

# DNS Resolver Observatory

<https://dnssecviews.net>

**Pouyan Fotouhi Tehrani\***, Eric Osterweil,  
Thomas C. Schmidt, Matthias Wählisch

\* [pft@acm.org](mailto:pft@acm.org)

# Motivation

- DNSSEC-enabled zones regularly change their keys
- Instantaneous changes @ authoritative name servers

**BUT**

- Users rely on recursive resolvers
- Recursive resolvers follow different policies
- Timing, caching, multiple signers, etc. influence propagation
- Infrastructure providers are interested to know how their services are observed by users (before, during, and after transitions)

# Motivation

- DNSSEC-enabled zones regularly change their keys
- Instantaneous changes @ authoritative name servers



We have been monitoring this in the past 15 years through SecSpider\* (see <https://secspider.net/>)

## BUT

- Users rely on recursive resolvers
- Recursive resolvers follow different policies
- Timing, caching, multiple signers, etc. influence propagation
- Infrastructure providers are interested to know how their services are observed by users (before, during, and after transitions)

\* Osterweil et al. "From the Beginning: Key Transitions in the First 15 Years of DNSSEC," *arXiv preprint* [arXiv:2109.08783](https://arxiv.org/abs/2109.08783) (2021).

# Motivation

- DNSSEC-enabled zones regularly change their keys
- Instantaneous changes @ authoritative name servers

} We have been monitoring this in the past 15 years through SecSpider\* (see <https://secspider.net/>)

**BUT**

- Users rely on recursive resolvers
- Recursive resolvers follow different policies
- Timing, caching, multiple signers, etc. influence propagation
- Infrastructure providers are interested to know how their services are observed by users (before, during, and after transitions)

That's why we are building the **Resolver Observatory!**

\* Osterweil et al. "From the Beginning: Key Transitions in the First 15 Years of DNSSEC," *arXiv preprint [arXiv:2109.08783](https://arxiv.org/abs/2109.08783)* (2021).

# Motivation

- DNSSEC-enabled zones regularly change their keys
- Instantaneous changes @ authoritative name servers

We have been monitoring this in the past 15 years through SecSpider\* (see <https://secspider.net/>)

**BUT**

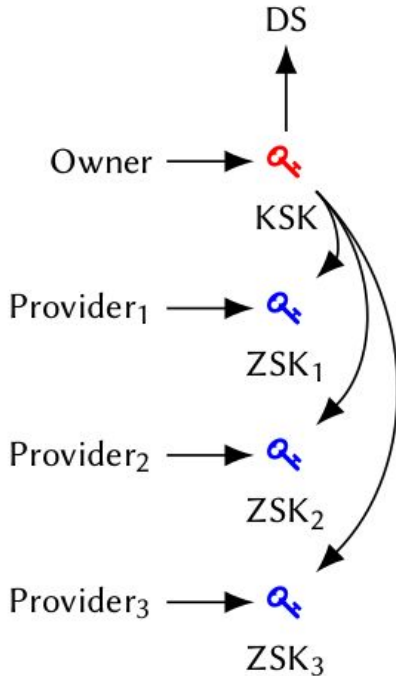
- Users rely on recursive resolvers
- Recursive resolvers follow different policies ← We already observe that the same host with multiple interfaces observe different responses for the same query!
- Timing, caching, multiple signers, etc. influence propagation
- Infrastructure providers are interested to know how their services are observed by users (before, during, and after transitions)

That's why we are building the **Resolver Observatory!**

\* Osterweil et al. "From the Beginning: Key Transitions in the First 15 Years of DNSSEC," *arXiv preprint [arXiv:2109.08783](https://arxiv.org/abs/2109.08783)* (2021).

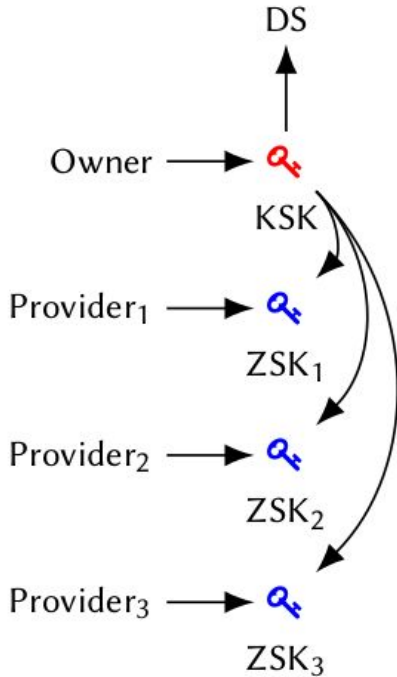
# Use Case: Multi-Signer DNSSEC

Common KSK Set, Unique ZSK Set per Provider

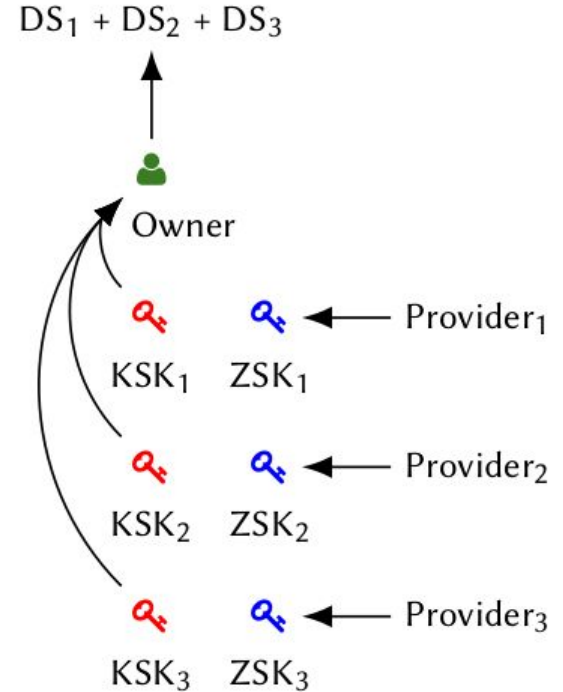


# Use Case: Multi-Signer DNSSEC

## Common KSK Set, Unique ZSK Set per Provider

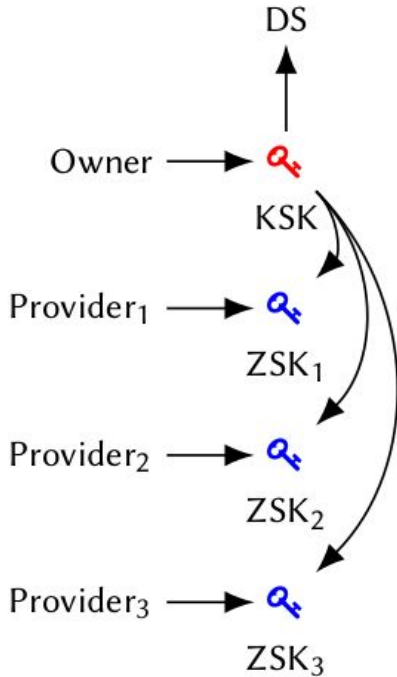


## Unique KSK Set and ZSK Set per Provider



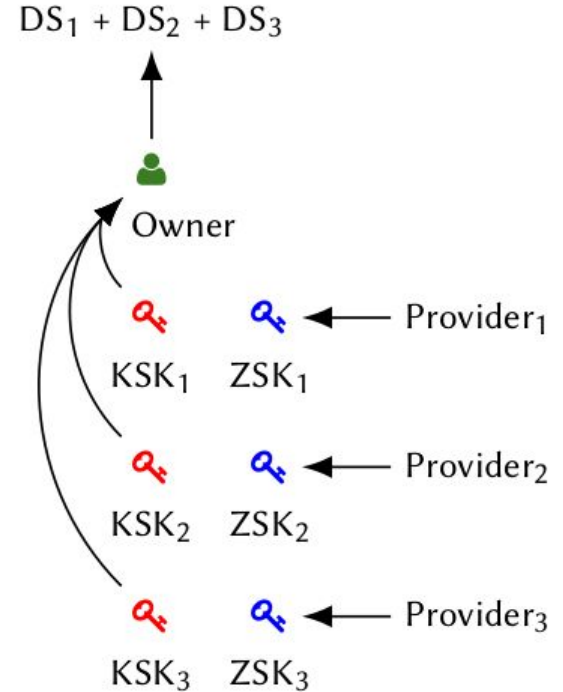
# Use Case: Multi-Signer DNSSEC

## Common KSK Set, Unique ZSK Set per Provider



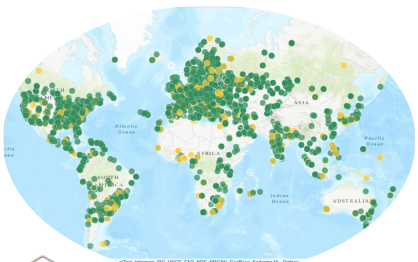
Complex key transition  
process models require  
coordination between  
providers and parent zone!

## Unique KSK Set and ZSK Set per Provider





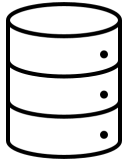
# System Overview



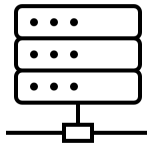
**RIPE Atlas**

Schedule Measurements

Database

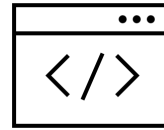


Persist



Back End

RESTful API



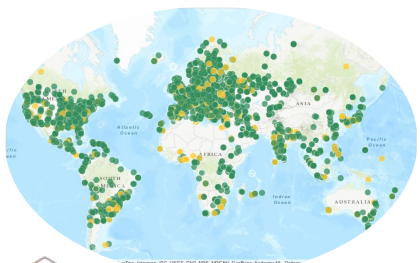
Front End

Register Zone



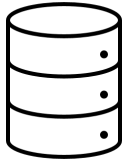
Infrastructure Operator

# System Overview

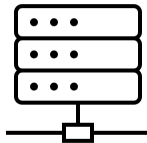


Schedule Measurements  
Listen for results

Database

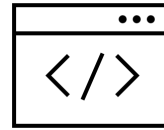


Persist and  
query



Back End

RESTful API



Front End


Register Zone  
Observe Stats



Infrastructure Operator

# Approach: Collect Data

1. Find zone apex
2. Schedule regular measurements via RIPE Atlas for following records:
  - DNSKEY
  - DS
  - NS
  - SOA
3. Parse and serialize data into the DB iff:
  - Response is valid
  - Response is signed



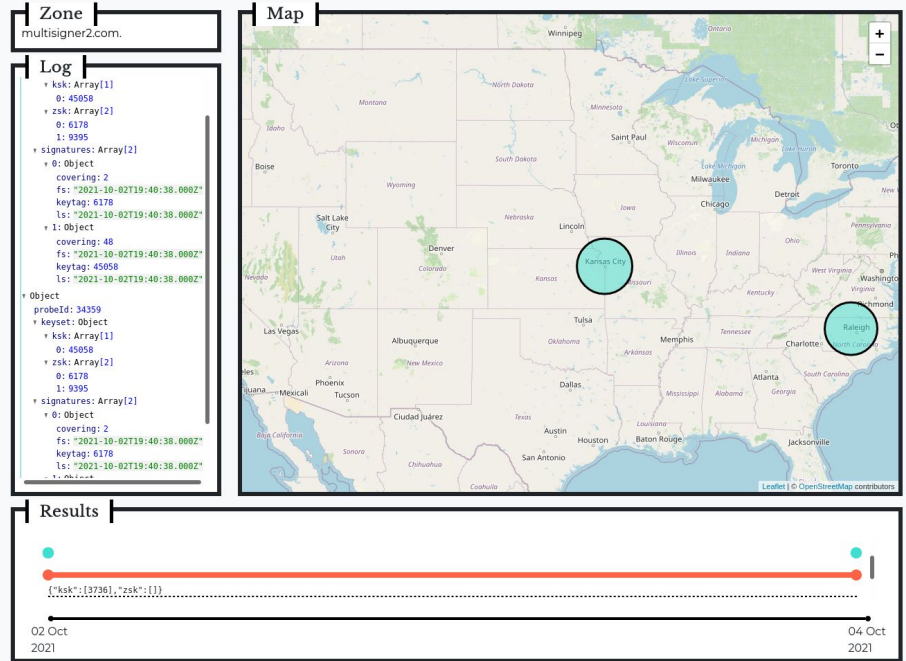
Executed by a set of random probes (currently only US)

Also record when each probes sees which RRSet and RRSIG

# Approach: Provide Analysis

For any given zone:

1. Calculate different combinations of *observed* DNSKEY sets and active keys in use.
2. Color code each combination and calculate when each probe sees which combination.

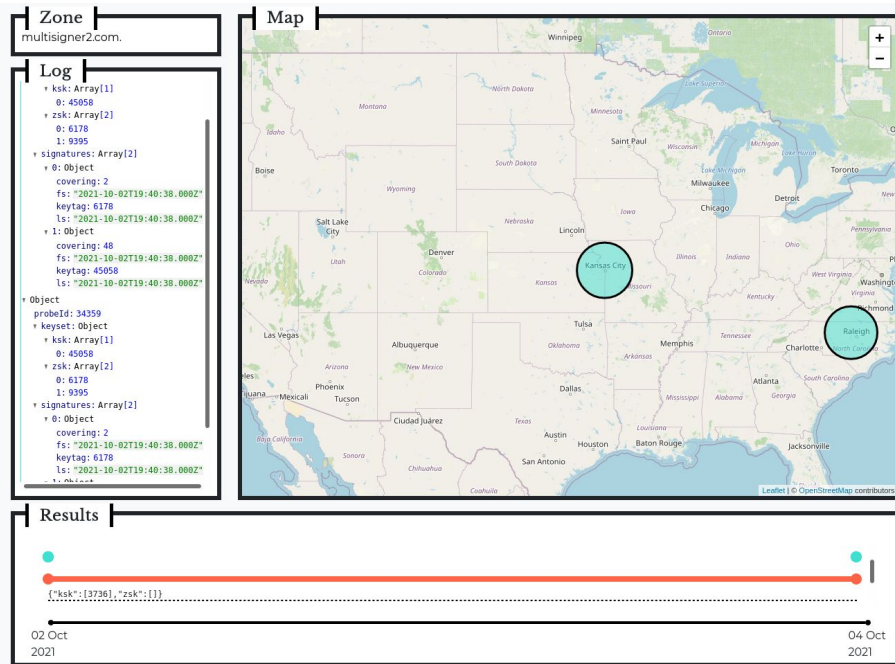


# Approach: Provide Analysis

For any given zone:

1. Calculate different combinations of *observed* DNSKEY sets and active keys in use.
2. Color code each combination and calculate when each probe sees which combination.

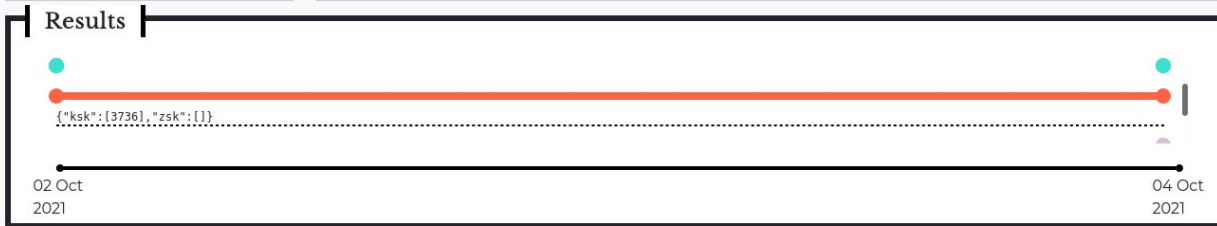
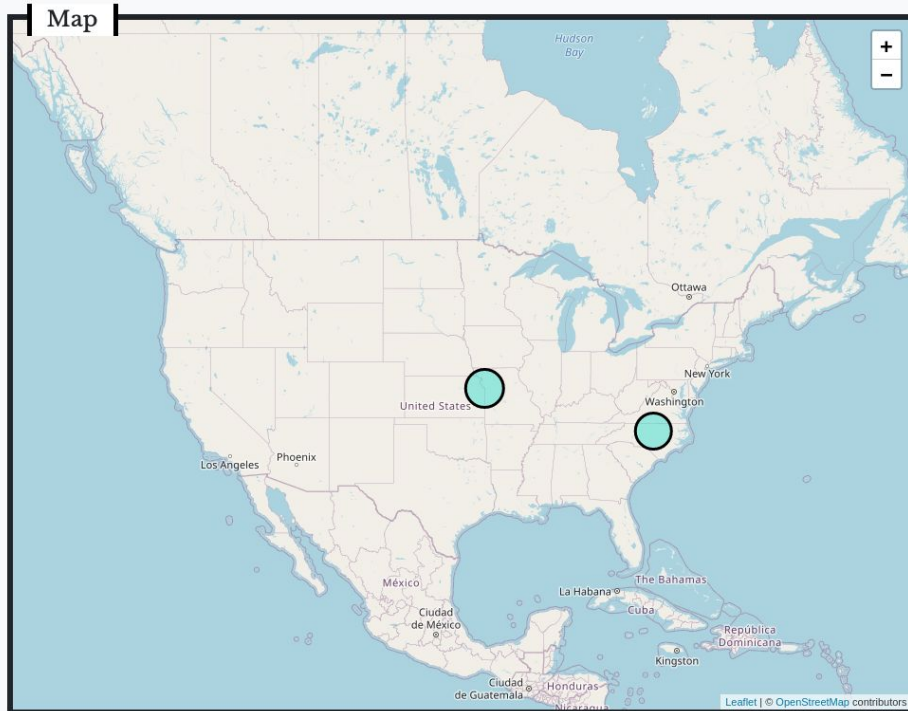
➔ Providers can see what recursive resolvers observe at any point in time and space\*



**Zone**  
multisigner2.com.

**Log**

```
Object
  probeId: 1001503
  keyset: Object
    ksk: Array[1]
      0: 45058
    zsk: Array[2]
      0: 6178
      1: 9395
  signatures: Array[2]
    0: Object
      covering: 2
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 6178
      ls: "2021-10-02T19:40:38.000Z"
    1: Object
      covering: 48
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 45058
      ls: "2021-10-02T19:40:38.000Z"
  Object
    probeId: 34359
    keyset: Object
      ksk: Array[1]
        0: 45058
      zsk: Array[2]
        0: 6178
        1: 9395
    signatures: Array[2]
      0: Object
        covering: 2
        fs: "2021-10-02T19:40:38.000Z"
        keytag: 6178
```

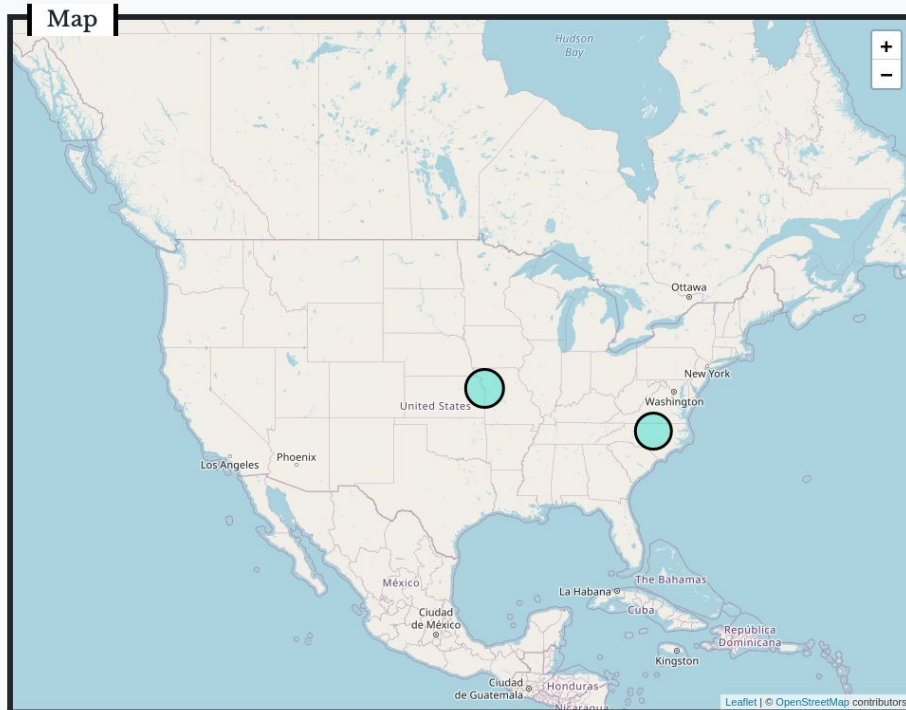


**Example:** multisigner2.com

Zone  
multisigner2.com.

Log

```
Object
  probeId: 1001503
  keyset: Object
    ksk: Array[1]
      0: 45058
    zsk: Array[2]
      0: 6178
      1: 9395
  signatures: Array[2]
    0: Object
      covering: 2
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 6178
      ls: "2021-10-02T19:40:38.000Z"
    1: Object
      covering: 48
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 45058
      ls: "2021-10-02T19:40:38.000Z"
  Object
    probeId: 34359
    keyset: Object
      ksk: Array[1]
        0: 45058
      zsk: Array[2]
        0: 6178
        1: 9395
    signatures: Array[2]
      0: Object
        covering: 2
        fs: "2021-10-02T19:40:38.000Z"
        keytag: 6178
```



- Key set:  
`{"ksk" : [45058],  
"zsk" : [6178, 9395]}`
- Active Keys:
  - ZSK: 6178
  - KSK: 45058



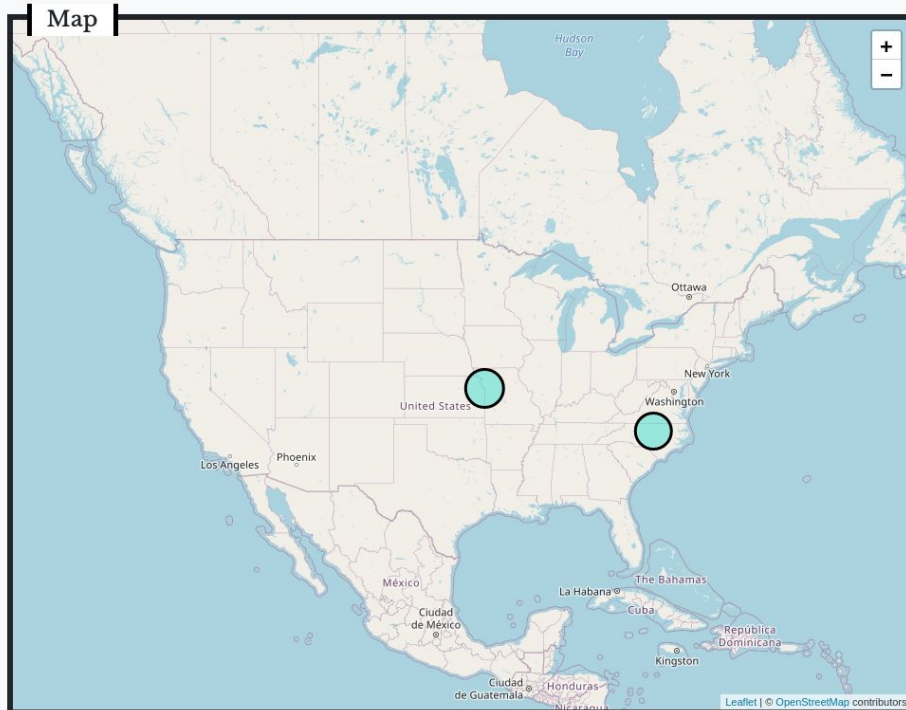
Example: multisigner2.com



Zone  
multisigner2.com.

Log

```
Object
  probeId: 1001503
  keyset: Object
    ksk: Array[1]
      0: 45058
    zsk: Array[2]
      0: 6178
      1: 9395
  signatures: Array[2]
    0: Object
      covering: 2
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 6178
      ls: "2021-10-02T19:40:38.000Z"
    1: Object
      covering: 48
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 45058
      ls: "2021-10-02T19:40:38.000Z"
  Object
    probeId: 34359
    keyset: Object
      ksk: Array[1]
        0: 45058
      zsk: Array[2]
        0: 6178
        1: 9395
    signatures: Array[2]
      0: Object
        covering: 2
        fs: "2021-10-02T19:40:38.000Z"
        keytag: 6178
```



Results

Observation point

```
{ "ksk": [3736], "zsk": [] }
```

02 Oct 2021

04 Oct 2021

- Key set:  
{ "ksk" : [45058],  
"zsk" : [6178, 9395] }
- Active Keys:
  - ZSK: 6178
  - KSK: 45058
- Key set:  
{ "ksk" : [45058],  
"zsk" : [6178, 9395] }
- Active Keys:
  - ZSK: [6178, 9395]
  - KSK: [3736, 45058]

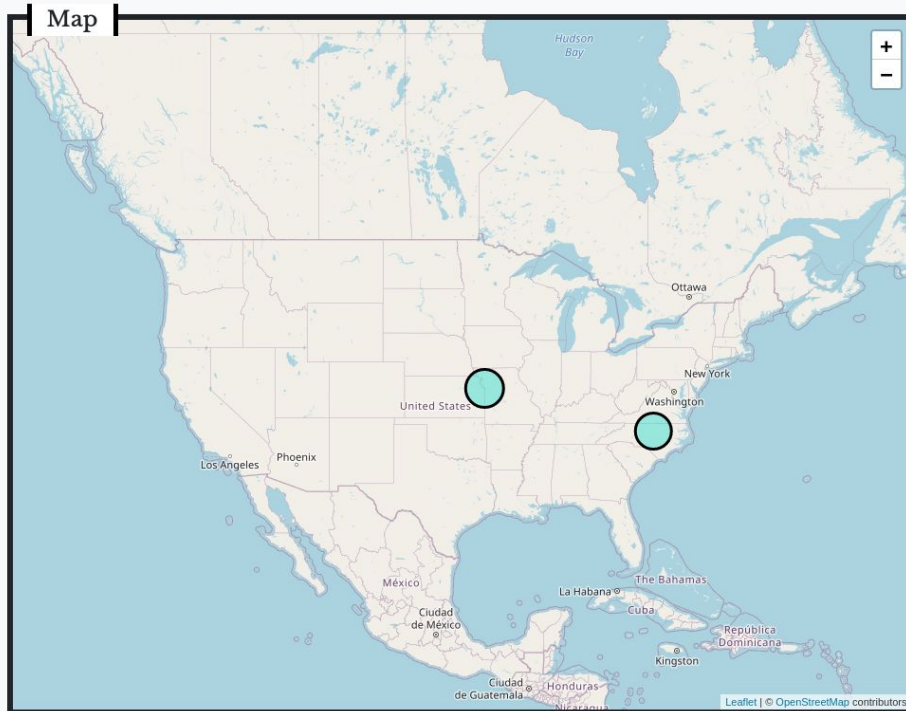
Example: multisigner2.com



Zone  
multisigner2.com.

Log

```
Object
  probeId: 1001503
  keyset: Object
    ksk: Array[1]
      0: 45058
    zsk: Array[2]
      0: 6178
      1: 9395
  signatures: Array[2]
    0: Object
      covering: 2
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 6178
      ls: "2021-10-02T19:40:38.000Z"
    1: Object
      covering: 48
      fs: "2021-10-02T19:40:38.000Z"
      keytag: 45058
      ls: "2021-10-02T19:40:38.000Z"
  Object
    probeId: 34359
    keyset: Object
      ksk: Array[1]
        0: 45058
      zsk: Array[2]
        0: 6178
        1: 9395
    signatures: Array[2]
      0: Object
        covering: 2
        fs: "2021-10-02T19:40:38.000Z"
        keytag: 6178
```



Results

Observation point

```
["ksk": [3736], "zsk": []]
```

02 Oct 2021 04 Oct 2021

- Key set:  
`{"ksk": [45058], "zsk": [6178, 9395]}`

- Active Keys:
  - ZSK: 6178
  - KSK: 45058

- Key set:  
`{"ksk": [45058], "zsk": [6178, 9395]}`

- Active Keys:
  - ZSK: [6178, 9395]
  - KSK: [3736, 45058]

Note: looks like a double signature transition, but it's not!

Example: multisigner2.com

# Method: Detecting Multi-Signer DNSSEC Heuristically

## Common KSK Set, Unique ZSK Set per Provider

- Same KSK set is in active use
- Different ZSK set is in active use

# Method: Detecting Multi-Signer DNSSEC Heuristically

## Common KSK Set, Unique ZSK Set per Provider

- Same KSK set is in active use
- Different ZSK set is in active use

Probe A observes at time  $t$ :

- DNSKEY: {"ksk" : [45058], "zsk" : [6178, 9395]} signed by 45058
- NS signed by **6178**

Probe B observes at time  $t$ :

- DNSKEY: {"ksk" : [45058], "zsk" : [6178, 9395]} signed by 45058
- NS signed by **9395**

# Method: Detecting Multi-Signer DNSSEC Heuristically

## Common KSK Set, Unique ZSK Set per Provider

- Same KSK set is in active use
- Different ZSK set is in active use

## Unique KSK Set and ZSK Set per Provider

- Different KSK and ZSK sets are in active use
- Same set of KSK and ZSK is observed together

# Method: Detecting Multi-Signer DNSSEC Heuristically

## Common KSK Set, Unique ZSK Set per Provider

- Same KSK set is in active use
- Different ZSK set is in active use

## Unique KSK Set and ZSK Set per Provider

- Different KSK and ZSK sets is in active use
- Same set of KSK and ZSK is observed together

Probe A observes at time  $t$ :

- DNSKEY: {"ksk": [45058, 2143], "zsk": [6178, 9395]} signed by **45058**
- NS signed by **6178**

Probe B observes at time  $t$ :

- DNSKEY: {"ksk": [45058, 2143], "zsk": [6178, 9395]} signed by **2143**
- NS signed by **9395**

# Method: Detecting Multi-Signer DNSSEC Heuristically

## Common KSK Set, Unique ZSK Set per Provider

- Same KSK set is in active use
- Different ZSK set is in active use

## Unique KSK Set and ZSK Set per Provider

- Different KSK and ZSK sets is in active use
- Same set of KSK and ZSK is observed together

### Caveats:

- What about standby keys? Do we care at all?
- What about ongoing transitions? Can they cause false-positives?
- What about anycast resolvers? Can they skew results?

Probe A observes at time  $t$ :

- DNSKEY: {"ksk" : [45058, 2143], "zsk" : [6178, 9395]} signed by **45058**
- NS signed by **6178**

Probe B observes at time  $t$ :

- DNSKEY: {"ksk" : [45058, 2143], "zsk" : [6178, 9395]} signed by **2143**
- NS signed by **9395**

# Conclusion

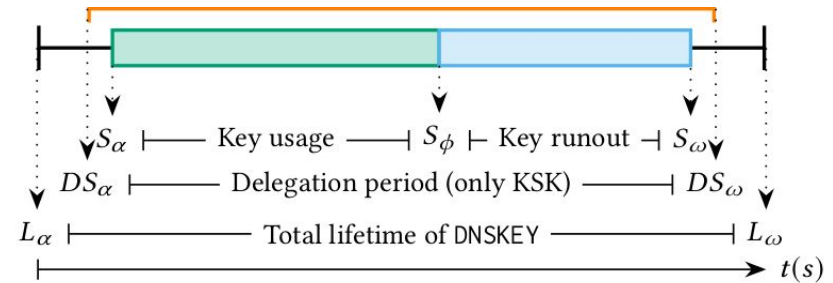
- There is a measurable discrepancy between records at authoritative name servers and what recursive resolvers deliver
- Resolver Observatory gives operators the opportunity to follow their DNSSEC deployment from the perspective of clients in real time
- Aggregated data can be used to improve deployment practices and figure out acceptance criteria

# Backup Slides



# Modelling DNSKEY lifetime

- **Total lifetime:** accumulative periods of time where key was included in a DNSKEY RR set and was covered by a signature
- **Key usage:** duration when key was in active use to generate new signatures
- **Key runout:** period when key was used only to verify existing signatures
- **Delegation period:** total amount of time when a secure delegation exists for the key



DNSKEY lifetime model (Osterweil et al., 2021)

# Challenges and Open Questions

- What is the proper unit of observation? A probe? An interface?
- What is robust heuristic to detect different key transitions with different process models in real-time?
- How can public recursive resolvers profit from this system?