Specification Modeling and Validation Applied to Network Security Gateways

Robert J. Hall
AT&T Labs Research
180 Park Ave, Bldg 103
Florham Park, NJ 07932
bob-2RE01_Demo-@channels.research.att.com

1. Overview

A network security gateway protects the computers of a home or small office from Internet-based attacks by remote adversaries. It allows all the protected machines to share a single connection to the Internet and may allow secure access, via a VPN tunnel, into a remote corporate network. It may provide other services as well.

Because its main job is security, it is critical to have high confidence that it meets its requirements. This is only partially obtainable through testing, because while testing can check that *expected* behaviors occur, attackers may exploit *unexpected* behaviors. Therefore, I am exploring tools and techniques for applying more sophisticated tools to validating the implementations of these devices. The primary difficulties are a lack of detailed knowledge of the behavior of standard platform components (such as the Linux operating system) and, of course, incompleteness in known requirements of the overall system. In this presentation, I will demonstrate how I use *executable specification modeling* and *lightweight formal methods tools* to help discover, validate, and refine requirements models. This process iteratively constructs a formal, executable model of (an abstraction of) the implementation and validates behaviors and properties, while suggesting experiments to perform on the implementation to reduce ignorance. The tool suite used is the Interactive Specification Acquisition Tools (ISAT) reactive system design suite[2].

2 Process

Model Creation. We must first construct an *executable specification model* of the implemented system in ISAT's P-EBF specification language[1], with the goal being a faithful abstraction of the implemented system, since the latter will generally be too detailed to reason about directly. Note that due to uncertainty surrounding some components, at least initially, we must make guesses in modeling them (e.g. "how does the Linux routing package work in detail?").

Model Validation. Using ISAT we can

- *simulate* scenarios. Simulation can easily cover behaviors that are difficult to test in the implementation, such as sending IP packets with a peculiar structure.
- *apply coverage metrics* to our library of scenarios. Gaps in the coverage suggest new cases that should be simulated. These may also lead to the discovery of ambiguities or infidelities in the underlying models of the imperfectly understood components.
- *prove properties* of the spec model. This can increase our confidence in the analogous properties holding of the implementation. For example, we could prove that "no packet coming in directly from the open Internet can be directed into the VPN tunnel." This may be impractical to prove of the implementation directly.
- *incrementally revalidate* a changed model when a fix is made. Since the process is fundamentally iterative, we must be able to resimulate, remeasure coverage, and reprove properties. ISAT tools support this as well.

Experimentation The key difficulties we face are "knowing *that* we don't know" and "knowing *what* we don't know". Once we can formulate questions (with help from modeling and validation phases) we can perform specific experiments on the implementation to reduce the knowledge gap. For example, a scenario inspired by a coverage gap may lead us to wonder "what does connection tracking mean for ICMP packets?" We then set up an experiment to test this aspect of the packet routing code.

References

[1] R.J. Hall; Specification, validation, and synthesis of email agent controllers: a case study in function rich reactive system design; in *Proc. 3rd Workshop on Formal Methods in Software Practice*, 2000, ACM.

[2] www.research.att.com/~hall/isat-project.html

Proceedings of the Fifth International Symposium on Requirements Engineering (RE'01)
1090-705X/01 $10.00 © 2001 **IEEE**

IEEE COMPUTER SOCIETY