Using Confidential Computing to protect DNS resolution

draft-arkko-dns-confidential-02.txt

IETF-112

Jari Arkko & Jiri Novotny
Ericsson Research

General Context

What can we do to further improve privacy?

- For services, infrastructure functions, network elements
- Beyond what's already been done

DNS as one example

- Domain name meta-data visible on the wire (even with encryption)
- Resolvers have the potential to see user's entire browsing history
- Large resolver services are an attractive target

Developments

- Work on encrypted DNS query protocols (e.g., DoH), discovery
- Practices, expectations, contracts (e.g., RFC 8932, Mozilla's TRR reqs)
- Other advances (e.g., eSNI, OHAI, draft-private-access-tokens, confidential computing, RATS)

Context of This Work

Build on all that work – and assume all communications encrypted

- What problems remain? Is there a next step?

- Our worry is that resolvers can be a major remaining source of leaks
  - Accidents, attacks, commercial use, authorities request
- We need to protect user's data in flight, at rest, or in use – we wanted to experiment with tech that could reduce leaks on the last two
- There are interesting opportunities and challenges, worth discussing

DNS Resolvers with Confidential Computing

Basic idea: run the resolver process inside a TEE, and do not provide any user-specific data outside the encrypted process

- Even to the DNS operator itself
- Or the owner of computer hardware, or the operating system

Requires a willing DNS operator to do this, of course

Desired security property: establish a security perimeter around the service such that it prevents easy collection of information

- Not perfect but complicates the life of the data collecting agency ☺

Trusted Execution Environment (TEE)

”An environment that enforces that any code within that environment cannot be tampered with, and that any data used by such code cannot be read or tampered with by any code outside that environment." [ietf-teep-architecture]

- Data confidentiality and integrity
- Code integrity
- Attestability

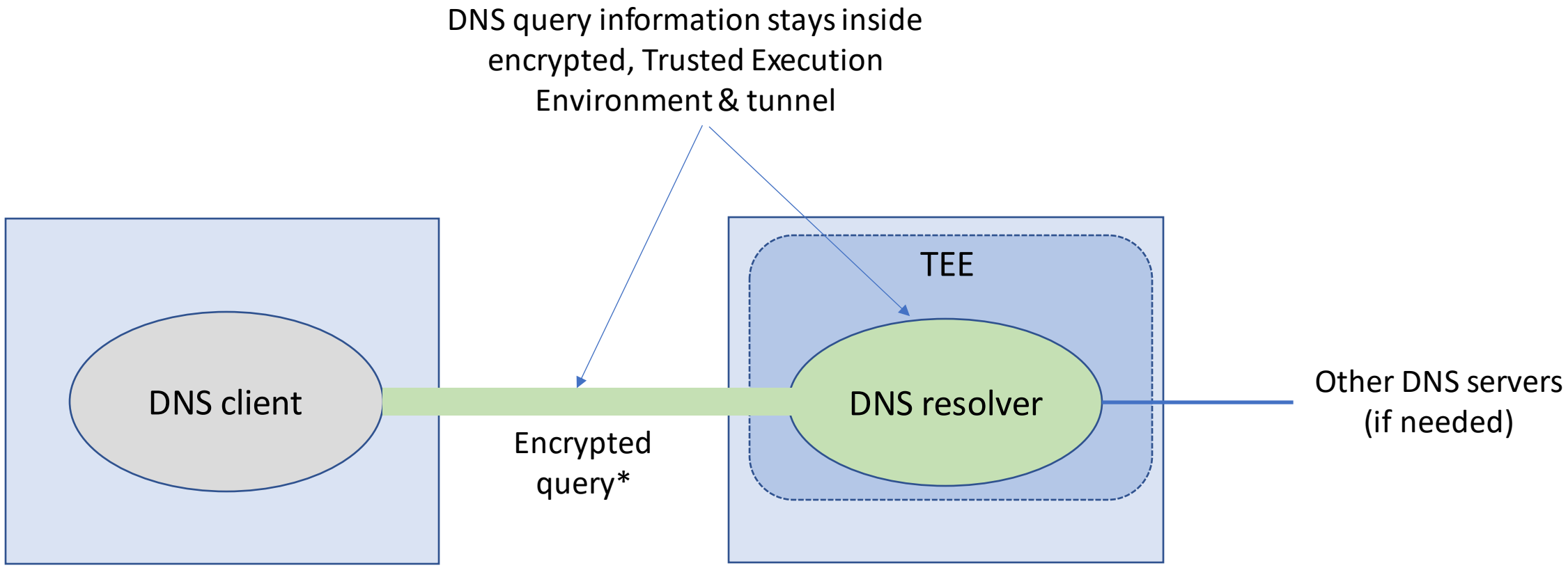
Attestation

Attestation is the process where a node can provide assurance that it is indeed running inside a particular type of TEE environment, with a specific configuration and application software.

Requires trusting a party – at least the CPU manufacturer, and an ability to understand what constitutes an acceptable configuration.

Highly heterogenous technology at this stage. Building blocks exist and evolve.

CC – DNS: Overview



* Can be built on top of DoT [RFC7858] or DoH [RFC8484]

Considerations (1)

- Upside: It is possible to hide all user-related information
  - We argue that we SHOULD do so
- Upside: Can provide evidence of compliance to an expected practice
  - “Trust but verify”
- Upside: There are potential incentives to use this
  - “Our DNS service does not leak your information, and we can show it”
- Upside: May change business practices, e.g., on use of data
  - Could still provide aggregate data, e.g., most popular lists, etc.
  - Can still perform geolocation etc. (inside TEE)

Considerations (2)

- Downside: Changes operational practices
  - Scaling, systems monitoring, debugging, DoS (can be addressed at least to an extent)
- Downside: Dependencies
  - Hardware, manufacturers, verifiers, who checks the software?
- Downside: Performance (?)
  - Impact varies – also some coming developments may change situation

Considerations (3)

- Observation: Not a cure for all problems, and attacks remain
  - If you thought $MANUFACTURER CPUs were secure, we have news for you
  - Also, side channel attacks, software issues
  - May still provide an additional layer of defense
  - Also, this is an additional hurdle for an attacker to have to overcome, e.g., $AGENCY
- Observation: Seems orthogonal to many other techniques
  - Communications encryption, data minimization, end-to-end encryption instead of letting $SERVER read mail, oblivious technologies, etc.

Reflections and Developments

- Areas with active developments
  - Hardware -- SGX3 / TDX, ARM-CCA, etc.
  - Communities and standards CCC, IETF RATS, etc.
  - Implementations, plenty e.g., for secrets management, secure containers, ML/AI, DBs
- Areas with more research needed
  - Operational impacts, debugging
  - Attestation – crucial component as we talked about earlier, and many possibilities and threat models apply here
- Applications
  - We are also looking at where we can apply this in mobile networks
  - Other possibilities include, e.g., draft-voit-rats-trustworthy-path-routing

Conclusion

- Data held by servers SHOULD receive at least as much security attention as communications do.

- Crucial for many Internet and infrastructure services
- Crucial for DNS resolvers, (potential leak of user's browsing history)

- Consider all applicable tools – one discussed here
- Nearing potential practical uses?

Thank you!

- Feedback to us very welcome – we are here to hear your thoughts
  - Feasible or science fiction?
  - Need to take X, Y, Z into account?