David Choffnes

Backtracking Intrusions
Samuel T. King, Peter M. Chen (UMich)

One-liner: This paper presents a technique that uses logging and graph visualization tools for tracing the origins of intrusions after they have been detected.

The best part about this paper is that it is rich with examples and graphs that helps the reader to understand exactly what the Backtracker tool provides. As the authors demonstrate, their tool provides a fairly complete view of how attacks take place and give administrators a rich set of information for analyzing such attacks. I agree with the authors that the OS is the level at which human-understandable logging should take place and as such, I find their implementation to be a very practical once. The use of VM tools, although at first it seemed wasteful with respect to overhead, ended up being quite useful, as the authors were able to use ReVirt to replay the attacks. I'm also glad that the authors used UML for virtualization, simply because I like the tool.

One of the glaring issues in this paper is that the authors had not yet implemented logging of several notable OS-level events. I'm curious as to why this was the case and what, if any, barrier there was to including them for the paper. The end result is that the authors have less of a claim on the effectiveness of their solution. Perhaps if the authors went to a blackhat site, downloaded some scripts and launched a larger set of attacks against the system, they could have made a stronger claim about how well it can trace attacks. I also would have liked to see the authors include automation for triggering backtracker when an intrusion was detected. Another major issue in the paper is that, for heavily-used machines such as servers, 1.2GB of logs per day (compressed!) and 9% runtime overhead is a bit too much to swallow. I don't think that the authors really made the case for accepting this level of overhead. As far as the minor concerns go, I'm not convinced that the method for marking events by a single counter will work on multiprocessor machines (which are common in servers), and global synchronization for modifying this counter might worsen performance. Also, considering that the authors have a good idea of when a graph is too large for an admin to interpret, I'd like to see them eventually add a mechanism to prune the graph dynamically according to some default graph size. Finally, everything was done in Linux, and it even relied on inodes. I'd like to have seen this be a little less architecture dependent.

Overall, this seems like a nifty tool, but it doesn't seem to be a killer app. Perhaps I would be more convinced if the authors had given me more of an idea about what admins will doe with this tool that will significantly impact security as we know it.