Considerations for Scan Detection Using Flow Data.

REDJACK

Overview

- Scans and scan detection – goals and objectives
- A review of Threshold Random Walk
- Real time vs. Flow based approaches
- Bi-flows and Oracles
- Extensions
  - to ICMP and UDP
  - indeterminate reduction to improve benign detection
- Beyond detection – actionable intelligence
- Comparisons with `rwscan`
- Conclusions and future directions

Scans and scan detection goals and objectives

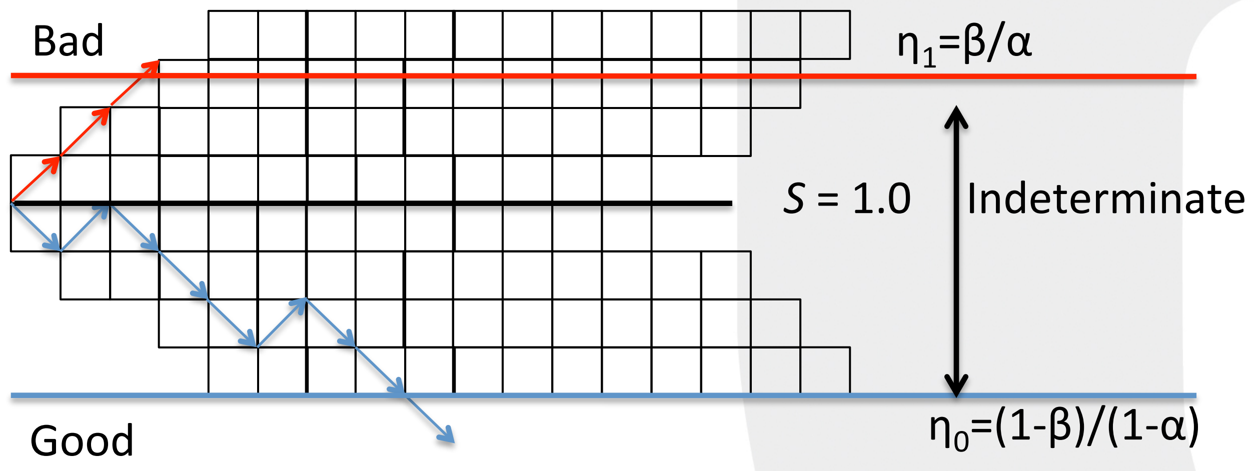
- At one time 90% of internet traffic was scanning
  - Now about 10% or so, so why do we care
- Still a viable propagation mechanism for malware
  - many newly compromised machines scan locally
  - scanning of entire internet happens, e.g. sip server
    - *Analysis of a "/0" Stealth Scan from a Botnet* – CAIDA
- Scan detection provides situational awareness
  - What is sought, who is looking on a global level
- Responses provide local inventory
- Interactions with scanners can identify compromise
  - actionable in many cases

Scan detection
Threshold Random Walk (TRW)

REDJACK

- Assumptions
  - good guys connect, bad guys don't (mostly, for both)
  - bad guys behavior random, targets random (hah! / huh?)
- Model both behaviors
  - analyze connection attempt sequence
  - choose between good guy / bad guy hypothesis
- Need probabilities for models
  - θ_0 – good guy connects
  - θ_1 – bad guy connects
- Score *S* starts at 1.0 (indeterminate)
  - Successful connection multiplies score by θ_1 / θ_0
  - Failed connection multiplies score by $(1-\theta_1) / (1-\theta_0)$

TRW scoring and classification



- α is the desired false positive rate (0.01 often used)
- β is the desired detection rate (0.99 often used)
- $\eta_1=\beta/\alpha$ and $\eta_0=(1-\beta)/(1-\alpha)$ set the bad and good thresholds for the score S
- For a given set of parameters, possible to calculate min *all hit* counts for good and min *all miss* counts for bad

TRW and oracles

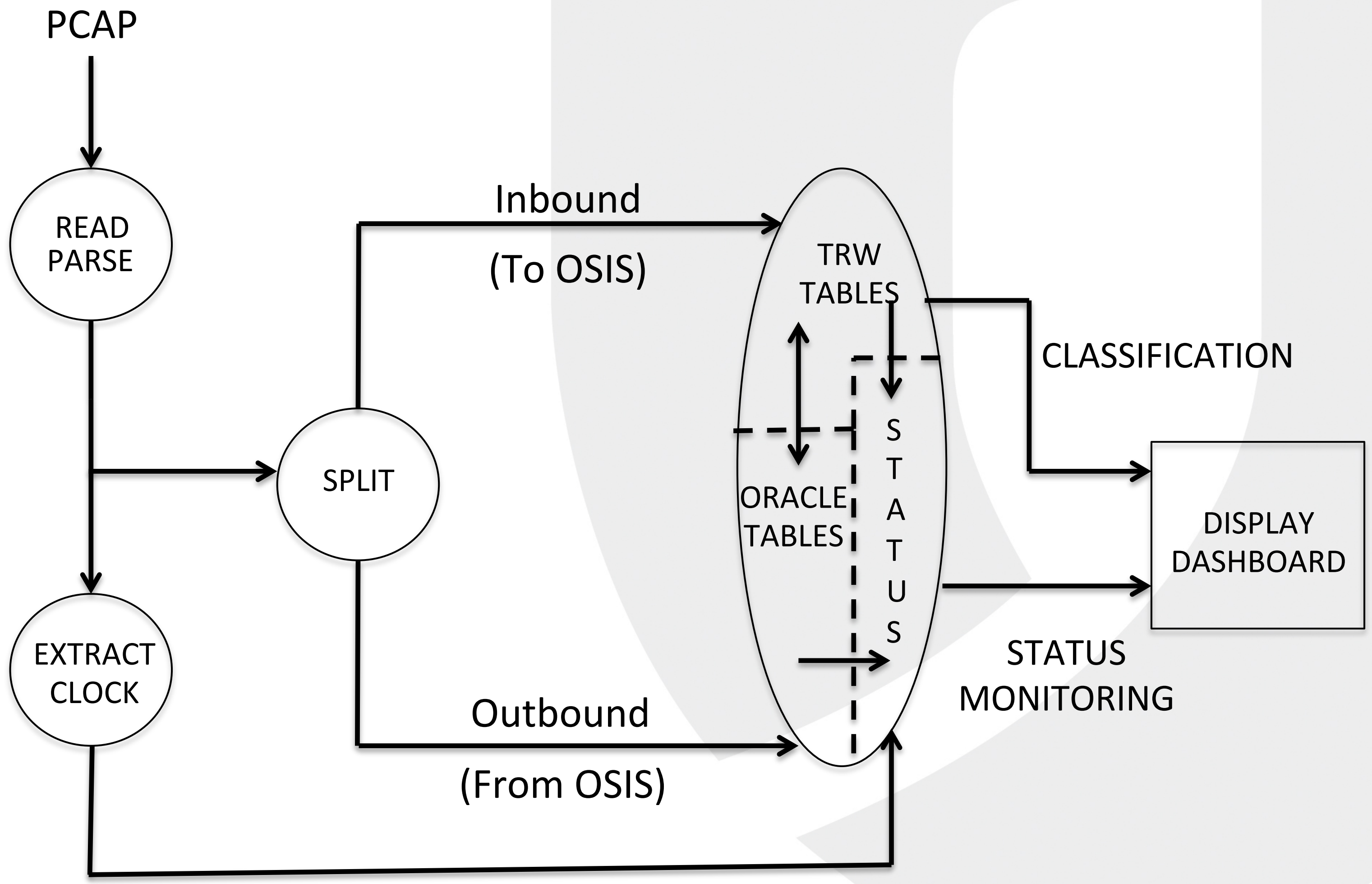
- In real time, hit/miss determination hard / expensive
  - Scan may be over before you can score
  - Use an *oracle* to predict connections
- An *oracle* tracks internal network services
  - Updated dynamically by outgoing traffic (or static)
    - For *ex post facto* analysis, oracle can be calculated from outbound traffic for an epoch, prior to inbound scan detection
    - Analysis of inbound traffic can be used to create an oracle if bi-directional traffic is not available
    - Both are effective with flow
  - Used to evaluate connection attempts
    - Works through temporary outages reducing false misses

Multiple oracles for multiple scan modes

REDJACK

- TRW primarily used for TCP scans
  - Service oracle from sources that lead with SYN/ACK
  - Include service (responsive port) for precision
  - Can deal with things like passive mode FTP
- UDP oracle possible, as well
  - Can infer UDP service ports over time
- ICMP (ping) oracle trivial from ping response flows
- Adding ports improves precision
  - Detects vertical scans / mixed mode scans
  - Host only oracles for non-SYN TCP, etc. work too.
  - Computation of appropriate θ_1 is interesting
    - Randomness assumptions probably not correct

"Real time" TRW workflow for prerecorded pcap data



Flow is liberating (somewhat)

- Can separate oracle maintenance and scan analysis
  - Can pre-compute oracle for analysis epoch
  - In the absence of outbound data, can infer consensus oracle from multiple complete connections
  - With enough state, can detect very slow scans
    - Can even detect distributed scans with a bit of thought
- TRW computation simplified with oracles
  - Per host target lists most difficult part
    - Cuckoo sets for {source, target, service [, mode]}
    - Bloom filters to eliminate duplicates
    - Short, linear, list of targets (indeterminates with many targets are very rare – can be special cased)
    - Sorted data (as with `rwscan`)

The dirty truth about indeterminates

- TRW requires minimum target count to classify a source
  - Lots of sources have all hits to too few targets
    - Regular users of your primary web site (nothing else) OK
  - Lots of singletons (one target, hit or miss, never again)
    - Can probably forget about them (or aggregate off line)
  - Partial results from multiple locations / epochs compose
    - Could put partial results in a DBMS & periodically compose
    - Detect very slow scans this way composing on source
    - Detect distributed scans composing on service
      - Look for aggregates with good coverage
  - The epoch over which the initial analysis is done sets the detectability threshold.
    - Probably want a continuous process with table maintenance

REDJACK

Beyond detection – what now?

- TRW in real time can be an active defense
  - Block scanners before they learn about you
  - With flow, it is too late (even in the pipeline)
- *Ex post facto* detection can
  - Identify possibly compromised machines
    - Significant exchanges between scanner / scanee bad sign
    - Even small exchanges are a danger sign
      - Link target service to vulnerabilities and prioritize fixes
  - Characterize scan targets to see “what’s hot”
    - Fix vulnerable machines based on scan interest
    - Whether machine has been successfully scanned or not.
  - Trends over time – repeat scanners, modes, services

Comparison with `rwscan` (I)

- Flow data from 14 months of a /22 in Canada
  - oracle is set of all active hosts
  - Implementation using cubags
    - Span bag – all inbound sources w active interval as data
    - Hit bag – all src/dst pairs w dst in oracle (# flows as data)
    - Miss bag – ditto for dst not in oracle
    - Project dst off hit / miss bags and roll up to dst counts
    - Join projected bags, span bag to give src, hit / miss counts
    - Compute TRW score and classify.
  - We took 0:13, `rwscan` took 3:15 (`malloc` ???)
    - Found 8000 more scanners, 75,000 more benign than `rwscan`
    - 400,000 indeterminate, mostly too few flows, some single target with many repeats and lots of flows (5% of total flows)

Comparison with rwscan (II)

- IARPA (OSIS) data from PREDICT
  - Streaming pcap implementation for comparison
    - No timings: different platforms and demo stream slowed
    - Flow at 1 pkt/flow from rwptoflow
  - Separate oracles for Hosts, TCP, ICMP
  - Results for background data (scenario 5b5)

	rwscan	Host	TCP	ICMP
Scanner	14	16	26	5
Benign	-	329	39	0

  - Host includes 1 UDP, 1 ping + 14 detected by rwscan
  - TCP includes 12 vertical, 2 mixed + 10_1 + 2_2 by rwscan
  - Only 1 ICMP detected by rwscan. Others less than 32 flows (Minimum for missile component)

Observations

- Stopping analysis on classification only good in real time
  - Can take action (block, whitelist, etc.) in real time
  - In batch mode lose information on volume, targets
- Benign classifications are important
  - Useful to know nice as well as naughty
    - Detect behavior changes
- Multiple oracles very useful.
  - oracle data is a cheap dynamic system inventory
- Confounding scan detection with backscatter analysis, etc. is not useful.
  - This is not an “either / or” case

Future Directions

REDJACK

- Refinement of θ parameters
  - oracle allows tightening of θ_0 (closer to 1.0)
  - What is the actual target density (θ_1)
- State maintenance for continuous operation
  - Management / pruning of indeterminate hosts
- oracle maintenance
  - Might link removal to DNS ttl?
  - New services / transient ports
- Consequences of scanning
  - Compromised host detection
  - Prioritization of patching – CVE/NVD linkage
- Distributed scans might be tractable

Conclusions

- Scan detection is still important
- Most useful in real time, but *ex post facto* is useful
- Can be done with flow – has some advantages
- Predictive oracles better than traffic matching
  - A miss should be a hit sometimes
  - Multiple oracles for multiple scanning modes work
- Management of “indeterminates” is important
- Diagnosing “benigns” is important
- `rwscan` needs to be replaced
  - Scan database needs more information
  - Need to feed operationally useful actions

Questions / Discussion

John McHugh
Senior Principal
RedJack, LLC
john.mchugh@redjack.com

I'll be around for the rest of the meeting.
Come talk to me.

REDJACK

Questions?