IETF Hackathon: Measurement & Analysis for Protocols Research Group (MAPRG)

IETF 105 Montreal
20-21 July 2019

Report updated: 26 July
David Plonka <plonka@akamai.com>
We describe a method of truncation and/or aggregation-based anonymization, *i.e.*, produce *anonymity sets* of WWW client addresses.

*e.g.*, for correlating web analytics with network topology, routing, service providers, and geographic or topological locations.
Background on kIP: IPv6 Address Activity Matrix: Identity Assignment

0 012345678901234567890123

000100011112332321122100 \(\Rightarrow\) 3 simultaneous IIDs, maximum

2001:db8::/64; Temporary SLAAC: 100%-- '!!!!!!' "fenceposts"

Legend:

\(!\) = infer /64 prefix assigned at the "fencepost" moments between intervals

IETF Hackathon - MAPRG
Background Example: Synthesize anonymous $k=2$ aggregates ($w=1d, i=1h$)
Hackathon Plan

• Problem:
  • **Related draft:** kIP: a Measured Approach to IPv6 Address Anonymization
    https://www.youtube.com/watch?v=qYtaKuzXaiM#t=59m55s

  • **Specific problem to solve:** Implement a modified PATRICIA trie or base-2 radix tree for “longest prefix match” making feasible for the tens to hundreds of billions of active IPv6 addresses used on the web today.

• **To solve it:** enhance aguri_tree to make portions of the tree *immutable.*
  https://www.iijlab.net/~kjc/software/#aguri

 This allows partitioning of the problem for map-reduce/cluster operation, splitting the active IP addresses into manageable subsets (files) to produce intermediate results that are subsequently combined using prior mode (non-immutable nodes).
Hackathon Plan

• Problem:

  Related draft: kIP: a Measured Approach to IPv6 Address Anonymization
  
  [https://www.youtube.com/watch?v=qYtaKuzXaiM#t=59m55s](https://www.youtube.com/watch?v=qYtaKuzXaiM#t=59m55s)

  Want a reference implementation for address aggregation, Internet-wide, esp. handling the freedom and variety of numbering (subnetting) options afforded by IPv6.

  This allows partitioning of the problem for map-reduce/cluster operation, splitting the active IP addresses into manageable subsets (files) to produce intermediate results that are subsequently combined using prior mode (non-immutable nodes).

• Specific problem to solve:

  Implement a modified PATRICIA trie or base-2 radix tree for "longest prefix match" making feasible for the tens to hundreds of billions of active IPv6 addresses used on the web today.

  To solve it: enhance [aguri_tree](https://www.iijlab.net/~kjc/software/#aguri) to make portions of the tree immutable.

  This allows partitioning of the problem for map-reduce/cluster operation, splitting the active IP addresses into manageable subsets (files) to produce intermediate results that are subsequently combined using prior mode (non-immutable nodes).
Hackathon Plan

• Problem:

  Related draft: kIP: a Measured Approach to IPv6 Address Anonymization

  https://www.youtube.com/watch?v=qYtaKuzXaiM#t=59m55s

  Specific problem to solve: Implement a modified PATRICIA trie or base-2 radix tree for "longest prefix match" making feasible for the tens to hundreds of billions of active IPv6 addresses used on the web today.

  To solve it: enhance aguri_tree to make portions of the tree immutable. This allows partitioning of the problem for map-reduce/cluster operation, splitting the active IP addresses into manageable subsets (files) to produce intermediate results that are subsequently combined using prior mode (non-immutable nodes).

Applications include address anonymization and creating anonymity sets, e.g., homogenous client populations for use, e.g., in match-making for high performance content delivery.
What got done

• <What you achieved? (key results)>
  • <New ideas - what team agreed on>
  • github pending: https://github.com/akamai

• What was novel?

  A PATRICIA or base-2 radix tree API permitting partitioning of aggregation problems for very large data sets, e.g., all active addresses, Internet-wide.
What got done

2001:db8:0:4000::/64
2001:db8:1001:400::/64
2001:db8:1001:a00::/64
2001:db8:1001:1300::/64
2001:db8:1001:1600::/64
2001:db8:1001:1800::/64
2001:db8:1001:2a00::/64
2001:db8:1001:3100::/64
2001:db8:1001:4500::/64
2001:db8:1001:4700::/64
2001:db8:1001:1000::/52
2001:db8:1001:1000::/53
What got done

$ ./agurify -v -p50 -k32 input_fragment_v6.txt ::/0

Aggregate to prefixes having at a minimum of k=32 simultaneously active addresses
This result is “wrong” because it aggregated beyond the bounds of this input file.
What got done

2001:db8:0:3fff:ffff:ffff:ffff:ffff added immutable...
2001:db8:1001:4701:: added immutable...
What got done

$ ./agurify -v -i -p50 -k32 input_fragment_v6.txt
20010db800003ffffff000000000000 added immutable...
20010db8100147000000000000000000 added immutable...

Produce an intermediate result, i.e., don’t aggregate to any prefix which could cover any address not appearing in this input file.
What got done

$ ./agurify -v -i -p50 -k32 input_fragment_v6.txt
20010db800003fffffffffffffffffffffff added immutable...
20010db8100147010000000000000000000 added immutable...
  2001:db8:0:4000::/64 335
  2001:db8:1001::/50 335
  2001:db8:1001:4500::/64 335
  2001:db8:1001:4700::/64 335

This is a useful intermediate result that can be processed, iteratively, toward the final result.
What we learned

• Lessons learned from this hackathon
  • Implemented IPv6 address “bignum” arithmetic rather than using GMP (GNU Multiple Precision Arithmetic Library)
  • New implementation/operation guidance?
    • Candidate best practice for aggregation-based IP address anonymization for privacy, *e.g.*, GDPR compliance
    • Aids investigation of address assignment practice, *e.g.*, to produce homogenous end user aggregates for matching with content in delivery networks
  • New work to take to WG?
    A reference implementation of kIP
Wrap Up

Team members:

Dave Plonka dave@plonka.us

More welcome! 😊

{based on code by Kenjiro Cho, Ryo Kaizaki}