be changed to simulate what-if scenarios using the same traffic traces. Most changes can be applied very effectively on to a model without rebuilding except the directly affected part. The same idea can be used to apply the effect of external phenomenon. For example, one can model power/link outage, massive interference causing packet getting transmitted in error, excessive volumes of cross traffic, etc by merely tweaking few transitions or invalidating some of the states. This opens the door for many applications from disaster recovery planning to resource allocation and optimization.

Challenges

Several opportunities exist to enable large-scale simulation with formal modeling. For large networks, modeling all layers with diverse parameters can render the model unmanageable. Building the model is the most expensive operation, and parallel processing can enable fast model generation.
Parallel processing can be used to: parallelize model-checking platforms (formal modeling domain, not here), or parameter selection and tuning (e.g. variable ordering), or decentralization of the simulation.
It is worth noting that current non-parallelized implementation can build the model for multi-layer configurations of 5K devices in less than 30 minutes [1,2]. While this seems satisfactory, it required manual tweaking of model building parameters (mainly variable ordering of BDDs and field encoding mechanisms). For a more general approach we have to automate this process and this requires, in turn, serious parallelizing of the model fine-tuning as well as the model construction operation itself.

Another source of complexity to the system stems from our need to model a more dynamic background status of the system. In other words, to model a realistic cross traffic, and actual network load, we need to 1) use actual sources and available network traffic traces, and 2) approximate these in a way that keep the model feasible to manage and analyze. Our prior work [4, 7] on traffic analysis gives us the ability to pinpoint the places to cut down traffic data without losing overall load fidelity.

References:

[1] Ehab Al-Shaer, Wilfredo Marrero, Adel El-Atawy, Khalid Elmansor, “Network Configuration in A Box: Towards End-to-End Verification of Network Reachability and Security”, In the 17th IEEE International Conference on Network Protocols (ICNP’09), Princeton, New Jersey, USA, 2009.
[2] Adel El-Atawy, Taghrid Samak, “End-to-end Verification of QoS Policies”, (NOMS’12), Maui, Hawaii, USA, April 2012.
[3] Alan Jeffrey and Taghrid Samak “Model Checking Firewall Policy Configurations”, IEEE International Symposium on Policies for Distributed Systems and Networks (Policy 2009) 20-22 July 2009 – London, UK
[4] Taghrid Samak, Dan Gunter, Valerie Hendrix, “Scalable Analysis of Network Measurements with Hadoop and Pig”, 5th Workshop on Distributed Autonomous Network Management System (DANMS), co-located with

NOMS 2012.
[5] Taghrid Samak and Ehab Al-Shaer, “Synthetic security policy generation via network traffic clustering”, The 3rd Workshop on Artificial Intelligence and Security, AISec, in conjunction with ACM/CCS 2010, ACM, October 2010
[6] Taghrid Samak, Adel El-Atawy and Ehab Al-Shaer, “Towards Network Security Policy Generation for Configuration Analysis and Testing”, Workshop on Assurable & Usable Security Configuration (SafeConfig), Colocated with ACM CCS 2009, Chicago, USA, November 9, 2009
[7] Adel El-Atawy, Taghrid Samak, Ehab Al-Shaer and Hong Li, “On Using Online Traffic Statistical Matching for Optimizing Packet Filtering Performance”, In the 26th Annual IEEE Conference on Computer Communications (INFOCOM’07), Anchorage, Alaska, USA, May 2007.

Primary authors: Dr EL-ATAWY, Adel (Google Inc.); Dr SAMAK, Taghrid (Lawrence Berkeley National Laboratory)

Co-author: Mr GUNTER, Daniel (Lawrence Berkeley National Laboratory)

Presenter: Dr SAMAK, Taghrid (Lawrence Berkeley National Laboratory)