A Formal Framework for Secure Routing Protocols

Chen Chen* Limin Jia° Hao Xu * Cheng Luo* Wenchao Zhou◇ Boon Thau Loo*
* *University of Pennsylvania* ° *Carnegie Mellon University* ◇ *Georgetown University*
{chenche,haoxu,chengluo,boonloo}@cis.upenn.edu, liminjia@cmu.edu, wzhou@cs.georgetown.edu

The Internet infrastructure, as it stands today, is highly vulnerable to attacks. The Internet runs the *Border Gateway Protocol* (BGP), where routers are grouped into Autonomous Systems (ASes) administered by Internet Service Providers (ISPs). Since route advertisements are not authenticated, ASes can, for example, advertise nonexistent routes or claim to own arbitrary destination addresses. In response to these vulnerabilities, there has been a considerable amount of work to design new architectures and protocols for a more secure Internet, including direct secure extensions (Secure-BGP, ps-BGP and so-BGP) and "clean-slate" Internet architectural redesigns such as SCION and ICING. However, *none* of the above formally analyzed their security properties; these protocols are typically evaluated primarily experimentally, and their security properties shown via informal reasoning. To address these limitations, we present a unified formal framework that allows a protocol developer or researcher to combine development and verification of secure routing protocols. We introduce the major components in our framework in the rest of the paper.

Protocol specification: Secure Network Datalog (SeNDLog) [2], a declarative language proved effective in implementing network protocols, is used to encode secure routing protocols. As a variant of Datalog, SeNDLog program consists of rules of the form h:-$b_1, ..., b_n$, where h (rule head) and b_is (rule body) are tuples stored in a relational database. Informally, each SeNDLog rule specifies that if all the body tuples are derivable, then the head tuple is derivable. To support distributed execution, each tuple has a location specifier indicating where the tuple is stored. The head tuple may specify a location that is different from the body tuples, in which case the derived head tuple will be communicated over the network. To support security operations such as asymmetric encryption or hash function, SeNDLog extends declarative networking with cryptographic libraries, which can be used as user-defined function in the rule body.

Compilation: The compiler we design, as part of a declarative engine for implementing and experimenting with network protocol[1], is key to bridging implementation and verification of secure routing protocol. It consists of two main parts: a *code generator* and a *verification condition generator*. The code generator translates SeNDLog specifications into imperative code, which can be executed for experimental analysis. On the other hand, the verification condition generator works on the abstract syntax tree of the SeNDLog specification. It syntactically extracts proof obligations, and outputs all necessary definitions, axioms and lemmas needed for proving user-specified security properties.

Performance evaluation: The performance evaluation of the secure routing protocols is performed using the RapidNet [1] declarative network toolkit. The SeNDLog specification is disseminated to all nodes participating in the secure routing protocol, and is compiled into imperative code executable both in simulation and as deployment. The RapidNet toolkit then automatically measures multiple performance metrics such as latency, bandwidth utilization, and convergence time.

Formal verification: Security guarantees provided by secure routing protocols can be conveniently verified. We develop a trace-based semantics of SeNDLog programs along with a sound program logic deriving security properties of programs running concurrently with adversarial codes. Users only need to specify the security properties to be verified, and the invariants of each head tuple. The verification condition generator then automatically constructs proof obligations based on the specified invariants. These proof obligations are discharged using a theorem prover. Currently we uses Coq, an interactive proof assistant, to manually complete the proof. We plan to explore possible automation in the proof generation as our future work.

Case study and demonstration: We will provide visualized demonstration through case studies on two example secure routing protocols, namely Secure-BGP (a well-recognized BGP variant) and SCION (a clean-slate new design of Internet routing infrastructure). We will present how they are specified in SeNDLog, and empirically evaluated in emulation. We will also show the verification and comparison of the security properties provided by the two example protocols.

Student authors: Chen Chen, Hao Xu, and Cheng Luo.

References

[1] S. C. Muthukumar, X. Li, C. Liu, J. B. Kopena, M. Oprea, and B. T. Loo. Declarative toolkit for rapid network protocol simulation and experimentation. In *Proc. SIGCOMM (demo)*, 2009.

[2] W. Zhou, Y. Mao, B. T. Loo, and M. Abadi. Unified Declarative Platform for Secure Networked Information Systems. In *Proc. ICDE*, 2009.