Snorting...

gdmr, October 2014

Progress

- Initial proof-of-concept up and running
- Pulledpork appears to work
  - With rules being automatically downloaded
- Snort itself appears to work
  - Logging to files
- Haven’t really looked at barnyard2 yet

Configuration

- “Sample” .conf files distributed with packages
- Rather large, and contain things which aren’t really “sample”
  - Distributed snort.conf is nearly 700 lines long
- Could perhaps template, but...
  - Rework with each new version
  - Rework with each change in requirements
- Edit and distribute in an RPM?
  - Inflexible
  - Likely to rot

Configuration

- So have written a fairly general pattern-matching and substitution engine
  - Matches on up to four (currently) strings
  - Can substitute any of them
  - Can clone lines and then substitute
  - Can comment or uncomment lines
  - Resource-driven through a tag-list
  - Works line-by-line, reading from stdin and writing to stdout

Configuration!

- Why not just generate some awk or sed?
- Not quite so simple...
  - Tool needs to know how to remove duplicates from substitution values (e.g. subnet lists)
  - Also needs to know how to turn whitespace-separated lists into snort lists, with [] and commas
- All rules are tried in turn and applied if there’s a match
  - Unless (resource-controlled) the rule says to terminate after its actions are done

lcfg-snort

- Configure() method
  - Takes the distributed pulledpork.conf and runs the configuration tool over it to produce a local version
  - Takes the distributed snort.conf and ditto
    - Kicks any running daemons if there’s any (non-comment) change
  - Generates a sysconfig file to control the init.d script
- Run() method
  - Runs pulledpork to generate new rules
    - pulledpork then kicks the daemons??
    - We don’t trust it to, so we kick them explicitly in the method!
- Start() method
  - Checks that the .conf files are in place, then uses Service() to start the daemons
- Stop() method
  - DOESN’T use Service(), as the init.d script is buggy
  - Finds the process pids and signals them all directly

Current test configuration

- Very much still a proof-of-concept!
  - Though done “properly” through component, headers, ...
- Current test .conf changes:
  - Fix up paths to match what’s in the RPM
  - Set oinkcode so we can download stuff
  - Set HOME_NET to define inside and outside
    - Uses INTERNAL_SUBNETS from live/subnets.h
  - Sundry other odds and ends
  - 17 match/subst rules affecting 129 lines in snort.conf and 21 lines in pulledpork.conf – that’s nearly 20% of the “sample” files
- Current test sysconfig stuff:
  - Defines interfaces
  - Sets up alerting mode (to “fast”)
  - Saves packets for wireshark etc
    - Which often can’t cope, as they’re “malformed”

Testing...

- Running on Forum and AT perimeter routers
  - On vlan42 and vlan160
  - Saving alerts and packets to /var/log/snort/<vlan>/
    - Very roughly 200MB/week
    - So very very roughly 4GB per 120 days
- Quite cpu-intensive
- Doesn’t seem to have affected network throughput
- Does seem to have made the machines less responsive interactively

Some numbers

- Snort alerts tallies have been pretty variable!
  - Anything from just about zero to almost 100k alerts per day
  - Vast majority against web servers of various kinds
  - Note though that snort has a limited view of traffic coming in through the edge
    - Because pcap sees packets before they get processed by iptables
    - All UDP and ICMP
    - But only TCP where there’s a filter hole
- Edge filters dropping 2-3M packets per day
  - that’s still around 0% of overall throughput!
  - Most-probed ports are for Microsoft things

Impressions of snort

- Basically sound in concept
- Lots of community activity
- Toolchain doesn’t seem totally integrated
  - Do the developers actually talk to each other?
  - Do they talk to the packagers?
- Reports on list of occasional fragility
- Single-threaded; multi-threading awkward
- Configuration is horrible, and clearly not designed for management tools
- Jargon!

What next?

- We have a prototype running, so principal deliverable has been delivered
- Wind up project?
  - Or throw some more effort at it?
  - Who??
- Enhance reporting and actions?
- Look at intrusion-prevention
  - Nervous about trusting perimeter to others
  - Throughput? Latency?
- Investigate other tools?
  - Suricata looks promising
    - Configuration seems cleaner at first glance
    - Somewhat snort-compatible
    - Multi-threaded
    - Use pulledpork again to obtain up-to-date rules?

What next?

- How do we make use of the reports?
  - Toss into syslog?
  - Barnyard2?
    - Looks like a spooler for a database of some kind
    - So still need to find or write a reporting tool
  - Something else??
- Procedures?
  - ...??