Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 9, 2015

J. Chroboczek
PPS, University of Paris-Diderot

April 7, 2015

Security considerations for the Babel routing protocol
draft-chroboczek-babel-security-considerations-00

Abstract

Where we stress that the Babel routing protocol is completely insecure.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Babel routing protocol [RFC6126] is a lightweight and robust routing protocol that aims at being applicable in a wide range of situations where familiar link-state routing protocols perform suboptimally, ranging from lossy and unstable radio networks through overlay networks and hybdrid networks (networks consisting of technologies with widely different performance characteristics).

Because of the wide applicability of Babel, no single security technology is likely to be acceptable to all the users of Babel. In particular, while symmetric cryptographic authentication technologies (such as the one described in [RFC7298]) solve many of the security issues of Babel in the vast majority of deployments, there may be applications of Babel where they are not applicable, either because their functionality is insufficient (e.g. no support for asymmetric cryptography) or because of implementation cost (not only CPU cost).

For that reason, RFC 6126 does not specify any particular "must implement" technology, and honestly mentions that "As defined in this document, Babel is a completely insecure protocol" (Section 6 of [RFC6126]). We would be opposed to defining a single "must implement" security mechanism, or including such a mechanism in the base Babel specification, at least until there is enough implementation and deployment experience to allow us to say "this is the right security mechanism for Babel".

Our position is consistent with the letter and the spirit of [BCP61]. BCP 61 stresses that security is necessary, but it does not specify how security is to be achieved. It does not mandate that a protocol should have a single "must implement" security mechanism, nor does it require that it should be included in the base specification.

In this document, we describe some of the attacks that are easily performed against an unsecured Babel router, and describe the mitigations and solutions known to us.

2. Active attacks

In this section, we describe some active attacks -- attacks that can be performed by an attacker that is able and willing to send Babel control traffic to Babel nodes.

2.1. Routing table poisoning

An attacker that is able to send packets containing Update TLVs can insert hostile entries into the victim's routing table. A routing table that contains such hostile routes is said to be poisoned.

2.1.1. Lower metric attack

An attacker that is in a sufficiently central position in a Babel routing domain can announce hostile routes that carry a lower metric than the authentic routes. Babel's route selection mechanism will prefer these routes to the legitimate but higher-metric routes, and therefore poison its routing table.

2.1.2. Higher seqno attack

An attacker that is unable to achieve a sufficiently low metric (presumably because it cannot reach a sufficiently central position in a Babel routing domain) can still poison routing tables by announcing a seqno that is higher than the seqno of the legitimate routes. While Babel's route selection algorithm will normally ignore higher-metric routes, a victim that is suffering temporary starvation (Section 2.5 of [RFC6126]) will, under some circumstances, temporarily switch to a higher-seqno route and therefore poison its routing table.

2.1.3. Replay attack

Even if Babel packets are authenticated, in the presence of static keying an attacker may capture enough authentified low-metric updates to perform routing table poisoning. The seqno mechanism does not do anything to protect against this attack, as Babel nodes do not ignore routes with an unexpected seqno; in any case, the seqno space is circular, and seqnos are reused after a few hours or at most days.

2.1.4. Amplification through routing table poisoning

An attacker that is able to perform routing table poisoning may announce a third party next hop (Section 4.4.8 of [RFC6126]), and therefore redirect a node's data traffic to a third party, which will potentially suffer a denial of service.

2.2. Amplification due to requests

The Babel protocol includes the ability to request a full routing table update by sending a "wildcard request". Wildcard request may be sent over multicast, and in a dense network a single request TLV may cause a significant amount of traffic, thus potentially performing a denial of service.

2.3. Covert channel

Babel is an extensible protocol. Babel's extension mechanism [BABEL-EXT] allows attaching extension data to almost any TLV in a Babel packet; this data will be silently ignored by an implementation that doesn't understand the extension, and can therefore be used as a covert channel that is propagated for just one hop.

Another approach consists in encoding covert information within one of the currently defined extensions, for example in the radio interference information carried by [BABEL-Z]. The advantage of this method is that the information will be propagated across the Babel routing domain by non-collaborating routers.

2.4. Mitigations and solutions

Some of the attacks in this section are avoided by using a cryptographic authentication mechanism with replay protection, such as the one defined in [RFC7298]. However, in some deployments such mechanisms may not be desirable, either due to implementation complexity or to the difficulty of deploying symmetric keys.

If the Babel traffic is protected by some lower-layer mechanism, the replay attack described in Section 2.1.3 above can be avoided by using a replay protection mechanism, such as the one described in Section 5.1 of [RFC7298], and which is independent of the rest of the protocol described in that document.

If Babel traffic is carried over IPv6, which is normally the case, Babel packets are sent from a link-local address. Since Babel nodes discard Babel packets that are not sent from a link-local address (Section 4 of [RFC6126]), and since link-local packets are unable to cross routers, this prevents all of the attacks in this section

unless an attacker is able to send traffic from a link directly attached to a Babel node.

No such natural protection is available when Babel traffic is carried over IPv4, which does not have an equivalent to IPv6 link-local addresses. However, no implementation of Babel known to us carries its control traffic over IPv4.

We are not aware of any solution or mitigation technique to the covert channel problem. As far as we can tell, there is nothing that can be done to protect against a covert channel if untrusted routers are allowed to join the routing domain.

3. Passive attacks

In this section, we describe some attacks that can be performed by an attacker that is unable or unwilling to send Babel protocol packets, but that is able to eavesdrop on a link that carries Babel control traffic.

3.1. Stable node identifiers

There are three ways in which Babel traffic can be used to identify a node. Babel Hello TLVs carry a unique (within the routing domain) 64 bit identifier, known as the "router-id"; Section 3 of [RFC6126] recommends that router-ids be allocated in modified EUI-64 format [RFC4291], presumably from a hardware address. Router-ids can therefore serve as a stable node identifier.

In addition, when Babel control traffic is carried over IPv6, it is sent from a link-local IPv6 address. Such addresses are usually generated from a hardware address, and can therefore be used as a stable node identifier.

Finally, Babel Update TLVs carry the set of prefixes announced by a node. The prefixes announced with a metric of 0 are prefixes of directly connected networks, which in some topologies can be used as a stable node identifier.

3.2. Mitigations and solutions

The stable identifier nature of a router-id can be mitigated by choosing router-ids randomly, and changing them periodically. However, the protocol does not allow changing router-ids gracefully: a Babel node that changes router-ids must tear down all of its neighbour associations. Current implementations are only able to change router-id at startup.

The same approach, with the same caveats, can be taken to change link-local interface addresses.

Whether the same approach can be used to rotate locally redistributed prefixes depends on the topology and the way the network is managed. At any rate, the Babel protocol allows rotating announced addresses in a graceful manner.

4. Conclusion

The Babel routing protocol, as defined in [RFC6126], is a completely insecure protocol. Due to the wide applicability of Babel, no single security mechanism is likely to satisfy all the needs of Babel's userbase, and hence no "must implement" security mechanism should be defined for Babel.

Implementors and users must be aware of this fact, and use security mechanisms or mitigation techniques that are adapted to the nature of their deployment. One such security mechanism can be readily integrated to the protocol [RFC7298] (sample code is available); alternatively, a lower-layer mechanism that is not vulnerable to replay attacks may be used.

5. References

[BABEL-EXT] Chroboczek, J., "Extension Mechanism for the Babel Routing Protocol", draft-chroboczek-babel-extension-mechanism-04 (work in progress), March 2015.

[BABEL-Z] Chroboczek, J., "Diversity Routing for the Babel Routing Protocol", draft-chroboczek-babel-diversity-routing-00 (work in progress), July 2014.

[BCP61] Schiller, J., "Strong Security Requirements for Internet Engineering Task Force Standard Protocols", BCP 61, RFC 3365, August 2002.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC6126] Chroboczek, J., "The Babel Routing Protocol", RFC 6126, February 2011.

[RFC7298] Ovsienko, D., "Babel Hashed Message Authentication Code (HMAC) Cryptographic Authentication", RFC 7298, July 2014.

Author's Address

Juliusz Chroboczek
PPS, University of Paris-Diderot
Case 7014
75205 Paris Cedex 13
France

Email: jch@pps.univ-paris-diderot.fr