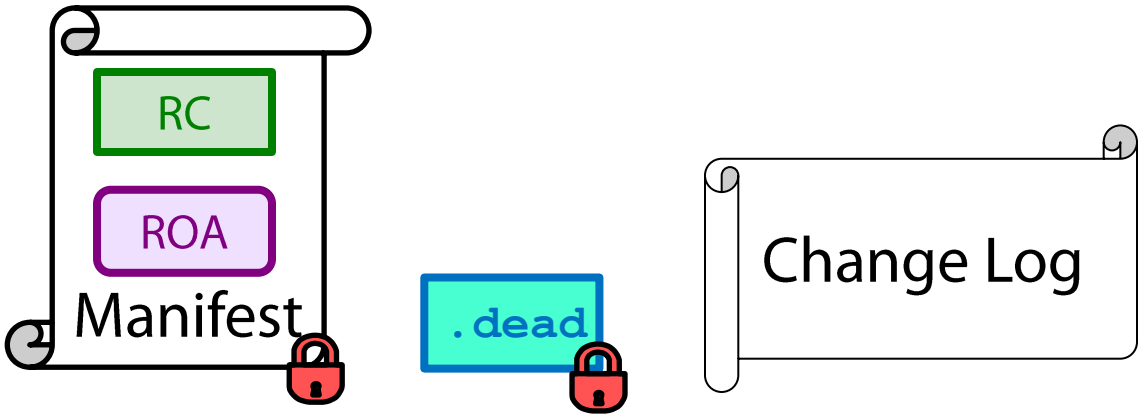
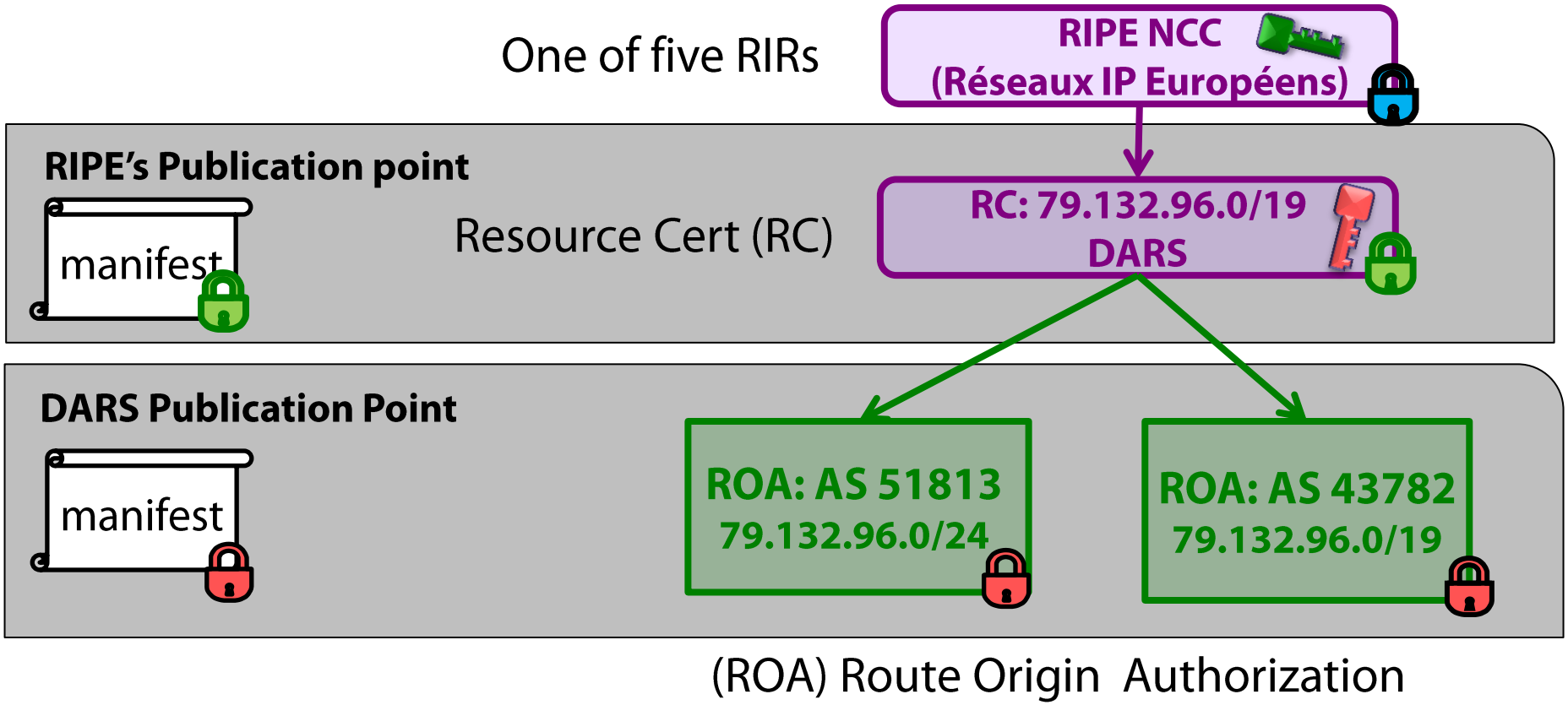
Proposal for signaling consent from whacked RPKI objects



BOSTON UNIVERSITY

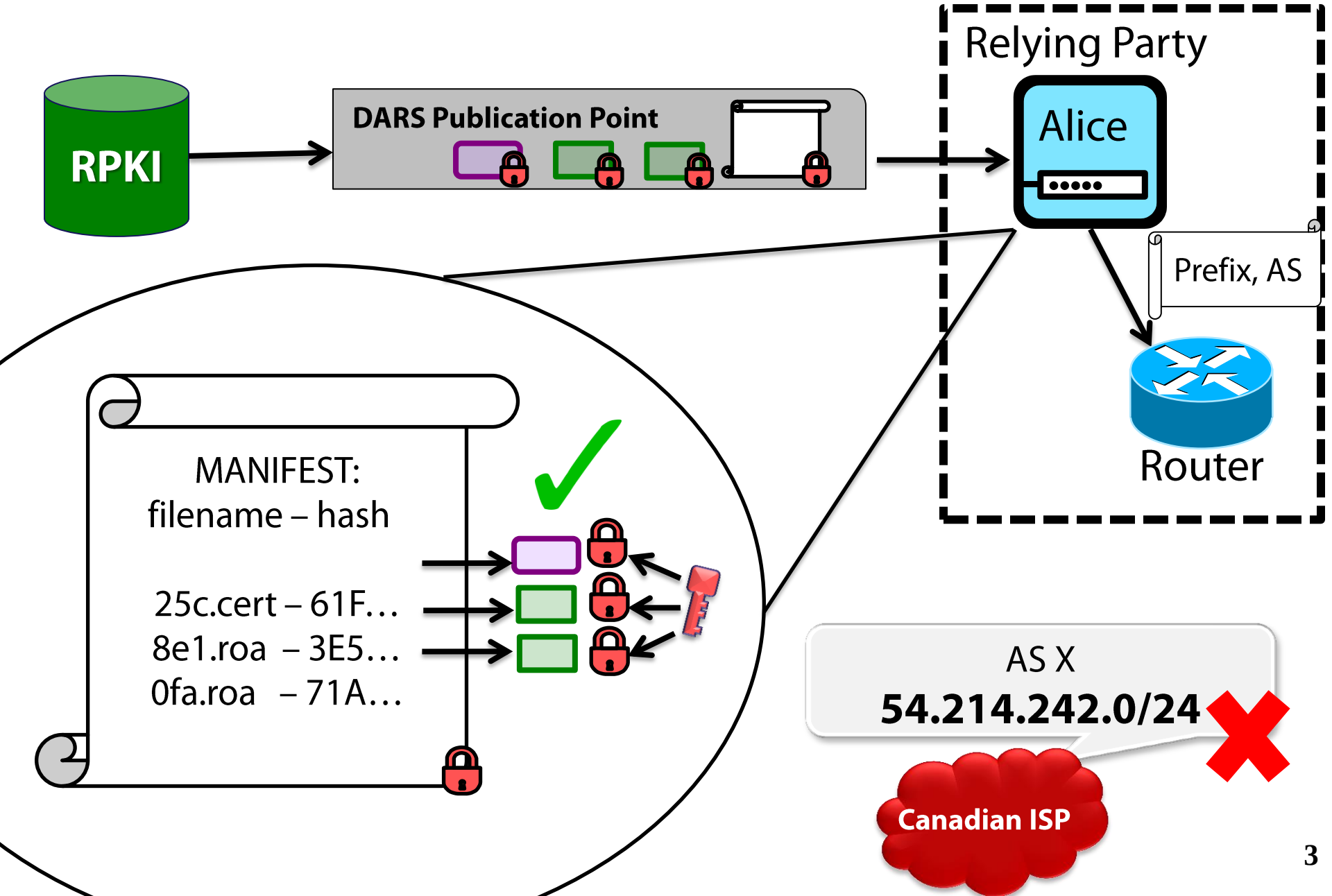
Sharon Goldberg
Danny Cooper, Ethan Heilman, Leonid Reyzin

structure of the RPKI [RFC 6480]

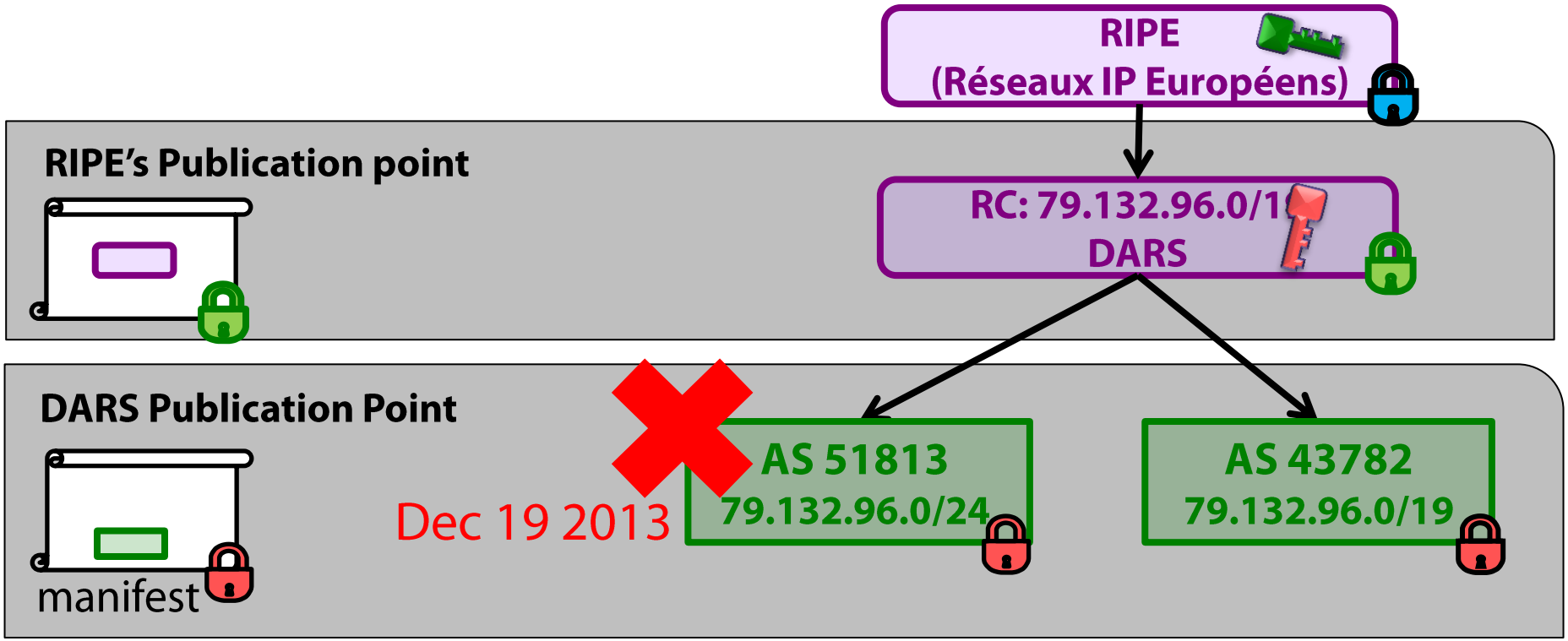


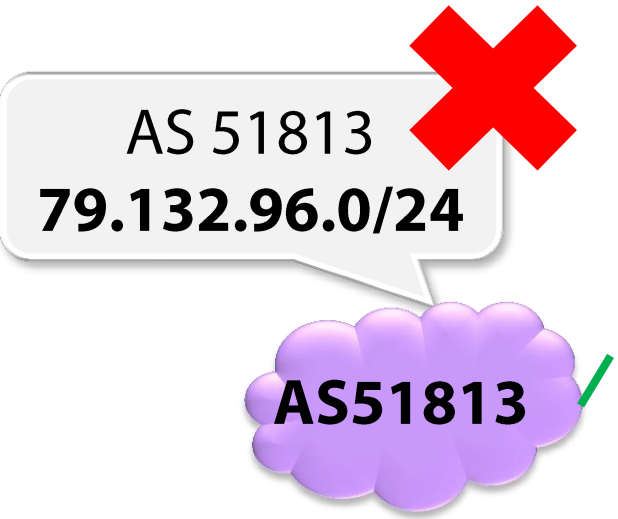
(ROA) Route Origin Authorization

how relying parties sync to the RPKI [RFC 6480]



RPKI authorities can unilaterally whack ROAs





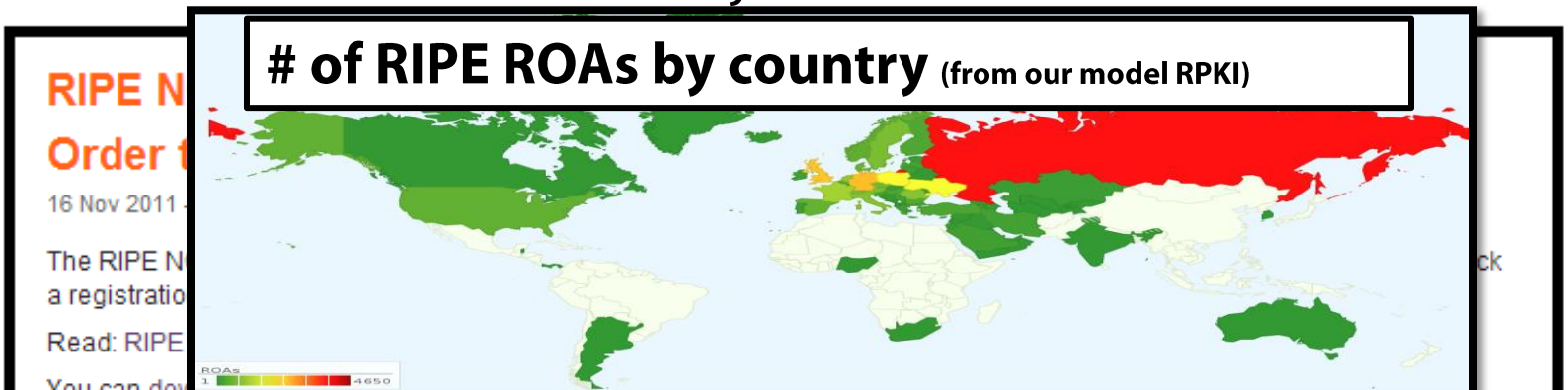
RPKI authorities can blackhole BGP routes. Why?

1. RPKI authorities can delete ROAs
2. Deleted ROAs can cause **invalid** BGP routes
3. RPs should **drop invalid** BGP routes to stop **sub**prefix hijacks.

IP prefix takedowns by whacking ROAs?

- Prior to the RPKI, authorities could allocate IPs but not revoke them.
- But RPKI authorities **can** revoke IP allocations!
- Creates a risk that the RPKI can be used for unilateral takedowns.
  - Law enforcement? Business disputes? Extortion?
  - The RPKI designed to secure routing, not enable takedowns.
  - **[Mueller-Kuerbis'11, Mueller-Schmidt-Kuerbis'13, Amante'12, FCC'13,…]**

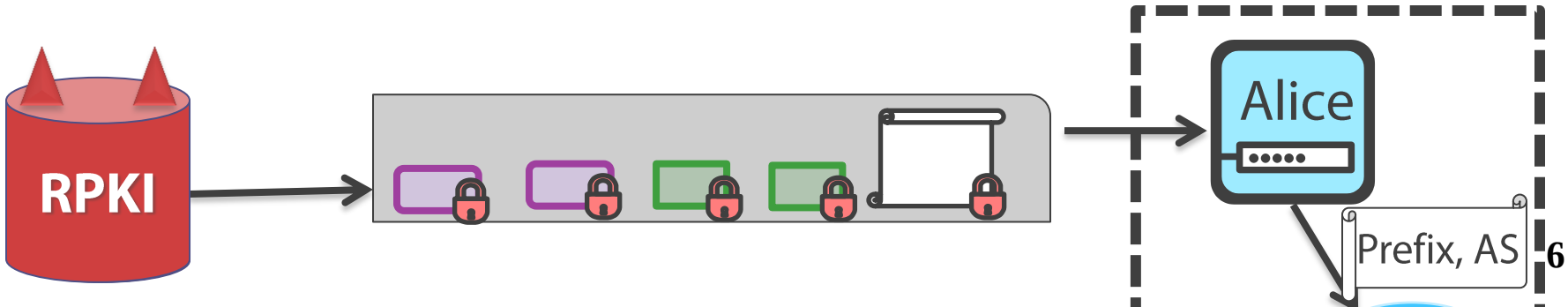
- States seem to want the ability to takedown IP prefixes…
  - Dutch court ordered RIPE to lockdown prefixes registration (Nov'11)
  - US court issued a writ of attachment on Iran's IP prefixes (June'14)
  - IP allocation does not reflect jurisdiction.



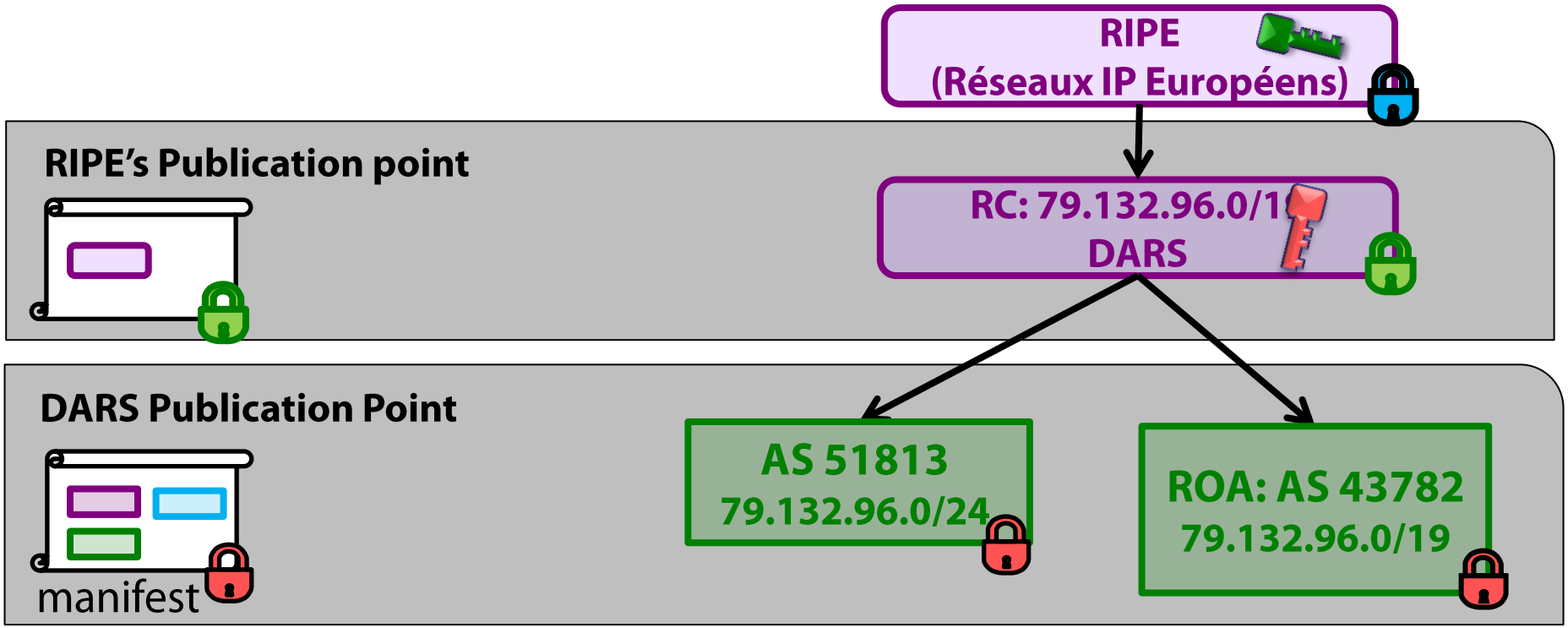
proposal : require consent to whack objects [SIGCOMM'14]

- **Design goals:**
  - **Consent**: Resource certs (RCs) consent to be whacked.
  - **Consistency**: Relying parties have consistent views of the RPKI.
  - **Transparency:** Relying parties audit RPKI & alarm on problems.
    - "Drop invalid" for prefixes that are not part of an alarm
    - Manually audit prefixes that are part of an alarm.

- **Threat Model:**
  - Similar to certificate transparency **[RFC 6962]**
  - Relying parties honestly audit the RPKI
  - Everyone else (incl. RPKI authorities) is untrusted



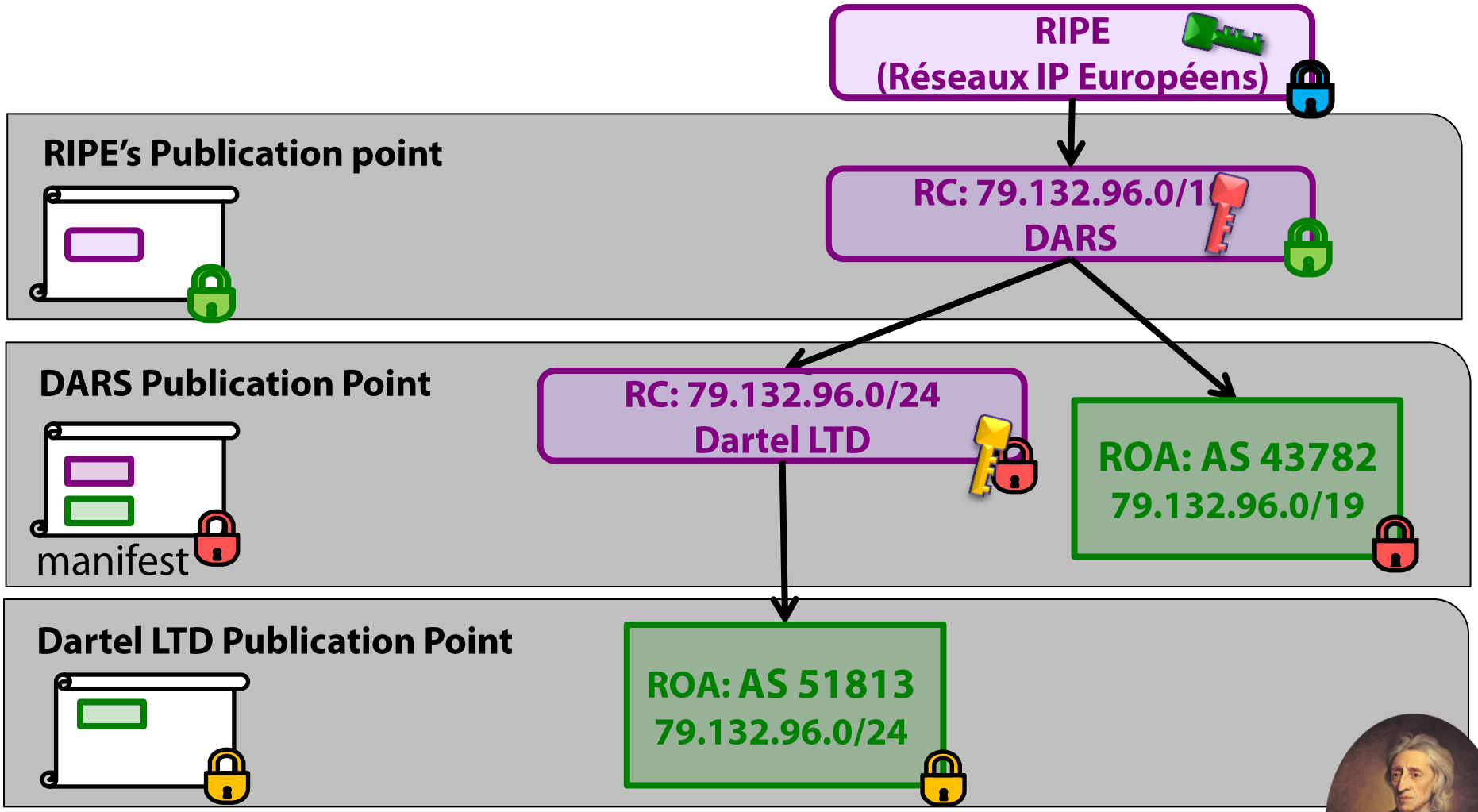
how to consent? introducing .dead objects

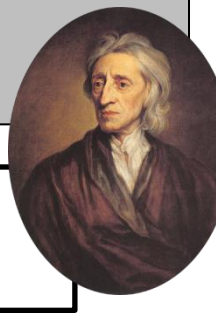


If an authority wants to revoke IP prefixes from a child RC,
it needs consent from that child & its impacted* descendant RCs.

*Descendants aren't always impacted by changes to the parent; ask me why later!

how to consent? introducing .dead objects

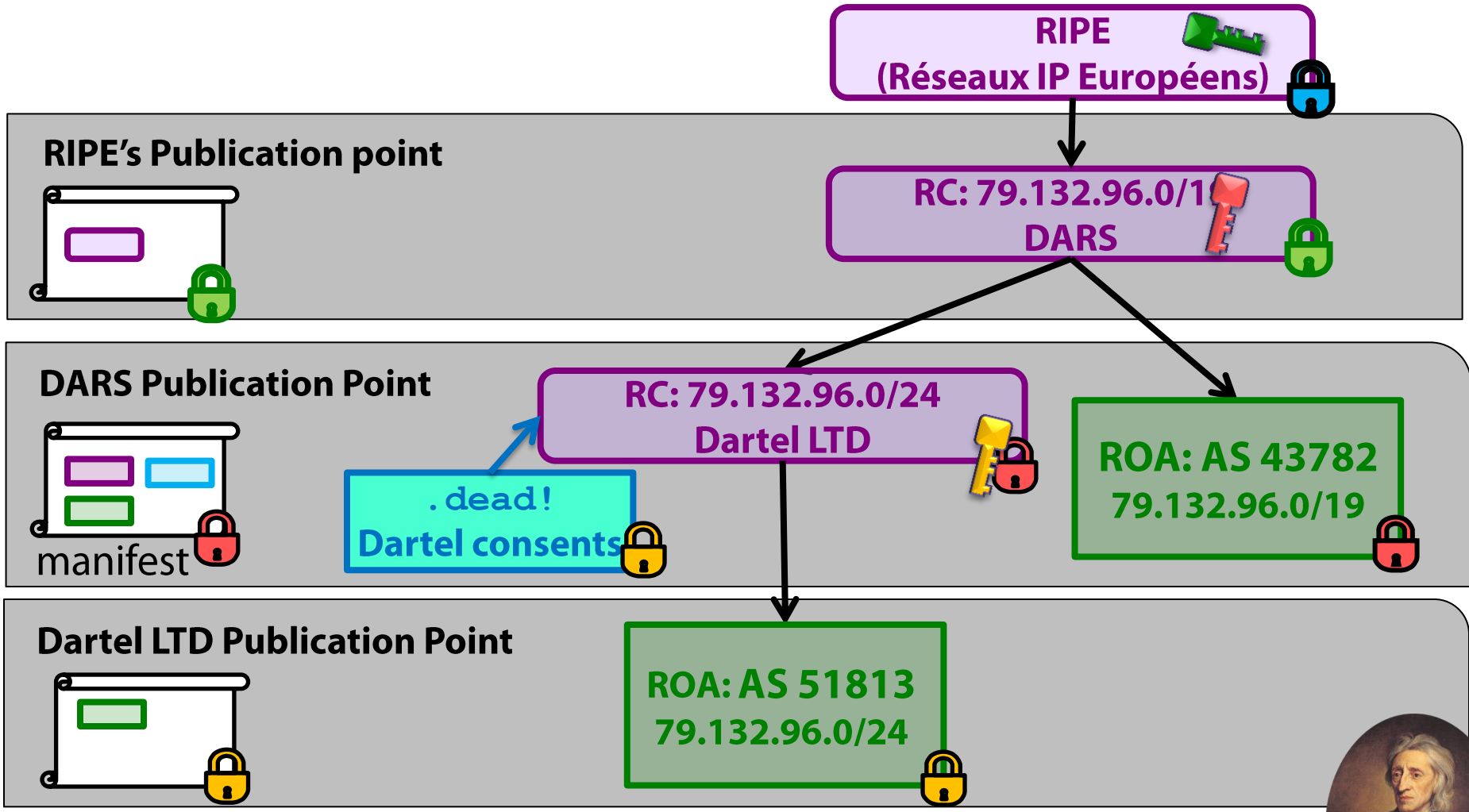


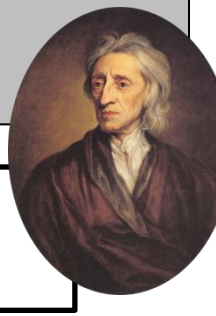


If an authority wants to revoke IP prefixes from a child RC,
it needs consent from that child & its impacted* descendant RCs.

*Descendants aren't always impacted by changes to the parent; ask me why later!

how to consent? introducing .dead objects

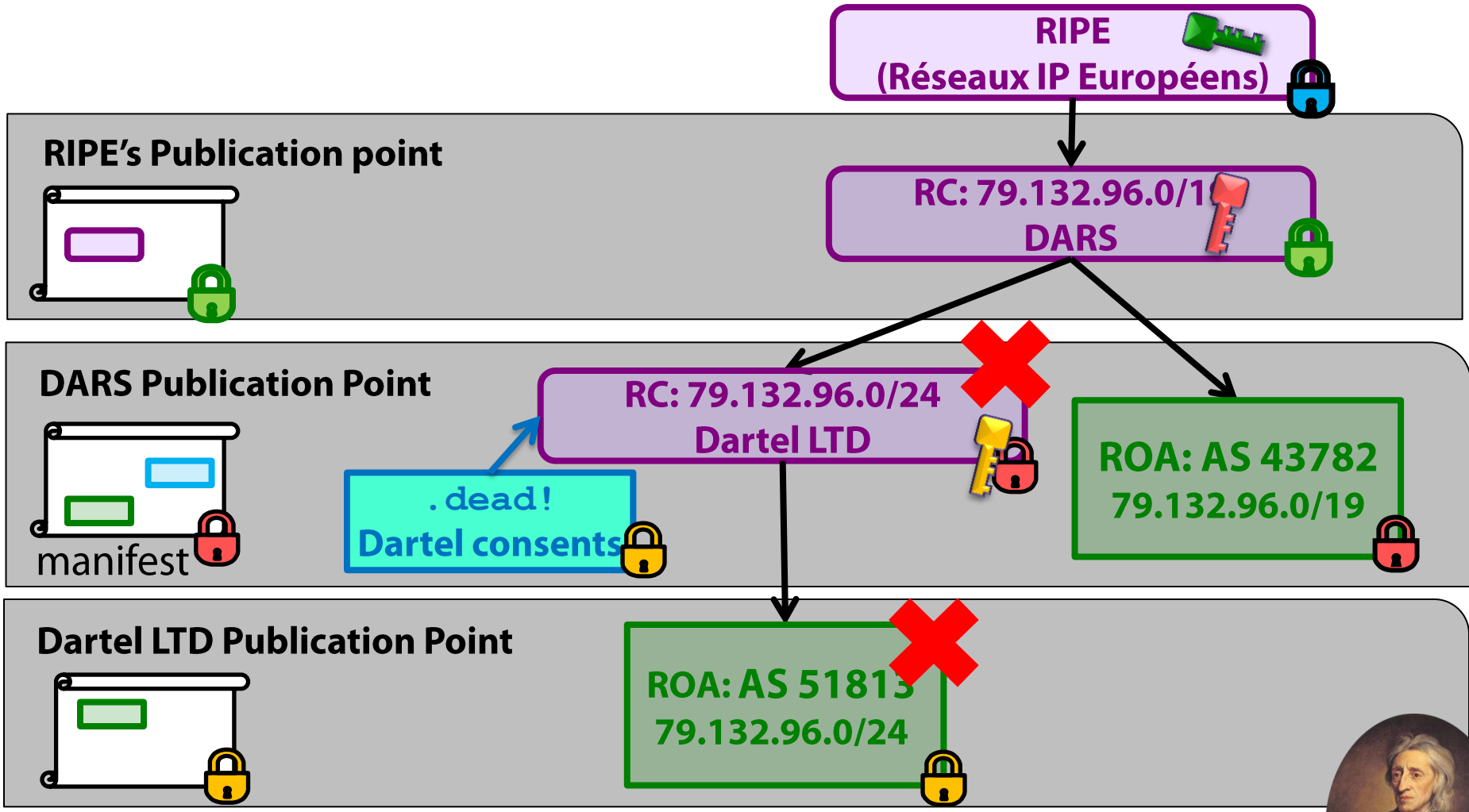


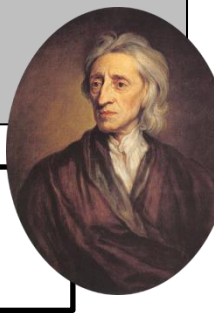


If an authority wants to revoke IP prefixes from a child RC,
it needs consent from that child & its impacted* descendant RCs.

*Descendants aren't always impacted by changes to the parent; ask me why later!

how to consent? introducing .dead objects

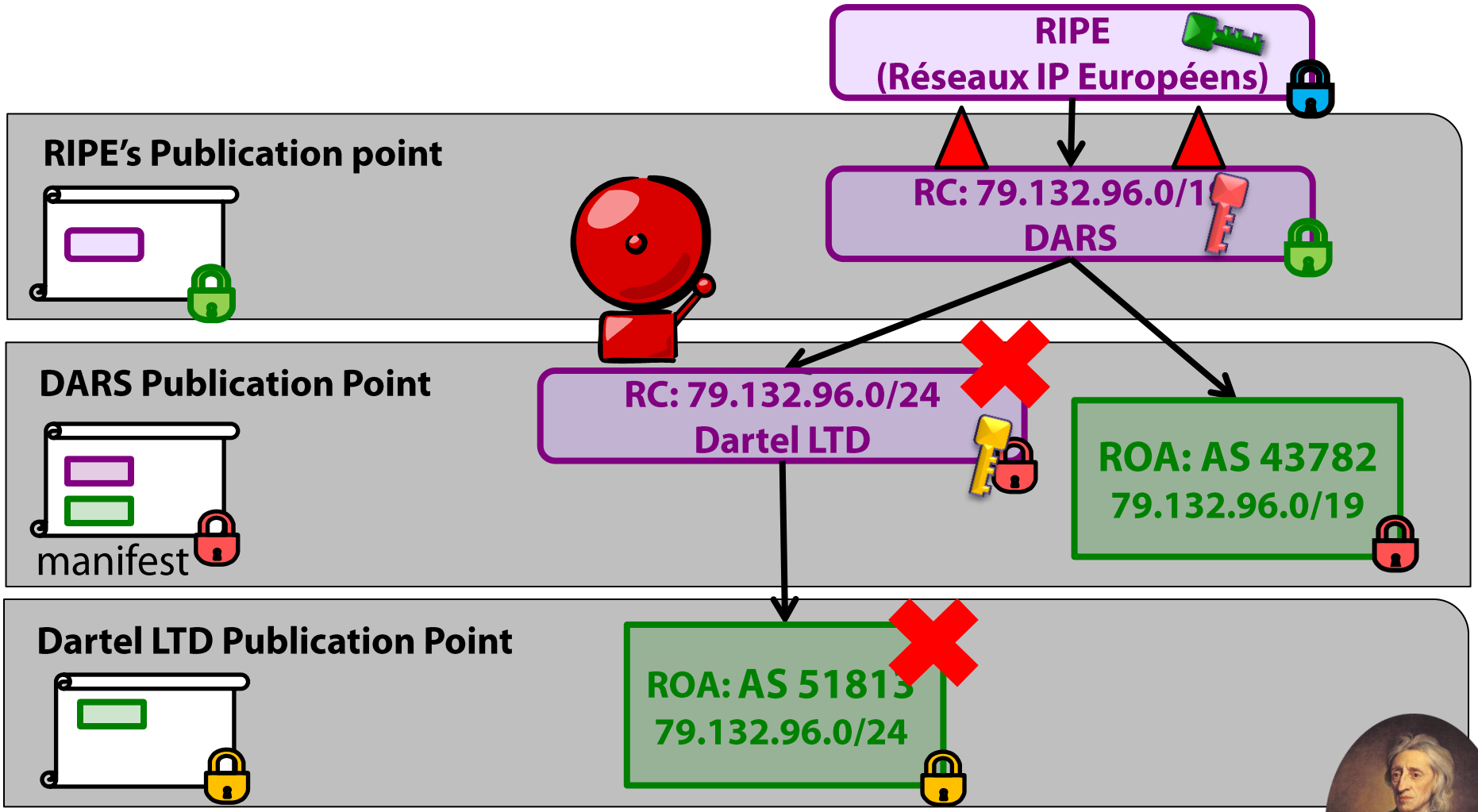


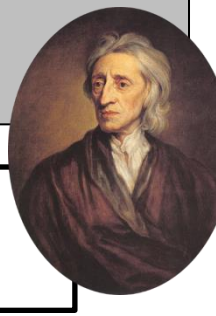


If an authority wants to revoke IP prefixes from a child RC,
it needs consent from that child & its impacted* descendant RCs.

*Descendants aren't always impacted by changes to the parent; ask me why later!

how to consent? introducing .dead objects

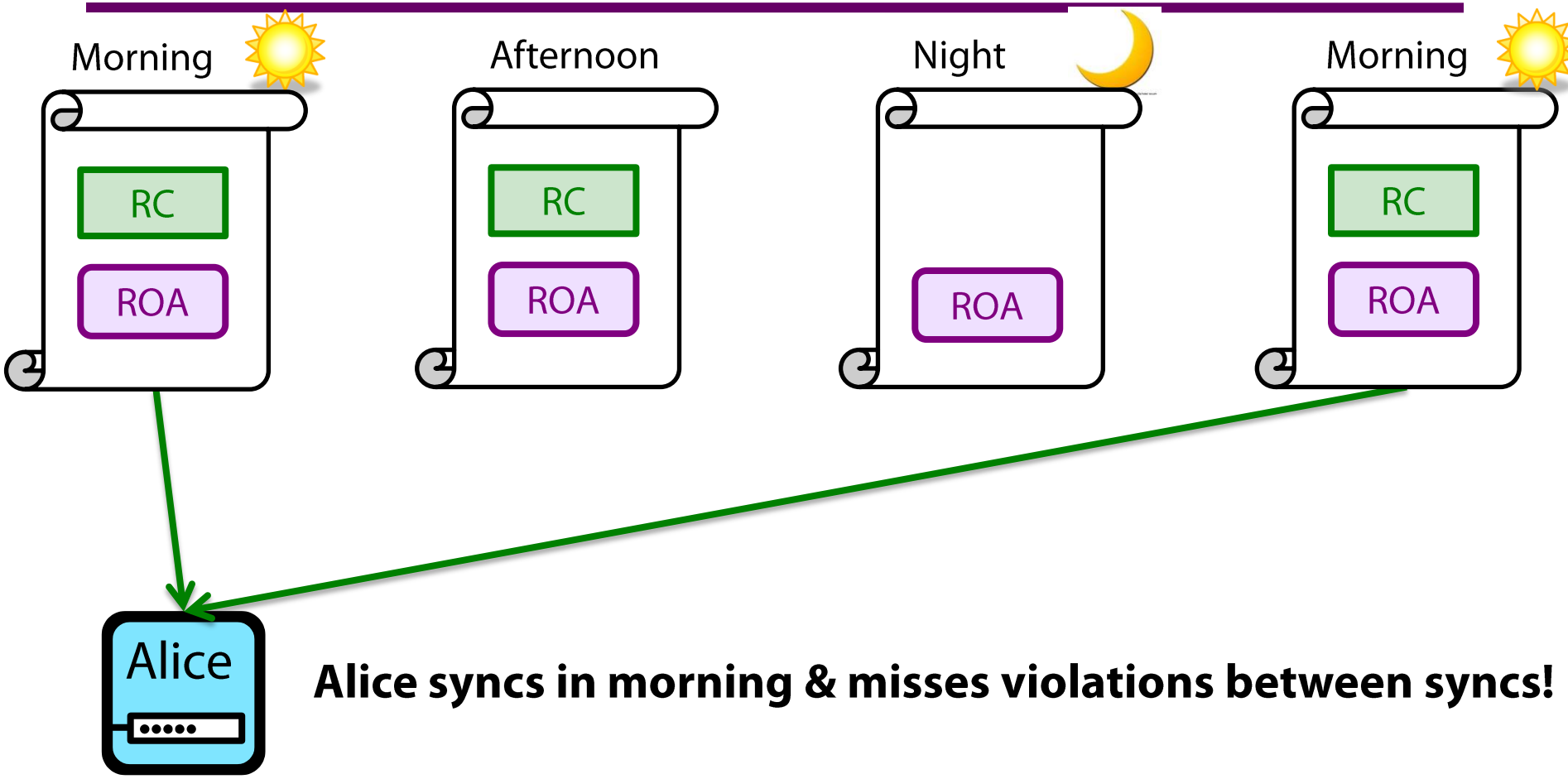




If an authority wants to revoke IP prefixes from a child RC,
it needs consent from that child & its impacted* descendant RCs.

*Descendants aren't always impacted by changes to the parent; ask me why later!

what about alarms between syncs?

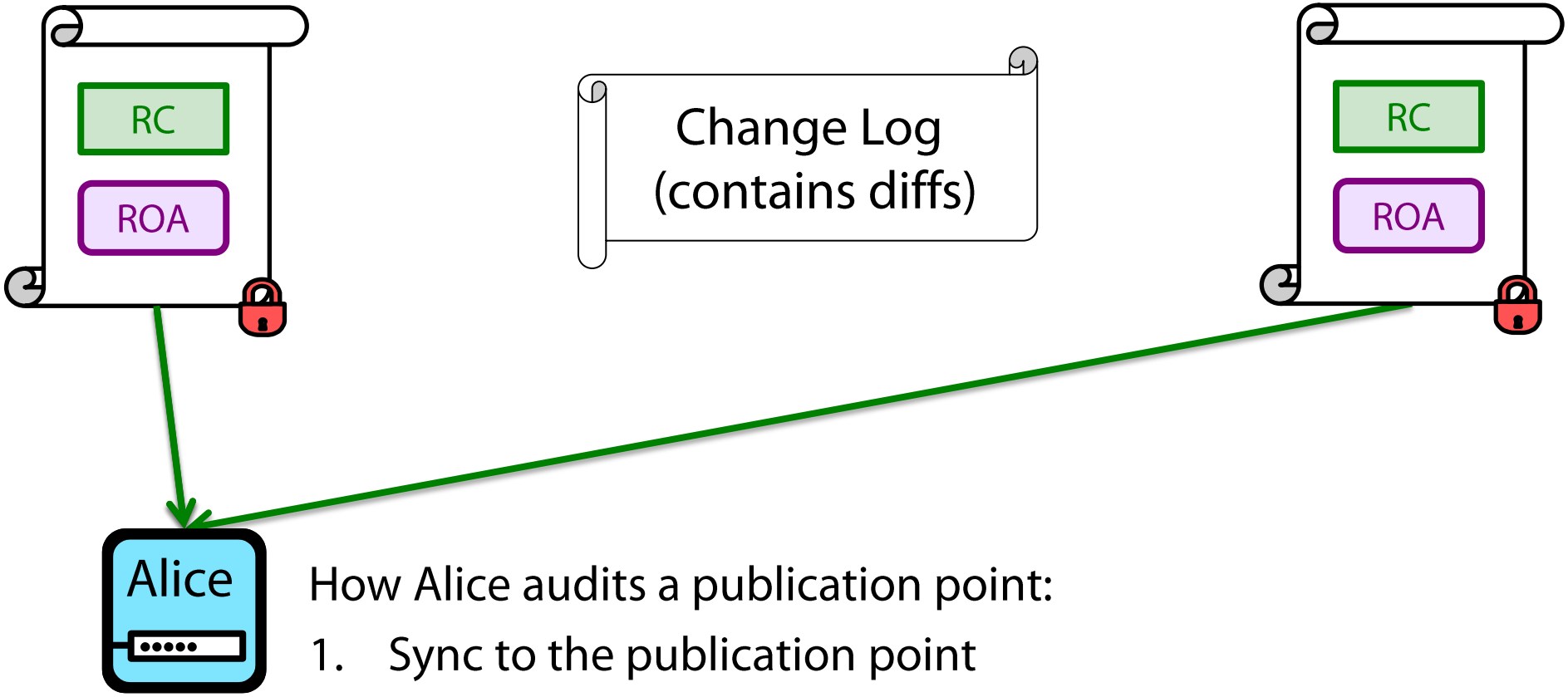


Alice syncs in morning & misses violations between syncs!

Why does Alice need to catch violations between syncs?

- So Alice can audit the RPKI
- So we can have consistency (explained later)

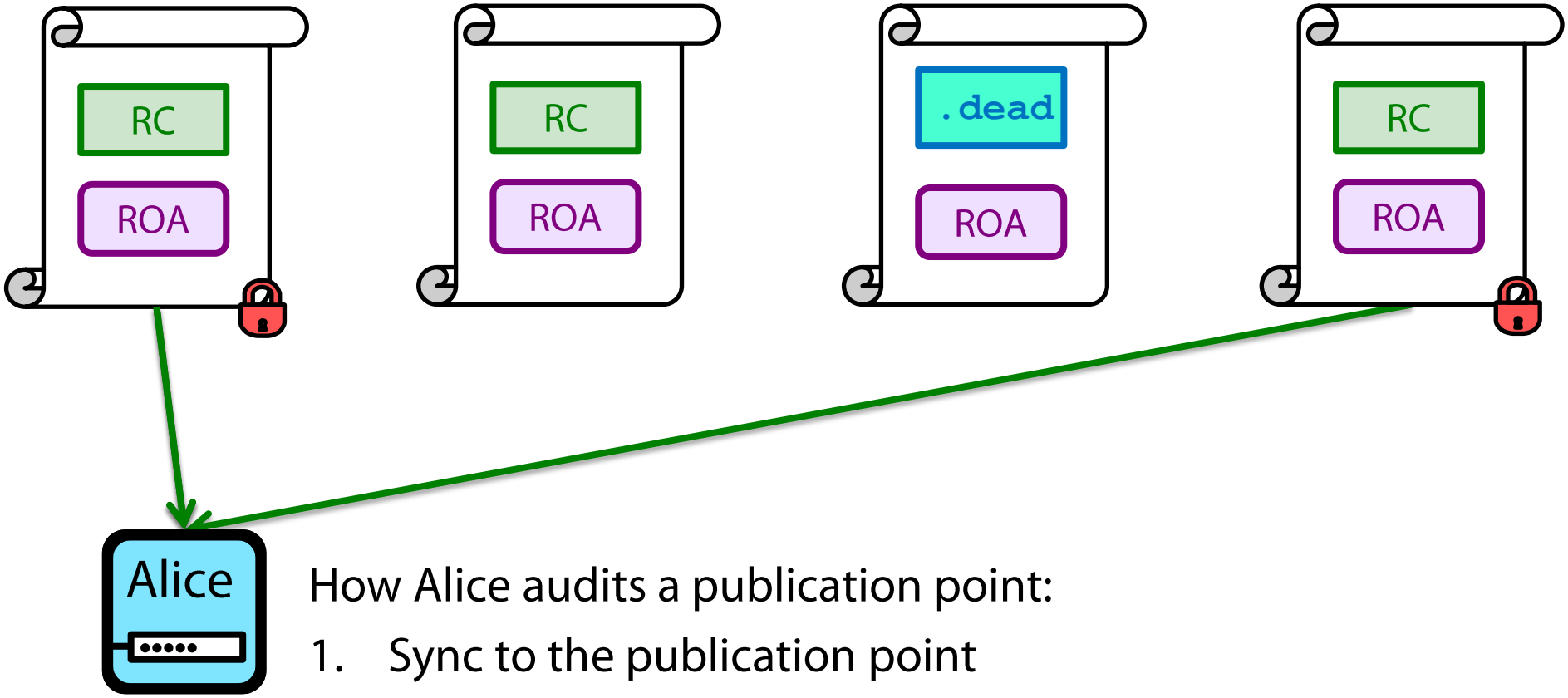
catching alarms between syncs!



How Alice audits a publication point:

1. Sync to the publication point
2. Use change log to reconstruct intermediate manifests

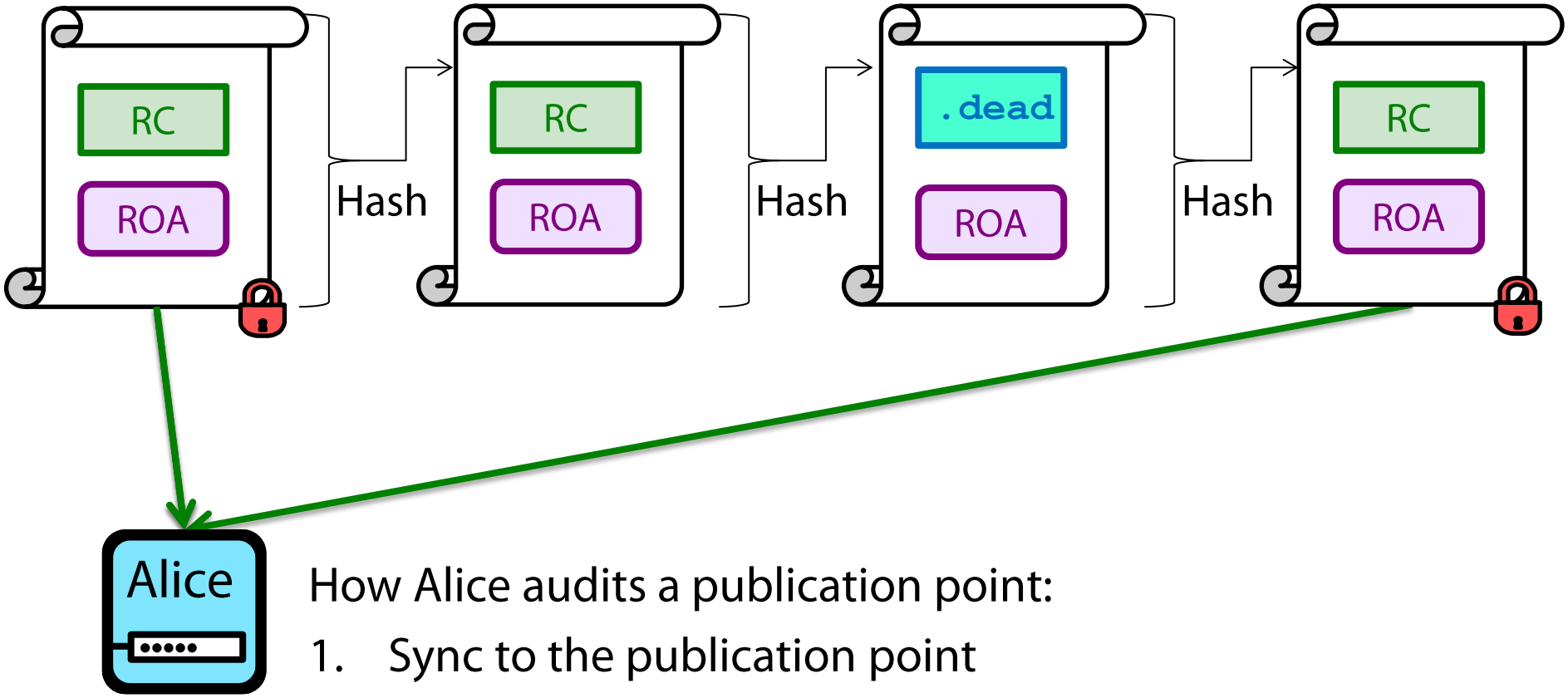
catching alarms between syncs!



How Alice audits a publication point:

1. Sync to the publication point
2. Use change log to reconstruct intermediate manifests

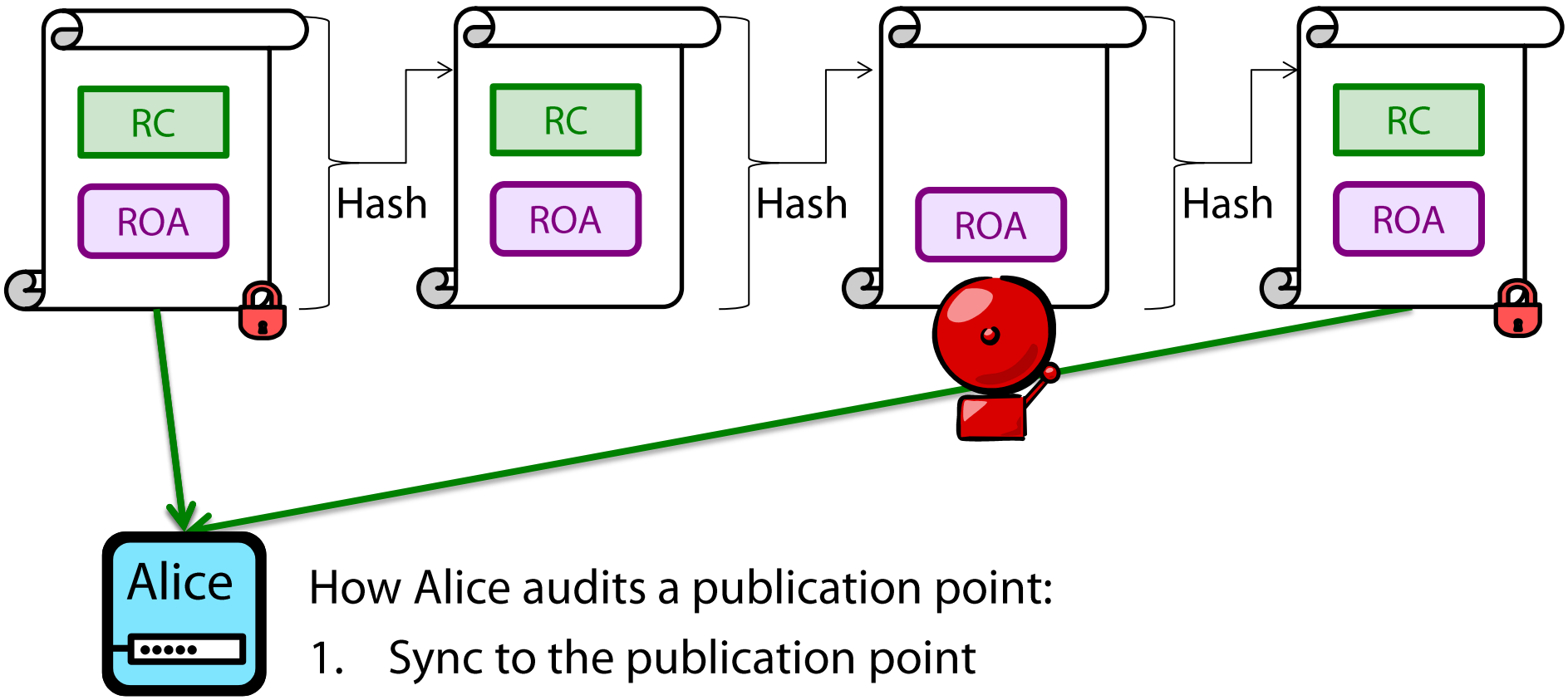
catching alarms between syncs!



How Alice audits a publication point:

1. Sync to the publication point
2. Use change log to reconstruct intermediate manifests
3. Verify the hash chain & signature of the latest manifest
4. Alarm if a consent violation is detected.

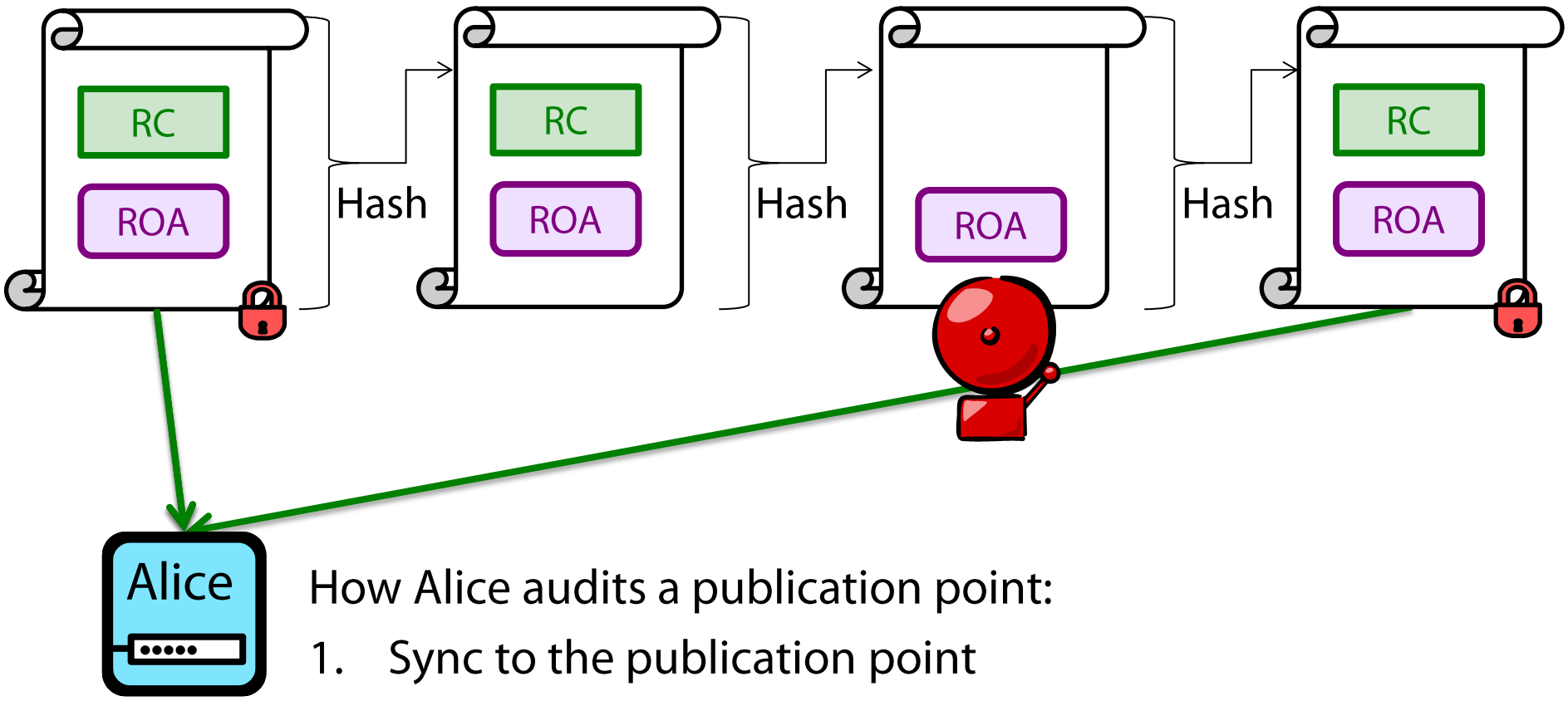
catching alarms between syncs!



How Alice audits a publication point:

1. Sync to the publication point
2. Use change log to reconstruct intermediate manifests
3. Verify the hash chain & signature of the latest manifest
4. Alarm if a consent violation is detected.

catching alarms between syncs!



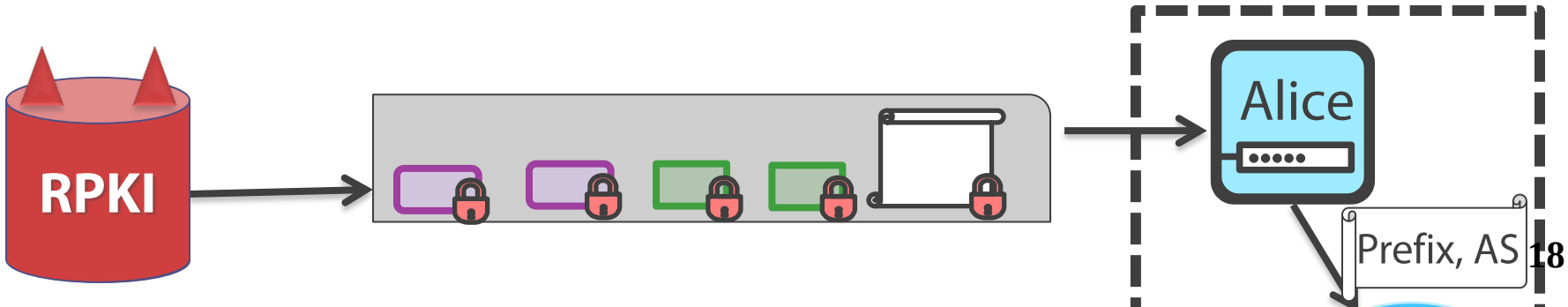
How Alice audits a publication point:

1. Sync to the publication point
2. Use change log to reconstruct intermediate manifests

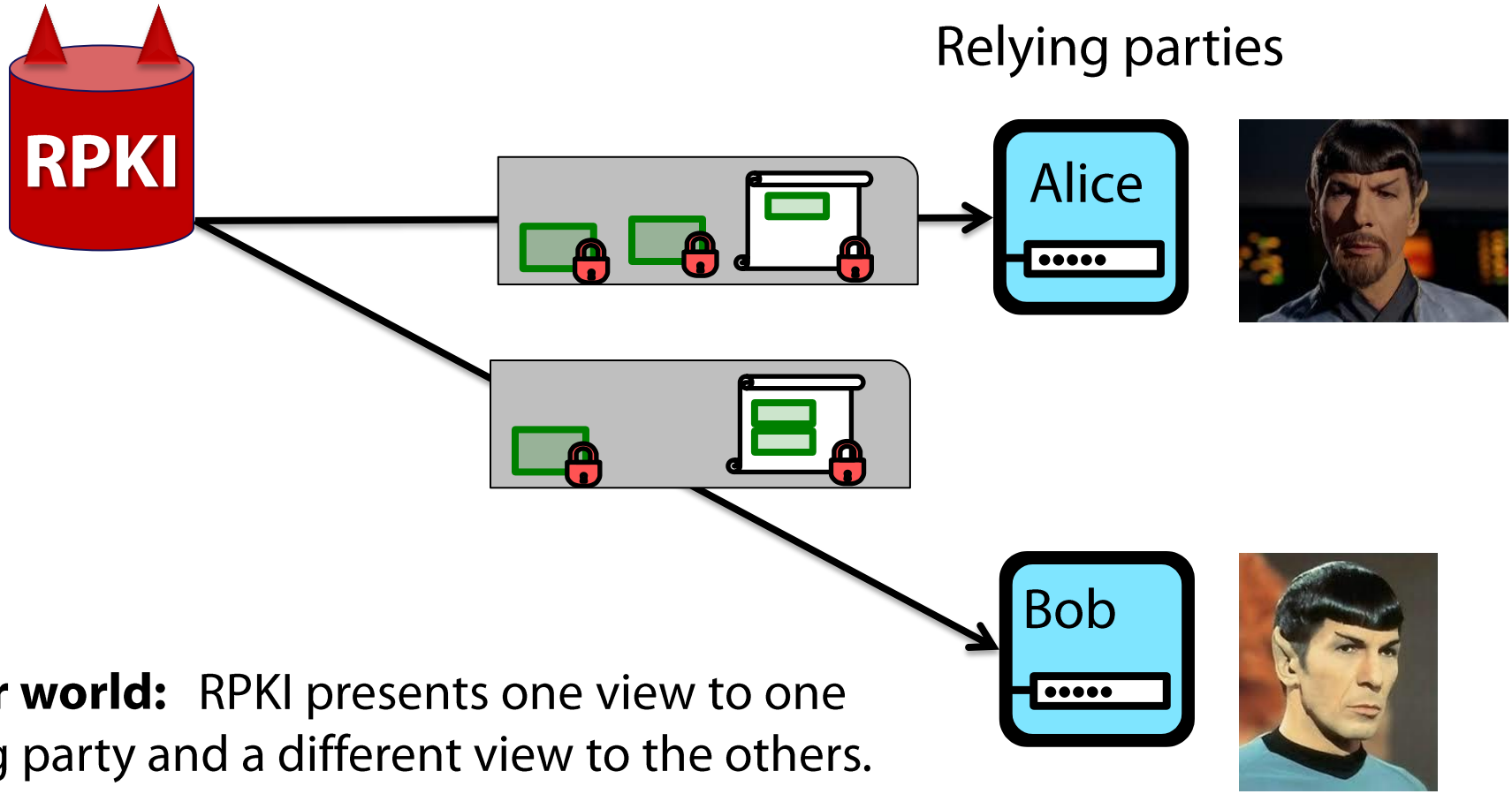
Valid Remains Valid. Our auditing algorithm makes sure that once a relying party has seen a valid resource cert (RC), that RC remains valid until it consents to be deleted/modified.

proposal : require consent to delete objects [SIGCOMM'14]

- **Design goals:**
  - ✓ **Consent**: **.dead** objects indicate consent to whack resource certs (RCs)
  - **Consistency**: Relying parties have consistent views of the RPKI.
  - ✓ **Transparency:** Relying parties audit RPKI & alarm on problems.
    - "Drop invalid" for prefixes that are not part of an alarm
    - Manually audit prefixes that are part of an alarm.



mirror worlds: inconsistent views of the RPKI

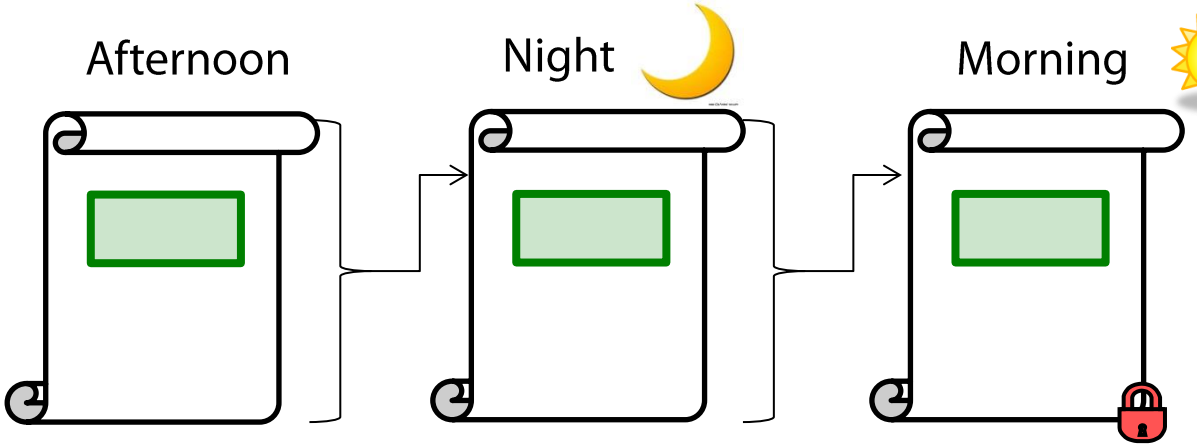


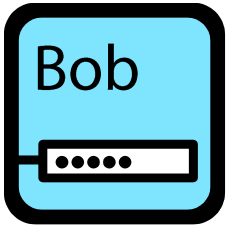
Mirror world: RPKI presents one view to one relying party and a different view to the others.

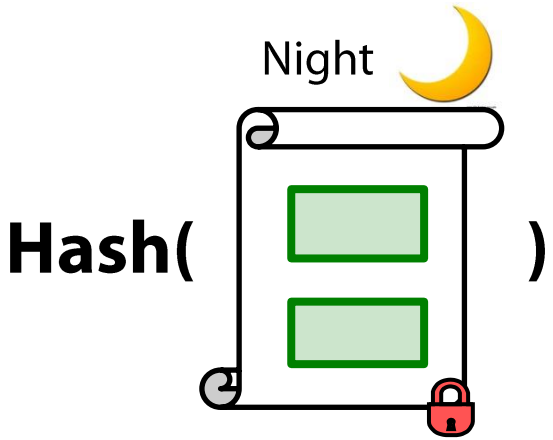
Why do we care?

- Auditing is less meaningful if Alice's view is different from everyone else's.
- Eg. Suppose Alice audits the RPKI to make sure her own ROAs are OK.

detecting mirror worlds using manifest hash chains







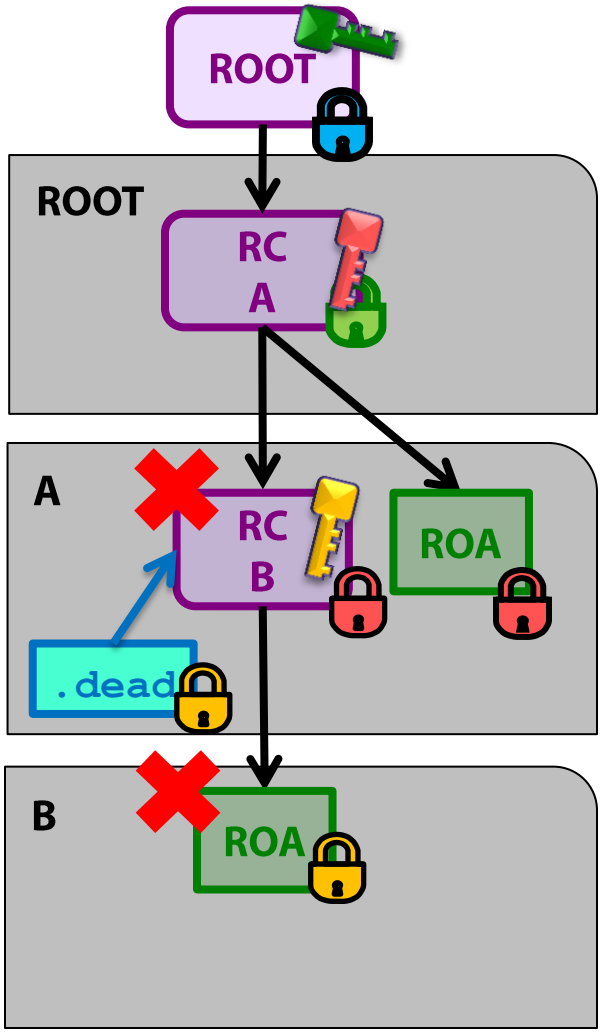
Bob sends a hash of his latest manifest & Alice finds it in her hashchain.

No mirror worlds!

If the consistency check passes, relying parties saw the same valid objects.

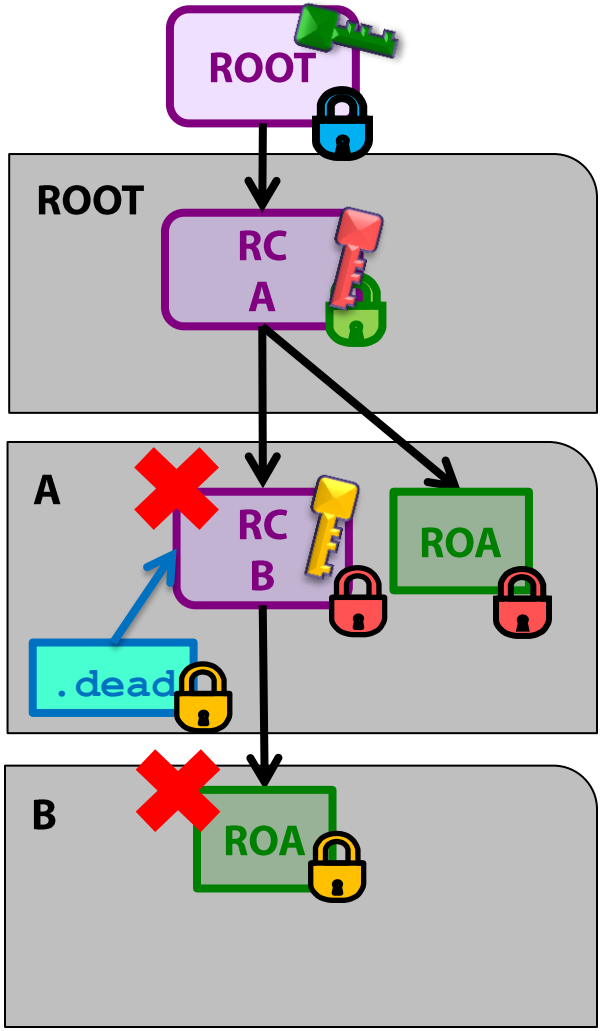
our proposal vs suspenders

our proposal [SIGCOMM'14]

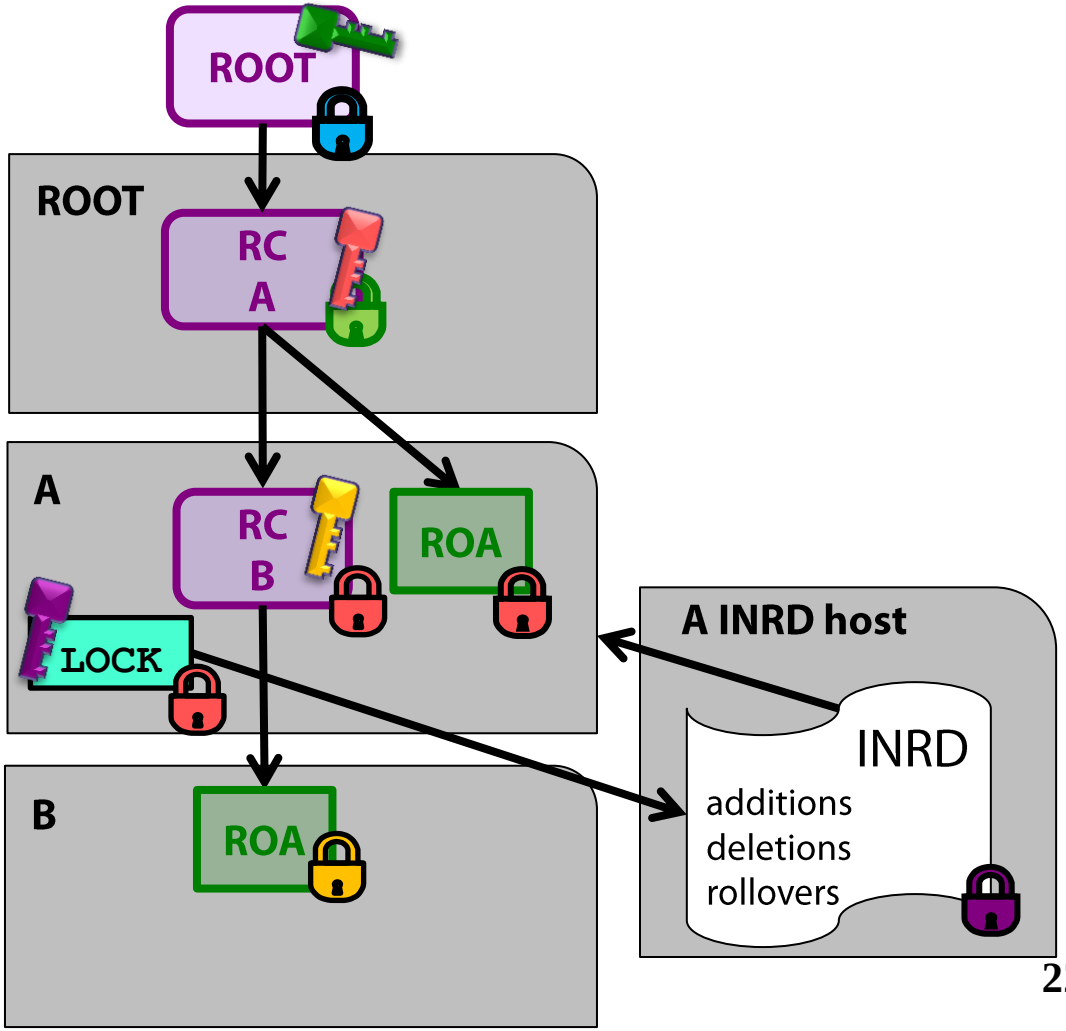


our proposal vs suspenders

our proposal [SIGCOMM'14]

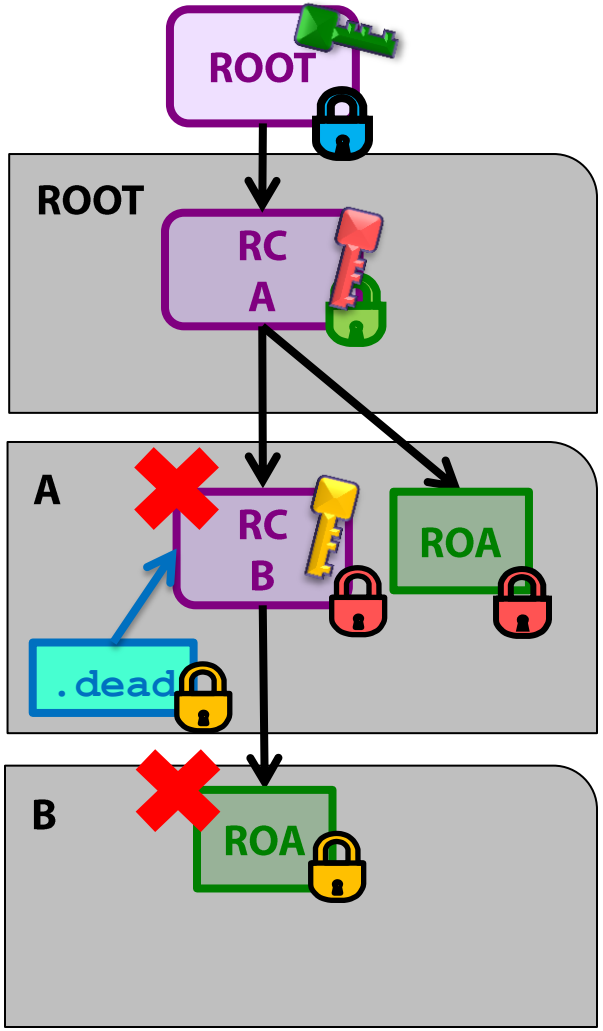


suspenders [draft-kent-sidr-suspenders-02]

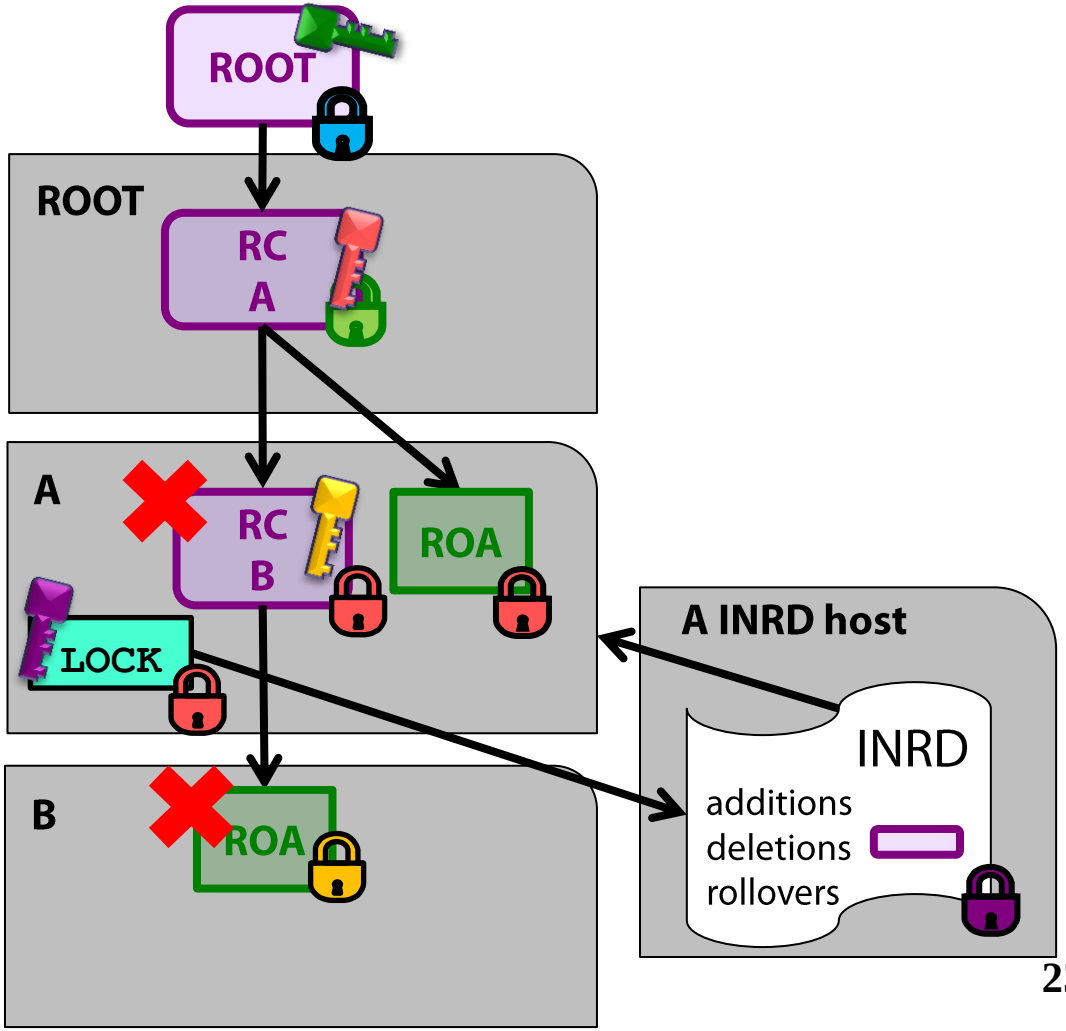


our proposal vs suspenders

our proposal [SIGCOMM'14]



suspenders [draft-kent-sidr-suspenders-02]



our proposal vs suspenders

		Our proposal	Suspenders
	Auditor:	Any Relying Party	
	Consent for whacking?	Yes: RCs	Yes: RCs & ROAs
	"Consent" for "ROA competition"?	No	Yes
	Consistency?	Yes	No
Requiren	Limited non-repudiation?	Yes	No?

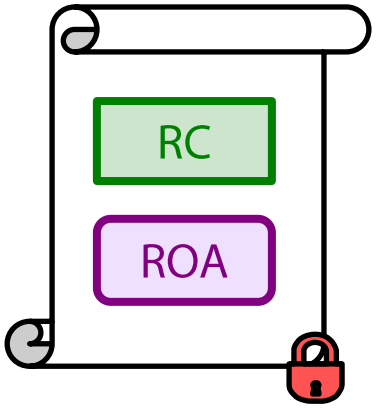
our proposal vs suspenders

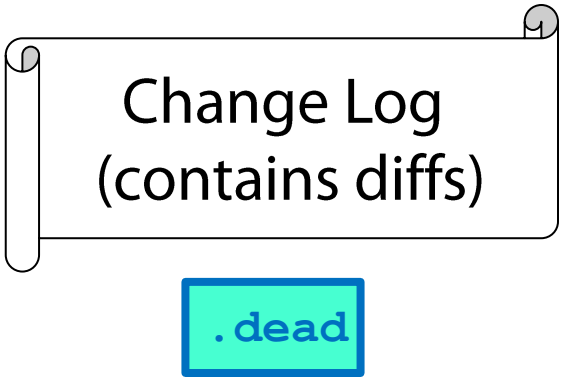
		Our proposal	Suspenders
	Auditor:	Any Relying Party	
Requiren	Consent for whacking?	Yes: RCs	Yes: RCs & ROAs
	"Consent" for "ROA competition"?	No	Yes
	Consistency?	Yes	No
	Limited non-repudiation?	Yes	No?
Design	New RPKI objects:	.dead .roll change logs	LOCK INRD
	Requires changes to manifests?	Yes	No

our proposal vs suspenders

		Our proposal	Suspenders
	Auditor:	Any Relying Party	
Requirem	Consent for whacking?	Yes: RCs	Yes: RCs & ROAs
	"Consent" for "ROA competition"?	No	Yes
	Consistency?	Yes	No
	Limited non-repudiation?	Yes	No?
Design	New RPKI objects:	.dead .roll change logs	LOCK INRD
	Requires changes to manifests?	Yes	No
	"Out of band" publication points?	Yes	No
	"Consenting" subjects need keys?	Yes	Yes
	Proofs of security goals:	Yes	No

Question for the room: What is the right set of requirements?





Questions ?

http://www.cs.bu.edu/~goldbe/papers/RPKImanip.html

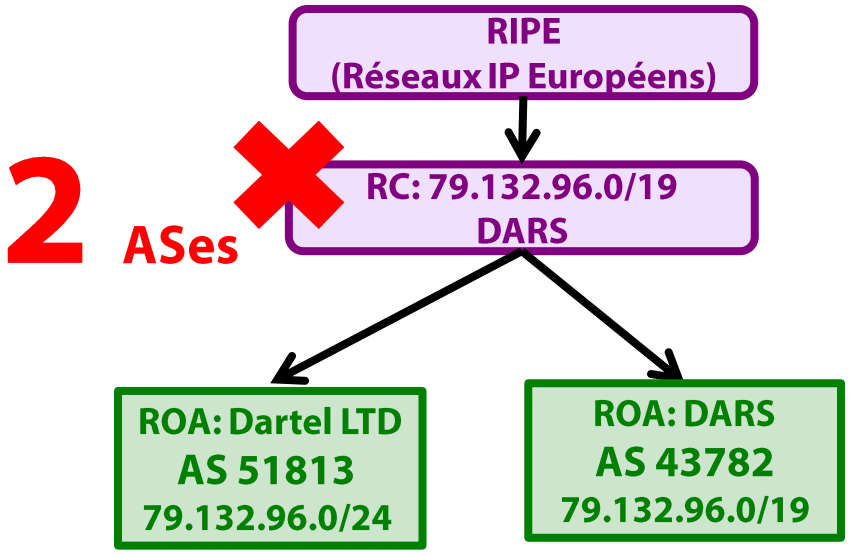
From the Consent of the Routed: Improving the Transparency of the RPKI.
Ethan Heilman, Danny Cooper, Leonid Reyzin, Sharon Goldberg
SIGCOMM'14, Chicago, IL. August 2014.

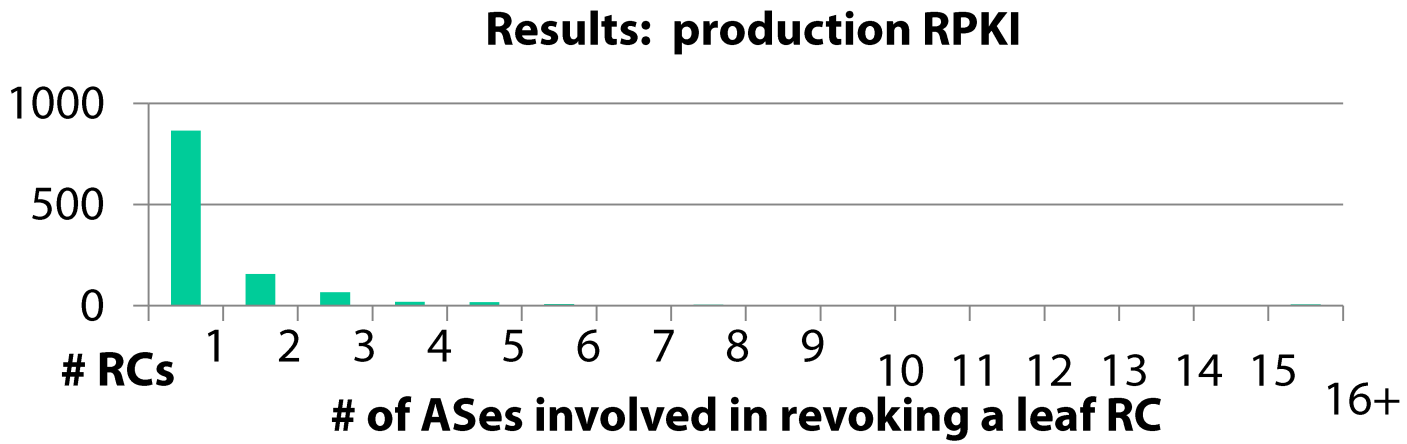
how many parties need to consent?

- How many ASes need to be involved when a leaf resource cert is revoked?

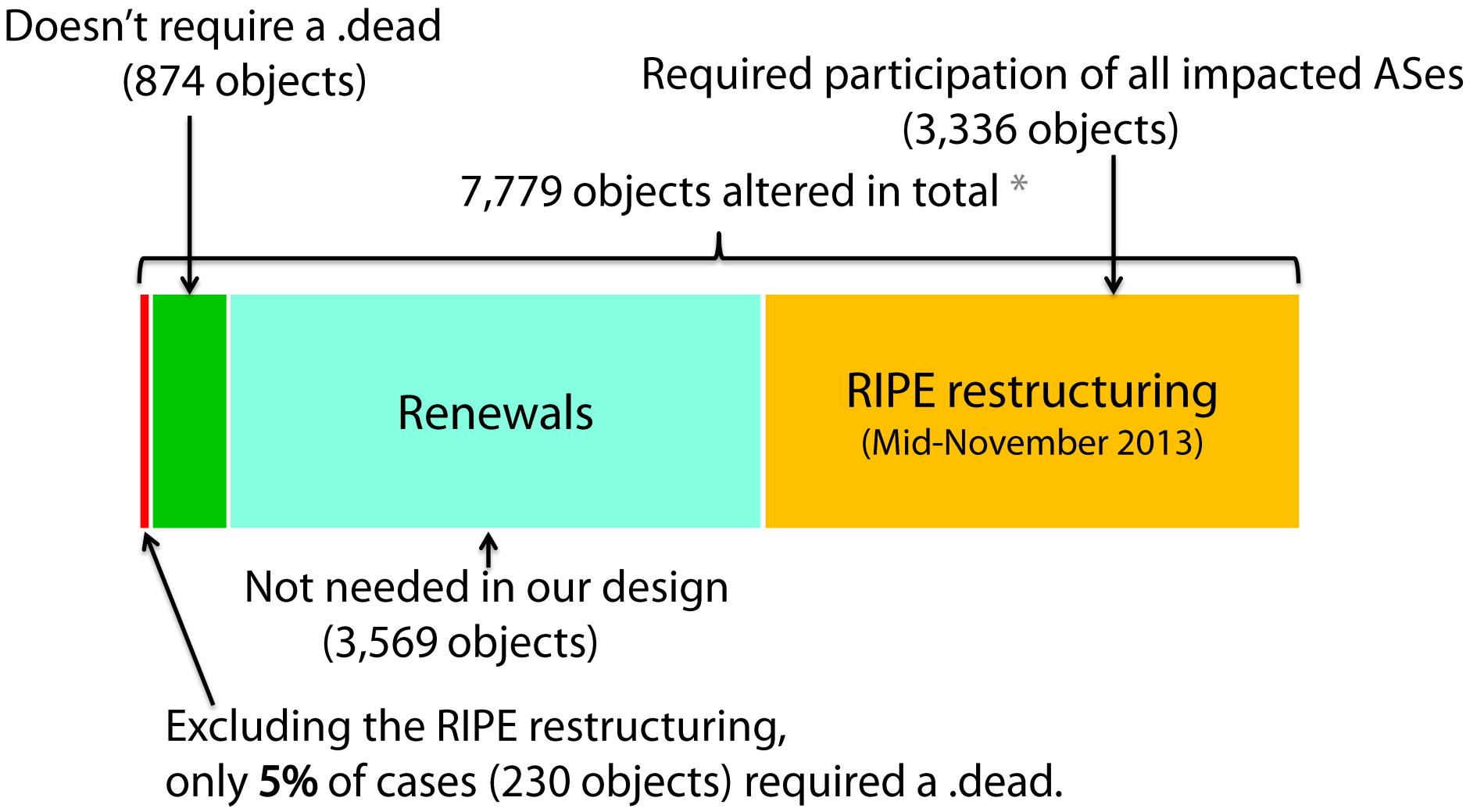
- Production RPKI
  - average **1.5** ASes / leaf RC

- Model fully-deployed RPKI
  - average **1.6** ASes / leaf RC
  - **99.3%** need **<10** ASes / leaf RC
  - **0.02%** need **>100** ASes / leaf RC





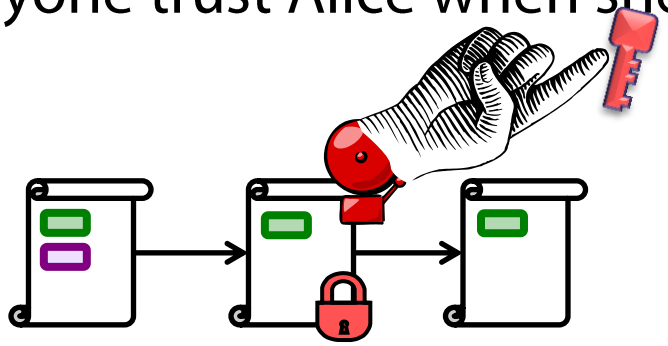
How often does would the RPKI need .deads?



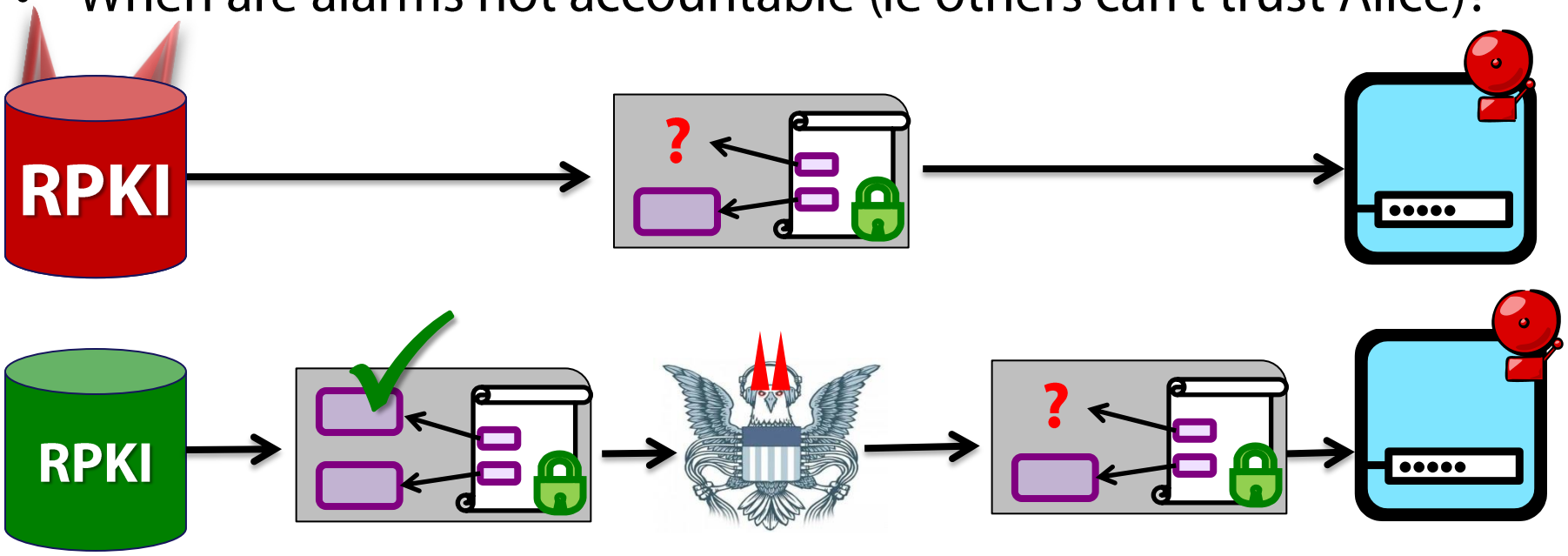
* all data from a ~3 month trace of the taken RPKI 2013/10/23 to 2014/01/21

Blaming authorities with accountable alarms.

- Why should anyone trust Alice when she raises an alarm?

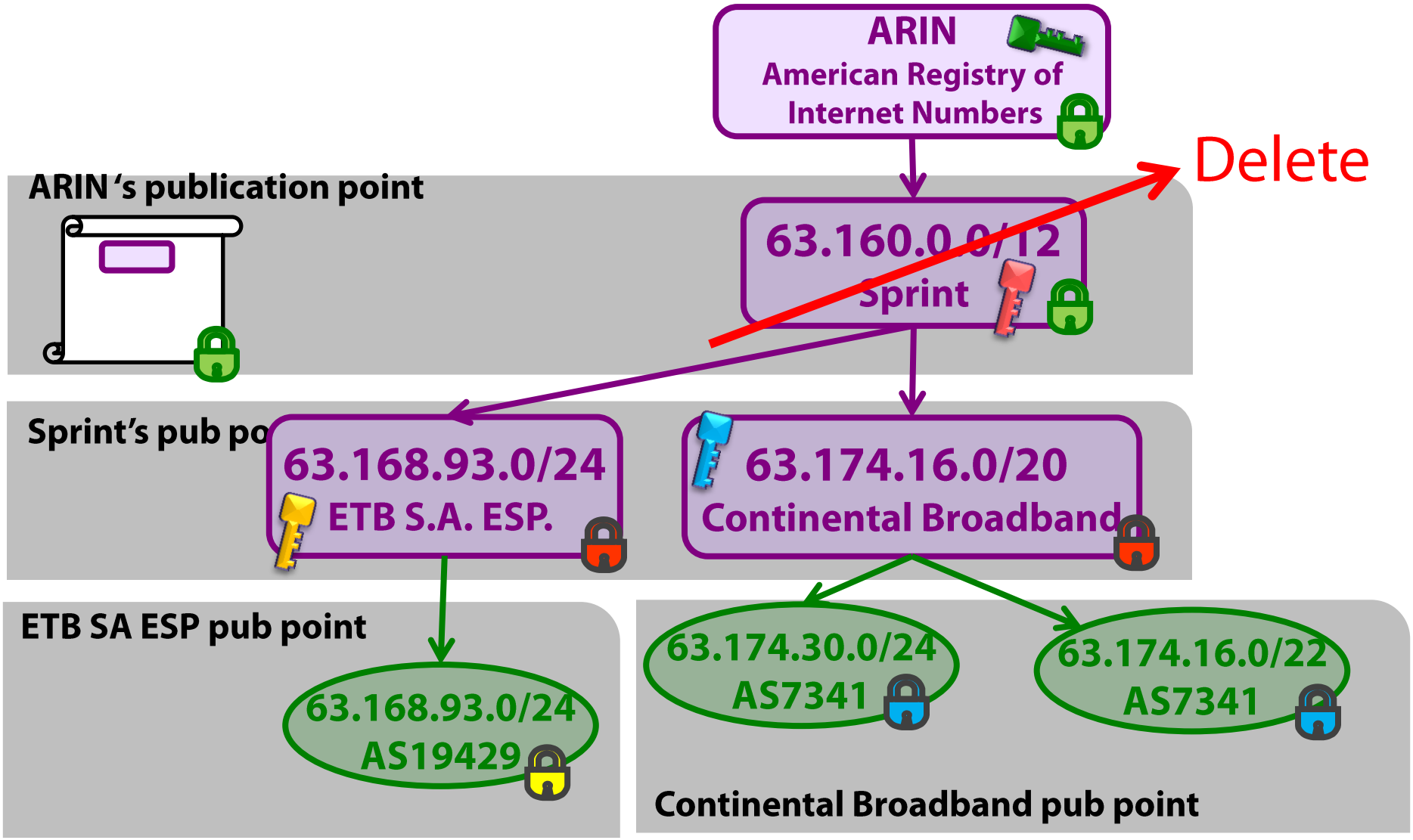


- When are alarms not accountable (ie others can't trust Alice)?

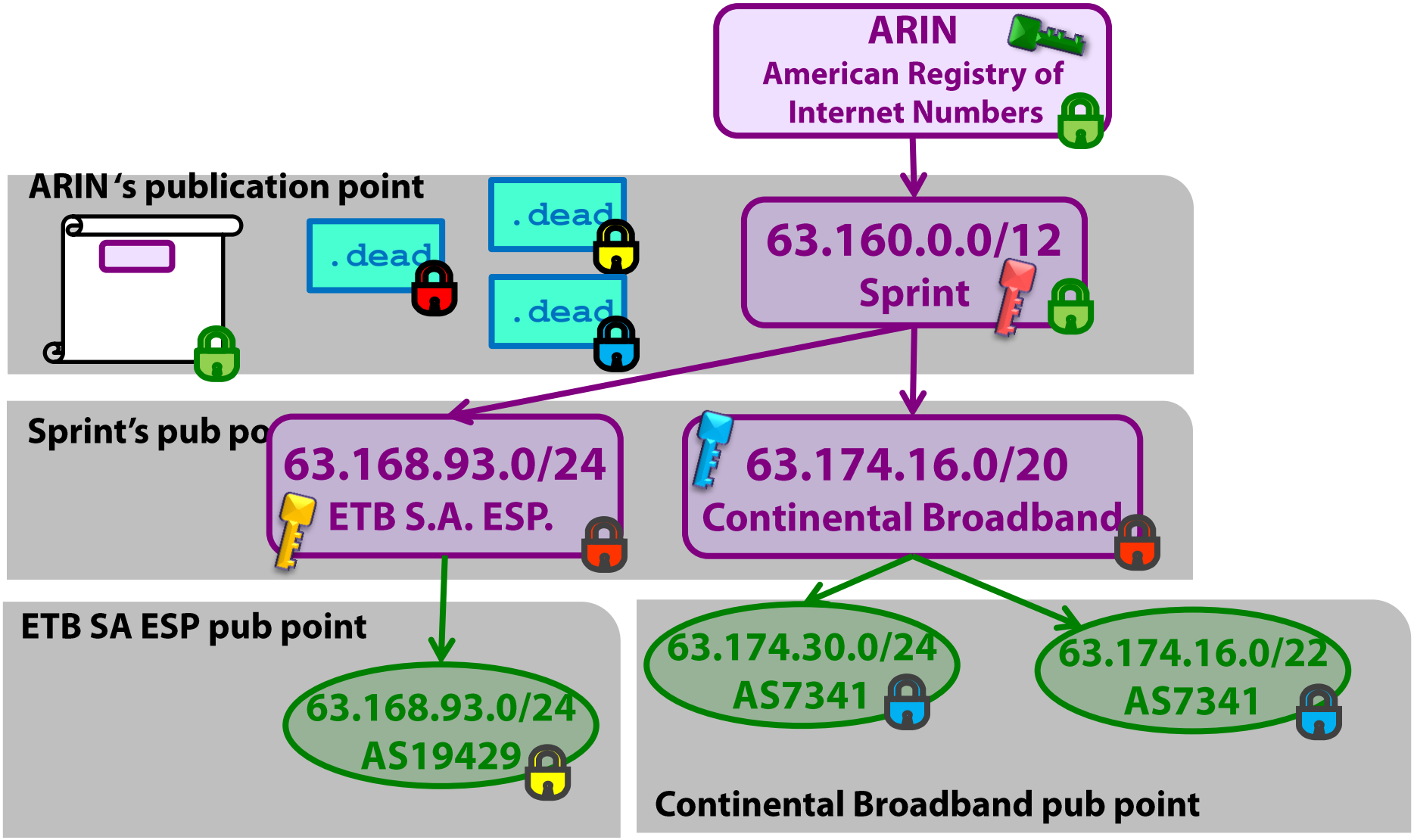


Alarms are accountable in every circumstance other than missing information.

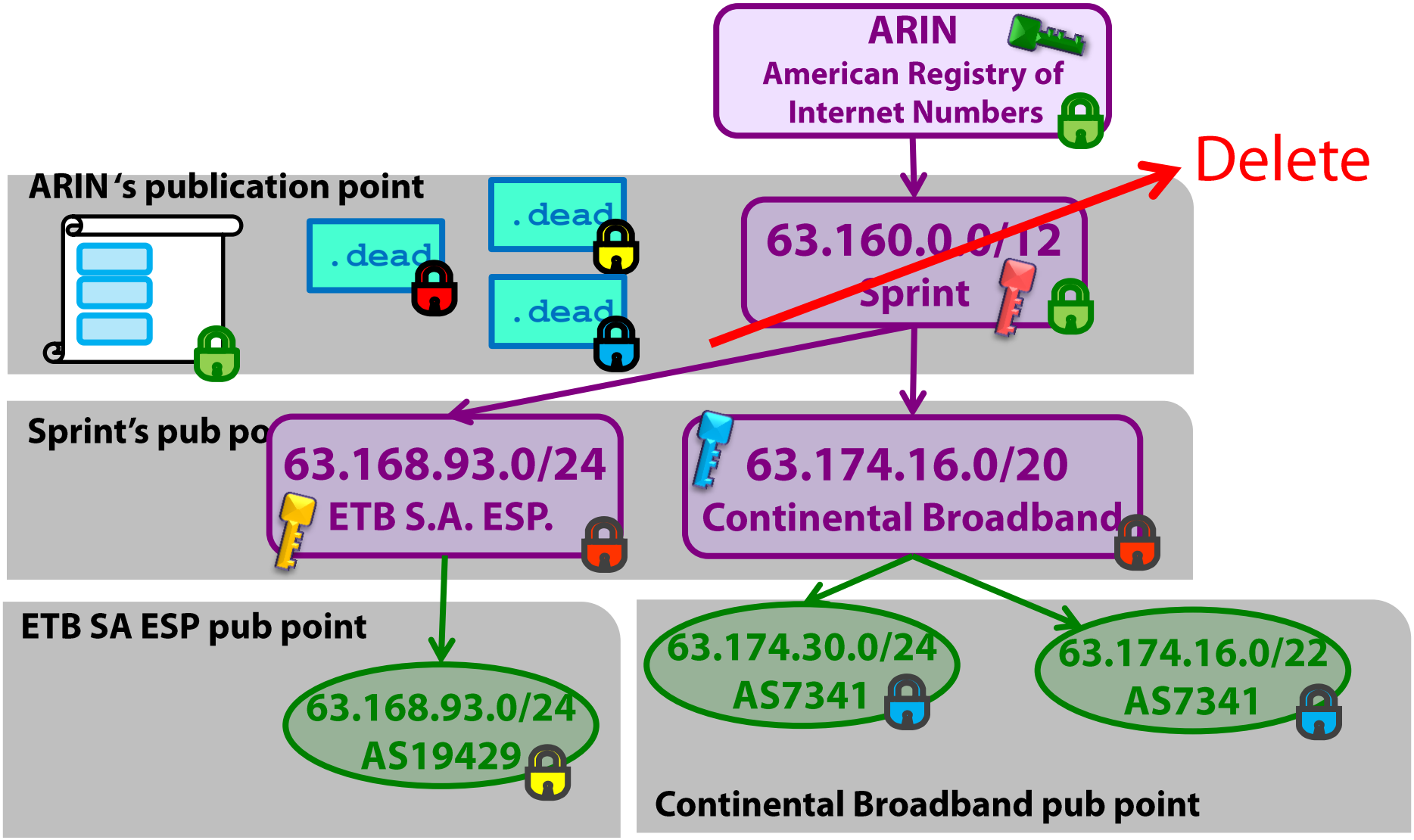
consent in a deep hierarchy: deletion



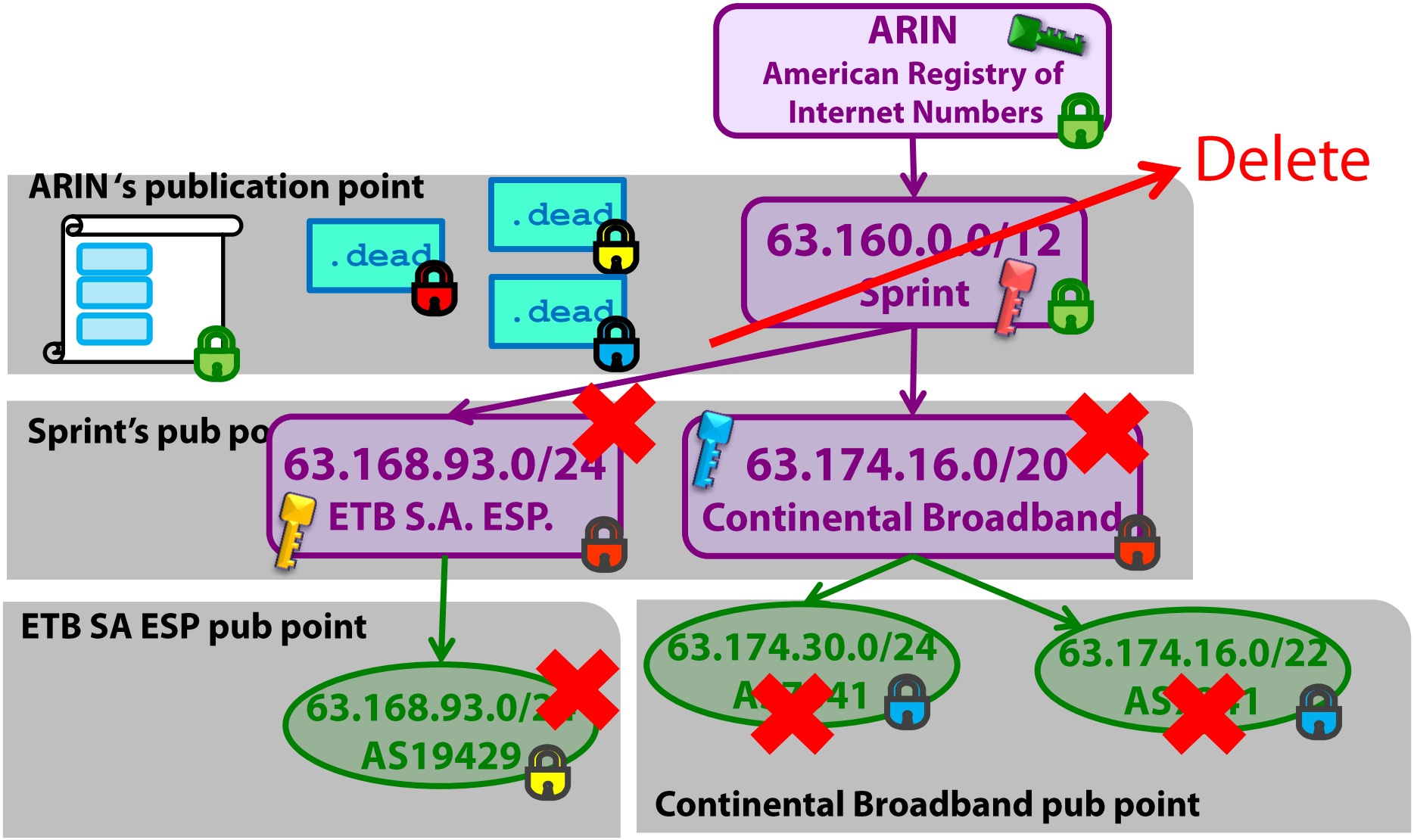
consent in a deep hierarchy: deletion



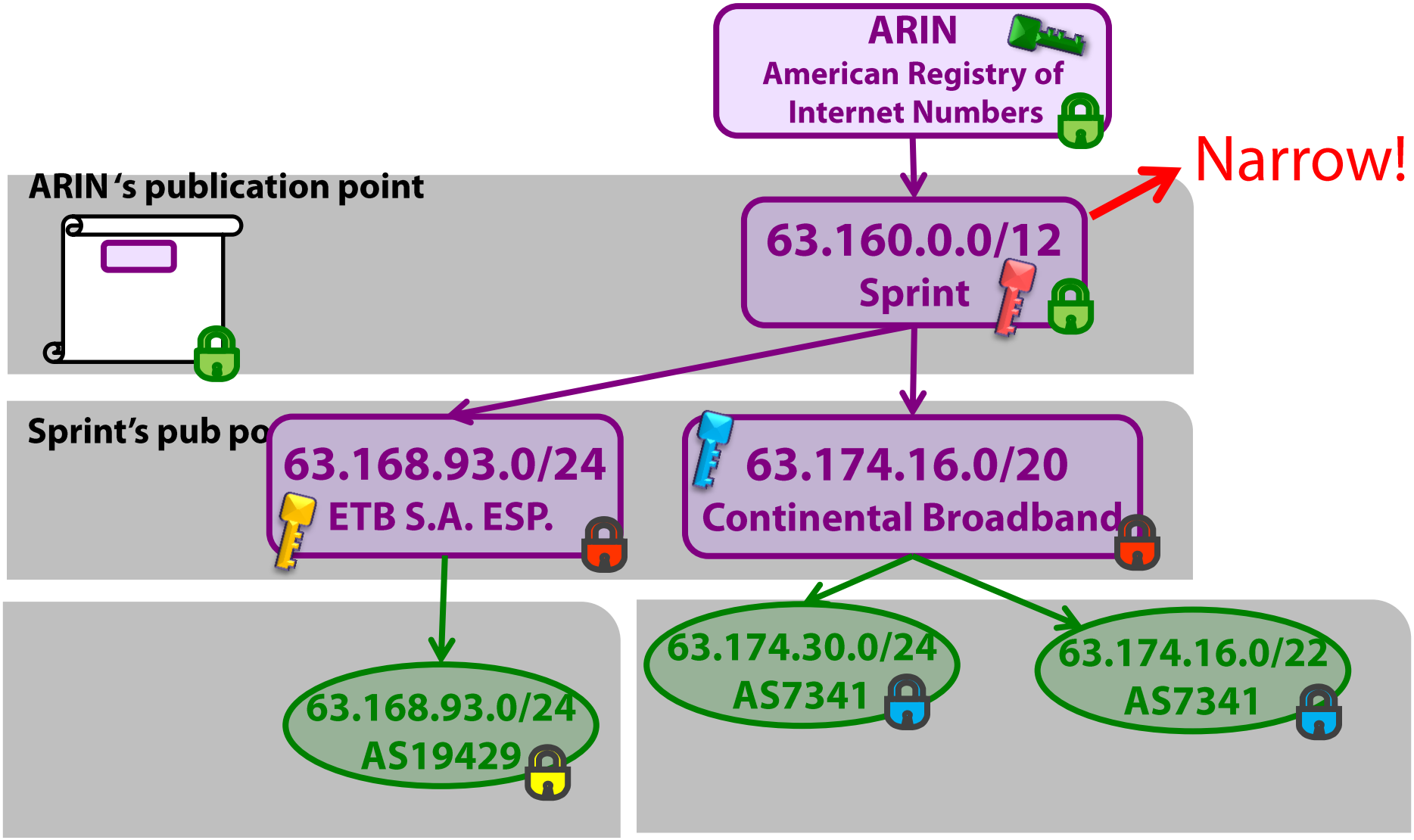
consent in a deep hierarchy: deletion



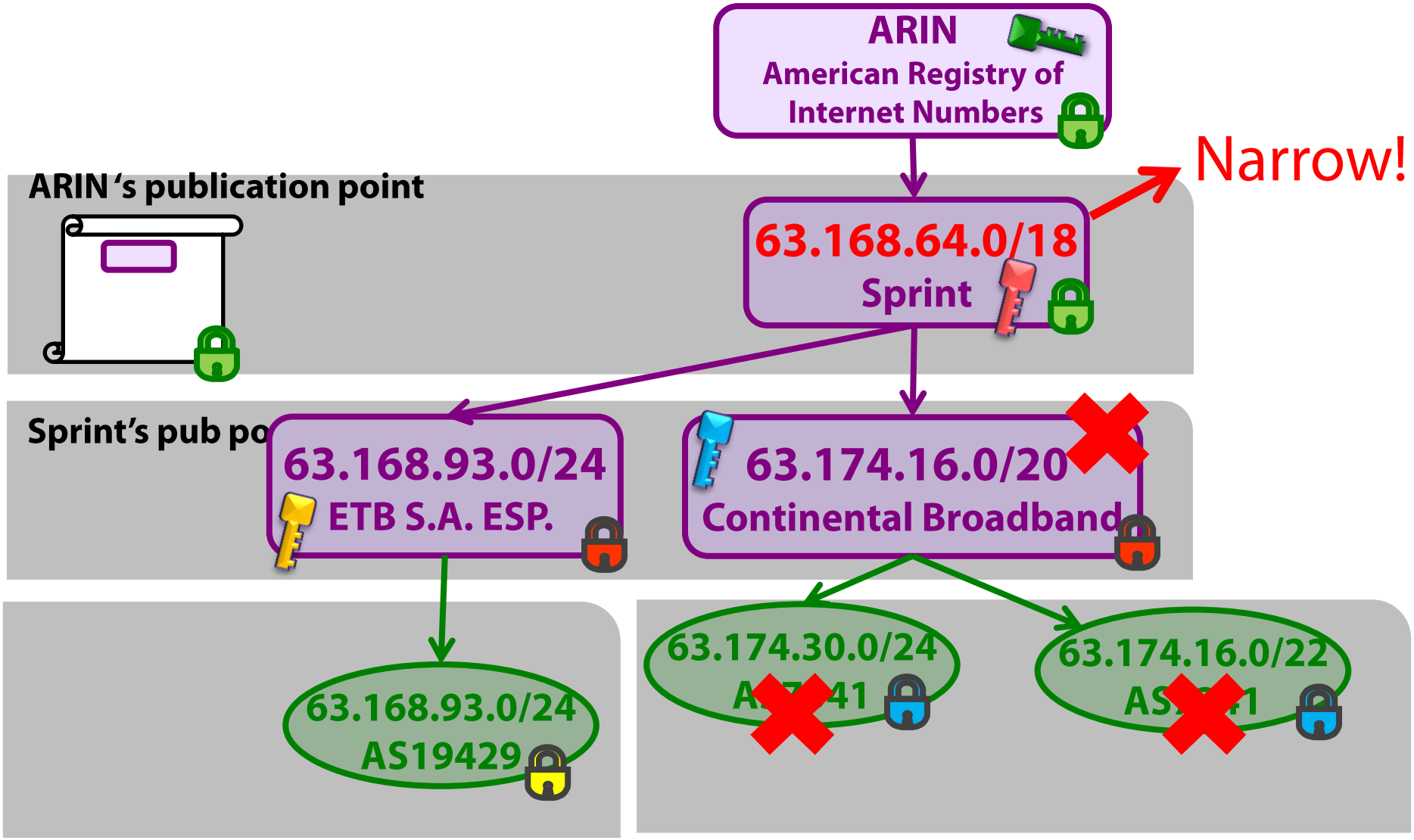
consent in a deep hierarchy: deletion



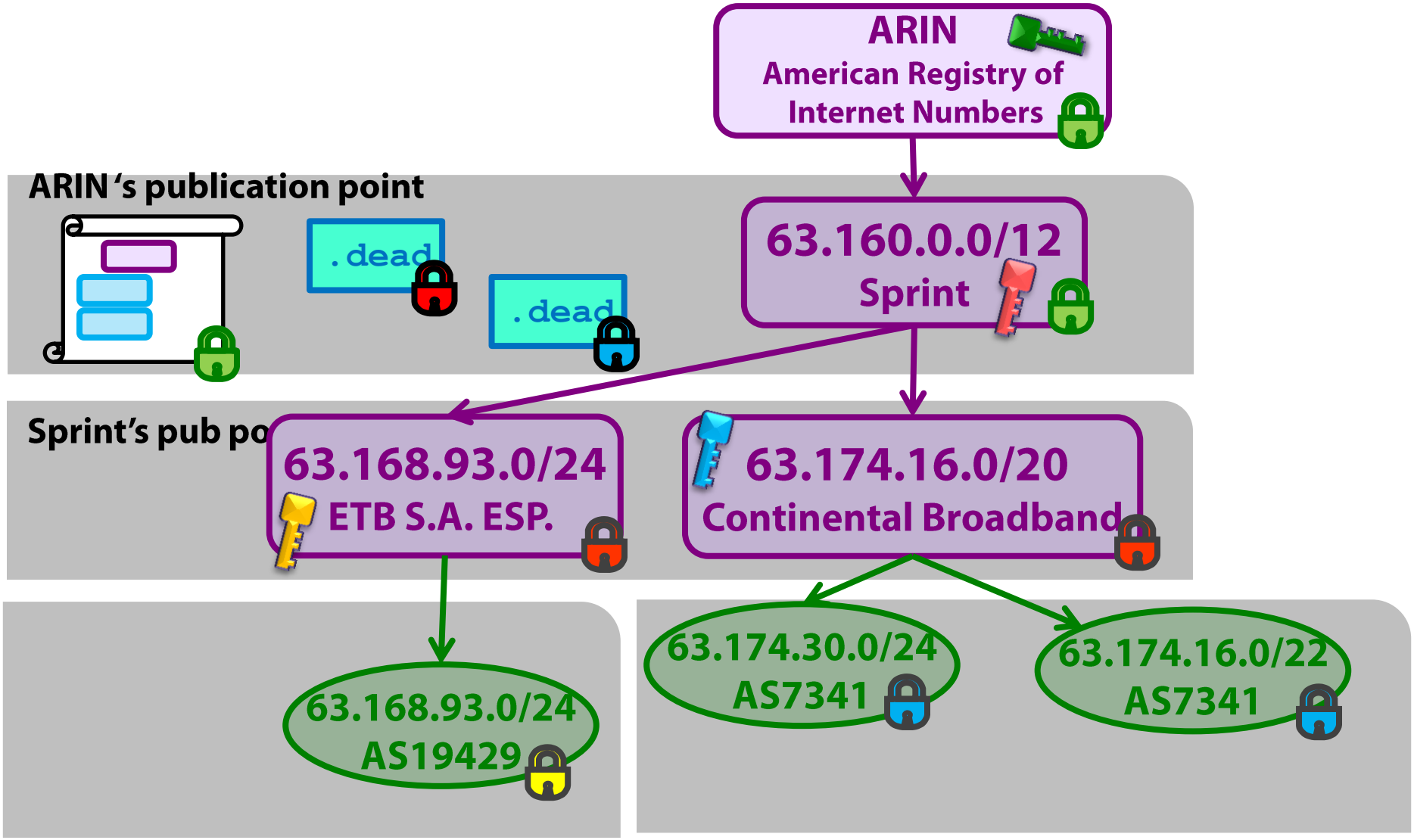
consent in a deep hierarchy: "address block narrowing"



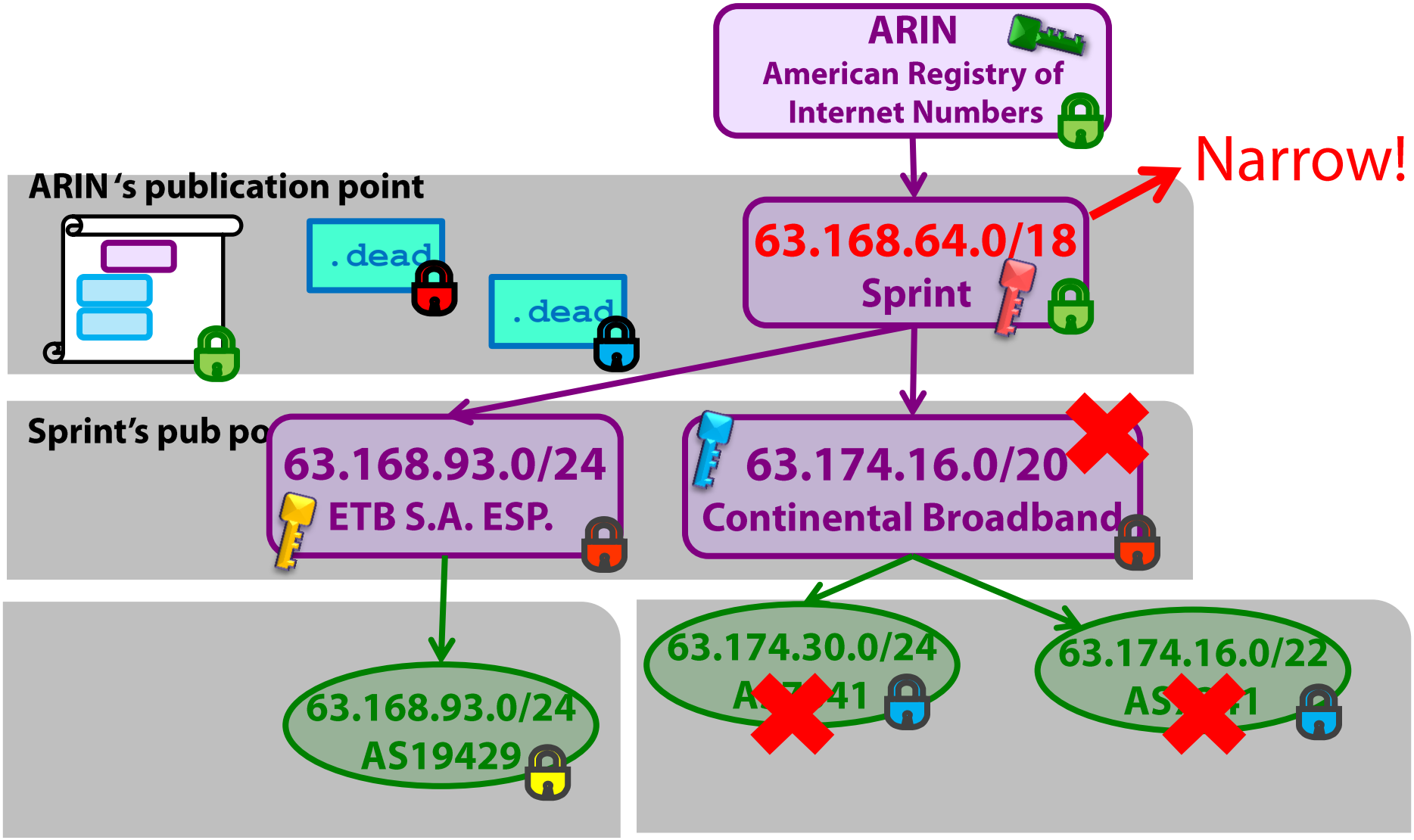
consent in a deep hierarchy: "address block narrowing"



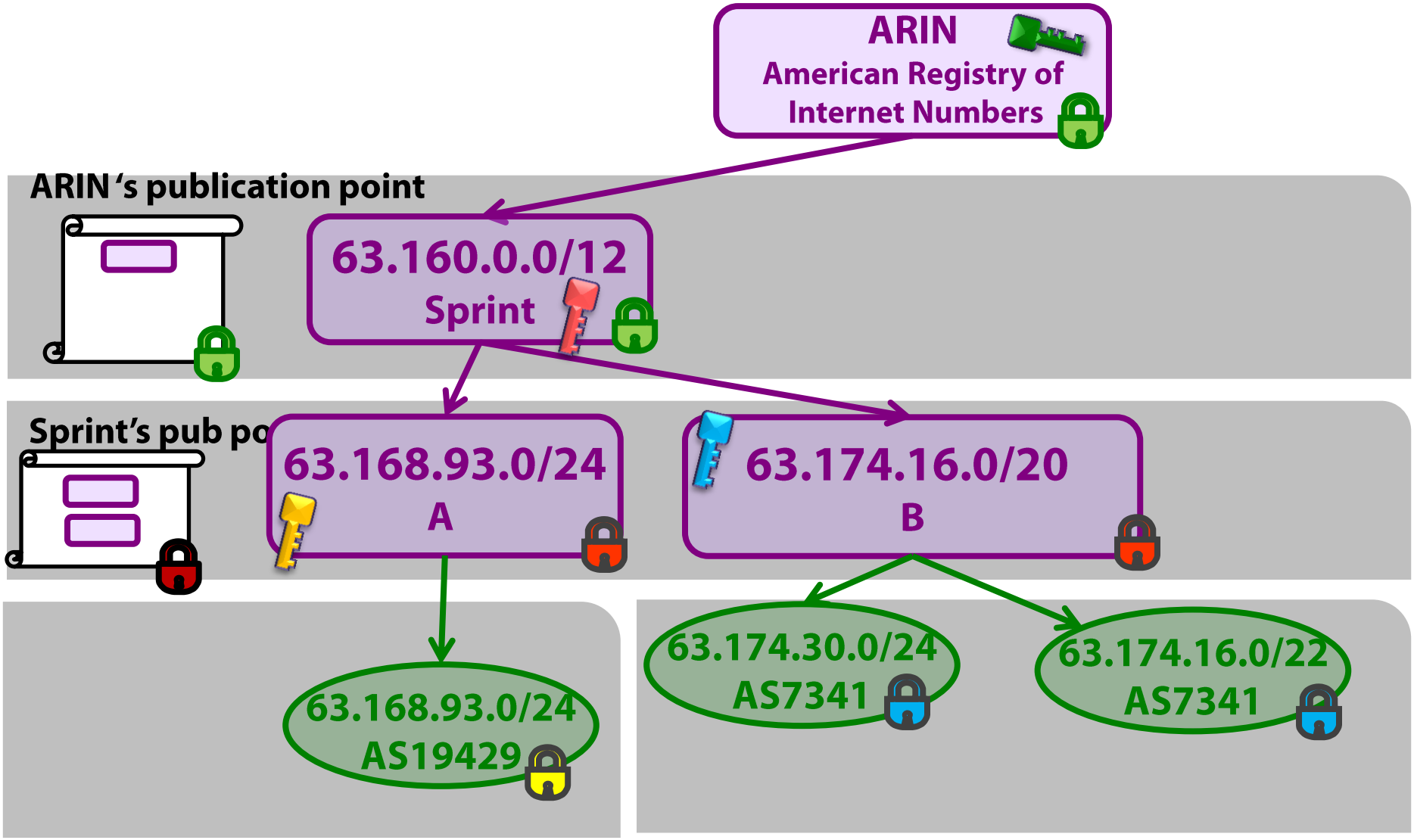
consent in a deep hierarchy: "address block narrowing"



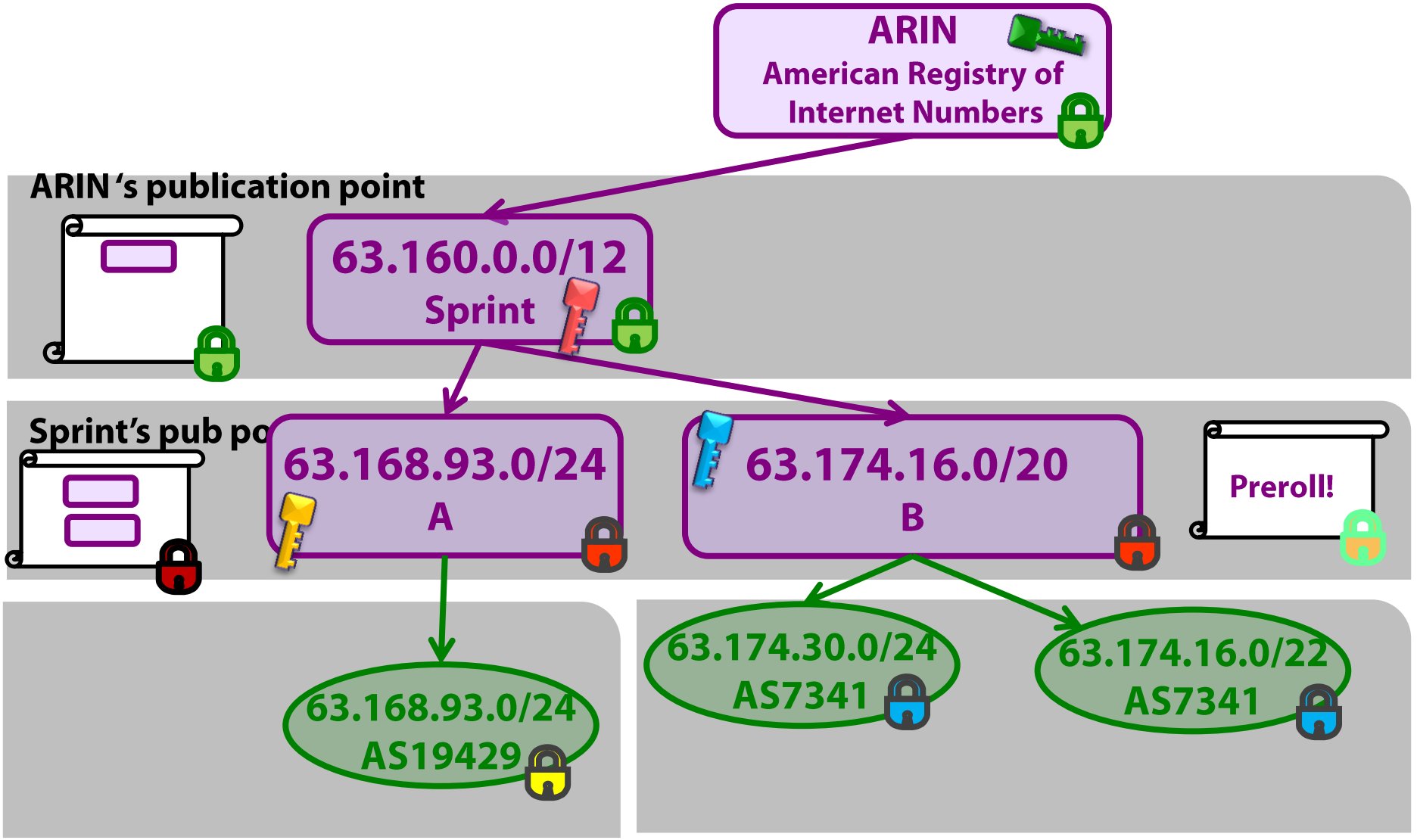
consent in a deep hierarchy: "address block narrowing"



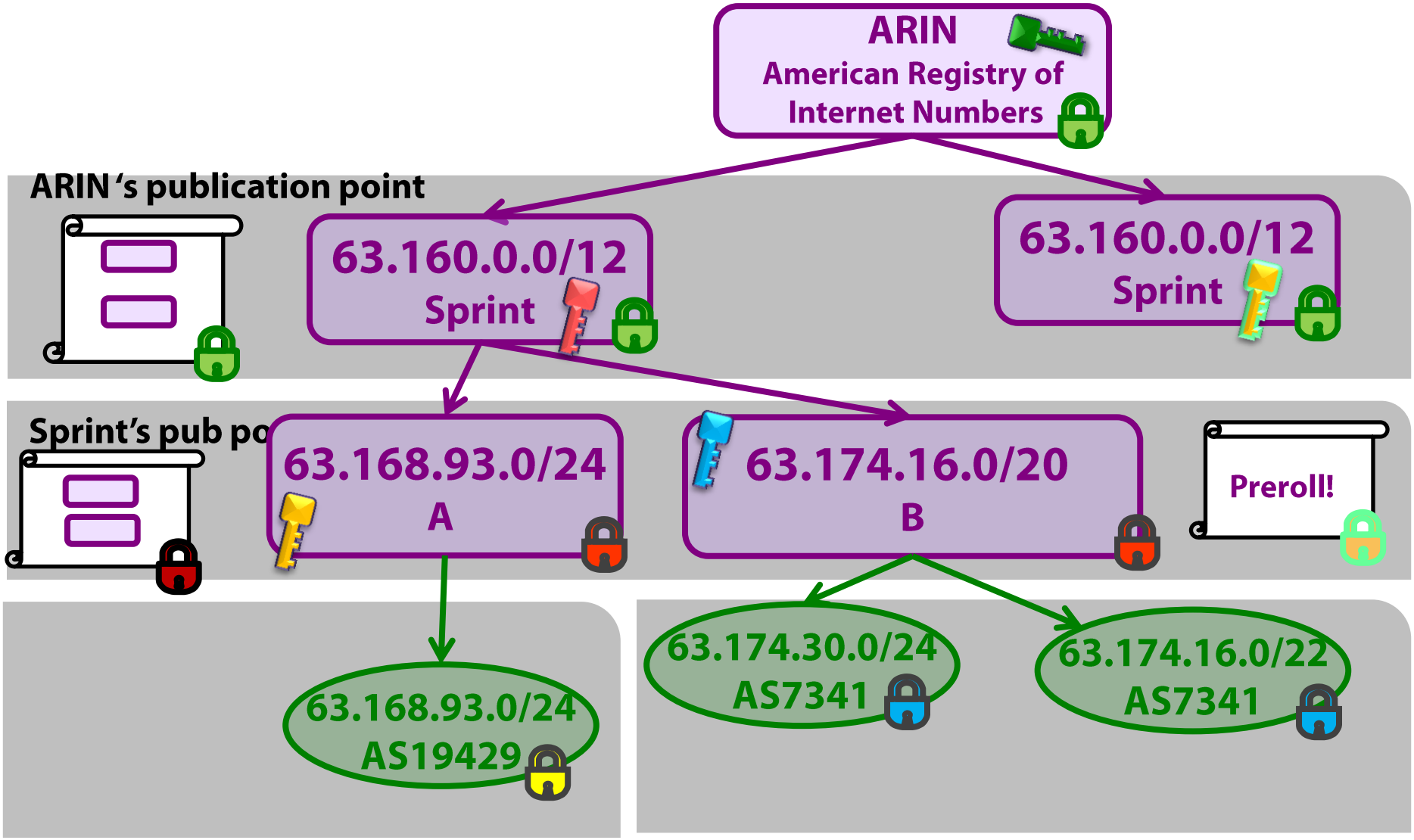
key rollover



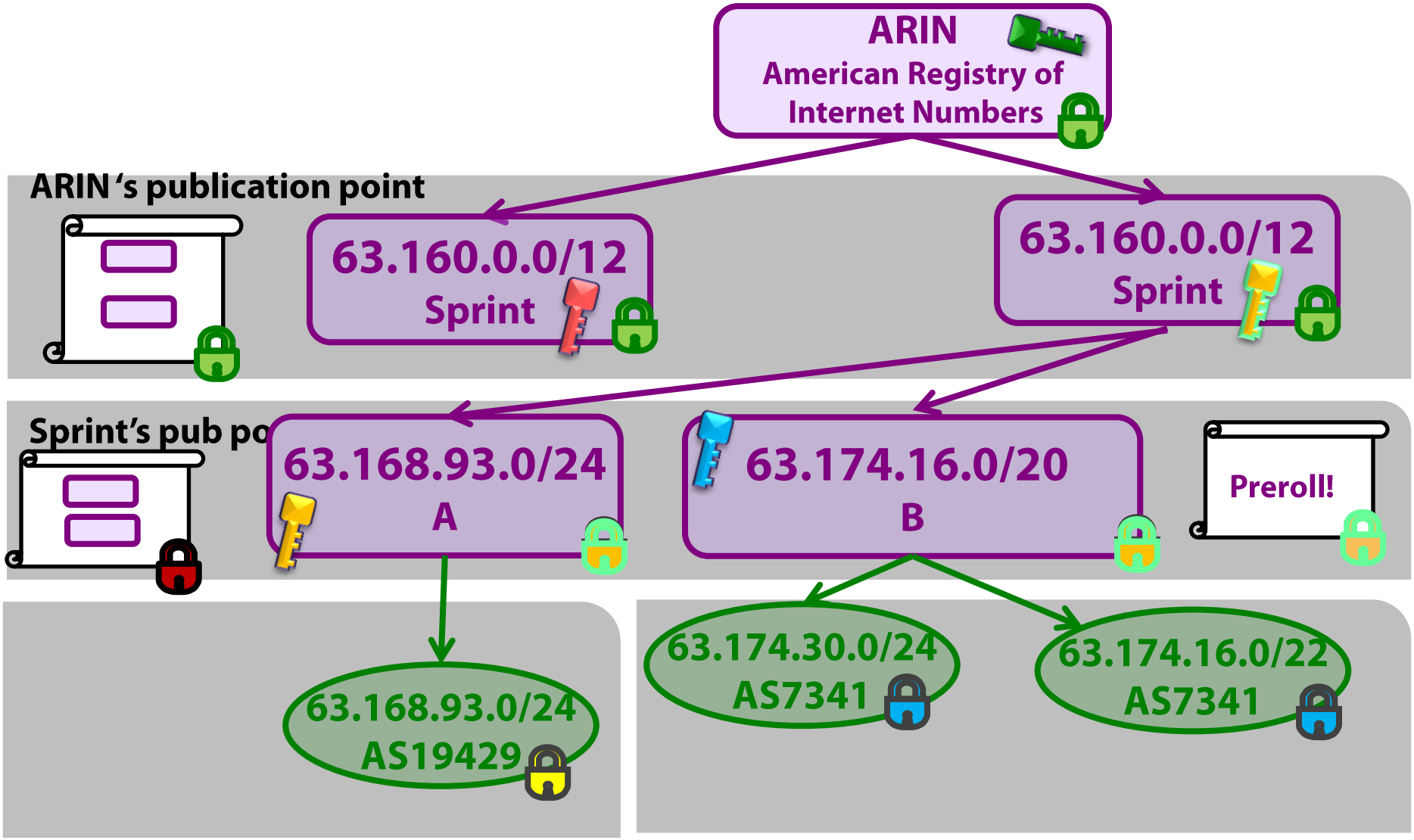
key rollover (step 0



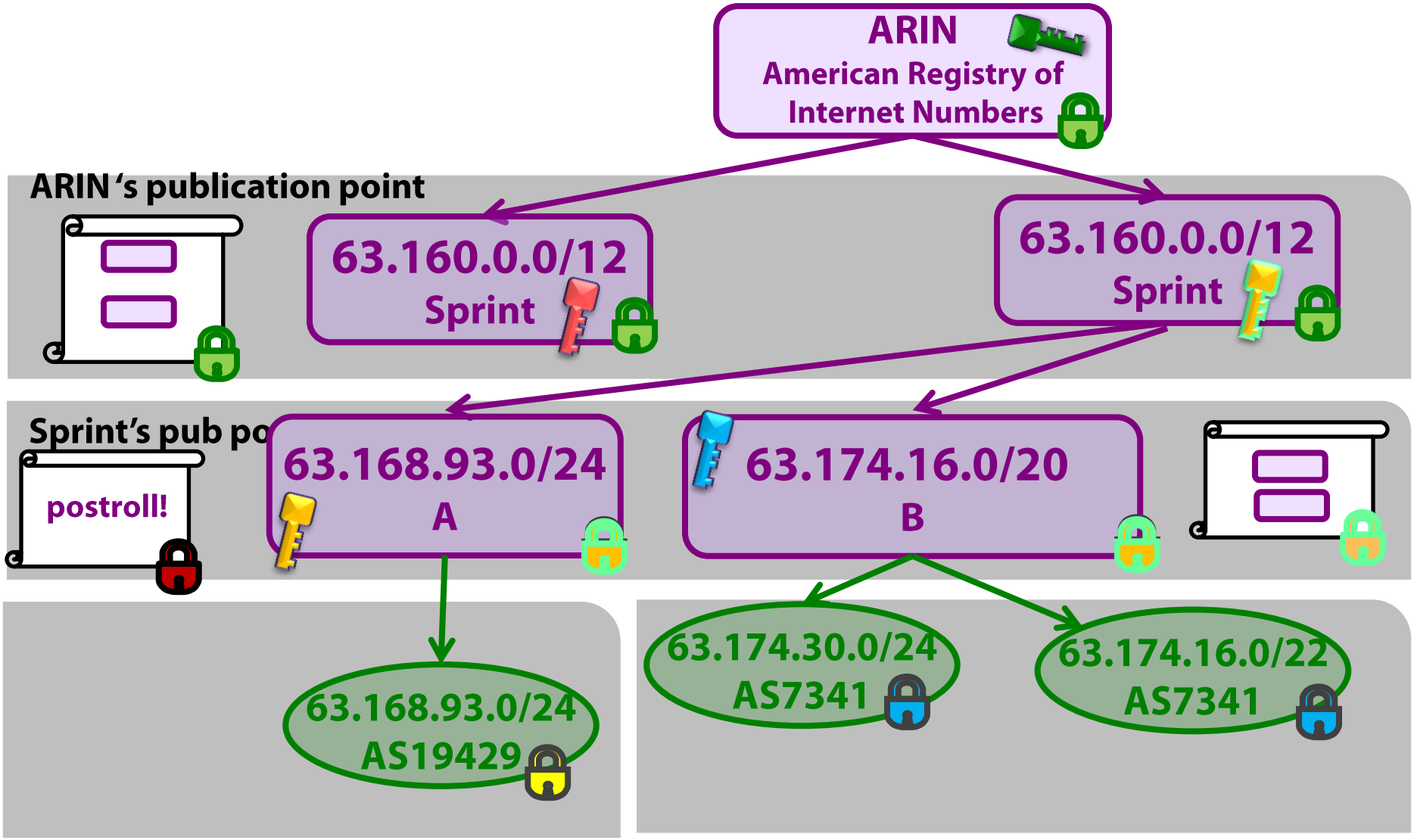
key rollover (step 1)



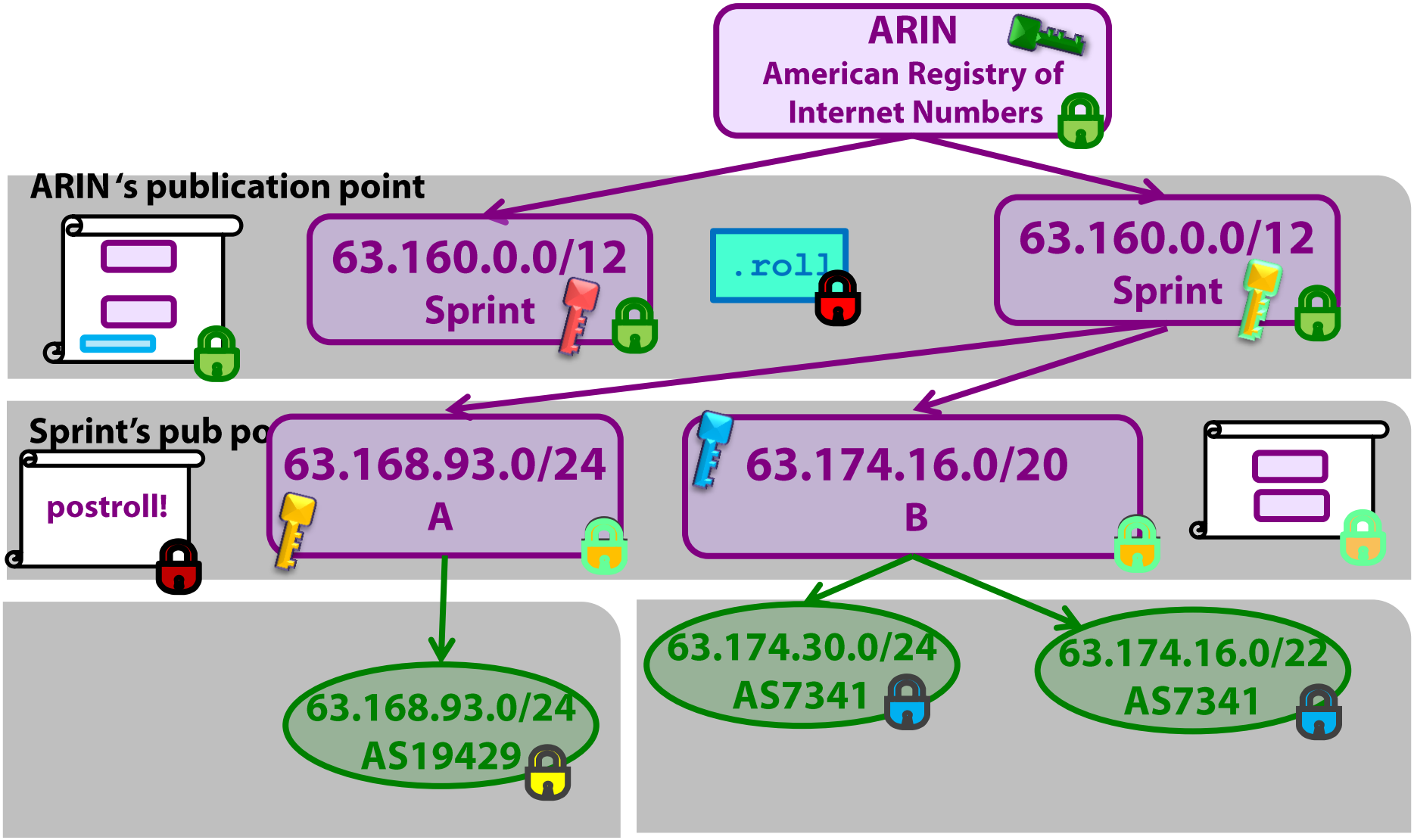
key rollover (step 2)



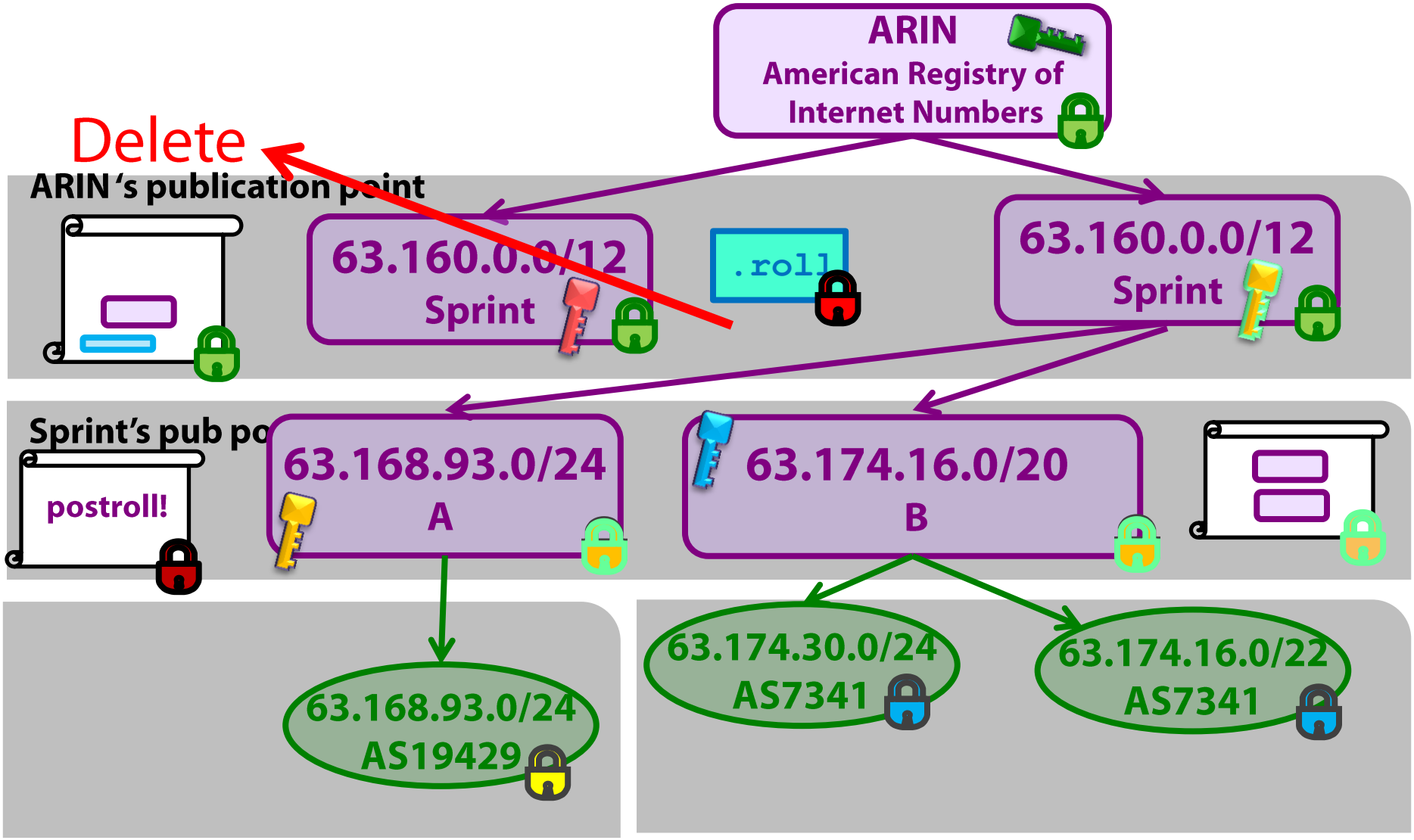
key rollover (step 2)



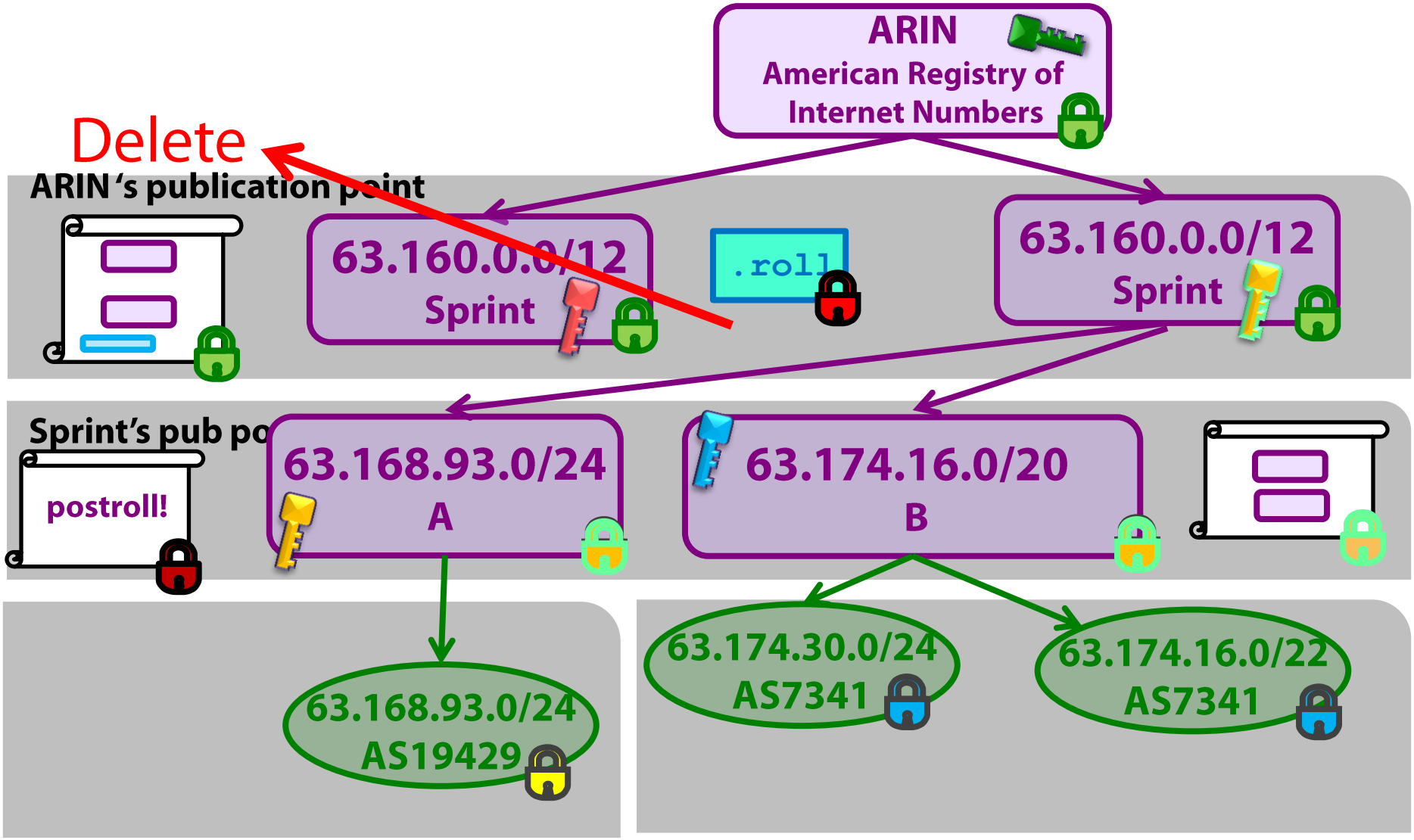
key rollover (step 3)



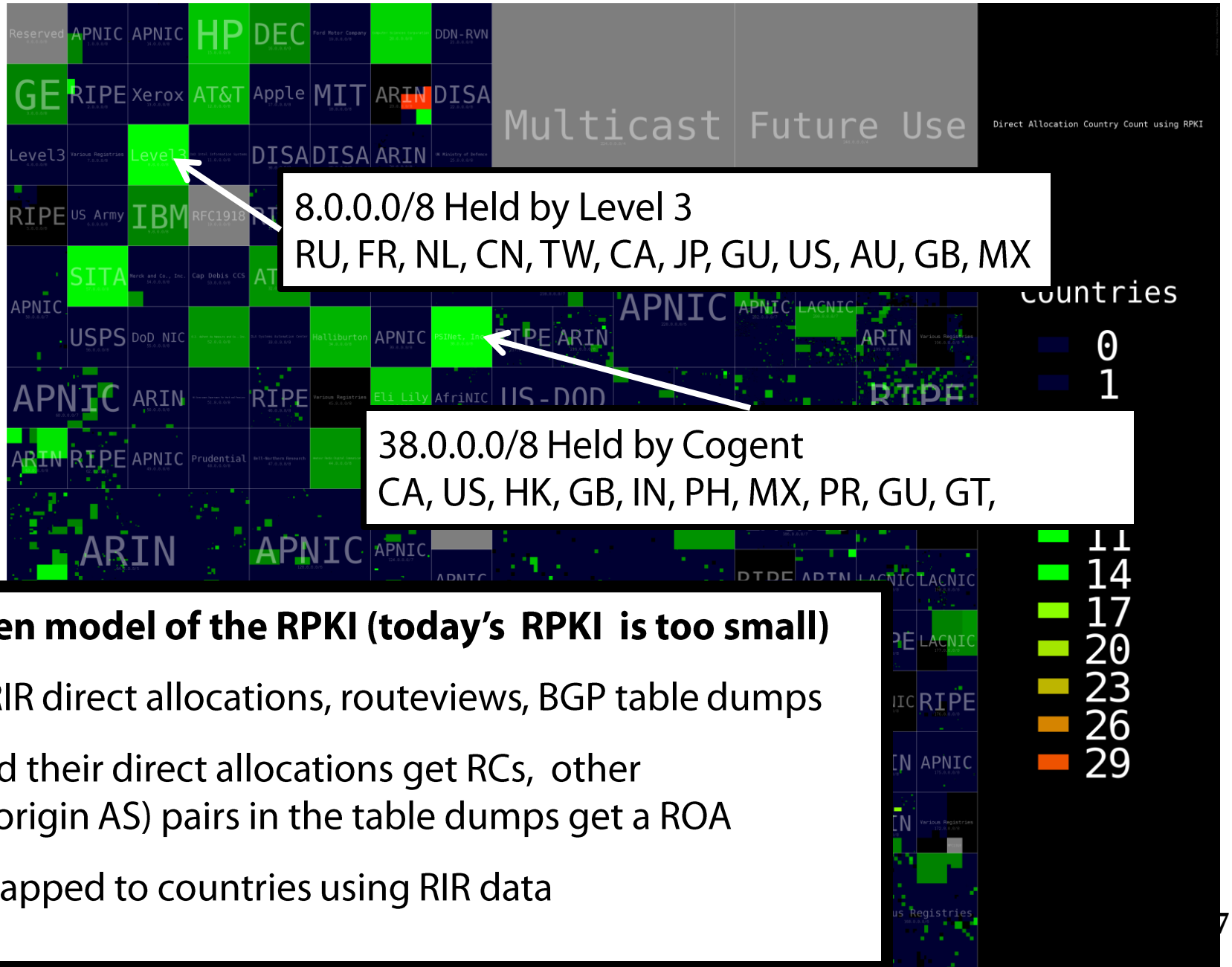
key rollover (step 3)



key rollover (step 3)



IPv4 address allocation does not reflect jurisdiction



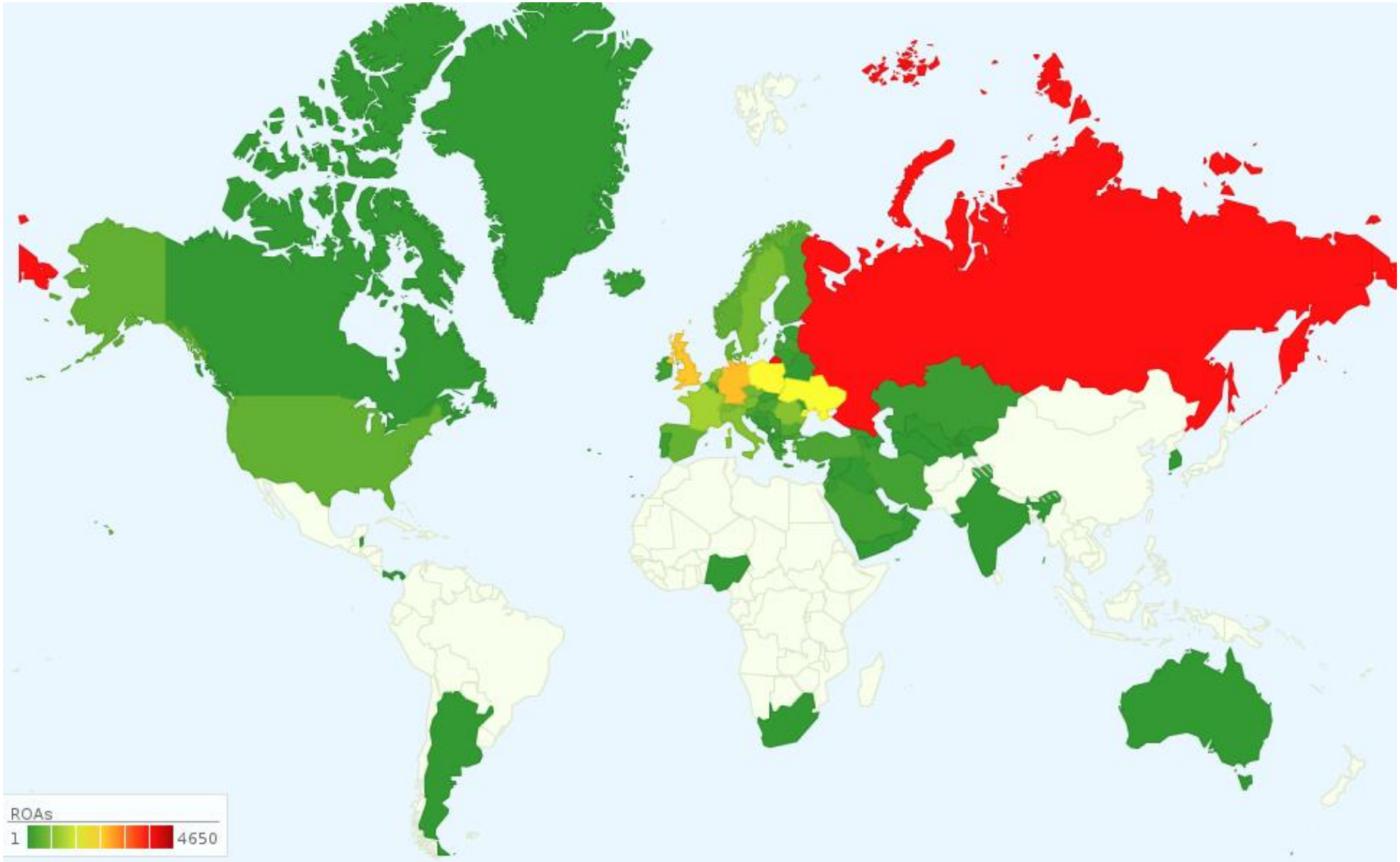
Data-driven model of the RPKI (today's RPKI is too small)

- ✧ Using RIR direct allocations, routeviews, BGP table dumps
- ✧ RIRs and their direct allocations get RCs, other (prefix,origin AS) pairs in the table dumps get a ROA
- ✧ ASes mapped to countries using RIR data

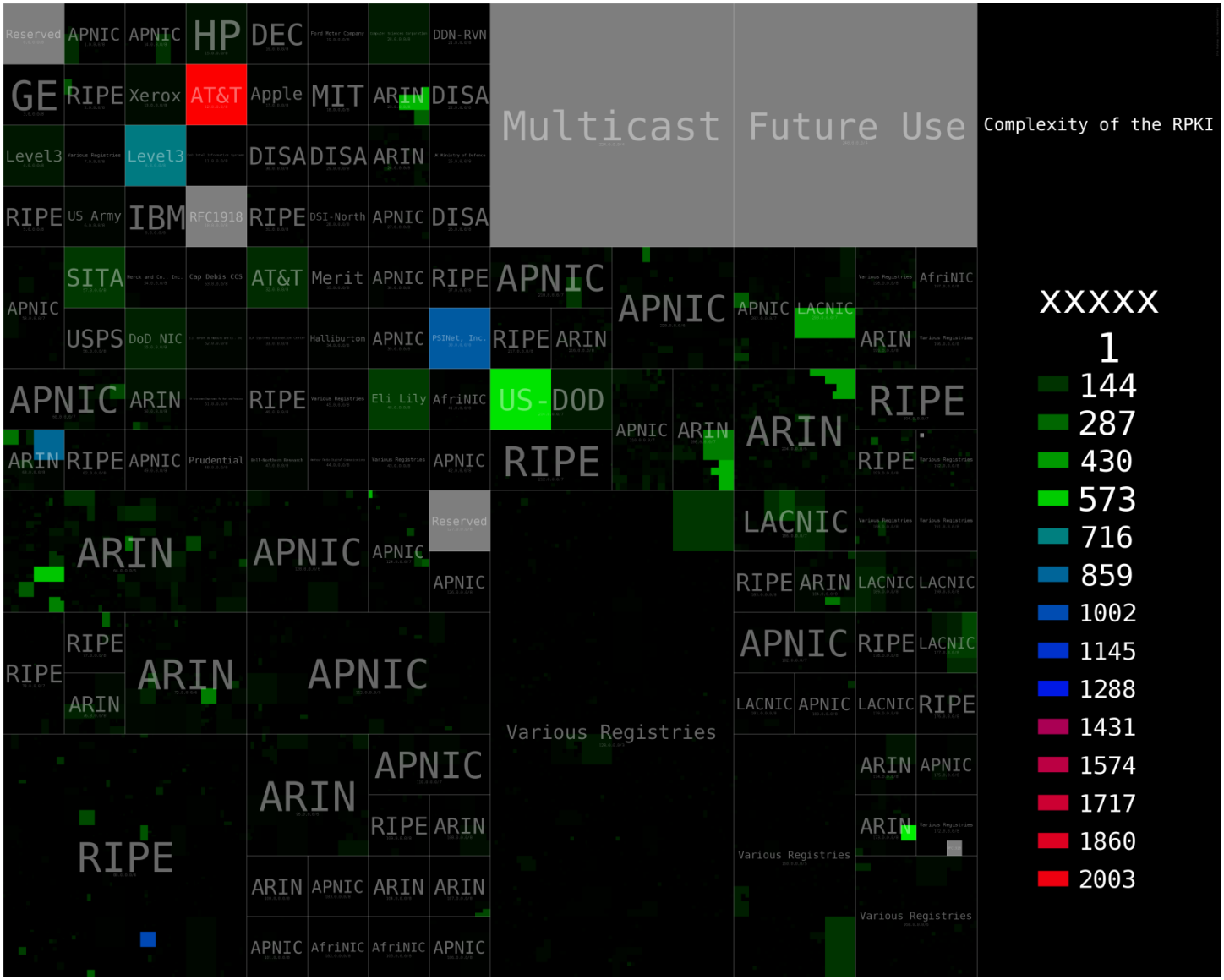
Countries

RC	Holder	Countries
8.0.0.0/8	Level3	RU, FR, NL, CN, TW, CA, JP, GU, US, AU, GB, MX
38.0.0.0/8	Cogent	PR, GU, GT, CA, US, HK, GB, IN, PH, MX
65.192.0.0/11	Verizon	CO, IT, US, AN, AS, GB, BS, EU, SG
208.0.0.0/11	Sprint	DM, CO, BB, VI, CA, BO, US, AS, EC, KY, ES
63.160.0.0/12	Sprint	PR, FR, CO, BB, CA, YE, US, AN, HN
93.170.0.0/15	ALFA Tel.	CZ, RU, BG, NL, US, LU, GB, KZ, UA, BY
64.86.0.0/16	Tata Comm.	GU, CO, CA, MH, US, HN, PH, ZW
206.48.0.0/16	France Tel.	FR, DM, CO, AW, CL, BR, BS, EU, KY
216.72.0.0/16	France Tel.	FR, GT, CO, VE, CL, HN, IL, BR, BS, EU
209.88.0.0/16	France Tel.	FR, DM, AW, CL, NA, IL, BR, BS, EU, ZW
192.71.0.0/16	Resilans	DK, NO, DE, US, CZ, GB, IN, EU, SE
63.245.0.0/17	Columbus	US, PR, NI, GT, CO, AN, GD, HN, BS, MX
61.28.192.0/19	Servcorp	FR, AE, CA, JP, US, NZ, AU, GB, TH, SG

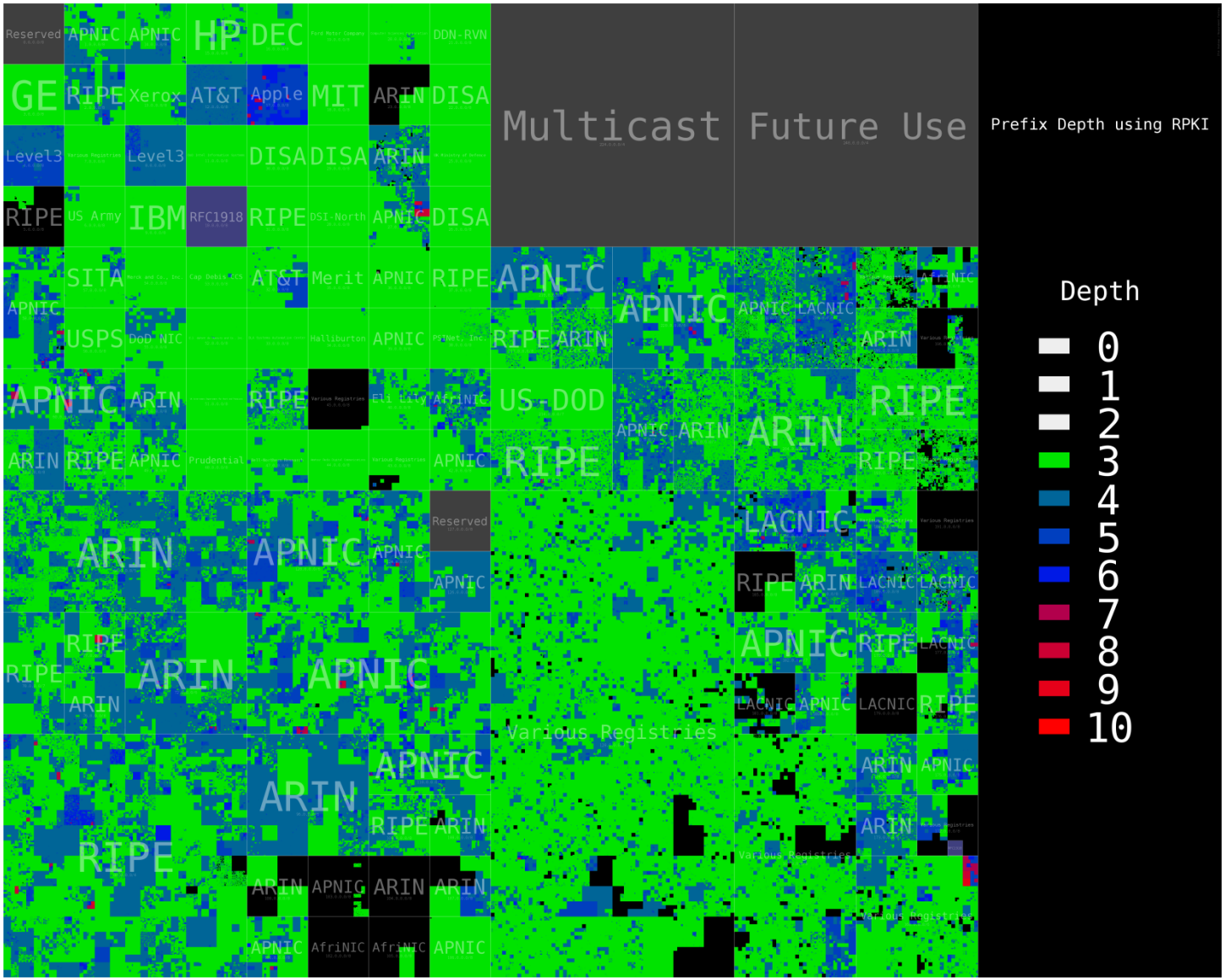
Countries covered by RIPE



Number of ROAs issued by each direct allocation



Depth of the RPKI



Depth	ROAs
3	118,028
4	108,043
5	10,863
6	293
7	9