Multi-Signer DNSSEC Models

Shumon Huque & Jan Včelák

May 12th 2019

DNS-OARC 30 Workshop; Bangkok, Thailand

Multi-Signer DNSSEC Models (abstract for reference)

Many enterprises today employ the service of multiple DNS providers to operate their authoritative DNS service. Two providers are fairly typical and this allows the DNS service to survive a complete failure of any single provider. Deploying DNSSEC in such an environment can have some challenges depending on the configuration and feature set in use. In particular, large enterprises often make use of a number of non-standardized DNS features, that necessitates having each provider independently sign the DNS zone data with a coordinated set of keys. We will present several operationally viable deployment models for multi signer DNSSEC. One of the goals of this talk is to generate interest in these models and encourage managed DNS providers to support them (encouragingly, several are already planning to do so), as this will solve an important deployment hurdle for enterprise DNSSEC. Additionally, it may be possible to leverage the multi-signer models to allow non-disruptive handoff of DNSSEC signed zones from one DNS operator to another. We now have an early implementation of some of the key management mechanisms needed to deploy the multi-signer models, and will share details of the implementation.

Background

- March 8th 2018: ”DNSSEC for a Large Enterprise” – OARC 28 Workshop, San Juan, Puerto Rico
- March 22nd 2018: “Multi-Provider DNSSEC Models” initial draft – IETF 101, London, U.K.
- Subsequent adoption by IETF dnsop working group, continuing work, testing of prototypes, discussions with providers, early implementations.

Traditional Multi-provider model

- Zone owner runs master server that maintains & signs zone data.
- Pushes out zone to multiple providers via DNS zone transfer.
- Providers serve the zone to the world.
- Zone owner holds signing keys – so managed DNS providers cannot serve false data, without detection by validating resolvers.

- Well understood model and is deployed successfully in the field.

Traditional Multi-provider model

- Zone owner runs master server that maintains & signs zone data.
- Pushes out zone to multiple providers via DNS zone transfer.
- Providers serve the zone to the world.
- Zone owner holds signing keys – so managed DNS providers cannot serve false data, without detection by validating resolvers.

- Well understood model and is deployed successfully in the field.

- **Notable limitation: doesn’t work with non-standardized DNS features that are quite widely used in the DNS industry today.**

Non-standard response mechanisms

- Often called “Traffic Management”
  - Global Server Load Balancing (GSLB), Probe & Failover records, custom programmed dynamic responses, etc.
- Often querier-specific or dependent on inspecting other state in the network
  - So answer and signature typically have to be determined at the authoritative server answering the query, at the time of the query, or both.

- Also known by other colorful names
  - P. Vixie “What the DNS is not”, ACMqueue, November 2009
  - DNS more brittle, harder to debug, layer violation, cost shifting, etc.
  - But …

Direction of DNS Evolution

- Long standing Tussle: DNS protocol purity vs. the reality of how extensively these mechanisms are already deployed.

- Paul Mockapetris, ICANN IDS 2018 keynote remark:
  - *"In the ultimate, the DNS should hold programs as well as data"*
  - Is that the next phase in the evolution of the DNS protocol?
  - Or are we already there?
  - https://www.icann.org/en/system/files/files/presentation-lessons-history-future-dns-13jul18-en.pdf

Multi-Signer Models

Multi-Signer DNSSEC models

- Each provider independently signs and serves zone data
- Zone owner typically uses provider specific zone management APIs to update zone content at each provider

Multi-Signer DNSSEC models

- Can support the non-standard DNS features *if* the provider is capable of signing the response data generated by these features.
- Common strategies for doing so:
  - On-the-fly signing (Online signing)
  - Pre-compute & sign all possible response RRsets, and then algorithmically determine at query time, which response + signature to return.

- Note: these may be attractive to organizations, even if they aren’t using non-standard features -- since they can forgo the task of running their own backend hidden master server infrastructure.

Key Management Requirements

- Main requirement: manage the contents of the DNSKEY and DS RRsets such that validation is always possible, no matter which provider you query and obtain the response from.

- **Strategy: each provider has to import the zone signing (public) keys of the other providers into their DNSKEY RRset.**

Validating Resolver Behavior

- Why do we need to cross-import Zone Signing Keys?
- How does a validating resolver behave in the presence of one zone that is independently signed by more than one party (provider)?
- We’ll discuss the case of 2 providers, A & B. But our models are generalizable to any number of providers.

From COM servers

Referral (Secure Delegation) for ”example.com”

Contains Signed DS records for both Provider A and Provider B’s KSKs

From COM servers

Referral (Secure Delegation) for ”example.com”

Contains Signed DS records for both Provider A and Provider B’s KSKs

From Provider A

DNSKEY response for example.com: **{KSK_A, ZSK_A}**

Signed by KSK_A

Cached in Resolver

From COM servers

Referral (Secure Delegation) for ”example.com”
Contains Signed DS records for both Provider A and Provider B’s KSKs

From Provider A

DNSKEY response for example.com: **{KSK_A, ZSK_A}**
Signed by KSK_A
Cached in Resolver

From Provider B

Response for “www.example.com AAAA”
Signed by ZSK_B
Problem: Cannot authenticate! ZSK_B is not present in cached DNSKEY response!

Obvious solution: Provider A’s DNSKEY response needs to include ZSK_B (and vice versa).

Validating Resolver Behavior

- State
  - Cached: {example.com DNSKEY: KSK_a, ZSK_a} -> signed by KSK_a
  - Response from Provider B: {www.example.com IN A} – signed by ZSK_b
    - ZSK_b missing from DNSKEY RRset

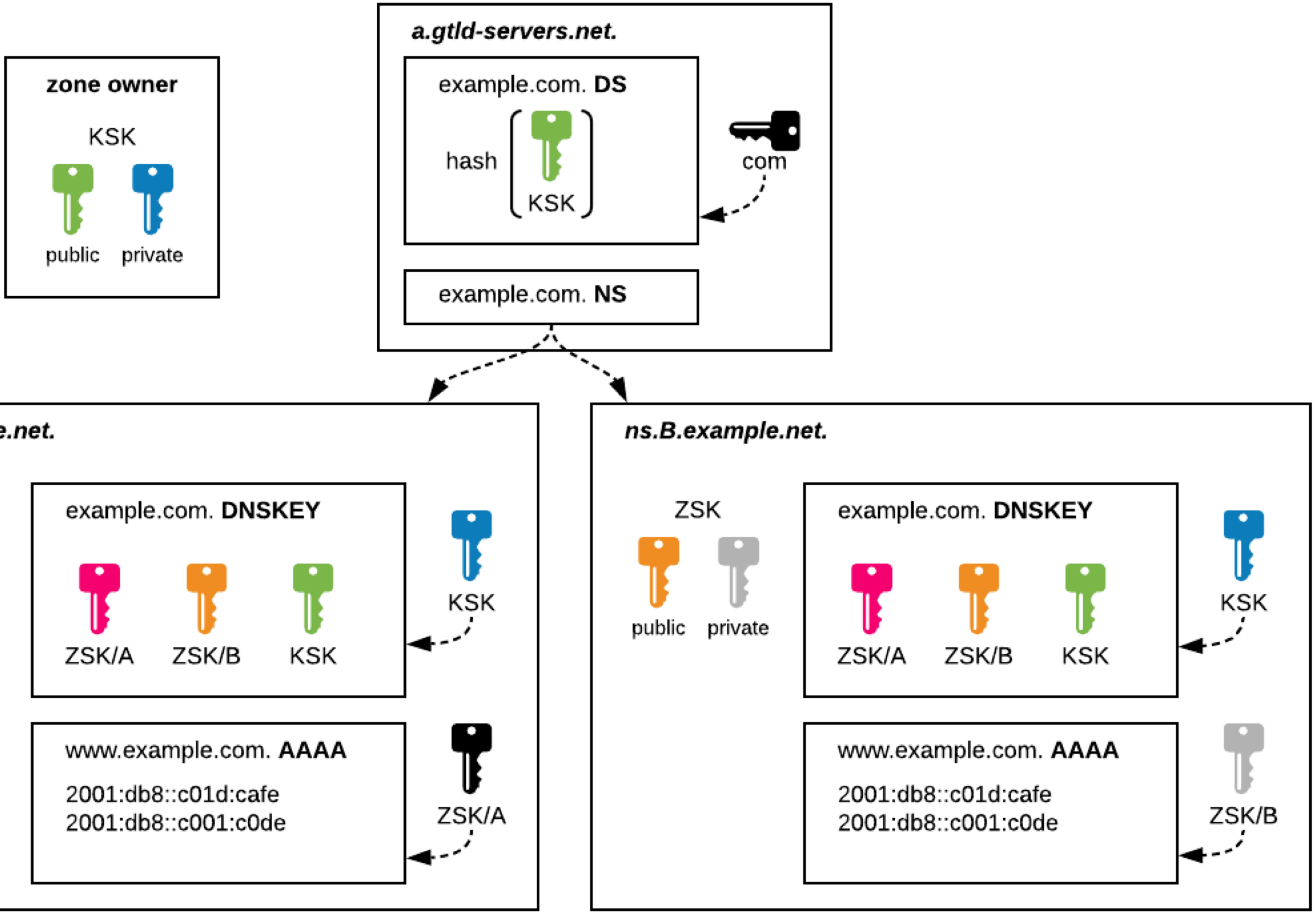
  - What is likely to happen: Resolver will likely treat the response from provider B as a possible attack, and keep retrying other servers for the zone, eventually hit provider A, and get a validatable response.
  - But if this takes too long, the resolver, or its downstream clients may have given up, or timed out.
  - We need to ensure that all servers for the zone always return validatable responses, and do so promptly.

Multi-Signer Model Details

Two Multi-Signer models

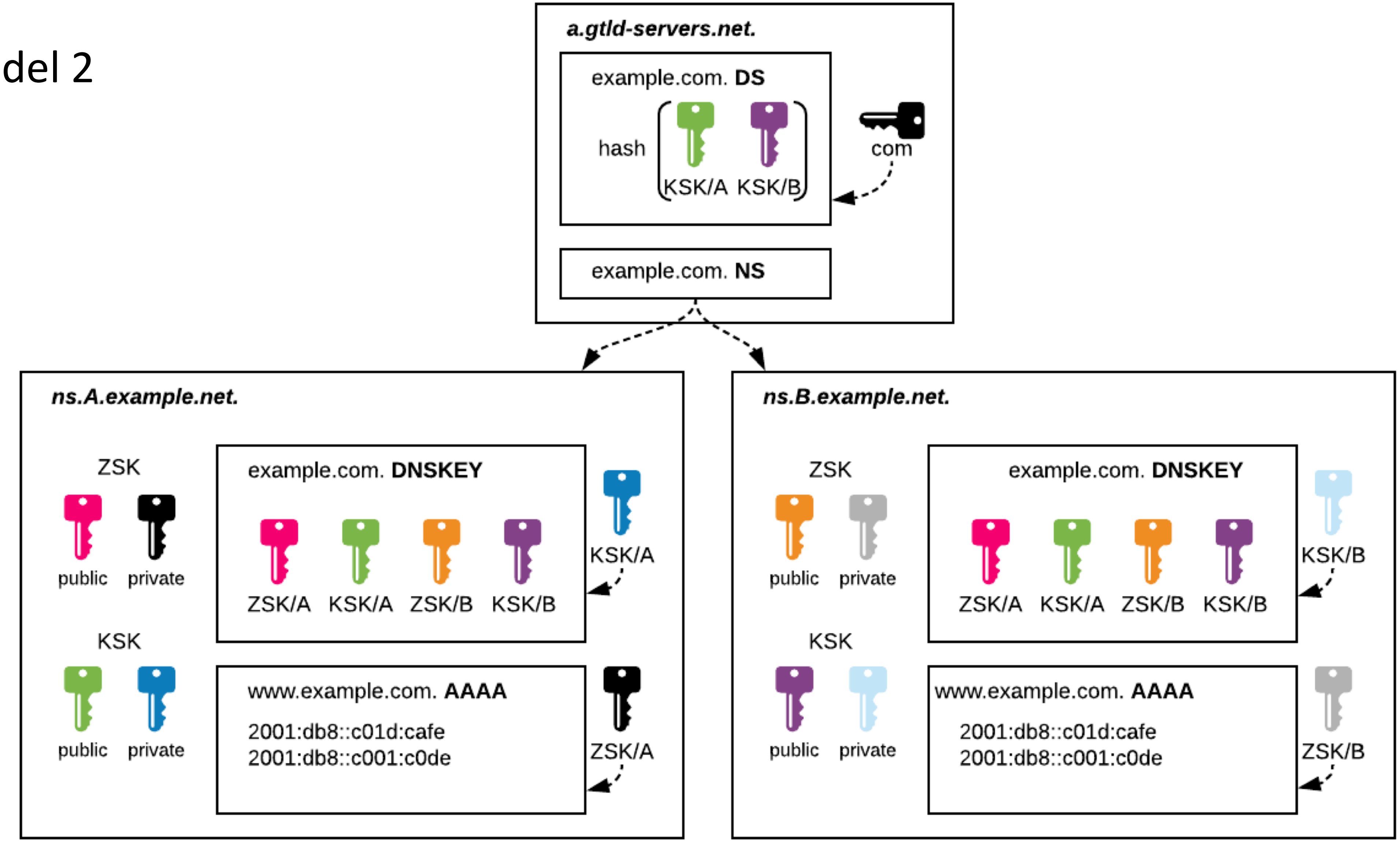
- **Model 1** — Common KSK, unique ZSK per provider:
    - Zone owner retrieves keys from providers, builds the DNSKEY record set, signs it, and distributes it to the providers.
    - Each provider serves pre-signed DNSKEY but signs other zone content independently.
- **Model 2** — Unique KSK and ZSK per provider:
    - Zone owner retrieves keys from providers, builds the DNSKEY record set, and distributes it to the providers. (No signing!)
    - Each provider signs the zone content independently.
- Zone owner always maintains DS records.

Model 1



zone owner
KSK
public
private
a.gtld-servers.net.
example.com. DS
hash
KSK
com
example.com. NS
ns.A.example.net.
ZSK
public
private
example.com. DNSKEY
ZSK/A
ZSK/B
KSK
KSK
www.example.com. AAAA
2001:db8::c01d:cafe
2001:db8::c001:c0de
ZSK/A
ns.B.example.net.
ZSK
public
private
example.com. DNSKEY
ZSK/A
ZSK/B
KSK
KSK
www.example.com. AAAA
2001:db8::c01d:cafe
2001:db8::c001:c0de
ZSK/B

Model 2


a.gtld-servers.net.
example.com. DS
hash
KSK/A
KSK/B
com
example.com. NS
ns.A.example.net.
ZSK
public
private
KSK
public
private
example.com. DNSKEY
ZSK/A
KSK/A
ZSK/B
KSK/B
KSK/A
www.example.com. AAAA
2001:db8::c01d:cafe
2001:db8::c001:c0de
ZSK/A
ns.B.example.net.
ZSK
public
private
KSK
public
private
example.com. DNSKEY
ZSK/A
KSK/A
ZSK/B
KSK/B
KSK/B
www.example.com. AAAA
2001:db8::c01d:cafe
2001:db8::c001:c0de
ZSK/B

Details

- Internet Draft:
- https://tools.ietf.org/html/draft-ietf-dnsop-multi-provider-dnssec-01

- Comments welcome.

Algorithm Considerations

Algorithm Considerations

- All providers should use the same signing algorithm
  - Ensures all providers have identical security postures.
  - Required for validators which require signature by each algorithm [1]
  - Required for validators with partial algorithm support.

- All providers should use the same NSEC algorithm:
  - Different algorithms may negatively impact aggressive NSEC caching.
  - NSEC3 should be used with the same parameters.

[1] But there is evidence that most modern validators do not in fact enforce this requirement.

Key Rollovers

Key Rollovers

- Slightly more complicated
    - Co-ordinated action of zone owner and the providers.
- But still fairly straightforward.
- Details are in the Internet Draft (Section 6: Key Rollover Considerations).
    - https://tools.ietf.org/html/draft-ietf-dnsop-multi-provider-dnssec-01#section-6

Prototypes, Testing, Implementation

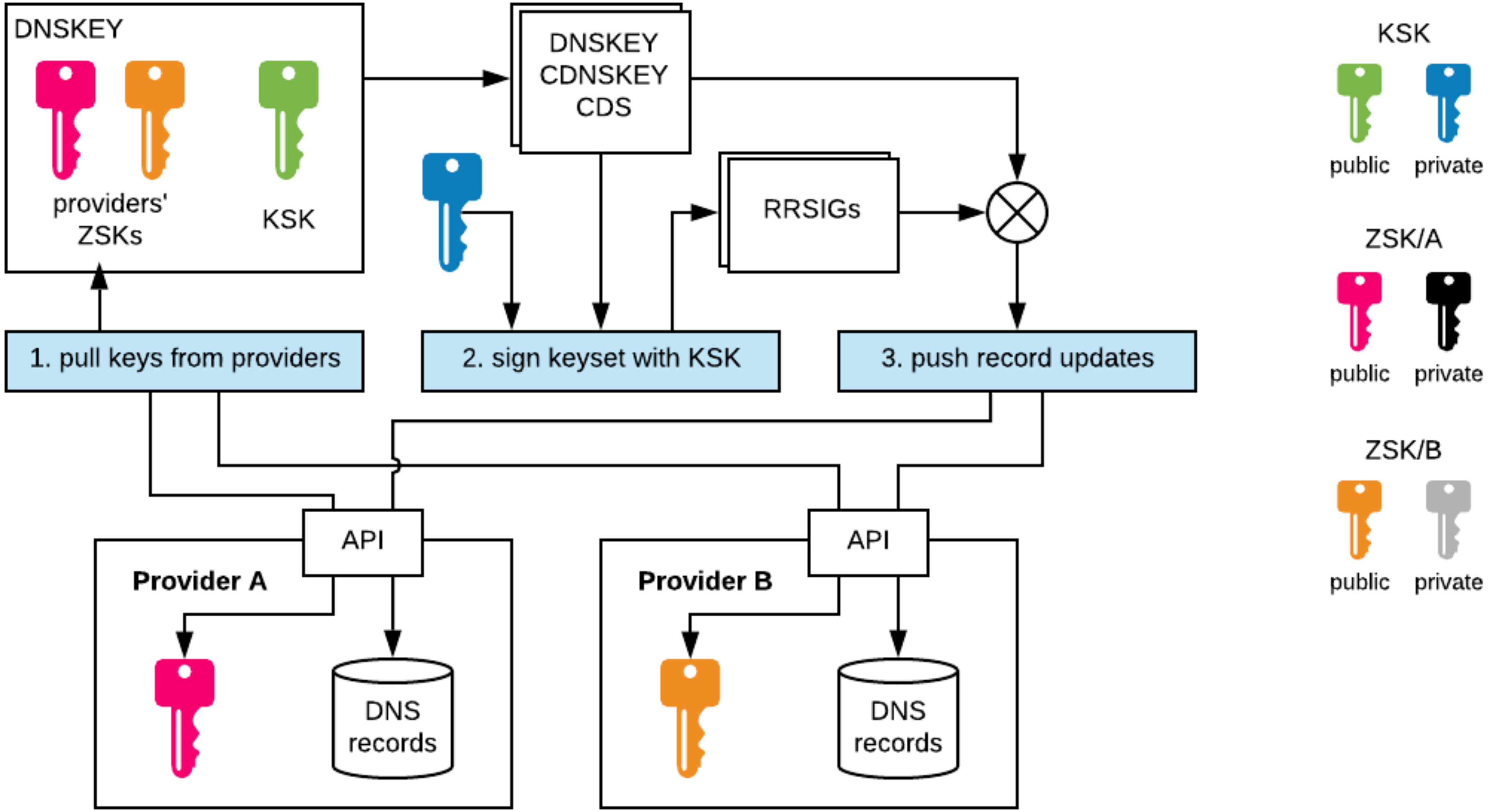
Prototypes deployed

- IETF 102 Hackathon (Montreal, July 2018)
- Prototypes of both models were successfully deployed using open source DNS servers.
- And tested with open source validating resolvers.

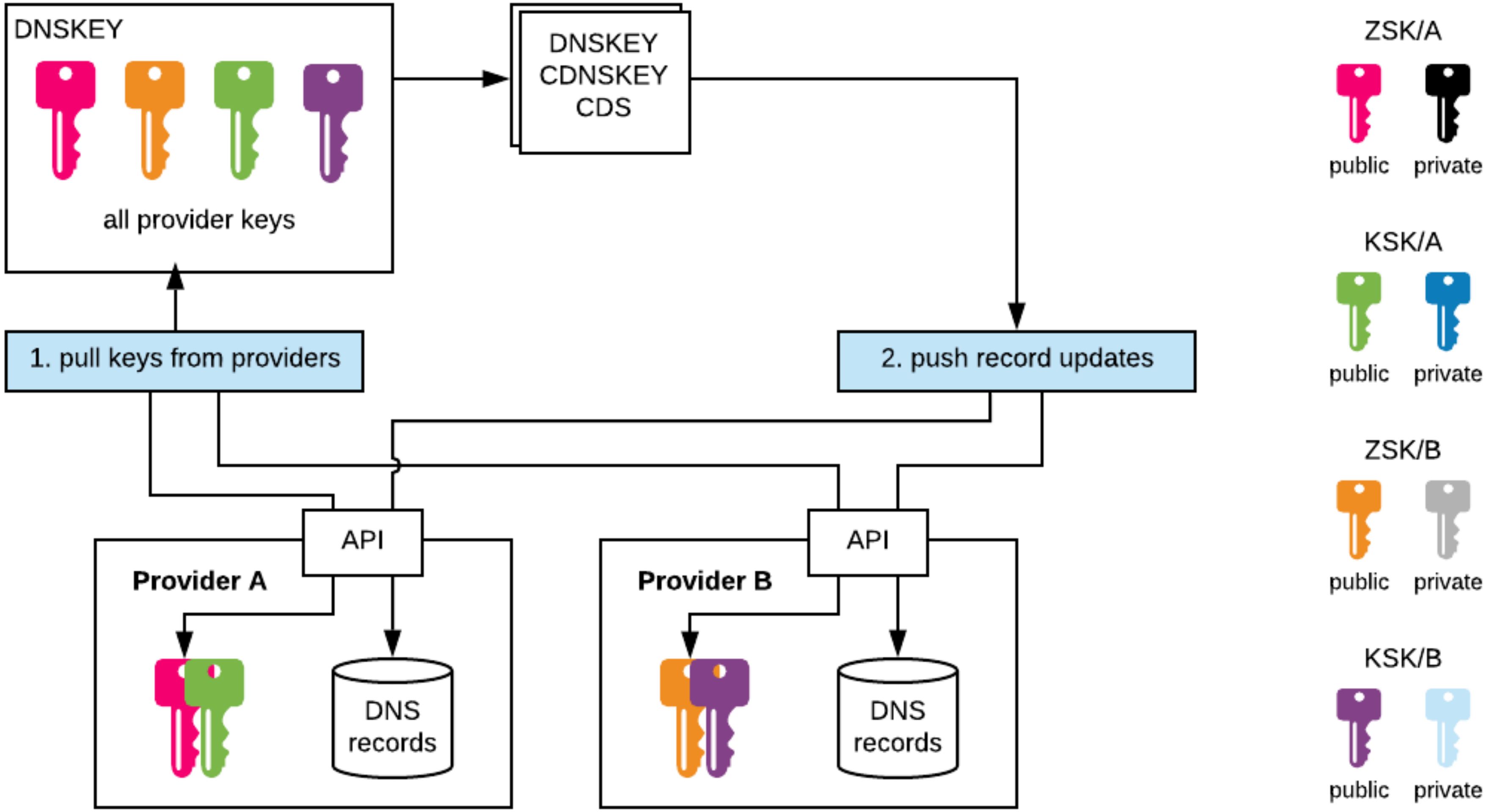
Key Management API Requirements

- Specific functions that need to be supported:
  - Query & authenticate the current DNSKEY RRset
  - For model 1, provide mechanism to import a signed DNSKEY RRset.
  - For model 2, provide mechanism to import a DNSKEY RR.
- Models are API agnostic
  - Should work with any managed DNS provider’s proprietary API (usually REST/HTTPS based) that supports these features.
  - Other API possibilities: DNS UPDATE, EPP

API for Model 1



API for Model 2



Open Source: ISC BIND

- Offline signing will likely just work – could benefit from tooling
- Configurations where BIND is actively maintaining the zone (“auto-dnssec”) should also work (initial tests work)
  - Dynamic key import accomplished by “**dnssec-importkey**” tool
  - Works out of the box for model 2.
  - Model 1, where an entire DNSKEY RRset (rather than individual DNSKEYs) needs to be imported, is a bit more challenging. Figuring that one out – may need small tweak in server code to work.

Open Source: Knot DNS and other

- Offline Signer (talk upcoming from CZ.NIC)
  - Should be able to support Model 1
- Model 2?

- Looking at other implementations:
  - PowerDNS
  - NSD probably not a candidate today, since it is not a signer.

Inter-Provider Handoff/Migration

Inter-Provider Handoff of signed zones

- How to cleanly transition a signed zone from one provider to another non-disruptively (i.e. breaking DNSSEC validation at any point in the process).
- A source of occasional frustration in the DNS industry.

- The Multi-Signer models could address this problem (assuming co-operating parties)
  - Multi-signer assumes a steady state configuration.
  - However, necessarily includes on-boarding (and off-boarding) providers into such a configuration, so inter-provider handoff is just a transitional state.
  - Standardizing a scheme for handoff will help.

Inter-Provider Handoff of signed zones

- Interest from the ICANN community in these models
- What specific technical and/or policy work needs to be done?

Other Issues

Compatibility of Traffic Mgmt Features

- Are these non-standardize/traffic management features really compatible across distinct providers?
- It depends on which feature and the extent of compatibility desired.
- GSLB – return response with “closest” server(s)
- Failover Pools
- Load Balancing (a bit trickier)
- Other?

- Augmented systems may be needed in some cases. Some providers have generic APIs to talk to external systems and integrate that into response decisions.

Enhancing the Traditional model?

- From an earlier slide: Traditional Zone Transfer model limitation:
- **doesn't work with non-standardized DNS features that are quite widely used in the DNS industry today.**

- Extend the DNS protocol to support the non-standard features?
- Goes back to: should DNS zones store program instructions & how to encode those instructions into new RRsets, and transmit via XFR.
- Have had discussions with a number of managed DNS providers
- Doesn't seem very likely to have support.
- Non-standard = secret sauce = competitive advantage

Response size considerations

- Both these models cause an increase in the size of the DNSKEY RRset
- Is this an issue, as a practical matter?
  - May depend on algorithm choice, keysize choice, number of providers, etc.
  - 2048-bit RSA may be an issue.
  - But ECSDSAP256SHA256 doesn’t appear to cause any concern.

- Non-DNSKEY responses are unchanged.

Response size considerations

A test with a 2LD and ECDSAP256SHA256, minimal responses, 2 providers.

Response1: 1 KSK, 2 ZSK – 1 per provider: 471 bytes

* 1 KSK: 80 bytes each (if compression is used, likely fixed size)
* 2 ZSK: 80 bytes each
* 1 RRSIG: 109 bytes (variable size signer in rdata dependent on zone name)

Response2: 1 KSK roll, 2 ZSK – 1 per provider: 580 bytes

* 2 KSK: 80 * 2
* 2 ZSK: 80 * 2
* 2 RRSIG: 109 * 2

Response3: 1 KSK roll, 2 ZSK rolls – 2 per provider: 740 bytes

* 2 KSK 80 * 2
* 4 ZSK 80 * 4
* 2 RRSIG 109 * 2

These responses are far smaller than the common Internet MTU.

In Summary ...

Summary

- Many organizations employ multiple providers.
- Many use non-standardized, traffic management features.
- We need to make DNSSEC work in such environments.
- The models presented here accomplish this.
- Can tackle the inter-provider handoff problem too.
- A number of large managed DNS providers are working on implementing these schemes today.
- More widespread adoption would be good.
- Open-source implementations should support multi-signer configurations too.

Questions and/or comments?