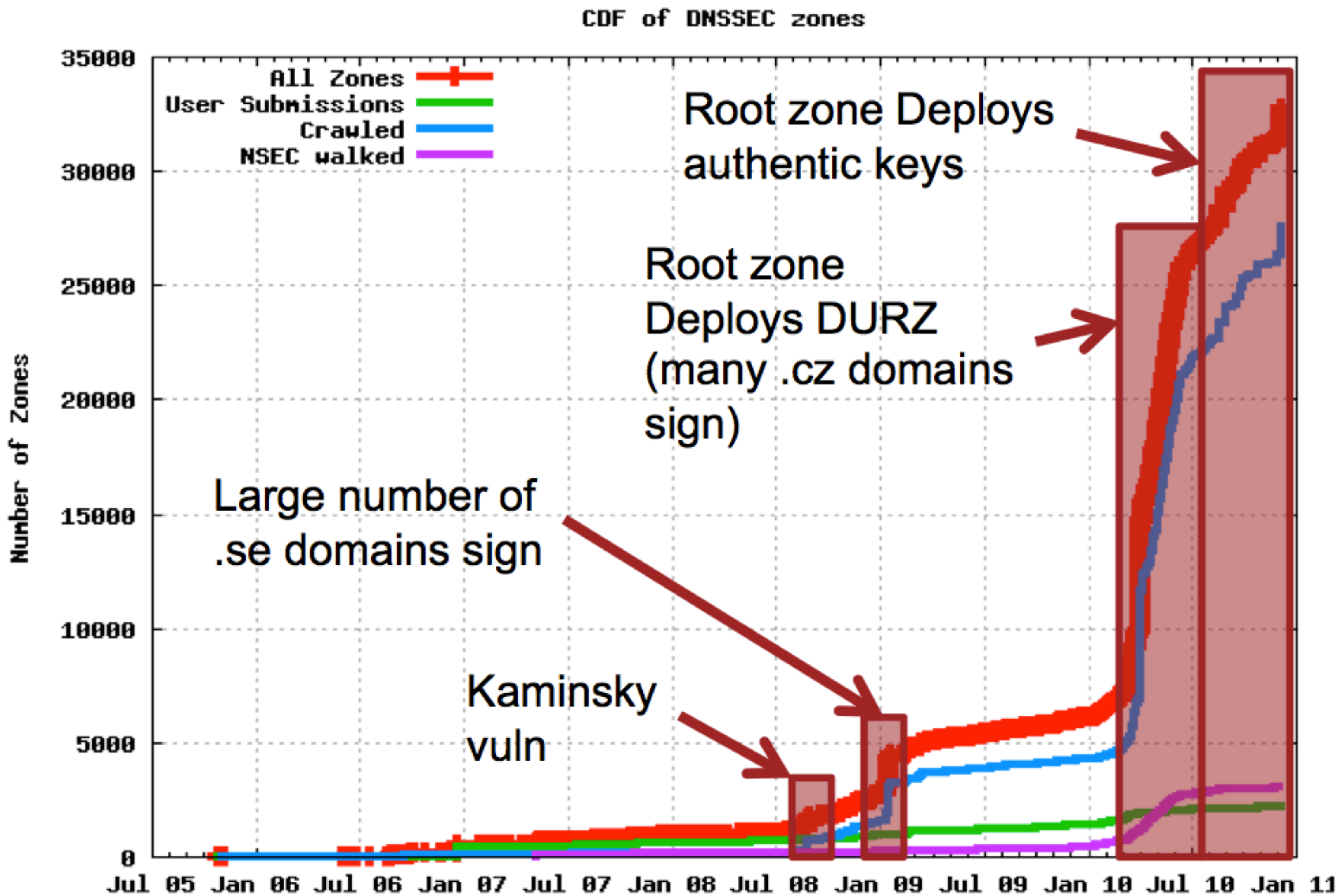
Verifying Keys through Publicity and Communities of Trust

Eric Osterweil
Dan Massey
Danny McPherson
Lixia Zhang

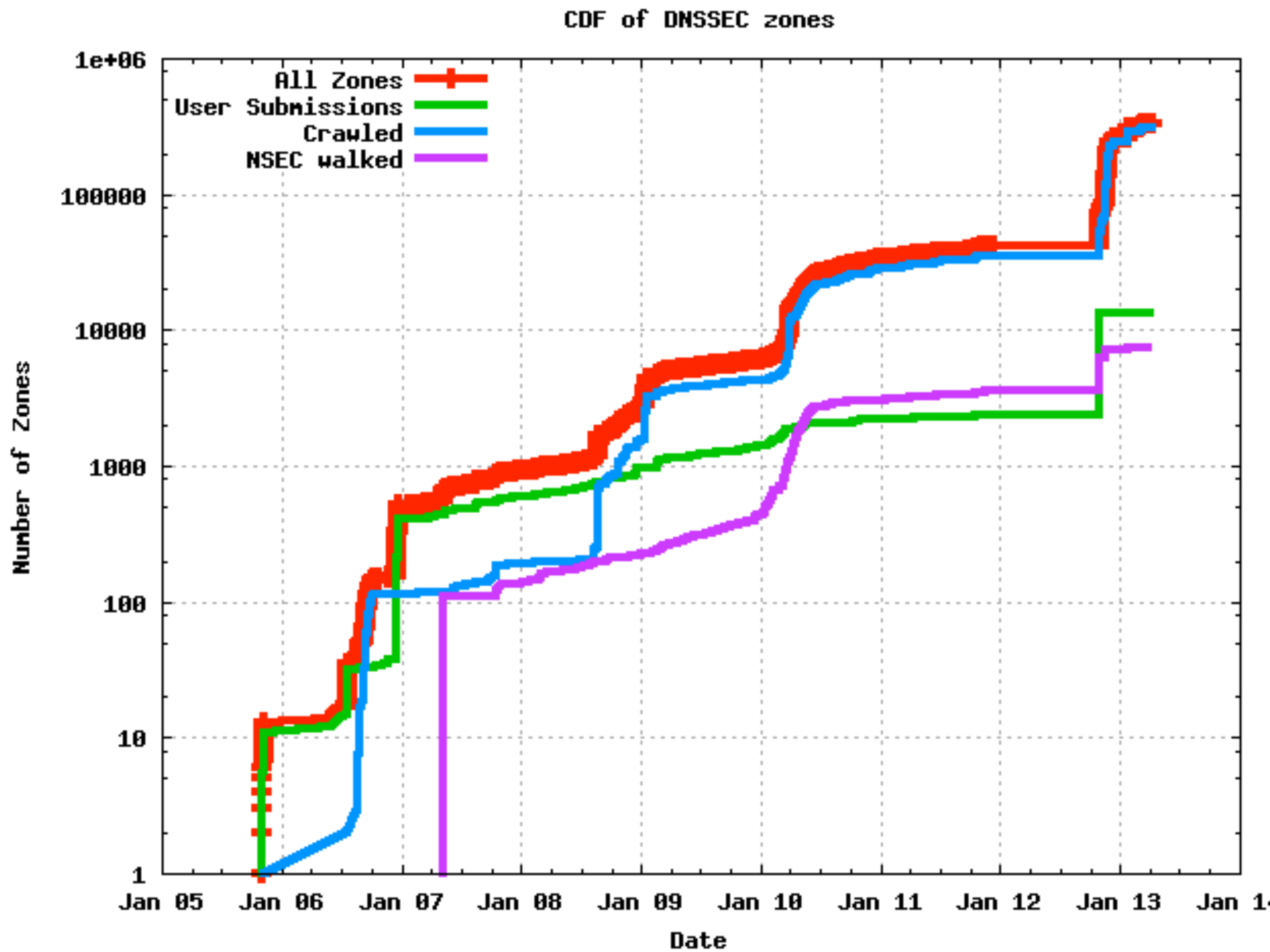
DNSSEC: Security for a Core Internet System

- DNS is a staple of today's online activities
  - Is there a pedestrian online activity that *doesn't* use DNS?
  - We use it to map unique names to network resources
  - It has long been a very robust system
- DNSSEC makes DNS the first core Internet system to protect itself and its data with hierarchical crypto
  - Protects DNS from cache poisoning and spoofing
  - 2010-2011, root and .net, and .com deployed DNSSEC
  - A straightforward design crypto-enhanced systems design
- The deployment has been growing, and standards are being built on DNSSEC: *DANE* (TLS, S/MIME, etc.)

Motivations Grow the Deployment (Graph From SecSpider)



Today we need a log-scale view :)
http://secspider.verisignlabs.com/growth.html



Some Challenges for DNSSEC Remain

- DNSSEC's early life has shown some stability concerns
  - We've *already* seen broken delegations (.gov, .arpa, .fr)
- DNSSEC faces architectural misalignments
  - Looking up unique names ≠ Verification of public keys
  - The design struggles with misconfigurations and partial deployment (though this may not be unique to DNSSEC)
- DNS is a core staple, and outages are not OK
  - If someone puts the wrong DS record in their zone, is that game over?
  - Network partitioning can break online delegations

Some Core Questions

- Is black and white verification the only option for dynamic Internet-scale systems, like DNS?
  - DNS has thrived because its design tolerates failures and misconfigurations
- What kind of verification can one derive for Internet-scale systems with dynamism like this?
  - Such a verification system *must* tolerate the Internet's chaotic setting
- Can any other verification model that is based on such a shaky operational foundation be trustworthy?
  - Moreover, can it be *better* than what we have now?

We Propose to Verify Using the Network… Public Data and Communities of Trust

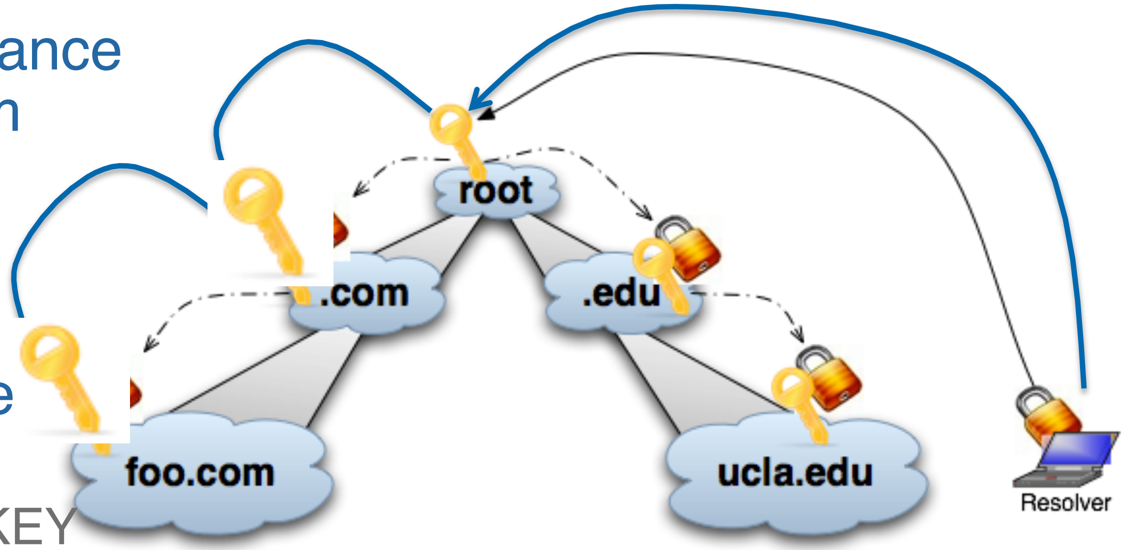
- Add distributed *redundant* measurements form *independent* paths as a new security substrate
  - Redundancy can overcome errors,
  - Publicity increases verifiability
  - Who to trust is subjective
- We propose the theoretical model *Public Data* to *augment* DNSSEC’s crypto substrate
- We implemented a candidate system called *Vantages* to demonstrate its feasibility

Outline

- DNSSEC background
- Public Data model and Vantages
- Measurements
- Conclusion

DNSSEC Crypto Key Learning + Verification

- First attempt to enhance core Internet system with crypto

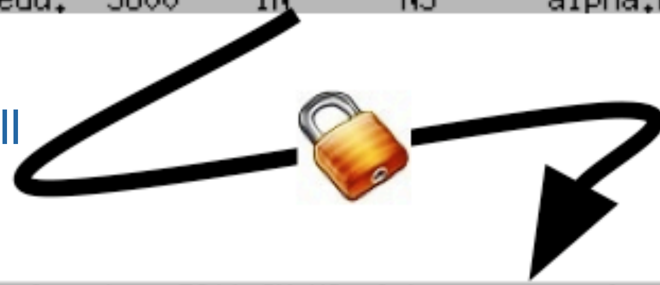


- DNSSEC zones create public/private keys
  - Public key is DNSKEY
- Zones sign all RRsets and resolvers use DNSKEYs to verify them
  - Each RRset has a signature attached to it: RRSIG
- Resolvers are configured with a single *root* key, and trust flows recursively down the hierarchy

Data Signing Example

Using a zone's key on a standard RRset (the NS)

```
secspider.cs.ucla.edu.   3600    IN      NS      zinc.cs.ucla.edu.
secspider.cs.ucla.edu.   3600    IN      NS      alpha.netsec.colostate.edu.
```

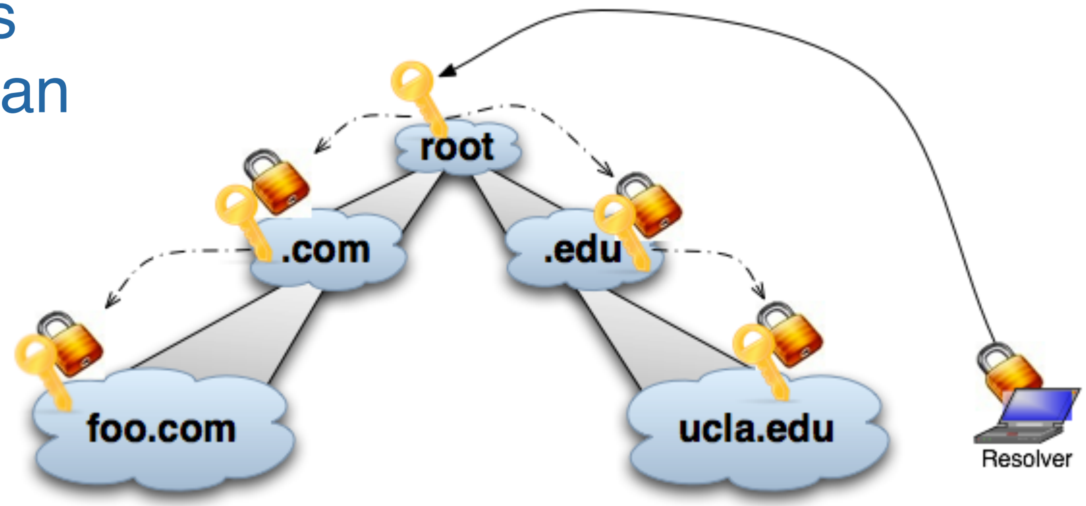


Signature (RRSIG) will only verify with the DNSKEY if *no* data was modified

```
secspider.cs.ucla.edu.  3600 IN NS alpha.netsec.colostate.edu.
secspider.cs.ucla.edu.  3600 IN NS zinc.cs.ucla.edu.
secspider.cs.ucla.edu.  3600 IN RRSIG NS 5 4 3600 20080324024800 (
                                20080322024800 44736 secspider.cs.ucla.edu.
                                E4msde1nzVlfGvwDo2X6jLU5d9Xrk37lrYRCZN6yq5ad
                                mABa3B3KgKl13u2VBXDujJZucHSwPQMBy+JOmotZOggf
                                SgQWUYm86v8G7ABHHcI+YFD3z3eqSoAoBAE5ysafop1u
                                g7tw1J4xd/IADIVeu1HnVIKRSycILXzvCwcaDWwAd610
                                9oJUBSMgWZjGzYeJO9RzOoUUqIqtn9PgVOzdTm+WnRC3
                                LEz50fdoP743QvPhe7RrF9w1KA3MOptTiQA++W8Gg085
                                NhbJ7MD99nEYaEv3+GuDCTkCy5ZOWoI/2Bcjq1NGBDLo
                                71lo6udu72i1tpyRfTEEQUirpInlZ9+IMw== )
```

Getting the Keys

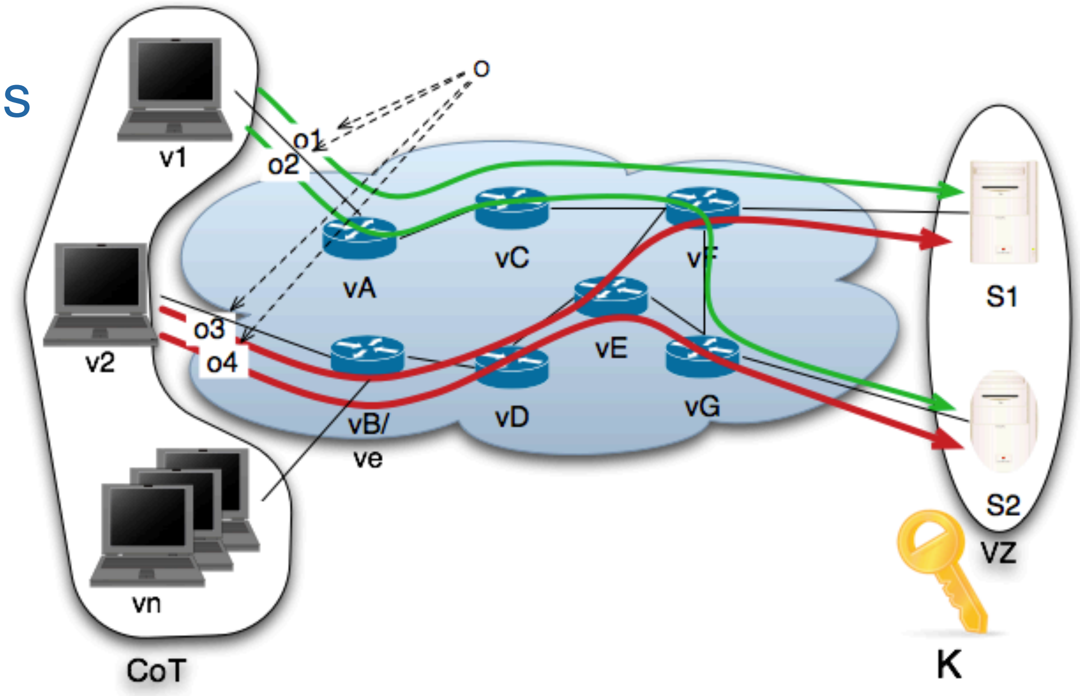
- Until a resolver gets DNSKEY(s), data can be spoofed
- Keys verified by secure delegations from parents to children
  - So resolvers know DNSKEYs are not being spoofed



- **DNSSEC's design needs the *full* hierarchy in order to verify keys**
  - No middle ground: either a key has a *verifiable* delegation, or you know nothing about it
- What if we just queried for crypto keys directly?

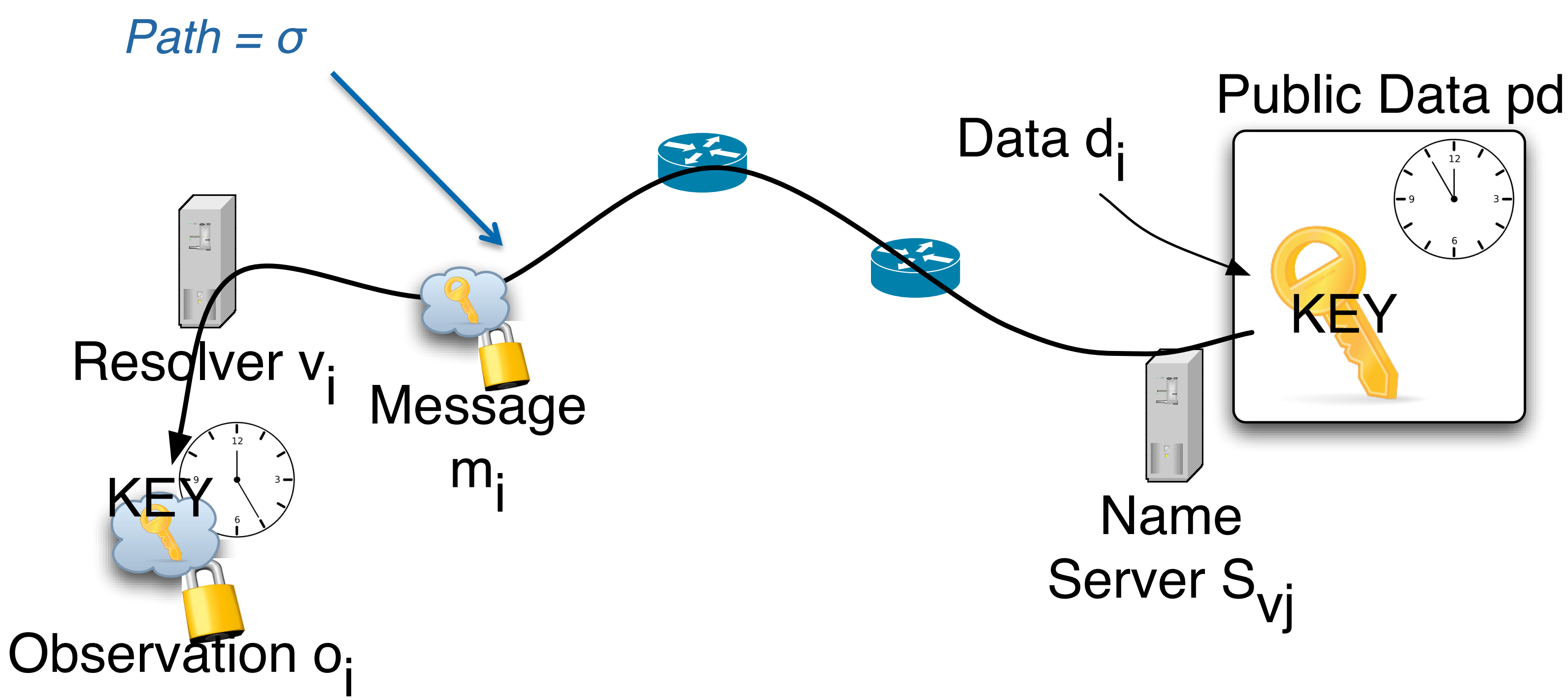
Public Data:
= Distributed polling + structured observations

- Verify DNSKEYs through Communities of Trust (CoTs)
  - *Consistency* and *redundancy* become the verification metric
- The network: topologically diverse vantages
  - G = (V, E)
  - $V = \{v_0, v_1, \ldots v_n\}$



- Observations: bind data to time and a network path
  - Path $\sigma_{(i,j)} = (v_i, \ldots, v_j)$
  - Data (such as a DNSKEY): d
  - Observation $o_i = (d_j, t, \sigma_{(i,j)})$

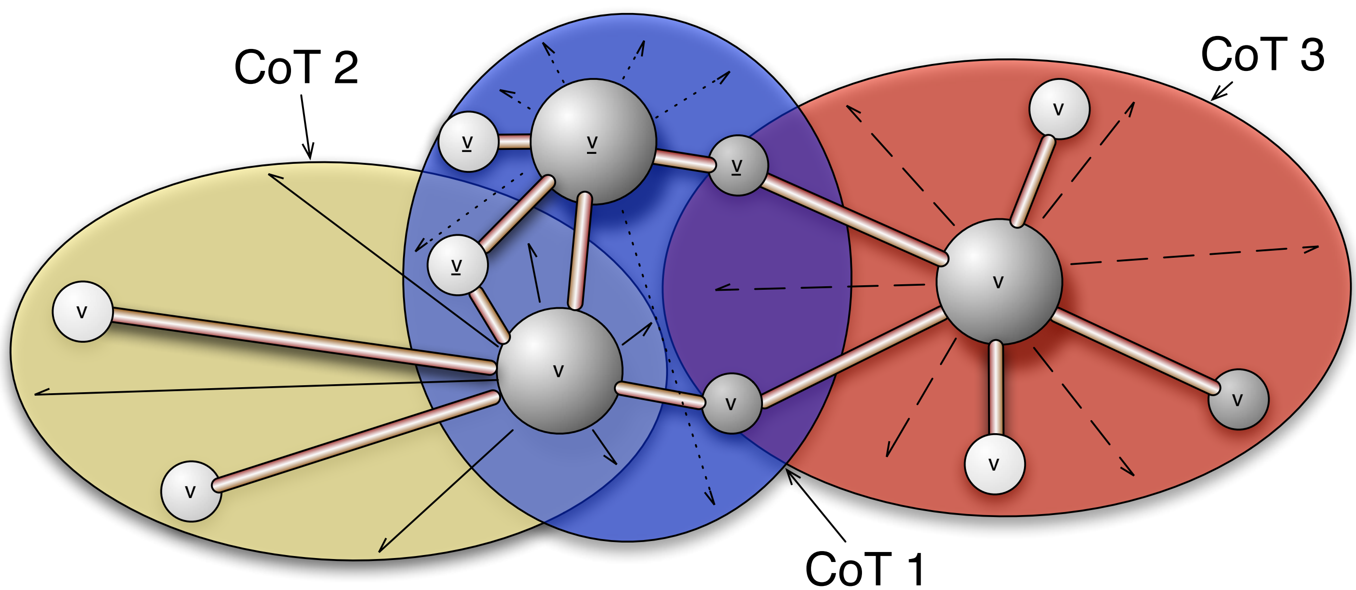
Public Data Model



- $pd_i = (d_i, tk)$
- $S_{vj} = \{pd_0, \ldots, pd_m\}$
- $m = (d_i, Sig_K(d_i))$
- …

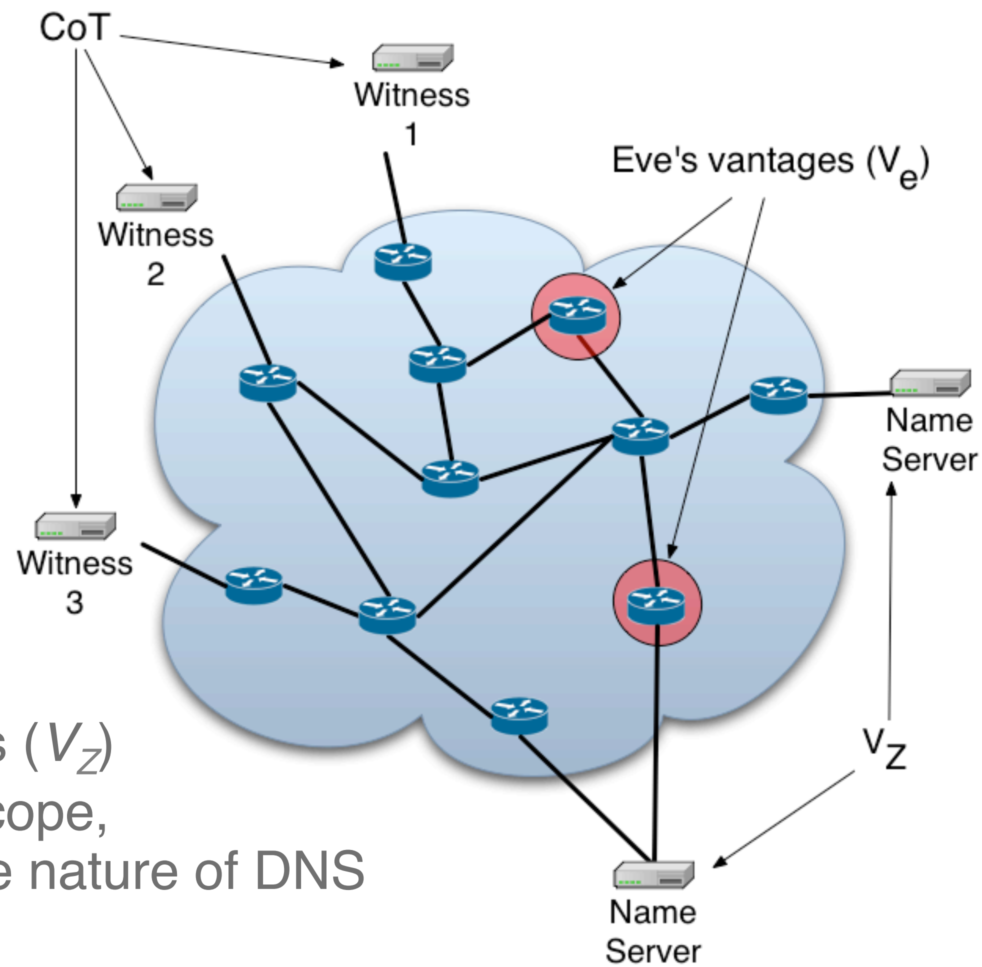
Peer-to-Peer CoTs

- P2P CoTs Compartmentalize
- CoTs are *manual*
  - Trust must be bootstrapped
- Observed data is signed by PGP key



Threat Model for Key Learning: Man in the Middle

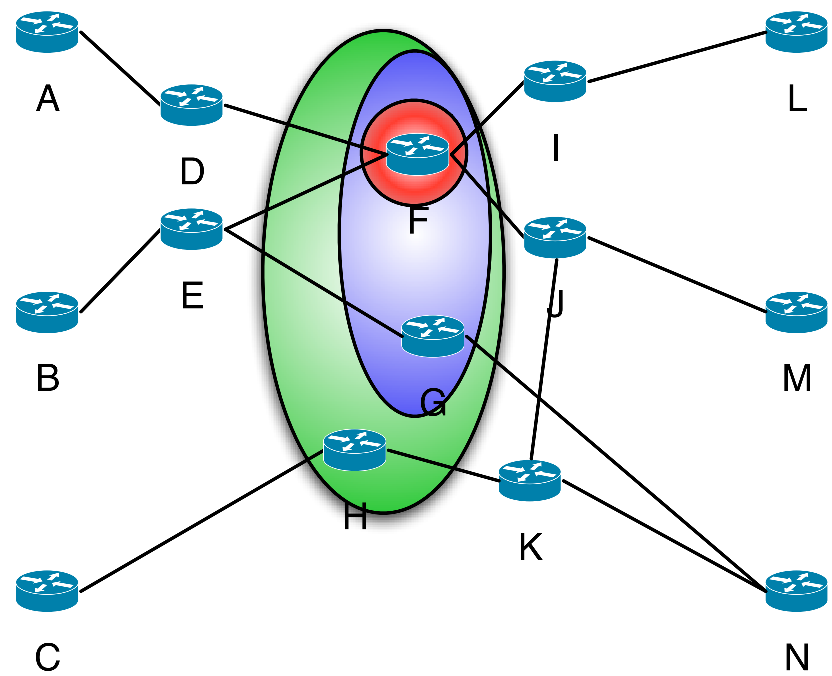
- To attack, Eve must see keys that are in transit
  - If she must own a vantage v_e in σ
- But, she can't arbitrarily attack just anyone
  - Attacks between a resolver v_i and a zone's name servers (V_Z)
  - Not a reduction of scope, this is dictated by the nature of DNS



- Eve must expend a *real* cost to own these vantages

What Eve *Really* Needs to Do

- To spoof, Eve must be in the right place at the right time
  - She must be able to intercept responses from all (or most) name servers
- The minimum set size for V_e to cut Alice off from the zone will be the *min-cut set* $V_{cut} = MinCut(v_i, V_Z)$
  - This is the lower bound on Eve's acquisition cost



Security Analysis: Attack Cost

- Eve must own vantage points (V_e) and be able to use them:
  Acquisition + usage costs

- Acquisition $c_a(V_e)$: can specific nodes even be purchased?
  - Core routers at AT&T may not be on sale like grandma's PC is
  - Eve may have to get her hands dirty (if she's able to)

- Usage $c_u(V_e, t)$: nodes in V_e may cost per hour, or may get reclaimed if detected
  - If renting nodes, then snooping is a function of rent
  - If Eve acquires her own nodes, operators may notice her

$$C(V_e, t) = c_a(V_e) + c_u(V_e, t)$$

Acquisition Cost: The Cat and Mouse Game

- Alice's best defense is to make her CoT as large and topologically diverse as possible
- Eve needs to know Alice's CoT (and all paths to V_Z's name servers)
  - Note: knowing *any* AS path is an open challenge [1]

- We evaluate three example types of adversaries
  1. General: does not know any path info
  2. Targeted: knows Alice's path to V_Z, but not her CoT's
  3. Nation State: will try to compromise the largest ISPs first

[1] Mao, Z. M., Qiu, L., Wang, J., and Zhang, Y. 2005. On AS-level path inference. *2005 ACM SIGMETRICS*

Eve’s Probability of Success

- General: the probability that Eve can subvert Alice’s min-cut set is (where *n* is the size of V_e):

$$Probability_s(V_e) = \binom{|V|}{n}^{-1} \times \binom{|V - V_{cut}|}{n - |V_{cut}|}$$

- Targeted: as Alice augments her min-cut set, the probability of compromise approaches the General case
- Nation State: the adversary is not focused on Alice’s CoT, but Alice’s chances are still augmented as she increases her min-cut set

Evaluation

- Simulated an AS-level topology using the Inet topology generator
  - Simulate 22,000 ASes
- Chose random ASes as V_Z nodes, and V_{CoT} nodes
  - Calculated min-cut set for V_Z and V_{CoT} combinations ranging from 2-11
- Used shortest path routing metric to represent routing
- Also deployed actual Vantages CoT
  - Vantages written in C++ with SQLite backed DB, uses GPG to verify witness communications
  - http://www.vantage-points.org/
- Constantly / automatically learns zones and polls
  - Aligns costs with benefits: verification aligns with needs

Actual Measured Min Cut-Set Sizes

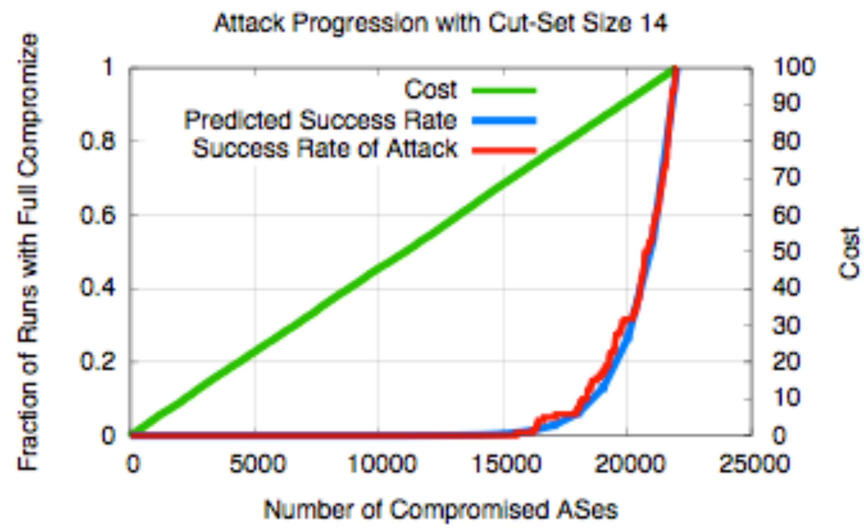
- Using a Vantages daemon peered with SecSpider, we get the following *actual* min cut-set sizes for major DNS zones
  - SecSpider's distributed key learning system, online since 2006

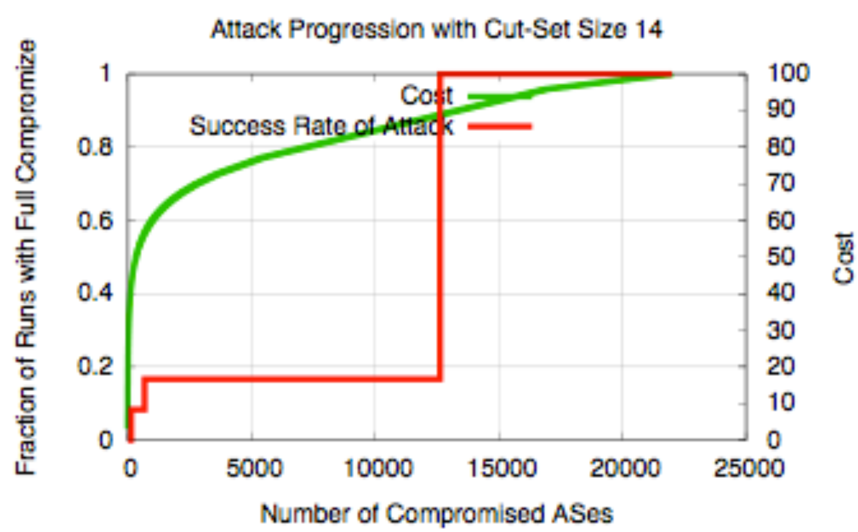
Actual Zone	Min Cut-Set Size
. (root)	27
.gov	18
.br	18
.bg	13
.org	11
…	…

- These are on par with, or better than, our simulated results

Simulated General Adversary

- Ran 10X10 simulations
  - CoTs = [1-10]
  - V_Z = [2-11]
- General Adversary
  - 90% ASes = 10%
- Nation State
  - 89% ASes = 20%





Conclusions

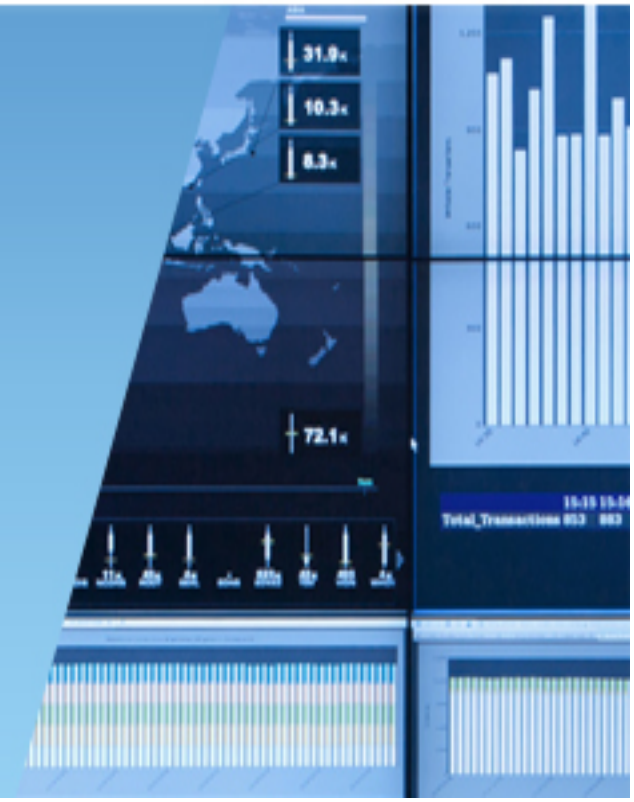
- With *Public Data*, we seek to add an orthogonal substrate to our systems: feasibility tested with *Vantages*
  - Large TLD failures did not black out Vantages' view of the tree
  - When the root's DURZ unblinded, Vantages automatically bootstrapped and learned it
- Fixing these problems in DNSSEC allows systems built *on* DNSSEC to inherit robustness!
  - DNSSEC must be robust to misconfigs and outages
  - People are adding services on DNS (DANE and more)
- Our Vantages deployment suggests its assurances are on par (or even *better* than) our simulated results

Check out our technical report:

http://techreports.verisignlabs.com/tr-lookup.cgi?trid=1110001&rev=1

VERISIGN™

Backup



General Lessons from Deployment Problems

- *"Distributing the authority for a [crypto-enhanced system] does not distribute the corresponding amount of expertise"*

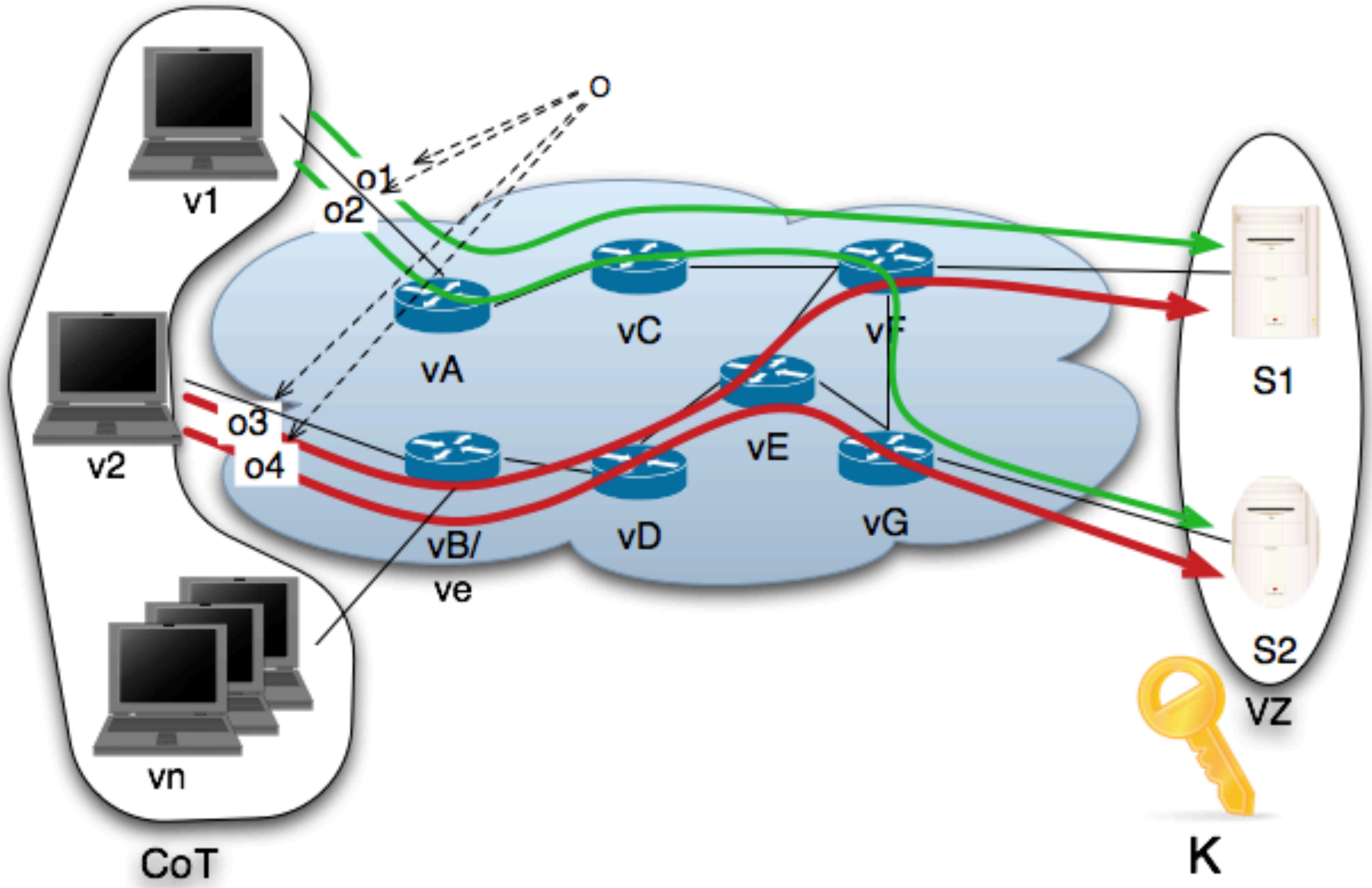
  -- Paul Mockapetris

- Simple designs do not always equate to simple operations
- Cryptography adds a lot of operational complexity
- Failing to consider operational realities can result in serious outages

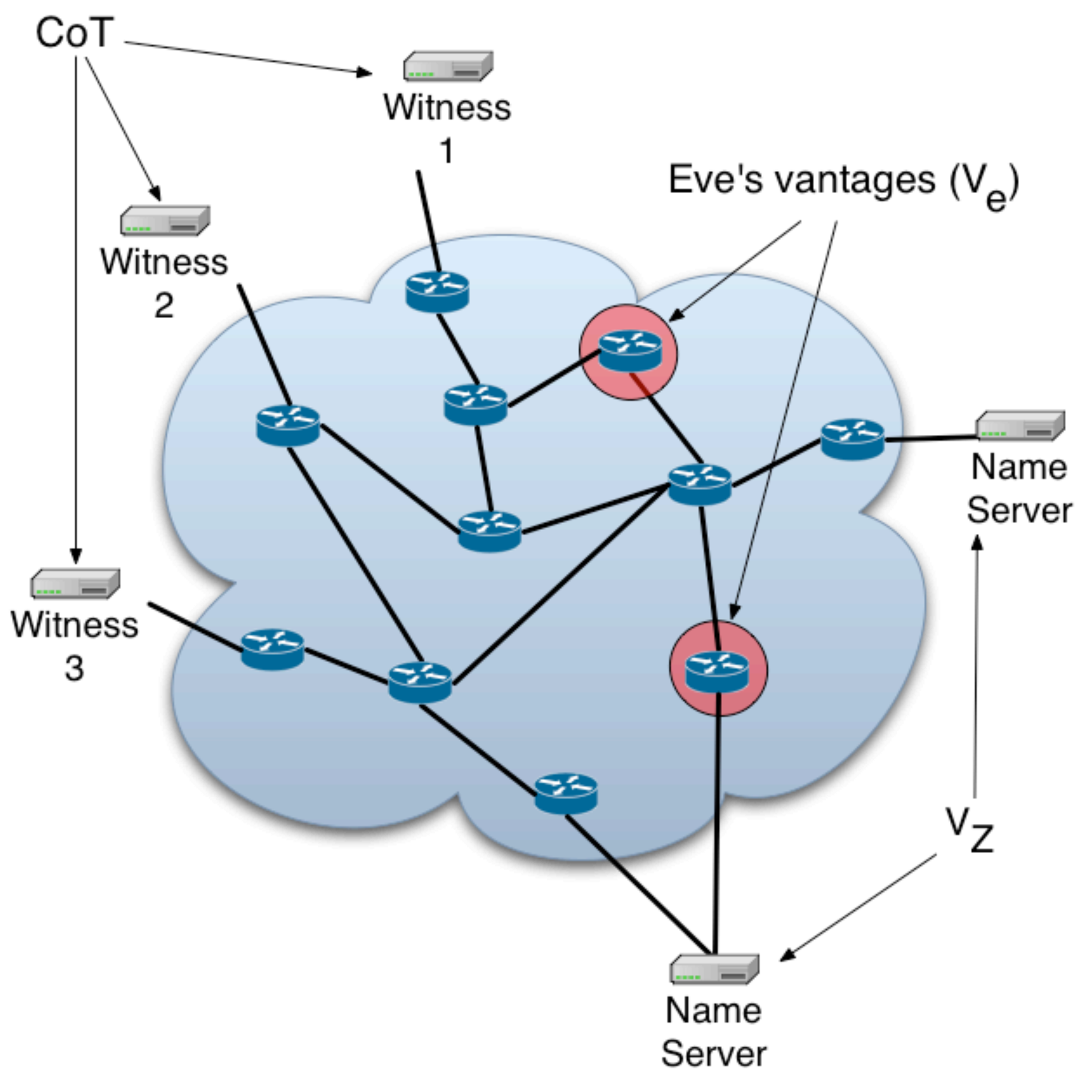
Public Data: Key Learning and Verification

- Motivated by measurements of the hierarchal model
- Goal: get proper keys for zones to resolvers
  - Avoid being spoofed *without* the hierarchy
  - Use redundancy for protection!
- Verification is now a *measurable* property of *publically* available data
  - The more *independent* measurements, the more secure
- Community of Trust (CoT): Trust is subjective
  - Cross-check what you see with what your friends saw
  - This is *not* the Web of Trust: observations, not attestations

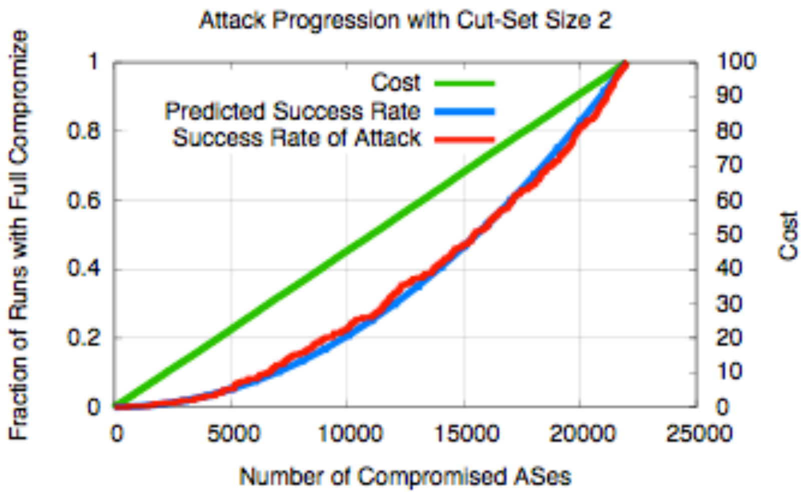
Public Data Model (again)

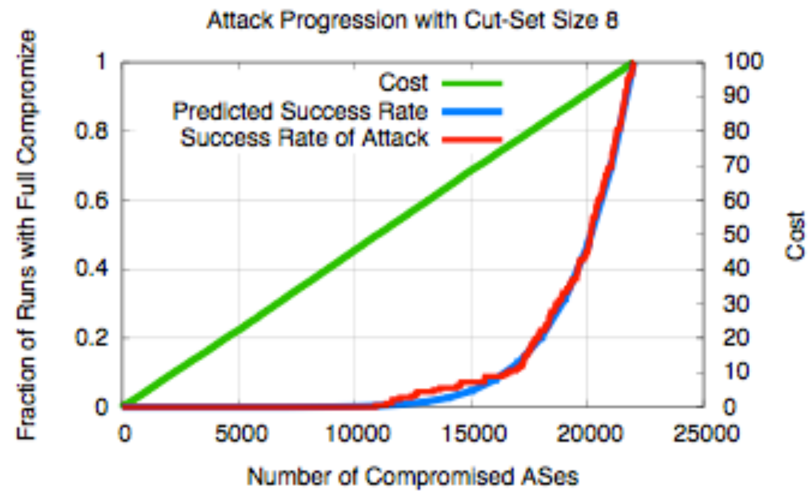


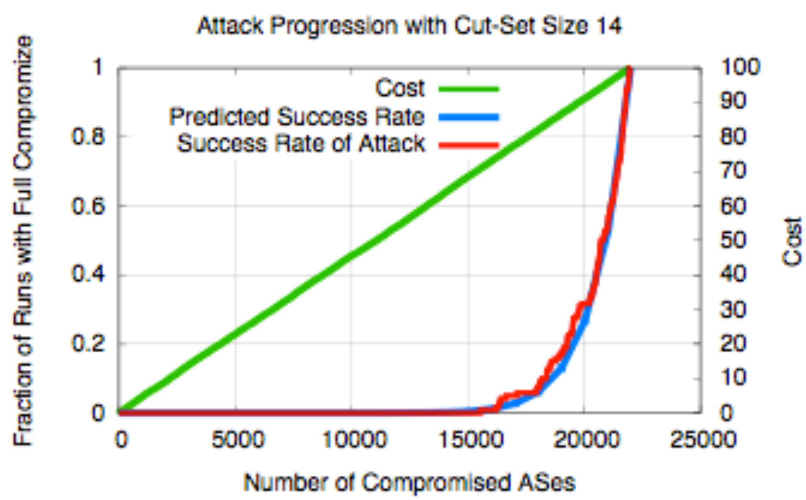
Public Data Model (again)



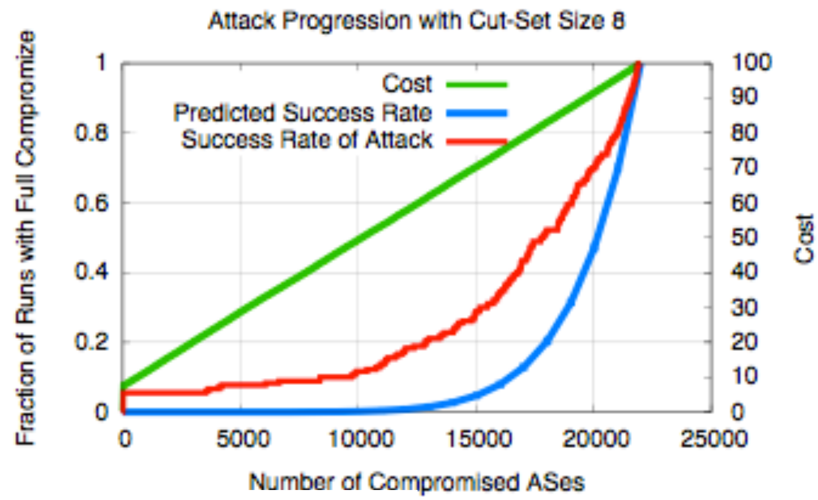
General Adversary

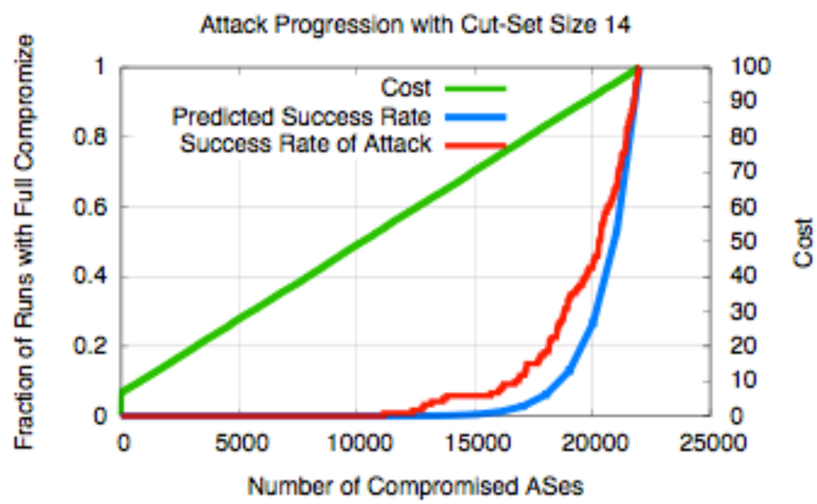




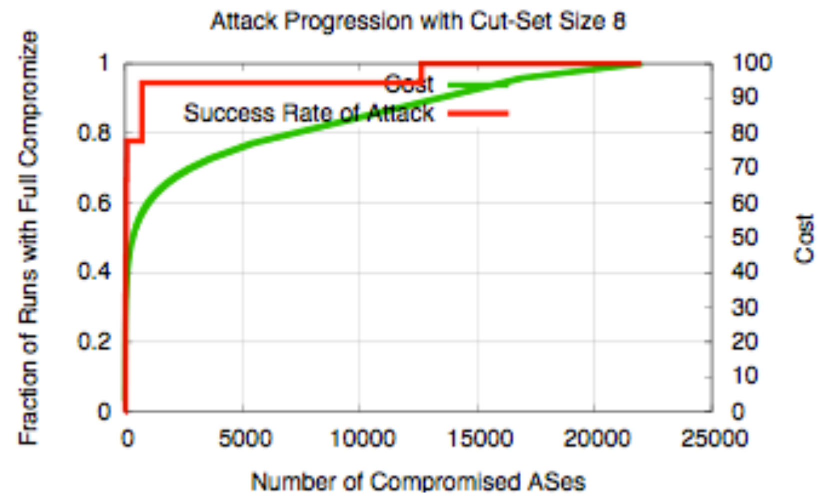


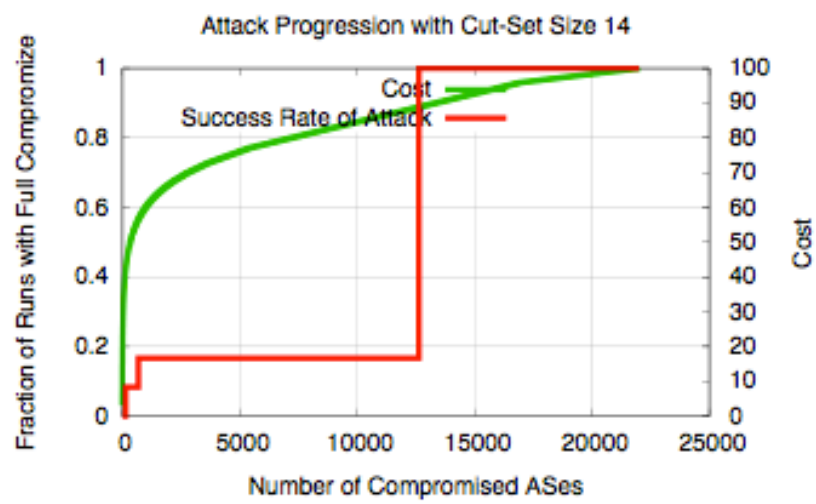
Targeted Adversary





Nation State Adversary





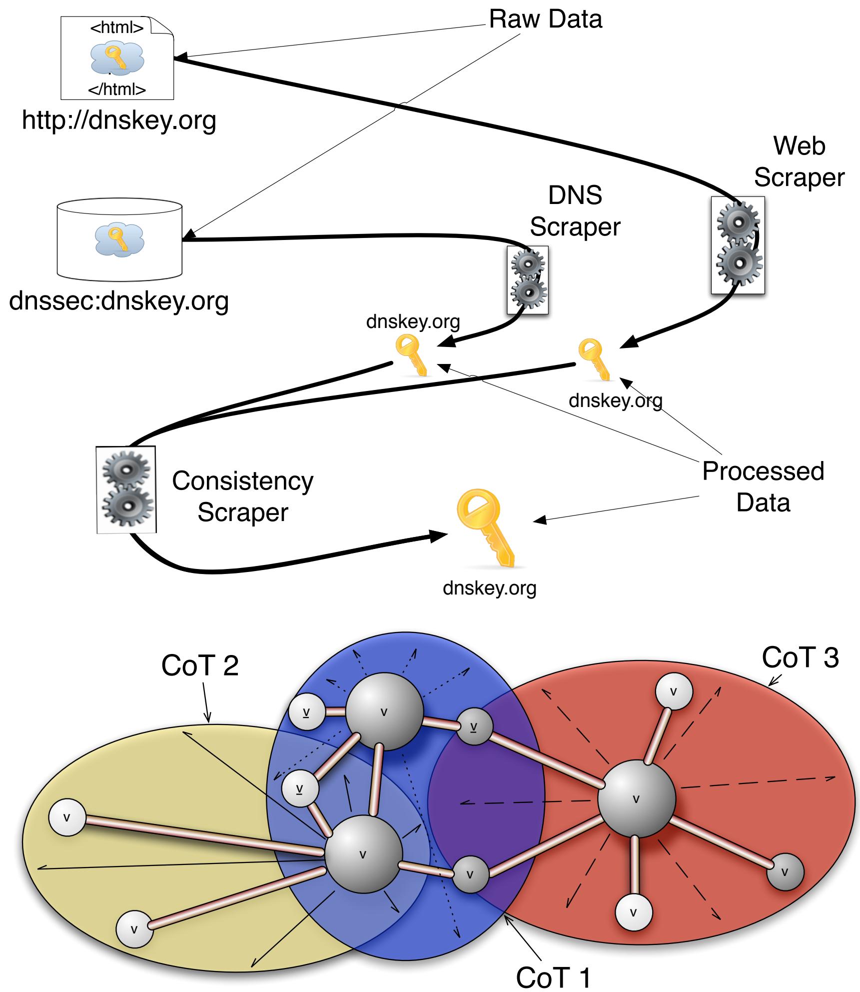
Vantages Implementation

- Written in C++ with SQLite backed DB, uses GPG to verify witness communications
  - Installs and can start running right away
  - http://www.vantage-points.org/
- Can be administered via web admin interface

- Automatically learns zones and polls every day

Peer-to-Peer CoTs

- Vantage daemons learn DNSKEYs from DNS or web pages
- Cross-check within CoT
- P2P CoTs Compartmentalize
- CoTs are *manual*
  - Trust must be bootstrapped
- Observed data is signed by GPG key



Vantages: A Public Data System

- Real system implementing Public Data needs some practical re-mappings
  - Some nodes may offer a *set* of observations (such as SecSpider), cull data from different protocols, etc.

- Everyone runs their own Vantage daemon
  - Peer-to-peer, choose your own CoT
  - Avoids the “who’s going to run it?” question