An Architecture for Developing Behavioral History

Mark Allman (ICSI), Ethan Blanton (Purdue),
Vern Paxson (ICSI)

SRUTI, MIT, July 2005

*"We like a long neck and a good old song,
Turn it up and then we’ll sing along."*

Example Problem

- Say some host H wants to access some network service on your host or network.

- What can we use to decide whether this request is from a good or bad actor?
  - IPsec / TCP-MD5
  - local cache of previous activity
  - knowledge of the local network
  - centralized databases of "behavior" (www.dshield.org)
  - central list of machine types (e.g., "dialup IPs")

Example Problem (cont.)

- All the information is either:
  - *narrow* in scope
  - *difficult* to obtain/setup

Proposed Architecture

- Goal: prevent unwanted traffic on the Internet
  - DDoS, worms, spam, scanners, etc.
  - *think in terms of commonality*

- Problem: network interactions are largely anonymous and self-contained
  - mostly true

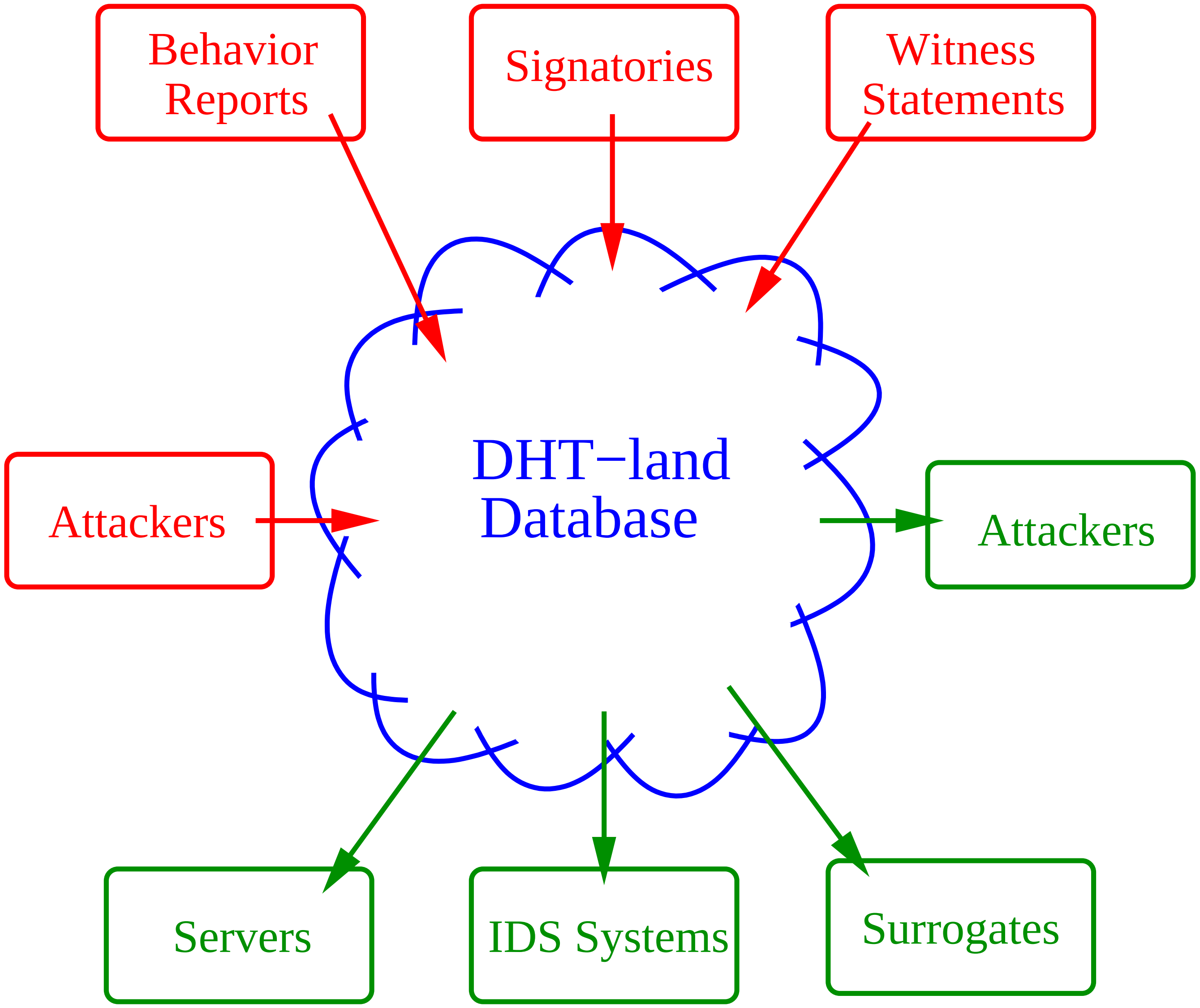
- Solution?: we have an *architectural approach* to track the history of bad "actors"

Proposed Architecture (cont.)

- Devise a database for accumulating information about malicious unwanted traffic:
  - Internet-scale
  - handles arbitrary "unwanted" traffic
  - distributed and robust
  - policy neutral
  - open

- Proposed as a community project because it’s big and complicated and could use help from smart people.

Proposed Architecture (cont.)



Proposed Architecture (cont.)

- Record insertion:

```
report = sign (report, my_opaque_key)
dht_database [bad_actor] += report
dht_database [my_opaque_key] += report
```

- Keys ... the kiss of death?
  - only used to correlate reports from the same entity
  - not tied to identity
  - no "PKI"

Behavior Reports

- We focus on attacks; could focus on other aspects of behavior

- Insert record with:
  - timestamp
  - actor identity
  - protocol and port number (optional)
  - behavior observed
  - behavior digest
  - signature

- Inserted whenever the reporter wishes
  - uh-oh!

Witness Statements

- Goal: generate an *audit trail* that offers evidence that some behavior report was not completely cooked up
  - ISP tracking packet digests
    - traceback, obits, etc.
  - mail server inserting a (signed) notation that a given message had traversed the server

- *A witness statement is not a judgment, but rather a statement of fact from a third-party*

Signatories

- Hosts that use particular database information in making policy decisions can sign those records to indicate their use

- Much like PGP’s "web of trust"
  - not quite the same because there is no hard and fast notion of *identity*

Policy

- The database provides a source of information that may or may not be used as part of *local policy decisions*
  - could deny access
  - could rate-limit access
  - could watch the traffic more closely
  - etc.

Trust

- The key problem with setting policy based on information from others is *trust*

- The information from the database may be wrong:
  - ▸ the reporter may have made an inaccurate assessment
  - ▸ the reporter may have intentionally lied
  - ▸ the information may be out-of-date

Trust (cont.)

- There will be bogus information in the database
- We address the problem of trust by using *locally-determined reputations*
- We can access an actor’s history, a reporter’s history, witness reports, witness history, etc. as fodder for reputation assessment

Trust (cont.)

- We can assess the *reputation* of various reporters:
  - do lots of entities *corroborate* some assessment?
  - have many entities signed reports?
  - does the audit-trail support the reported behavior?
  - do we have local evidence that is consistent with the reported behavior?
  - (we might even know the identity of a reporter!)

- All these can be gamed
  - we need research into reputation assessment schemes
  - some work done, much more needed

Trust (cont.)

- We may need to adjust our principals:
  - e.g., openness
  - e.g., instead of witnesses maybe we need *expert witnesses*

- May need new notions:
  - e.g., *ringers*

Overhead

- Even small networks like ICSI’s are visited by thousands of hosts every day

- Lookup for every transaction?
  - computational burden
  - bandwidth burden
  - causes delay

- Cache?
  - well... maybe...

Surrogates

- Calculating reputations is burdensome

- Introduce *surrogates* to help
  - hosts on the network that constantly monitor the database, calculate reputations, etc.
  - make synthesized information quickly and easily available
  - local or global

Surrogates (cont.)

- Surrogates can take away *local* control, which is a fundamental notion to the system
  - surrogate could publish algorithms
  - the database is still available, so surrogates could be periodically *audited*

Issues

- Many, many more issues....
  - see the paper

Conclusions and Future Work

- We have sketched an architecture that we think the community could improve and implement
- However, the entire talk has been future work
- Questions? Comments? Thoughts?
- Useful? Not?