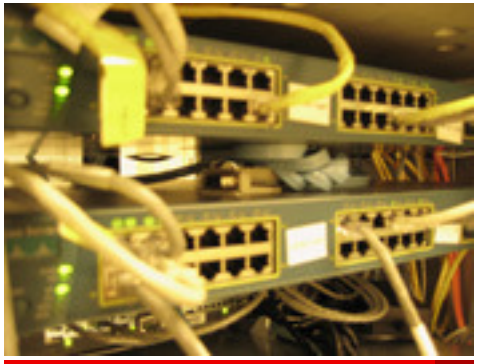
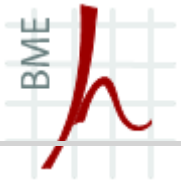


BENCHMARKING METHODOLOGY FOR IPV6 TRANSITION TECHNOLOGIES

IIJ Lab seminar

Dr. Gábor Lencse
senior research fellow
Department of Networked Systems and Services
Budapest University of Technology and Economics
lencse@hit.bme.hu

Tokyo, Japan
10. 10. 2017.



Introduction

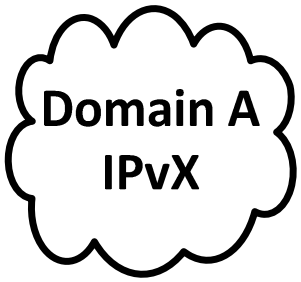
- Benchmarking
  - accurately measure some standardized performance characteristics in order to obtain reasonable and comparable results
  - RFC 2544
    - Benchmarking Methodology for Network Interconnect Devices
    - IP version independent
  - RFC 5180
    - IPv6 Benchmarking Methodology for Network Interconnect Devices
    - IPv6 specific, but excludes IPv6 transition technologies
  - RFC 8219 (new)
    - Benchmarking Methodology for IPv6 Transition Technologies

IPv6 Transition Technologies

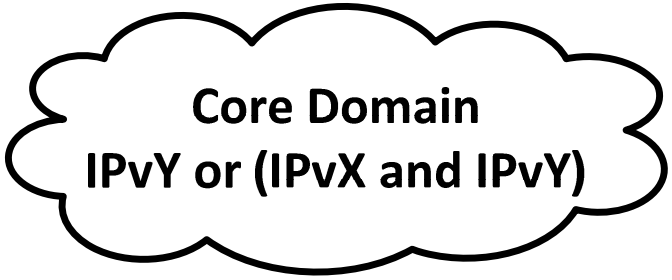
- Transition period from IPv4 to IPv6
  - Lasts for several years or decades
  - IPv4 and IPv6 are incompatible
  - Several “IPv6 transition technologies” can be used in various communication scenarios to enable communication
    - e.g. IPv6-only clients and IPv4-only servers: DNS64+NAT64
  - 26 IPv6 transition technologies were collected in:
    - G. Lencse and Y. Kadobayashi, “Survey of IPv6 Transition Technologies for Security Analysis”, IEICE Communications Society Internet Architecture Workshop, Tokyo, Japan, Aug. 28, 2017, *IEICE Tech. Rep.*, vol. 117, no. 187, pp. 19-24.
- In RFC 8219, the high number of technologies are classified into a small number of categories

Classification of IPv6 trans. techn.

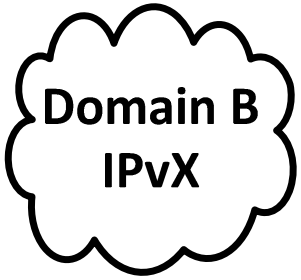
- Building blocks for the model of a production network regarding IPv6 transition:
  - Domain A: IPvX-specific domain



  - Core domain: IPvY-specific or dual-stack (IPvX and IPvY) domain



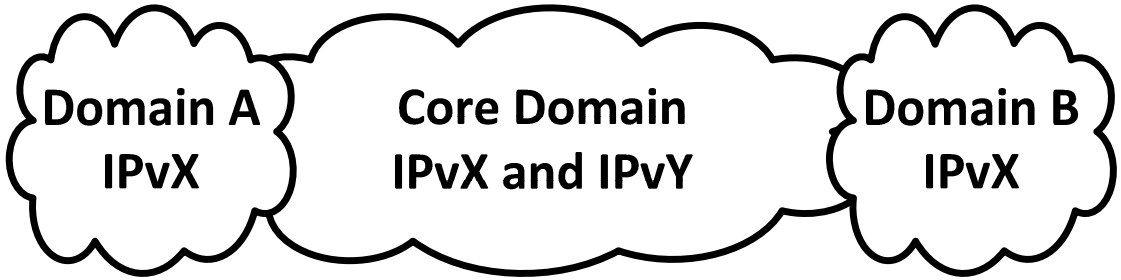
  - Domain B: IPvX-specific domain



Note: X,Y are part of the set {4,6}, and X≠Y.

Dual Stack

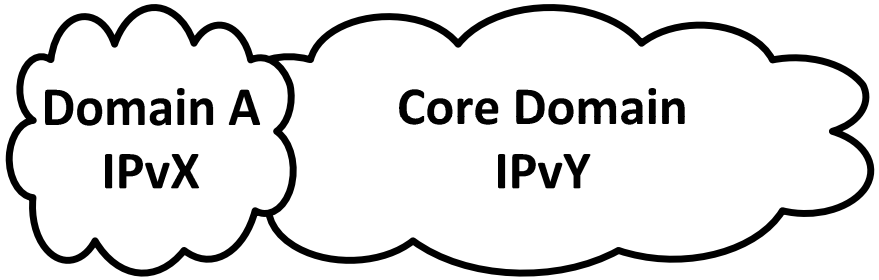
- Devices in the core domain implement both IP protocols.



- RFC 2544 and RFC 5180 can be used
- No new methodology is necessary

Single translation

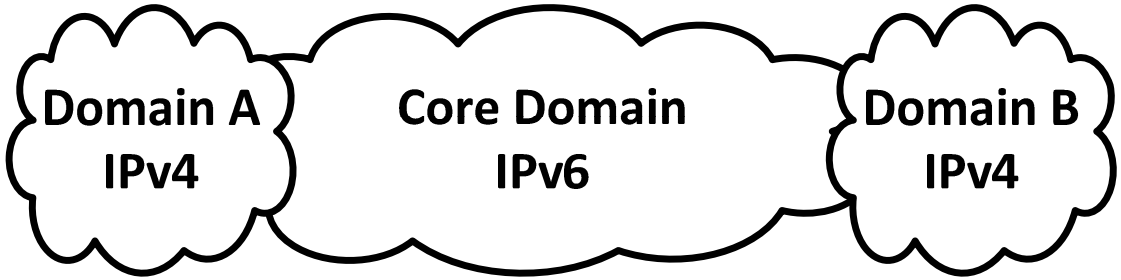
- The production network has only two domains
  - Domain A is IPvX specific
  - The Core Domain is IPvY specific
  - IPvX packets are translated to IPvY at the edge between Domain A and the Core domain.



- Example IPv6 transition technologies
  - Stateful NAT64 (RFC 6146), SIIT (RFC 7915), IVI (RFC 6219)

Double translation

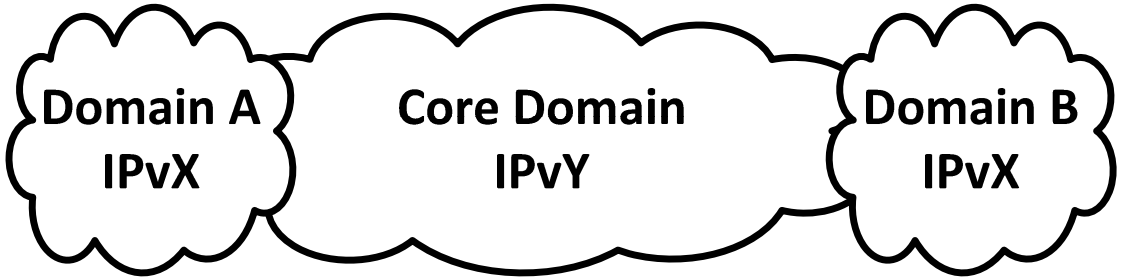
- The production network has all three domains
  - Domains A and B are IPv4 specific
  - The core domain is IPv6 specific
- Two translations are used for core network traversal
  - Domain A → Core Domain: IPv4 to IPv6 translation
  - Core Domain → Domain B: IPv6 to IPv4 translation



- Example IPv6 transition technologies
  - 464XLAT (RFC 6877), MAP-T (RFC 7599)

Encapsulation

- The production network has all three domains
  - Domains A and B are IPvX specific
  - The core domain is IPvY specific
- Encapsulation is used for core network traversal
  - Domain A → Core Domain: IPvX packets are encapsulated into IPvY packets
  - Core Domain → Domain B: de-encapsulation is performed



- Example IPv6 transition technologies
  - DS-Lite (RFC 6333), MAP-E (RFC 7597), 6to4 (RFC 3056)

DNS64 does not fit into any category

- The categories are general enough to include most of the IPv6 transition technologies
- DNS64 does not fit into any of them
  - It has to be handled individually – we do so
- Theoretically there is another one: DNS46
  - Defined by a long expired Internet Draft:
    - D. Liu, H. Deng, “NAT46 consideration”, expired Internet Draft,
    - https://tools.ietf.org/html/draft-liu-behave-nat46-02
  - Implementations exist
    - OpenWrt
    - Cisco ASA 5510
    - Brocade ServerIron ADX
  - But we do not have deployment info – we do not address it.

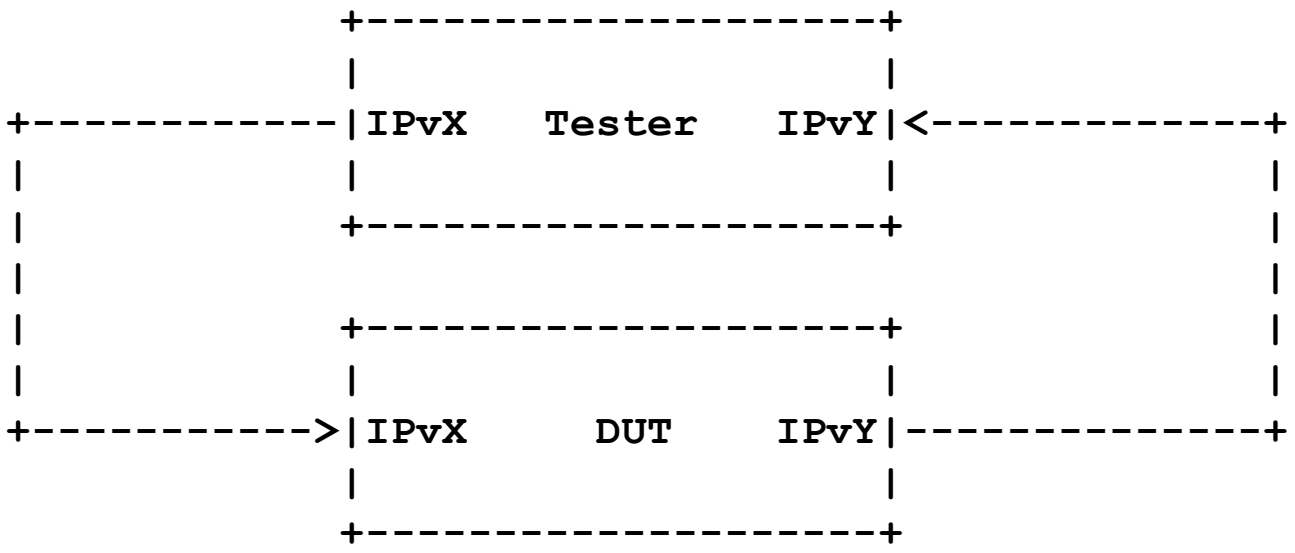
BENCHMARKING METHODOLOGY FOR SINGLE/DOUBLE TRANSLATION AND ENCAPSULATION TECHNOLOGIES

Applicability of RFC 2544 Testers

- RFC 2544 compliant Testers are produced and sold worldwide
- Problem: packet format and packet size are changed due to
  - Translation
  - Encapsulation
- But: the main points of the measurement procedures can be kept
  - The methods are adapted (e.g. packet sizes)
  - Two different measurement setups are defined
    - In one of them, the RFC 2544 Testers are applicable with some limitations
  - Complementary measurement procedures are defined (for stateful case)

Test Setup 1

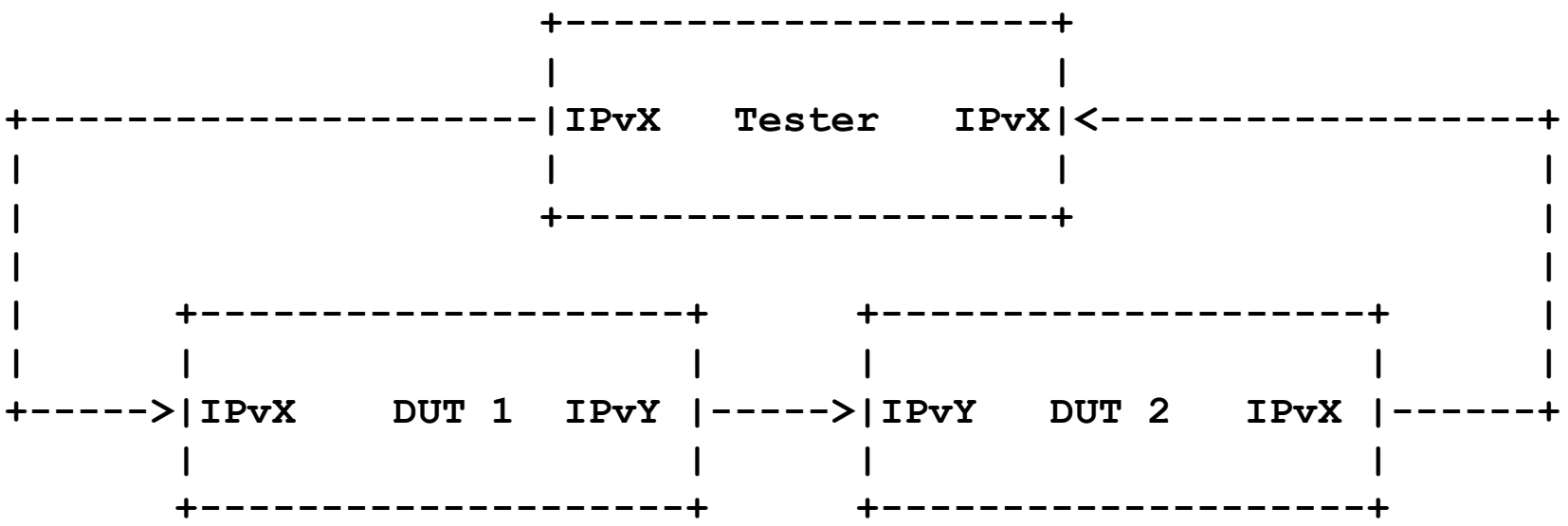
- For single-translation technologies, the test setup defined in RFC 2544 is reused as follows



  - Where X,Y are part of the set {4,6}, and X≠Y.
  - We also call it as “single DUT setup”

Test Setup 2

- For double-translation and encapsulation technologies, the following test setup is used



  - X,Y are part of the set {4,6}, and X≠Y.
  - DUT2 performs the inverse operation of DUT1
    - e.g. if DUT1: 4 → 6 translation, then DUT2: 6 → 4 translation
  - We also call it as “dual DUT setup”

Test Setup 2 (properties)

- Advantage
  - RFC 2544 Testers can be used ☺
- Disadvantage
  - Hides possible asymmetric behavior ☹
    - Solution: Test Setup 1 SHOULD also be used

Test Traffic

- Only unicast traffic is used (multicast is out of scope)
- Both translation and encapsulation change the frame sizes, which should be considered, when:
  - Maximum frame rate is calculated
    - Media speed could be exceeded due to overhead
  - Maximum/minimum frame size is calculated
    - Minimum may also be critical!
      - E.g. with NAT64, IPv6 minimum Ethernet frame size should be 84
        - » 84-20=64 for the IPv4 side
  - MTU is set in the devices

Traffic Setup

- Following the recommendations of RFC 5180
  - All tests described SHOULD be performed with bidirectional traffic.
  - Unidirectional traffic tests MAY also be performed for a fine-grained performance assessment.
    - E.g. when downloading a web page, the answers contain much more data than the request
    - Small, but important technical detail for Stateful NAT64
      - First packet in 6 → 4 direction establishes the connection
      - After that, the device can forward packets in the other direction
  - In principle, UDP traffic should be used
    - We shall address TCP specific questions, too

Coexistence

- In a network, where translation/encapsulation is used, native IPv4 or IPv6 traffic may also be present
- Recommended conditions
  - IPvX only traffic (where the IPvX traffic is to be translated/encapsulated by the DUT)
  - 90% IPvX traffic and 10% IPvY native traffic
  - 50% IPvX traffic and 50% IPvY native traffic
  - 10% IPvX traffic and 90% IPvY native traffic

Procedures: single translation, stateless

1. Throughput
   - RFC 2544 Throughput test
     - https://tools.ietf.org/html/rfc2544#section-26.1
2. Latency
   - Similar to RFC 2544, but redefined
     - https://tools.ietf.org/html/rfc8219#section-7.2
3. From RFC 5481
   1. Packet Delay Variation (PDV) is RECOMMENDED
      - https://tools.ietf.org/html/rfc8219#section-7.3.1
   2. Inter Packet Delay Variation (IPDV) is OPTIONAL
      - https://tools.ietf.org/html/rfc8219#section-7.3.2

Procedures: single translation, stateless

4. Frame Loss Rate
   - RFC 2544 Frame Loss Rate test
     - https://tools.ietf.org/html/rfc2544#section-26.3
5. Back-to-Back Frames
   - RFC 2544 Back-to-Back Frames
     - https://tools.ietf.org/html/rfc2544#section-26.4
6. System Recovery
   - RFC 2544 System Recovery test
     - https://tools.ietf.org/html/rfc2544#section-26.5
7. Reset
   - RFC 6201 Reset test
     - https://tools.ietf.org/html/rfc6201#section-4

Procedures: single translation, stateful

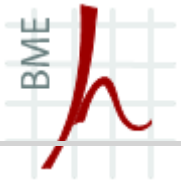
- All the stateless tests plus the following ones
  - RFC 3511 Concurrent TCP Connection Capacity test
    - https://tools.ietf.org/html/rfc3511#section-5.2
  - RFC 3511 Maximum TCP Connection Establishment Rate test
    - https://tools.ietf.org/html/rfc3511#section-5.3
  - Although not in RFC 8219, but further measurements are possible concerning the speed of the different state table management steps.
    - (Considered as future research topic.)

Procedures: double translation

- The same tests as for single translation technologies
  - Note: both stateless and stateful may be relevant
    - E.g. for 464XLAT
      - CLAT is stateless 4 → 6 translation
      - PLAT is stateful NAT64
- But different test setups are possible
  - Dual DUT setup facilitates the usage of RFC 2544 Testers
  - Single DUT setup tests are also RECOMMENDED to uncover possible asymmetric behavior.

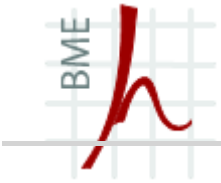
Procedures: Encapsulation

- The same tests as for double translation technologies
- Notes
  - no stateful tests
  - In the single DUT setup, the Tester SHOULD be able to send IPvX packets encapsulated as IPvY in order to test the de-encapsulation efficiency.



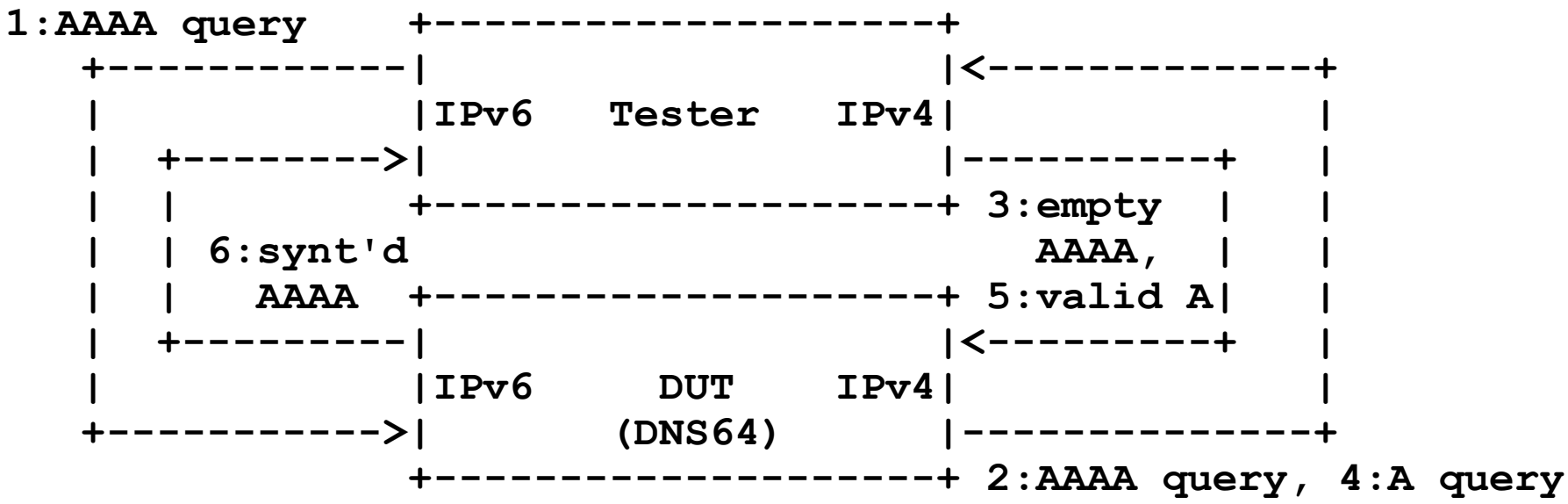
Protocol Types above IP

- UDP SHOULD be used for all the stateless tests.
- TCP SHOULD be used for all the RFC 3511 stateful tests.



BENCHMARKING METHODOLOGY FOR DNS64 SERVERS

Test and Traffic Setup



- Roles:
  - **Tester**: DNS64 client + Authoritative DNS server
  - **DUT** (Device Under Test): DNS64 server
- The Tester can be implemented either by a single device or by two devices.

Measurement traffic

1. Query for the AAAA record of a domain name (from client to DNS64 server)
2. Query for the AAAA record of the same domain name (from DNS64 server to authoritative DNS server)
3. Empty AAAA record answer (from authoritative DNS server to DNS64 server)
4. Query for the A record of the same domain name (from DNS64 server to authoritative DNS server)
5. Valid A record answer (from authoritative DNS server to DNS64 server)
6. Synthesized AAAA record answer (from DNS64 server to client)

Notes to measurement traffic

- If the DNS64 server implements caching and there is a cache hit, then step 1 is followed by step 6
  - (steps 2 through 5 are omitted).
- If the domain name has a AAAA record, then it is returned in step 3 by the authoritative DNS server, steps 4 and 5 are omitted, and the DNS64 server does not synthesize a AAAA record but returns the received AAAA record to the client.
- IP version used between the Tester and the DUT
  - IPv6 MUST be used between the client and the DNS64 server (messages 1 and 6)
    - recall: a DNS64 server provides service for an IPv6-only client
  - either IPv4 or IPv6 MAY be used between the DNS64 server and the authoritative DNS server (messages 2-5).

Measurement procedure

- Objective:
  - To determine DNS64 performance by means of the maximum number of successfully processed DNS requests per second.
- Procedure (innermost cycle)
  - send a specific number of DNS queries at a specific rate to the DUT
  - (concurrently) count the replies from the DUT that are both
    - received in time
      - within a predefined timeout period from the sending time of the corresponding query, having the default value 1 second
    - valid
      - contain a AAAA record.

(continued on next slide)

Measurement procedure (search)

- Procedure (cont.)
  - If the count of sent queries is equal to the count of received replies, the rate of the queries is raised, and the test is rerun.
  - If fewer replies are received than queries were sent, the rate of the queries is reduced, and the test is rerun.

- Note
  - The above wording follows that of RFC 2544
  - In practice, binary search is used

Measurement timing and result

- Timing
  - The duration of each trial SHOULD be at least 60 seconds.
  - No higher timeout time than 1 second SHOULD be used.
- Rationale: otherwise GAMING is possible! (See it later.)

- Result
  - The maximum number of processed DNS queries per second is the fastest rate at which the count of DNS replies sent by the DUT is equal to the number of DNS queries sent to it by the test equipment.

Measurement final results

- Random events MAY influence the measurement
  - E.g. if the DNS64 server is executed by a computer, and the OS does something else, too.
- The test SHOULD be repeated at least 20 times, and the median and 1st/99th percentiles of the number of processed DNS queries per second SHOULD be calculated.
  - (See it later why median and not average!)
  - Note:
    - for less than 100 measurements, 1st and 99th percentiles are the same as minimum and maximum, respectively.

Details and parameters

- Caching
  - First, all the DNS queries MUST contain different domain names (or domain names MUST NOT be repeated before the cache of the DUT is exhausted).
  - Then, new tests MAY be executed when domain names are 20%, 40%, 60%, 80%, and 100% cached.
  - Ensuring that a record is cached requires repeating a domain name both
    - "late enough" after the first query to be already resolved and be present in the cache
    - "early enough" to be still present in the cache.

  See possible solution in:

  G. Lencse, M. Georgescu, and Y. Kadobayashi, “Benchmarking Methodology for DNS64 Servers”, *Computer Communications*, vol. 109, no. 1, pp. 162-175, September 1, 2017, DOI: 10.1016/j.comcom.2017.06.004

Details and parameters

- Existence of a AAAA record
  - First, all the DNS queries MUST contain domain names that do not have a AAAA record and have exactly one A record.
  - Then, new tests MAY be executed when 20%, 40%, 60%, 80%, and 100% of domain names have a AAAA record.

- Note: caching and the existence of a AAAA record are orthogonal
  - All their combinations MAY be examined
  - The testing with 0% cached domain names and with 0% existing AAAA records is REQUIRED, and the other combinations are OPTIONAL.

Examination of “gaming”

- If a DNS64 implementation is able to store the requests and reply them later, “gaming” is possible
- Calculations:
  - t_{Test} test duration, r_T sending rate, t_{TO} time out
  - Pass condition:
    - DUT answers $t_{Test}*r_T$ number of queries within $t_{Test}+t_{TO}$ time.
  - Thus a rate of r_{DUT} may be enough:

$$r_{DUT} = \frac{t_{Test} * r_T}{t_{Test} + t_{TO}} = r_T \frac{1}{1 + \frac{t_{TO}}{t_{Test}}}$$

  - Therefore:
    - $t_{TO} << t_{Test}$ is necessary for correct result!
  - E.g. if t_{Test}=60s and t_{TO}=1s then:

$$r_{DUT} = \frac{60}{61} r_T = 0.9836 r_T$$

Examination of “gaming”

- Measurement results

Mtd64-ng DNS64 performance: number of answered queries per second, 1 worker thread, 1 s timeout.

Duration (s)	5	10	30	60
Average	3448.1	3160.9	2970.8	2926.3
Median	3448	3161	2971	2926
1st Perc. (min)	3442	3157	2967	2923
99th Perc. (max)	3452	3164	2974	2929
Standard Deviat.	2.81	1.77	1.92	1.66
MoE 99.9	2.07	1.31	1.41	1.22
Median Abs. Dev.	2.97	1.48	1.48	1.48
Calculated rate	2873.3	2873.6	2875.2	2878.0

Median vs. average

- Average (also called mean or arithmetic mean)
  - is more inclusive and less sensitive to noise,
  - but more sensitive to outliers.
- Median is more representative if the distribution is significantly skewed or multimodal
- In our case, IT IS!

(see next slide for full size)

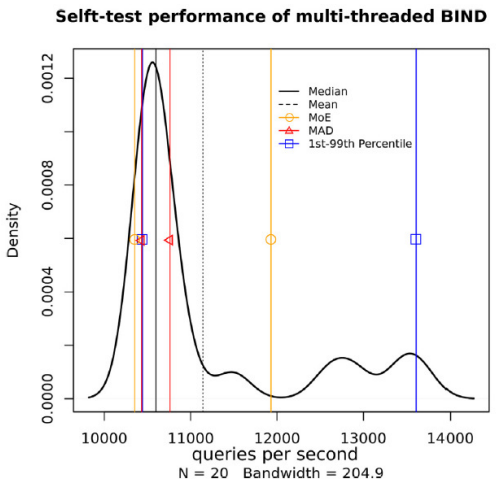


Fig. 5. BIND9 authoritative DNS server performance results: number of successfully answered AAAA record requests per second (**multi-threaded**, 800 MHz CPU clock frequency).

- We recommend the median for summarizing the results and the 1st and 99th percentiles as indices of dispersion.

Median vs. average: example

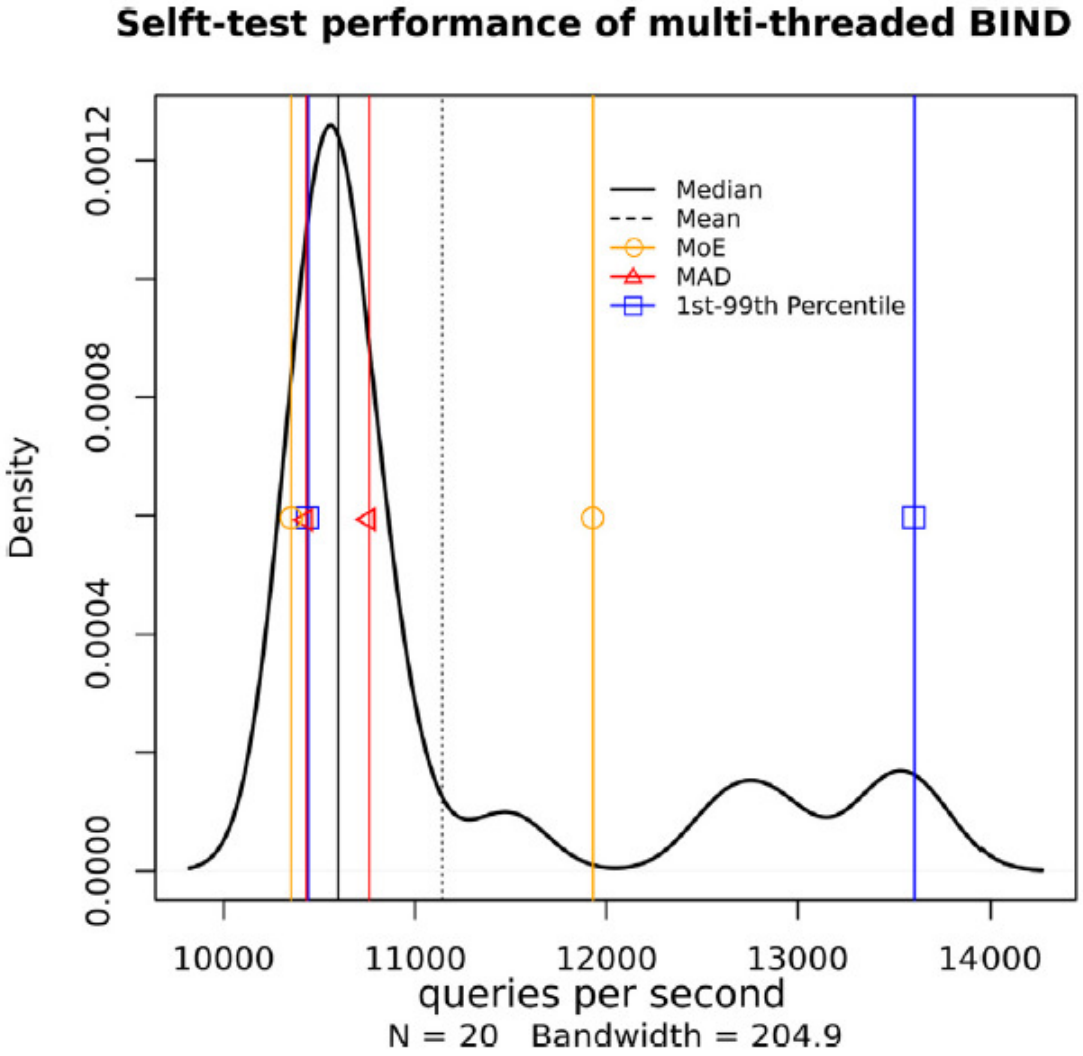


Fig. 5. BIND9 authoritative DNS server performance results: number of successfully answered AAAA record requests per second (**multi-threaded**, 800 MHz CPU clock frequency).

Example for nearly proper results

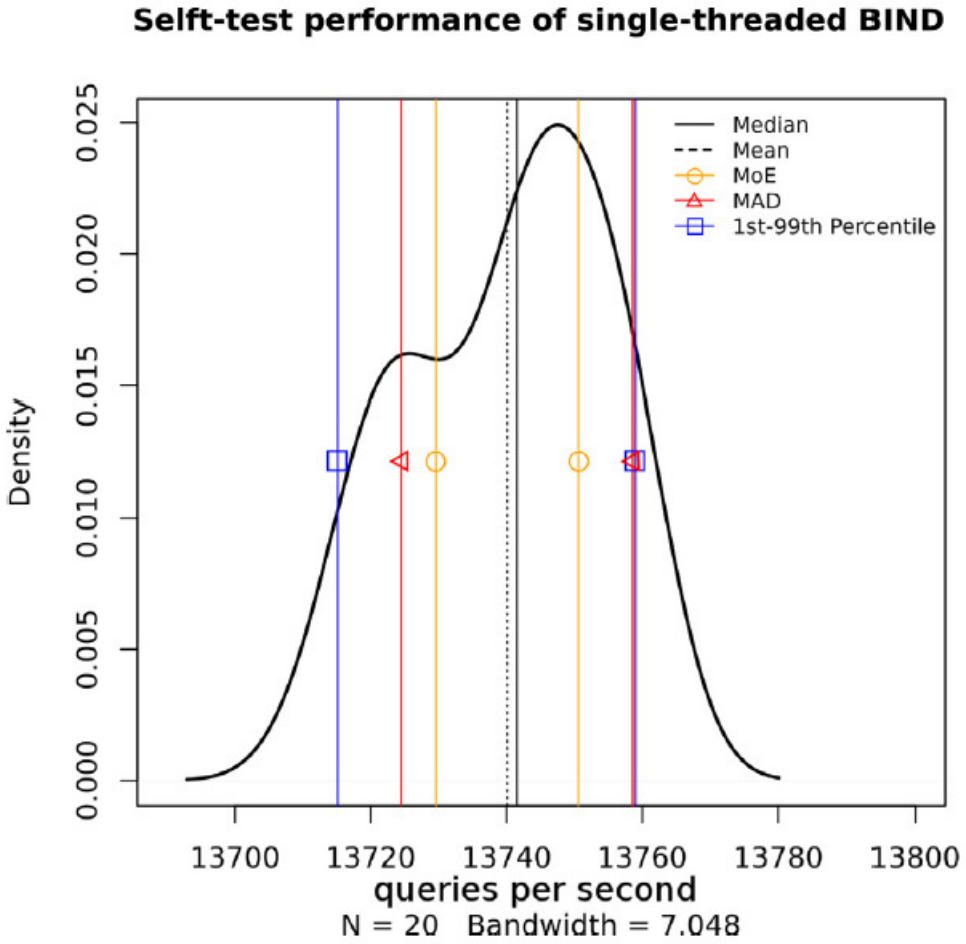


Fig. 4. BIND9 authoritative DNS server performance results: number of successfully answered AAAA record requests per second (**single-threaded**, 800 MHz CPU clock frequency).

Note: the scale for the horizontal axis is from 13,700 to 13,800!

Measurement tool: dns64perf++

- Implements Tester/Measurer only
- Needs an authoritative DNS server as Tester/AuthDNS,
  - E.g. BIND, NSD, YADIFA, etc.
- Free Software under GPLv2 license
- 1st version
  - implemented compulsory DNS64 tests
  - 1 thread for sending and 1 thread for receiving
  - documented in:

    G. Lencse, D. Bakai, “Design and implementation of a test program for benchmarking DNS64 servers”, *IEICE Transactions on Communications*, vol. E100-B, no. 6. pp. 948-954, June 2017. DOI: 10.1587/transcom.2016EBN0007

Measurement tool: dns64perf++

- Later extensions
  - support for testing caching

    G. Lencse, “Enabling Dns64perf++ for Benchmarking the Caching Performance of DNS64 Servers”, *Journal of Computing and Information Technology*, vol. 26, no 1, pp. 19-28. July 2018, DOI: 10.20532/cit.2018.1004078
  - correction of the sending timer algorithm

    G. Lencse and A. Pivoda, “Checking and Increasing the Accuracy of the Dns64perf++ Measurement Tool for Benchmarking DNS64 Servers”, *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 7. no. 1. pp. 10-16. DOI: 10.11601/ijates.v7i1.255
  - support for 2*n threads
    - Currently under testing
- Available: https://github.com/bakaid/dns64perfpp

Measurement tool: dns64perf++

- Limitations
  - Built on TCP/IP socket interface for packet sending
  - Executed as a user process
  - Moderate accuracy

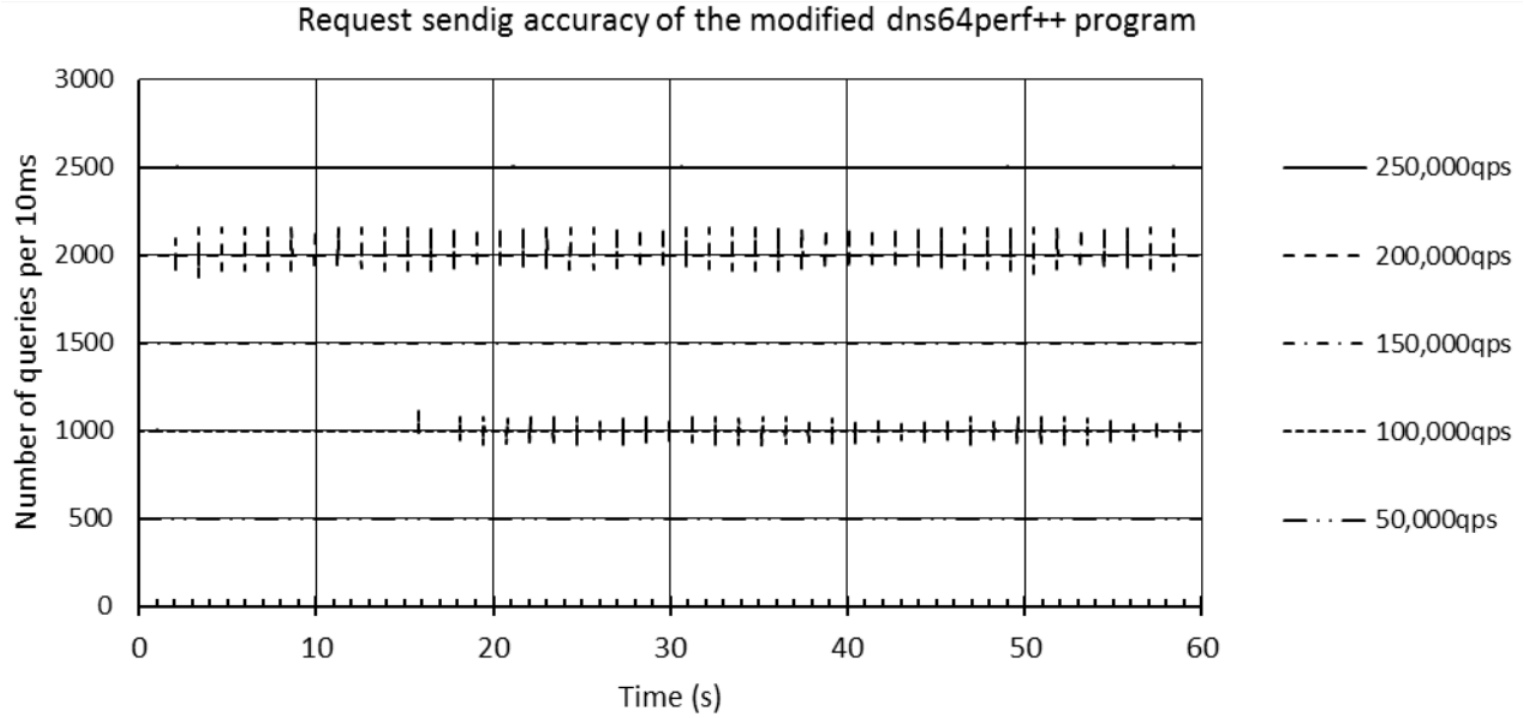


Fig. 5 Number of AAAA record queries sent in a 100ms long interval by the modified **`dns64perf++`** program – *using 10ms cells*.

Measurement tool: dns64perf++

- The next step could be:
  - Reimplementation using DPDK (Intel Dataplane Development Kit) instead of the socket interface API
  - Anyone interested in?

Benchmarking tests

- Tested DNS64 implementations
  - BIND, ~~TOTD~~, PowerDNS, Unbound, ~~mtd64-ng~~
- Tests were carried out using NICT StarBED, Japan in 2017
  - Results have been published recently:
    - G. Lencse and Y. Kadobayashi, "Benchmarking DNS64 Implementations: Theory and Practice", *Computer Communications* (Elsevier), vol. 127, no. 1, pp. 61-74, September 1, 2018, DOI: 10.1016/j.comcom.2018.05.005
- Currently testing several authoritative DNS implementations
  - BIND, NSD, YADIFA, Knot DNS, etc.
  - FakeDNS (special-purpose program)

Summary

- The high number of IPv6 transition technologies were classified into a small number of categories
- Dual Stack is trivial
- Single/Double translation and Encapsulation
  - Handled more or less together
- DNS64
  - Handled separately
  - Measurement tool “dns64per++” also exists
  - Benchmarking tests have been carried

Recommended reading

1. M. Georgescu, L. Pislaru and G. Lencse, "Benchmarking methodology for IPv6 transition technologies“, *IETF RFC 8219*, Aug. 2017, DOI: 10.17487/RFC8219
2. G. Lencse, M. Georgescu, and Y. Kadobayashi, “Benchmarking methodology for DNS64 servers”, *Computer Communications*, vol. 109, no. 1, pp. 162-175, September 1, 2017, DOI: 10.1016/j.comcom.2017.06.004
3. G. Lencse, D. Bakai, “Design and implementation of a test program for benchmarking DNS64 servers”, *IEICE Transactions on Communications*, vol. E100-B, no. 6. pp. 948-954, June 2017. DOI: 10.1587/transcom.2016EBN0007
4. G. Lencse, “Enabling dns64perf++ for benchmarking the caching performance of DNS64 servers”, *Journal of Computing and Information Technology*, vol. 26, no 1, pp. 19-28. July 2018, DOI: 10.20532/cit.2018.1004078
5. G. Lencse and A. Pivoda, “Checking and increasing the accuracy of the dns64perf++ measurement tool for benchmarking DNS64 servers”, *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 7. no. 1. pp. 10-16. DOI: 10.11601/ijates.v7i1.255

All these and further papers are (or will be) available from:
http://www.hit.bme.hu/~lencse/publications/

Questions?

THANK YOU FOR YOUR ATTENTION!

This presentation was supported by the International Exchange Program of the National Institute of Information and Communications Technology (NICT).

Dr. Gábor Lencse
Senior Researcher
Department of Networked Systems and Services
Budapest University of Technology and Economics
lencse@hit.bme.hu