Scalability, Fidelity, and Containment in the Potemkin Virtual Honeyfarm (Review)

Ken Murphy
This paper details the implementation of the Potemkin honeyfarm. A honeyfarm is a collection of honeypots. Honeypots are used to serve as fake servers. The goal is to have an attacker attack the honeypot instead of your actual servers doing real work. A honeypot usually has systems in place to gather data and log everything that the attacker is doing for later analysis which helps understand your adversaries better. There are many honeypot systems available currently but they all have their particular drawbacks. Low-interaction honeypots are one type, they simply emulate network services and do not keep any state. This may make it easier for a skilled attacker to realize they're attacking a honeypot, netting you no new information. Another type is high-interaction honeypots which emulate an entire system. This makes the environment more real however they require more resources and do not scale well. The authors have implemented a new honeypot system, knows as Potemkin. Their system combines the best of both systems.

The Potemkin honeyfarm system is a hybrid of the two standard approaches. When a new attacker connects the network gateway automatically dispatches it to the correct honeyfarm server. The server automatically instantiates a new IP address and prepares itself for running the required service. It does this in two ways, the first is flash cloning. This method copies the memory of a previously prepared base image and continue execution. This method removes the start up cost of bringing up a virtual machine from scratch, which an attacker might find as odd. The second method is delta virtualization. This method uses copy on write, as the memory needed probably already exists. This methods also removes the start up costs. Both methods are used in tandem in order to create new virtual machines almost instantly.

There are many things to think about when creating a honeyfarm. The first and foremost is what if an attacker uses it to launch further attacks against third parties. The Potemkin honeyfarm uses carefully implemented rules to track packets, identify non malicious patterns, and allow/deny based on those observations. This greatly reduces the possibility of the honeyfarm being used to attack other people. Their gateway server is the brains of this operation and implements all the rules. One interesting thing about it is that it can identify certain behaviors used as attackers, like a port scan. Under normal circumstances this would require the honeyfarm to create a ton of new VMs to handle each service request. The authors thought of this possibility beforehand and implemented detection to prevent this from happening. The gateway can proxy the scans and take action based on the severity of the behavior.

The hardware the authors used for testing are Dell 1750s and SC1425s. They use Debian with Xen as the VMM. A graph on page 175 shows the required virtual machines
based on a /16, peaking at around 9000. The authors faced many challenges while implementing the honeyfarm. The first is actually making sure the honeyfarm gets any traffic. As it does not perform any actual work or put itself out there, it only attracts random attacks. The second is detection, many attackers will simply stop attacking once they realize they aren’t attacking anything of value. This is especially true if it is detected as a honeypot.