Overlay-based Active Monitoring and Security

Bobby Bhattacharjee  Tuna Guven
Christopher Kommareddy  Richard La
Mark Shayman  Vahid Tabatabaee

University of Maryland
A minor aside

- The *Schnell* attack on TCP

with Rob Sherwood

- Attack network core by causing well-provisioned servers to send lots of traffic (GBs) into the core by sending fake TCP ACKs
Is this feasible?

- The ACK estimating etc. has been implemented
  
  real attack: 128 Kbps user causes server to send 32 Mbps.

- Good news: there is an elegant fix (See TR)

- Bad news: There are probably other Schnells …
  
  … and of course all other well known attacks

- Lot of fixes require Internet-wide deployment of new functionality

Not clear if this is feasible or practical, in the short or the long term
Inter-domain Monitoring and Security using an Overlay

- Monitor and stop attacks at the source of the attack
  
  source ≡ first domain not entirely controlled by attacker

- Most efficient solution — attacks are stopped before they can do much damage

- Does not require Internet-wide deployment

- Shares the cost of attack monitoring and prevention
Approaches

- Firewall at the domain egress(es)
Approaches

- Monitor at each host
Approaches

- Overlay-based

Overlay nodes monitor at internal routers
Solution components — new ideas

- **Coordinate** and Correlate information between nodes

- **Local Oracle**
Solution components — new ideas

- Coordinate and Correlate information between nodes

- Local Oracle
Local Oracle (Hardware)

- Pass-through processor on NIC with a physically secure key $\mathcal{K}$
  
  Cannot be controlled via host software

- Passive monitor of all network traffic

  Logs (compressed) all traffic [headers+snippet]
Local Oracle (Hardware)

- Pass-through processor on NIC with a physically secure key $\mathcal{K}$
  Cannot be controlled via host software

- Passive monitor of all network traffic

  Log requires 1 MBytes storage per minute of data (avg.)

  worst case 1 order of magnitude worse.

- Log dumped to sender when packet with $\mathcal{K}$ intercepted

  Consider adding rudimentary filtering instead of log dump?
Local Oracle (Hardware)

- Pass-through processor on NIC with a physically secure key $\mathcal{K}$ Cannot be controlled via host software
- Passive monitor of all network traffic

  Log requires 1 MBytes storage per minute of data (avg.)

  worst case 1 order of magnitude worse.

Attackers (can) know of the oracle, but cannot modify its operation
What can such a system do …? 

● Detect different attacks — DoS, malicious packets
  
  – More capable than single node systems

  – Aggregation of local information towards root \(\rightarrow\) correlation

  – Adaptively locate problems towards leaves \(\rightarrow\) refinement

● Complete single packet traceback (using local oracle)
  
  does not require global deployment
So, is distributed monitoring really necessary?

- Consider current hardware
  
  OK, say *only* 1 Tbps access link [\(\sim 1 \text{ ns/avg.packet}\)]
  
  Even Gbps links must be serviced in 320 ns
  
  SDRAM access times [10 ns*]; expensive
  
  L1 caches [<1 ns access]; prohibitively expensive

- Implications:
  
  Extremely limited per packet processing
  
  Infeasible to keep per flow state
  
  Incomplete information [sampling]
So, is distributed monitoring really *necessary*?

**Answer:** Yes.

Multi-node solutions provide exponential benefit.
Example: Detection of a single DoS flow

- Assume binary tree topology, one op. per packet [worst case for multi-node]

- Assume N flows, mapped to $k$ bins
  
  Single node, in one round
  
  reduces # of suspected nodes to $N/k$

- Suppose, instead, we have $t$ overlay nodes (anywhere on path)
  
  Worst case, in one round + 1 prop. delay
  
  # suspected flows reduced to $\frac{N}{2^t k^t}$

  Overhead: 1 bit/packet inline, or $O(t)$ extra comm.
**Example: Detection of a single DoS flow**

- Assume **100K flows, 1024 bins**
  
  Single node, in one **round**

  # of suspected flows — 100

- With overlay monitoring, suppose **1M flows** and only **100 bins** per node

  
  \[
  \begin{array}{ccc}
  \text{# monitors:} & 2 & 3 \\
  \text{# suspected flows:} & 244 & <1 \\
  \end{array}
  \]

- With **1000 bins per node**, **3 nodes** can detect 1 in 8 billion flows in 1 round of detection + communication
Summary: General Approach

- Overlay Communication infrastructure — provides general primitives such as multicast, naming useful beyond monitoring/security

- Specific statistical tests implemented in a distributed manner using comm. primitives over input data primarily borrow from existing literature

- Input data locally generated for specific tests/attacks defined by environment, node capabilities, range of attacks
Current work and Future Directions

- Tests for various types of DoS attacks, and also a traceback mechanism

- Ideally, we’d like to BUILD the local oracle hardware

- Extend current work to handle multiple egresses

- Fully develop general approach with multiple examples of tests and distributed statistical computations

- Develop more tests — possibly extending into virus detection