Welcome to BlackHat!

Windows Security 2003 Briefings
Seattle, February 26-27, 2003

Timothy M. Mullen
Ryan L Russell
AnchorIS.Com, Inc.
Enforcer

Automated Worm Mitigation
For Private Networks
In Our Last Episode...

- Tim wants to shut down Nimda-infected boxes attacking him out of self-defense
- Lawyers assure him he will be thoroughly arrested
- We write a demo tool, and run it on three machines on the stage at Black Hat Vegas 2002
- It is, of course, quite legal to do this on a network you own
- Security people continue to come up with reasons why it might go wrong, without looking at what we proposed
The Problem

- Despite careful planning, policy, and firewall maintenance, someone manages to loose a worm on your Internal network.
- Copies of the worm propagate faster than you can stamp them out manually.
- You start advertising to the world that you’re infected (Microsoft and Sapphire, anyone?)
Proposed Solution

Under user-defined control:
- Detect worm activity
- Determine ownership of infected host
- Check rules for dealing with that host
- Exploit or log into the host
- Install agent software
- Agent mitigates the infection
Detect Worm Activity

- Use well-understood IDS technologies
- We need to only detect known code, so signatures are usually easy
- Need to have a small hook to get IDS output in real time
- Snort works great
Rules

- The system needs to have a record of what IP addresses it is allowed to fix.
- Needs rules for what kinds of action to take depending on which host is infected and which worm it is.
Dynamic Agent Install

- Use same or similar exploit as worm to execute code on the infected host
- When that’s not possible, utilize stored authentication credentials to install agent
Agent Execution

- Once the agent is executing on the infected host, it connects to the central station for instructions.
- Agent capabilities may include firewalling, rebooting, claiming mutexes, removing files and modifying the registry, or even full patch installation.
- Agent logs to central station at each step so there is a record of changes made.
Challenges

We have a harder job than the worm author

- Worms can get away with only working properly 50% of the time, we can’t
- We need more resources than worms (agent doesn’t fit in 376 bytes)
- We have to care about having permission to modify the box, even slightly, worms don’t
Architecture

Worms will always have a certain amount of lead time over the mitigation method. The worms have to be analyzed, and a strategy for neutralization has to be developed. By having a well-planned architecture already in place, delays are minimized.
Architecture (cont.)

- Have to be able to quickly distribute new IDS rules
- Code that contains “exploits” for each new worm must be upgraded
- Agent should be flexible enough to handle most situations, but it gets uploaded every time to each new infected box, so upgrades are easy
Constant Maintenance

The real value for a system like this is in the maintenance. For each new worm, an analysis must be performed, a variation on the exploit written, IDS rules created, and a mitigation strategy chosen.
Enforcer Architecture

Enforcer contains several logical components. These are:

- Sensor
- Manager
- Console
- Dispatcher
- Agent
Sensor

- Standard Snort software with custom Snort Output Plugin
- Logs each alert to the Manager for processing
Manager

- Receives alert information for the Sensor
- Sends commands to the Dispatcher to handle mitigation
- Communicates with the Console to display logs and accept configuration changes
- Contains all the logic to determine if a particular IP address is infected, if it is allowed to be handled, and with what method
Console

- Used to display logs from the Manager
- Used to make changes to the logic for handling mitigation
Dispatcher

- Accepts commands from the Manager containing IP address, worm type, and mitigation method to use
- Exploits or logs into infected host to install Agent
- Monitors Agent progress, logs all steps to Manager
Agent

- Contacts Dispatcher once uploaded to infected host and executed
- Receives command from Dispatcher for which mitigation method to use, and implements it
Demo