Whither Deprecating TCP-MD5?

A light dose of reality vs. IETF process.

IEPG – March 2018

Jeffrey Haas <jhaas@juniper.net>

A new protocol is brought before the Security ADs

YOUR TRANSPORT SECURITY CONSIDERATIONS ARE NOT ADEQUATE!

Our story…

- Control plane protocols are often carried over simple transport layers such as UDP or TCP.
- Control planes are good targets for attack and their disruption or subversion can have serious operational consequences.
    - TCP RST attacks against BGP routers were the original motivation for
      RFC 2385, TCP-MD5.

Security Properties We Want for the Control Plane

- The security property of greatest concern to most protocol engineers is data integrity. (RFC 4949)
  - What a protocol sends and receives should not be meddled with.
  - (“Do not meddle in the affairs of wizards, for they are subtle and quick to anger.”)
- Data authentication is also a property that is desired.
  - You have a stream of routing data that you believe hasn’t been meddled with, but do you know who it came from?
- Data confidentiality might be desired.
  - Protocol engineers are often agnostic about this.
    Arguably, they don’t have enough skin in the game.
  - Operators may care or not, depending on protocol and circumstance.
    Pragmatism with respect to making operations more difficult matters!
  - Security professionals would prefer no one sees anything they’re not supposed to.
    This is often reasonable, but not pragmatic.

Pragmatic (noun)

- **2:** relating to matters of fact or practical affairs often to the exclusion of intellectual or artistic matters **:** practical as opposed to idealistic.

  -- Merriam-Webster dictionary

What’s in the toolbox?

- For datagram protocols:
  - The packets can carry their own authentication, integrity, etc. E.g. IGP authentication fields.
  - DTLS (RFC 6347) can provide authentication, integrity, and confidentiality as a generic plumbing layer. There’s a cost though.
  - IPsec

What’s in the toolbox?

- For stream protocols:
  - TCP-MD5 (RFC 2385). Provides integrity, but doesn’t protect against IP header stuff. Deprecated due to being weak.
  - TLS (RFC 5246). Well deployed.
  - IPsec. Largely just works (see next slide), but has interesting caveats.
  - TCP-AO. (RFC 5925) Addresses many of the deficiencies of TCP-MD5, and adds key agility.

IPsec headaches

- ‘[…] then the specification of IPsec is tantamount to saying "turn off security" within this community’ – RFC 5406
- “All variants of IPsec have problems with NAT boxes” – RFC 5406
  - Although tunnel mode may work fine.
- Key management:
  - Yay, IKE! (simplifies things)
  - Boo, IKE! (“simplifies” things. Doesn’t scale. Slow session establishment. Bootstrapping issues, which are messy for routers; part of the motivation for the closed karp Working Group.)

TLS headaches

- Certificates are great for authentication!
- Certificate validity makes for headaches for very long lived connections.
  - BGP sessions could last for years!
  - Expiration, rollover, etc.
  - What to do about CRL or similar?
- Doesn’t protect TCP or IP header.

We have the tools in the toolbox, so what’s the issue?

- Proper use of these mechanisms requires prior thought.
  - Routing experts are not security experts.
  - Especially during initial code work, security “gets in the way”.
  - Developers generally would prefer to just open a socket, call connect() and get to work.
- The more transparent to the programmer a security mechanism is, the more likely it is to get used.
  - TCP-MD5 often involves just poking a ioctl() or similar.
  - IPsec modes often managed outside of the user TCP stack. E.g. tunnel mode.
  - TLS will usually push more of the complexity to the programmer. (Although stunnel, etc…)

TCP-AO

- Are there any implementations?
  - Despite being a very good answer to a number of headaches, there have to be implementations to realistically recommend using it!
- draft-bonica-tcp-auth-06 has vendor implementations to provide *something*, but there are interop issues.

Confidentiality Makes Operations Harder.

- The number one thing asked for by vendors when there are protocol issues between different types of equipment is a tcpdump.
  - It is possible to decrypt things if you have enough information, but this is at best a dark art.
- Cryptographic mechanisms that interfere with the streaming from applications to optimize compression, etc. may interfere with protocol keepalive timers.
  - As it is, pretty much every BGP developer on the planet is a entry-level expert in TCP headaches, especially windowing.
- Interferes with some Non-Stop Routing implementations.

Adding Security After the Fact

- Some mechanisms are easier to add in later than others. Unsurprisingly, these are the ones that didn’t require a lot of work to put in the first place.
  - TCP-MD5, TCP-AO easy. IPsec in tunnel mode, the user stack hides it from the user. No protocol change is usually required.
  - TLS will require a substantial amount of new code. Dealing with the new exception cases is “fun”. The protocol must now accommodate it.

TLS after the fact

- Protocols such as SMTP and PCEP added in TLS after the fact.
  - This was done by adding a new ability to “upgrade” a connection into a TLS protected one using the STARTTLS command in each protocol.
  - However, this is also vulnerable to attacks on its own since it’s not secured up front. https://www.eff.org/deeplinks/2014/11/starttls-downgrade-attacks
- The protocol also needs a good place to allow such a thing to be done.
  - Where does this go into BGP (RFC 4271)?
  - Ditto for LDP? (https://tools.ietf.org/html/draft-nslag-ietf-deprecate-md5-00.html)
  - BMP (RFC 7854), which is completely passive one-way?
  - TCP-ENO may help

TLS Operational Consequences

- Managing key chains for simple protocols such as the IGPs, TCP-MD5 is fairly simple.
  - Can be done locally on a given router. Likely to be centrally managed via provisioning system.
- Certificates for TLS require a completely different piece of infrastructure and arguably contribute to fragility in routing.
  - Internet of Things will have similar issue!
  - Automatic certificate management (acme) may simplify this.
- Certificate infrastructure is great for authentication and thus great for API use!
  - But is it really good for securing long lived resources like routing protocols?

Pragmatism

- Where these things leave us is “the Right Way” to do things, vs. what has been done.
- Drafts thus get to IESG review and transport security is missing and some flavor of ”this is coded and deployed” happens. IPsec or other appropriate optional text gets appended to the spec
    - The security “fig leaf”.
- TCP-AO is a good fit for many protocols, but the fact that it isn’t implemented keeps reducing us back to the same conversations.
- Code work must go on, and protocol implementers aren’t security people.
    - Meanwhile, actual security people are driven to alcoholism or other destructive behaviors.

What Should We Do?

- Transport security considerations have to be discussed UP FRONT. Adding it in after the fact doesn’t really work well.
- IETF protocol authors could use some simple boilerplate for common profiles of security applications.
  - These need to discuss bootstrapping, performance, what attack surface is being protected, and operational consequences.
  - RFC 3352 isn’t a lot of help here.
  - Similar to MIB boilerplate years ago.
- Early security review to help pick the right profiles.
- Encourage vendors to make “easy mode” internal APIs for their stacks to ease future protocol development.
  - Security has to be a ”required feature”.

Bibliography

- RFC 2385 - Protection of BGP Sessions via the TCP MD5 Signature Option.
- RFC 3352 - Guidelines for Writing RFC Text on Security Considerations
- RFC 4949 - Internet Security Glossary, Version 2.
- RFC 4953 - Defending TCP Against Spoofing Attacks
- RFC 5925 - The TCP Authentication Option
- RFC 5246 - The Transport Layer Security (TLS) Protocol, Version 1.2
- RFC 5406 - Guidelines for Specifying the Use of IPsec Version 2
- RFC 6347 - Datagram Transport Layer Security Version 1.2
- RFC 8253 - PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)
- Automatic Certificate Management Environment (ACME) https://tools.ietf.org/html/draft-ietf-acme-acme-10
- TCP-ENO: Encryption Negotiation Option https://tools.ietf.org/html/draft-ietf-tcpinc-tcpeno-18