# Security considerations for the Babel routing protocol

Denis Ovsienko (Custodian Data Centres)
denis@custodiandc.com
IETF 95 Buenos Aires, April 4 2016

# Ways to see a network protocol

- as a user (i.e. operator)
- as an implementer
- as a researcher
- as a designer

# Observations: tcpdump

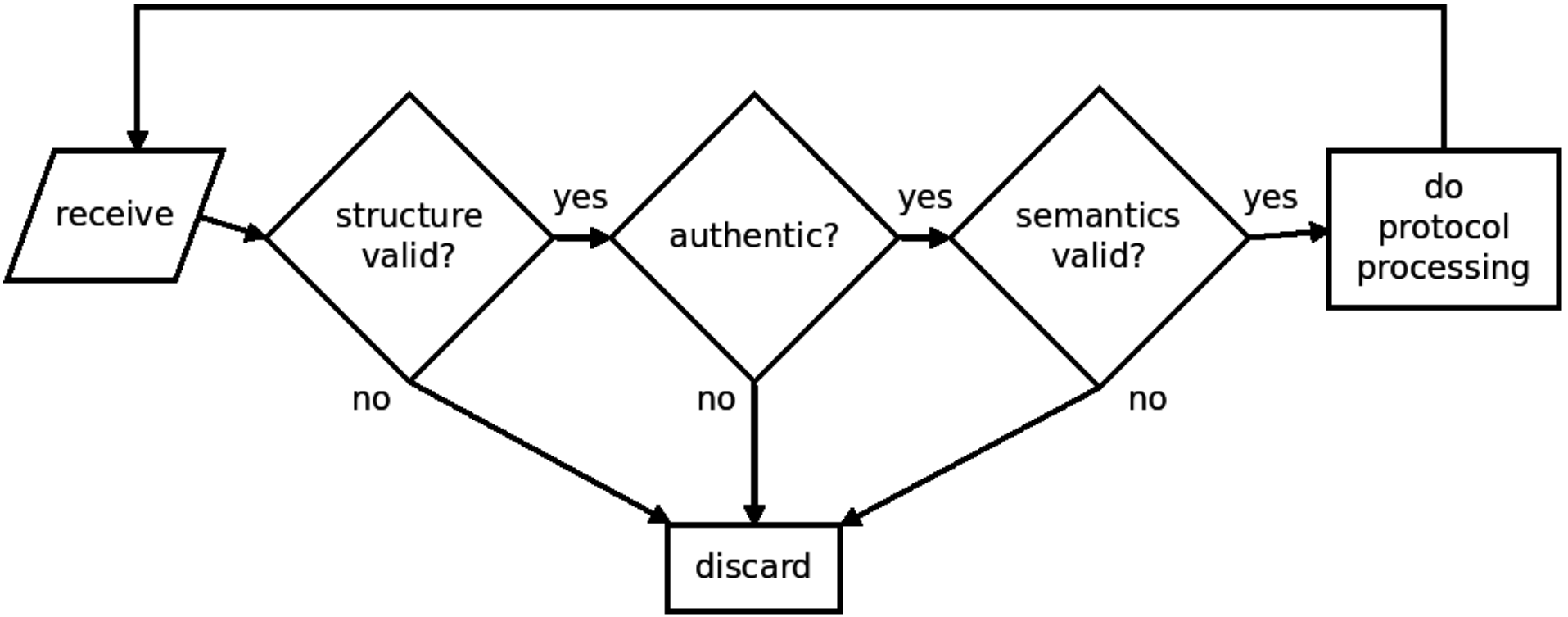
| source file | last update (*) | kbytes | SLOCCount |
|---|---|---|---|
| `print-babel.c` | 2014 | 24 | 594 |
| `print-bgp.c` | 2015 | 96 | 2268 |
| `print-isoclns.c` | 2015 | 104 | 2558 |
| `print-olsr.c` | 2015 | 23 | 493 |
| `print-ospf.c` | 2007 | 39 (**) | 991 (***) |
| `print-ospf6.c` | 2013 | 30 (**) | 798 (***) |
| `print-rip.c` | 2012 | 9 | 188 |

(*) any encoding-specific update, not necessarily the latest specification
(**) +10 of `ospf.h`
(***) +221 of `ospf.h`

# Observations: bugs and vulnerabilities in network protocols

- These problems more often seem to be implementation-specific.
- Fuzzing does not change the amount of problems in a software but drastically increases the amount of known ones.
- Generic counter-measures: keep specifications clear and uniform, expect rigid encodings to cause issues, keep future protocol extensions in mind.
- More particular counter-measures: don't depend on an implied context to validate a packet; design the encoding with layered input validation in mind.

# Layered input validation



# Origins of RFC 7298

1. To protect this OSPF code from fuzzing I need to understand the authentication trailer it seems to implement...
2. OK, done with the OSPF code and RFC 5709/6506 but they both derive from RFC 4822 so let's study that as well...
3. Let's implement RFC 4822 to see if I got the concept...
4. Looks like a similar approach could make my small mesh network project on Babel more secure...
5. Perhaps the Babel community would want the result as an extension?

# Cryptographic agility

- Carpenter, B., Aboba, B., Ed., and S. Cheshire, "Design Considerations for Protocol Extensions", RFC 6709, September 2012.
- RFC 7298 took the above into account.
- Housley, R., "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", BCP 201, RFC 7696, November 2015.
- RFC 7298 remains in line with BCP 201.

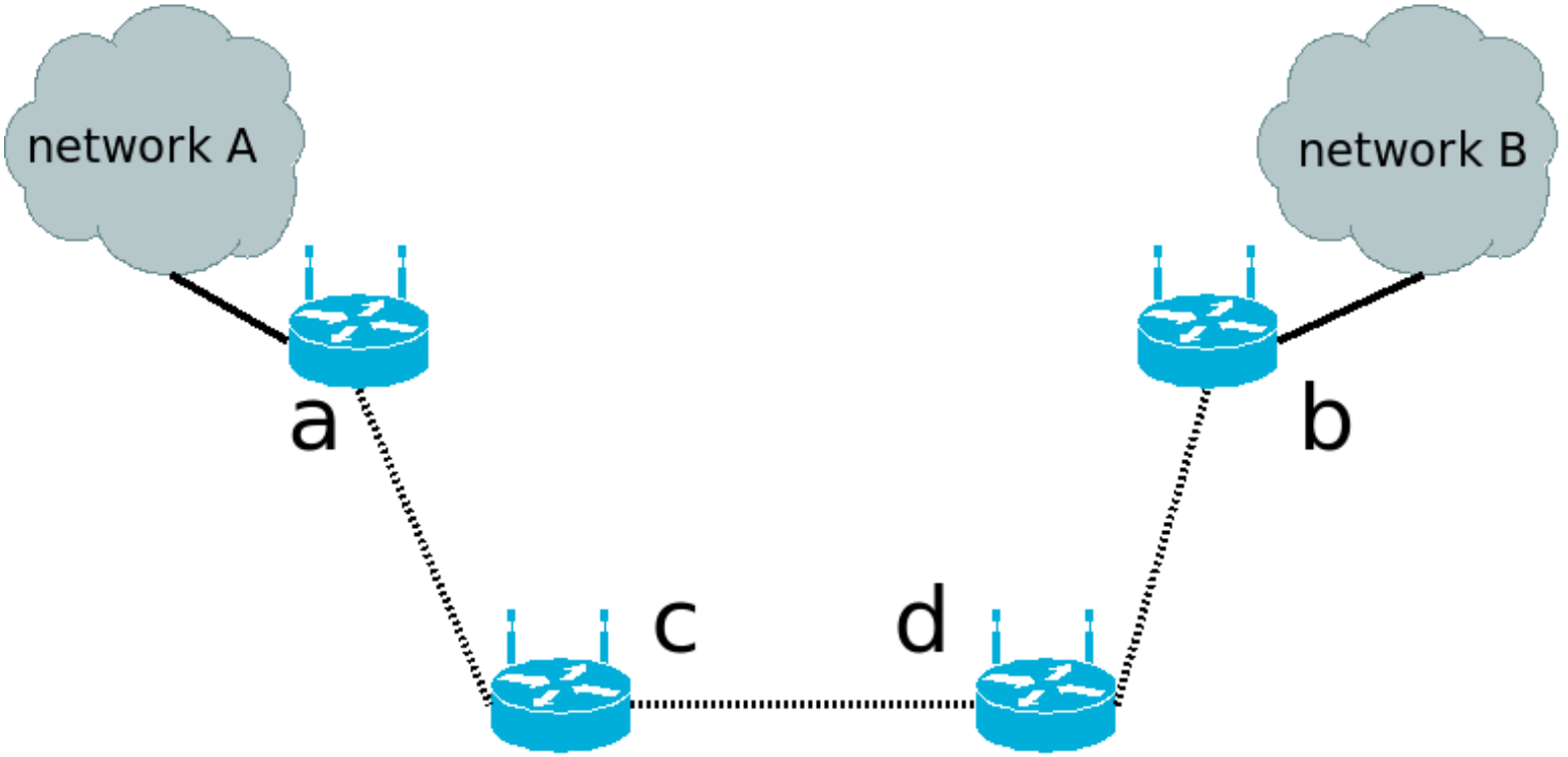
# How exactly is it in line?

- Algorithm ID does not need to be on the wire.
- In that case an IANA registry does not apply.
- Parameter (ICV) sizes must be identified.
- Roughly equal algorithms in the MTI suite.
- Ideally two independent MTI algorithms.
- Which must be public and well-studied.
- Should be clear how to switch between them.

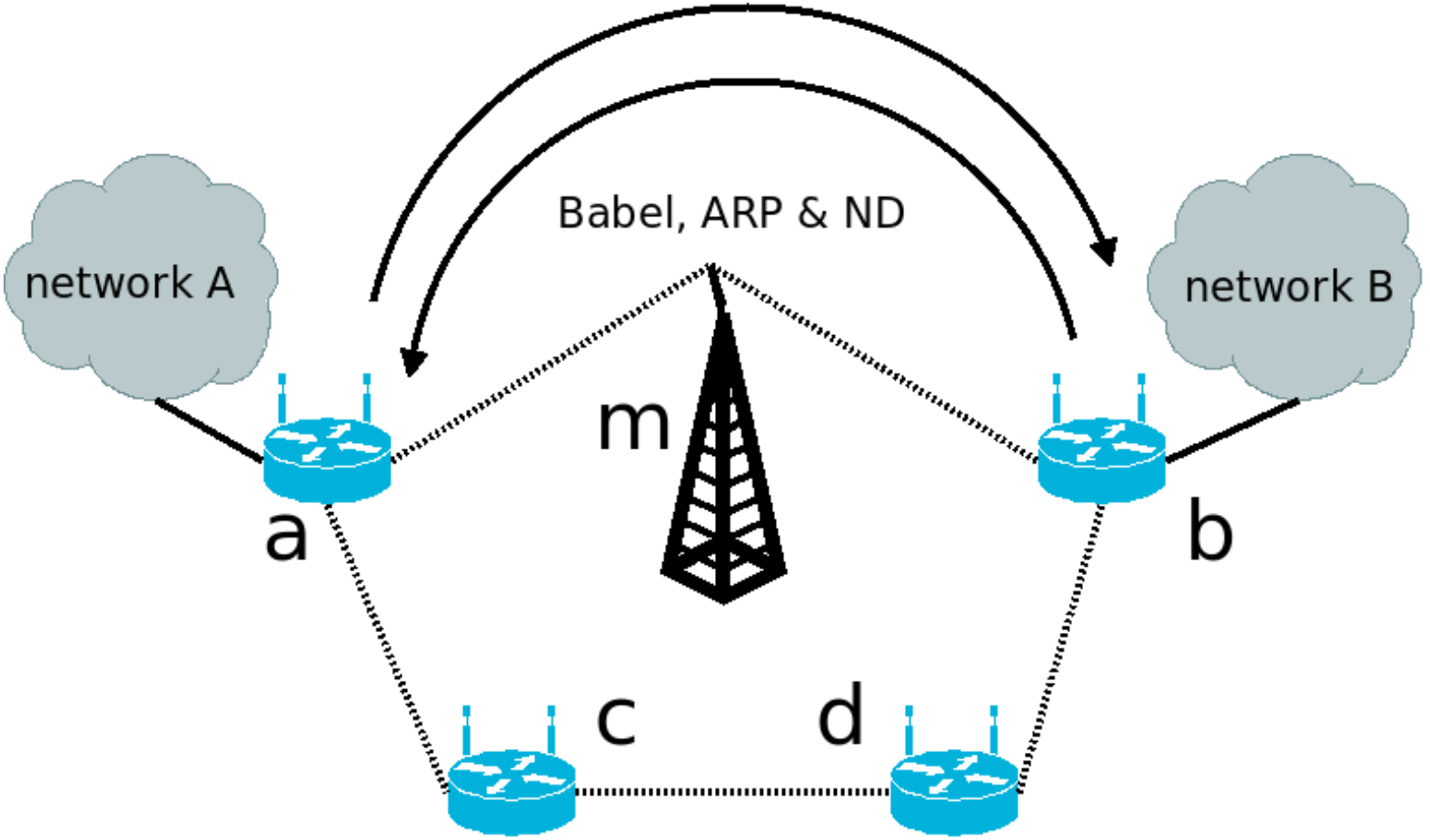
# So is RFC 7298 a perfect solution?

- Short answer: no, it is not perfect.
- Barbir, A., Murphy, S., and Y. Yang, "Generic Threats to Routing Protocols", RFC 4593, October 2006.
- The following example demonstrates just one attack from the above document that is easy to explain but difficult to solve (acknowledged in RFC 7186 Section 4.5 and RFC 7298 Section 8.e).

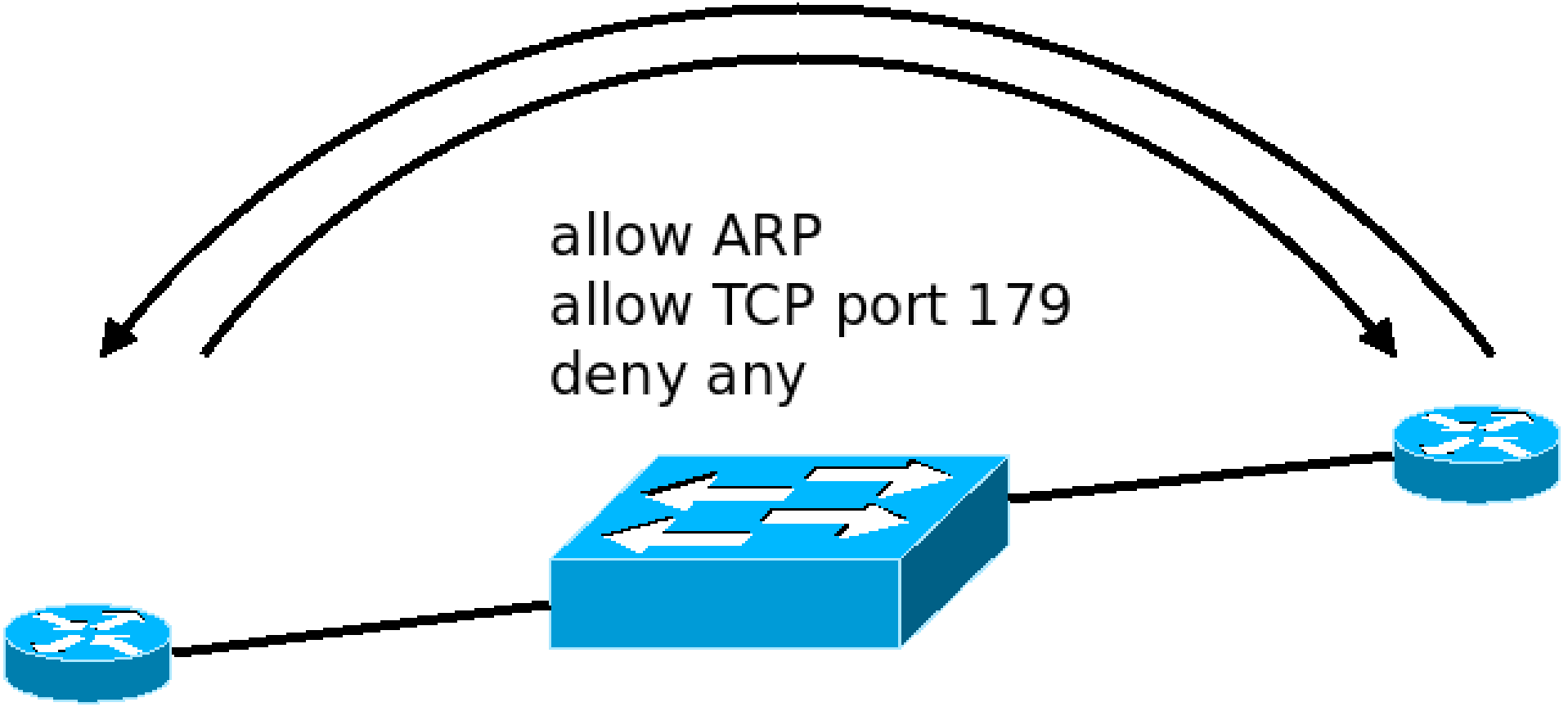
# Spoofing: the original topology



# Spoofing: the attack



# Spoofing: slightly different setup



# So is RFC 7298 a perfect solution?

- Long answer: the solution space of this class of authentication mechanisms is known to be smaller than the full space of the problem, which is not specific to Babel or mesh routing protocols or routing protocols in general. Better results require a different class of solution.

# Conclusions

- RFC 7298 to my best understanding is roughly as secure as existing Standards Track specifications.
- A better class of authentication requires full-time cryptography experts in the WG.
- The current design of Babel looks fine for fuzzing resistance, let's keep it this way.
- RFC 4593, RFC 6709 and BCP 201 are very relevant documents to consider in future works.
- The present Babel encoding and extension spec incorporate useful design points, let's understand and preserve them.

Thank you!