ï»¿

           Guidance for External Pre-Shared Key (PSK) Usage in TLS

Abstract

   This document provides usage guidance for external Pre-Shared Keys
   (PSKs) in Transport Layer Security (TLS) 1.3 as defined in RFC 8446.
   It lists TLS security properties provided by PSKs under certain
   assumptions, then it demonstrates how violations of these assumptions
   lead to attacks. Advice for applications to help meet these
   assumptions is provided. This document also discusses PSK use cases
   and provisioning processes. Finally, it lists the privacy and
   security properties that are not provided by TLS 1.3 when external
   PSKs are used.

Table of Contents

1. Introduction

   This document provides guidance on the use of external Pre-Shared
   Keys (PSKs) in Transport Layer Security (TLS) 1.3 [RFC8446]. This
   guidance also applies to Datagram TLS (DTLS) 1.3 [RFC9147] and
   Compact TLS 1.3 [CTLS]. For readability, this document uses the term
   "TLS" to refer to all such versions.

   External PSKs are symmetric secret keys provided to the TLS protocol
   implementation as external inputs. External PSKs are provisioned out
   of band.

   This document lists TLS security properties provided by PSKs under
   certain assumptions and demonstrates how violations of these
   assumptions lead to attacks. This document discusses PSK use cases,
   provisioning processes, and TLS stack implementation support in the
   context of these assumptions. This document also provides advice for
   applications in various use cases to help meet these assumptions.

   There are many resources that provide guidance for password
   generation and verification aimed towards improving security.
   However, there is no such equivalent for external PSKs in TLS. This
   document aims to reduce that gap.

2. Conventions and Definitions

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" in this document are to be interpreted as described in
   BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all
   capitals, as shown here.

3. Notation

   For purposes of this document, a "logical node" is a computing
   presence that other parties can interact with via the TLS protocol.
   A logical node could potentially be realized with multiple physical
   instances operating under common administrative control, e.g., a
   server farm. An "endpoint" is a client or server participating in a
   connection.

4. PSK Security Properties

   The use of a previously established PSK allows TLS nodes to
   authenticate the endpoint identities. It also offers other benefits,
   including resistance to attacks in the presence of quantum computers;
   see Section 4.2 for related discussion. However, these keys do not
   provide privacy protection of endpoint identities, nor do they
   provide non-repudiation (one endpoint in a connection can deny the
   conversation); see Section 7 for related discussion.

   PSK authentication security implicitly assumes one fundamental
   property: each PSK is known to exactly one client and one server and
   they never switch roles. If this assumption is violated, then the
   security properties of TLS are severely weakened as discussed below.

4.1. Shared PSKs

As discussed in Section 5.1, to demonstrate their attack, [AASS19] describes scenarios where multiple clients or multiple servers share a PSK. If this is done naively by having all members share a common key, then TLS authenticates only group membership, and the security of the overall system is inherently rather brittle. There are a number of obvious weaknesses here:

1. Any group member can impersonate any other group member.

2. If a PSK is combined with the result of a fresh ephemeral key exchange, then compromise of a group member that knows the resulting shared secret will enable the attacker to passively read traffic (and actively modify it).

3. If a PSK is not combined with the result of a fresh ephemeral key exchange, then compromise of any group member allows the attacker to passively read all traffic (and actively modify it), including past traffic.

Additionally, a malicious non-member can reroute handshakes between honest group members to connect them in unintended ways, as described below. Note that a partial mitigation for this class of attack is available: each group member includes the Server Name Indication (SNI) extension [RFC6066] and terminates the connection on mismatch between the presented SNI value and the receiving member's known identity. See [Selfie] for details.

To illustrate the rerouting attack, consider three peers, A, B, and C, who all know the PSK. The attack proceeds as follows:

1. A sends a ClientHello to B.

2. The attacker intercepts the message and redirects it to C.

3. C responds with a second flight (ServerHello, ...) to A.

4. A sends a Finished message to B. A has completed the handshake, ostensibly with B.

5. The attacker redirects the Finished message to C. C has completed the handshake with A.

In this attack, peer authentication is not provided. Also, if C supports a weaker set of ciphersuites than B, cryptographic algorithm downgrade attacks might be possible. This rerouting is a type of identity misbinding attack [Krawczyk] [Sethi]. Selfie attack [Selfie] is a special case of the rerouting attack against a group member that can act as both a TLS server and a client. In the Selfie attack, a malicious non-member reroutes a connection from the client to the server on the same endpoint.

Finally, in addition to these weaknesses, sharing a PSK across nodes may negatively affect deployments. For example, revocation of individual group members is not possible without establishing a new PSK for all of the members that have not been revoked.

4.2. PSK Entropy

Entropy properties of external PSKs may also affect TLS security properties. For example, if a high-entropy PSK is used, then PSK-only key establishment modes provide expected security properties for TLS, including establishment of the same session keys between peers, secrecy of session keys, peer authentication, and downgrade protection. See Appendix E.1 of [RFC8446] for an explanation of these properties. However, these modes lack forward security. Forward security may be achieved by using a PSK-DH mode or by using PSKs with short lifetimes.

In contrast, if a low-entropy PSK is used, then PSK-only key establishment modes are subject to passive exhaustive search attacks,

which will reveal the traffic keys. PSK-DH modes are subject to
active attacks in which the attacker impersonates one side. The
exhaustive search phase of these attacks can be mounted offline if
the attacker captures a single handshake using the PSK, but those
attacks will not lead to compromise of the traffic keys for that
connection because those also depend on the Diffie-Hellman (DH)
exchange. Low-entropy keys are only secure against active attack if
a Password-Authenticated Key Exchange (PAKE) is used with TLS. At
the time of writing, the Crypto Forum Research Group (CFRG) is
working on specifying recommended PAKEs (see [CPACE] and [OPAQUE] for
the symmetric and asymmetric cases, respectively).

5. External PSKs in Practice

   PSK ciphersuites were first specified for TLS in 2005. PSKs are now
   an integral part of the TLS 1.3 specification [RFC8446]. TLS 1.3
   also uses PSKs for session resumption. It distinguishes these
   resumption PSKs from external PSKs that have been provisioned out of
   band. This section describes known use cases and provisioning
   processes for external PSKs with TLS.

5.1. Use Cases

   This section lists some example use cases where pairwise external
   PSKs (i.e., external PSKs that are shared between only one server and
   one client) have been used for authentication in TLS. There was no
   attempt to prioritize the examples in any particular order.

   * Device-to-device communication with out-of-band synchronized keys.
      PSKs provisioned out of band for communicating with known
      identities, wherein the identity to use is discovered via a
      different online protocol.

   * Intra-data-center communication. Machine-to-machine communication
      within a single data center or Point of Presence (PoP) may use
      externally provisioned PSKs; this is primarily for the purpose of
      supporting TLS connections with early data. See Section 8 for
      considerations when using early data with external PSKs.

   * Certificateless server-to-server communication. Machine-to-
      machine communication may use externally provisioned PSKs; this is
      primarily for the purposes of establishing TLS connections without
      requiring the overhead of provisioning and managing PKI
      certificates.

   * Internet of Things (IoT) and devices with limited computational
      capabilities. [RFC7925] defines TLS and DTLS profiles for
      resource-constrained devices and suggests the use of PSK
      ciphersuites for compliant devices. The Open Mobile Alliance
      Lightweight Machine-to-Machine (LwM2M) Technical Specification
      [LwM2M] states that LwM2M servers MUST support the PSK mode of
      DTLS.

   * Securing RADIUS [RFC2865] with TLS. PSK ciphersuites are optional
      for this use case, as specified in [RFC6614].

   * 3GPP server-to-user equipment authentication. The Generic
      Authentication Architecture (GAA) defined by 3GPP mentions that
      TLS PSK ciphersuites can be used between server and user equipment
      for authentication [GAA].

   * Smart Cards. The German electronic Identity (eID) card supports
      authentication of a card holder to online services with TLS PSK
      [SmartCard].

   * Quantum resistance. Some deployments may use PSKs (or combine
      them with certificate-based authentication as described in
      [RFC8773]) because of the protection they provide against quantum
      computers.

   There are also use cases where PSKs are shared between more than two

entities. Some examples below (as noted by Akhmetzyanova, et al.
[AASS19]):

* Group chats. In this use case, group participants may be
   provisioned an external PSK out of band for establishing
   authenticated connections with other members of the group.

* IoT and devices with limited computational capabilities. Many PSK
   provisioning examples are possible in this use case. For example,
   in a given setting, IoT devices may all share the same PSK and use
   it to communicate with a central server (one key for n devices),
   have their own key for communicating with a central server (n keys
   for n devices), or have pairwise keys for communicating with each
   other (n^2 keys for n devices).

5.2. Provisioning Examples

   The exact provisioning process depends on the system requirements and
   threat model. Whenever possible, avoid sharing a PSK between nodes;
   however, sharing a PSK among several nodes is sometimes unavoidable.
   When PSK sharing happens, other accommodations SHOULD be used as
   discussed in Section 6.

   Examples of PSK provisioning processes are included below.

* Many industrial protocols assume that PSKs are distributed and
   assigned manually via one of the following approaches: (1) typing
   the PSK into the devices or (2) using a trust-on-first-use (TOFU)
   approach with a device completely unprotected before the first
   login took place. Many devices have a very limited UI. For
   example, they may only have a numeric keypad or even fewer
   buttons. When the TOFU approach is not suitable, entering the key
   would require typing it on a constrained UI.

* Some devices provision PSKs via an out-of-band, cloud-based
   syncing protocol.

* Some secrets may be baked into hardware or software device
   components. Moreover, when this is done at manufacturing time,
   secrets may be printed on labels or included in a Bill of
   Materials for ease of scanning or import.

5.3. Provisioning Constraints

   PSK provisioning systems are often constrained in application-
   specific ways. For example, although one goal of provisioning is to
   ensure that each pair of nodes has a unique key pair, some systems do
   not want to distribute pairwise shared keys to achieve this. As
   another example, some systems require the provisioning process to
   embed application-specific information in either PSKs or their
   identities. Identities may sometimes need to be routable, as is
   currently under discussion for [EAP-TLS-PSK].

6. Recommendations for External PSK Usage

   Recommended requirements for applications using external PSKs are as
   follows:

1. Each PSK SHOULD be derived from at least 128 bits of entropy,
    MUST be at least 128 bits long, and SHOULD be combined with an
    ephemeral key exchange, e.g., by using the "psk_dhe_ke" Pre-
    Shared Key Exchange Mode in TLS 1.3 for forward secrecy. As
    discussed in Section 4, low-entropy PSKs (i.e., those derived
    from less than 128 bits of entropy) are subject to attack and
    SHOULD be avoided. If only low-entropy keys are available, then
    key establishment mechanisms such as PAKE that mitigate the risk
    of offline dictionary attacks SHOULD be employed. Note that no
    such mechanisms have yet been standardized, and further that
    these mechanisms will not necessarily follow the same
    architecture as the process for incorporating external PSKs
    described in [RFC9258].

2. Unless other accommodations are made to mitigate the risks of
    PSKs known to a group, each PSK MUST be restricted in its use to
    at most two logical nodes: one logical node in a TLS client role
    and one logical node in a TLS server role. (The two logical
    nodes MAY be the same, in different roles.) Two acceptable
    accommodations are described in [RFC9258]: (1) exchanging client
    and server identifiers over the TLS connection after the
    handshake and (2) incorporating identifiers for both the client
    and the server into the context string for an external PSK
    importer.

3. Nodes SHOULD use external PSK importers [RFC9258] when
    configuring PSKs for a client-server pair when applicable.
    Importers make provisioning external PSKs easier and less error-
    prone by deriving a unique, imported PSK from the external PSK
    for each key derivation function a node supports. See the
    Security Considerations of [RFC9258] for more information.

4. Where possible, the main PSK (that which is fed into the
    importer) SHOULD be deleted after the imported keys have been
    generated. This prevents an attacker from bootstrapping a
    compromise of one node into the ability to attack connections
    between any node; otherwise, the attacker can recover the main
    key and then re-run the importer itself.

6.1. Stack Interfaces

   Most major TLS implementations support external PSKs. Stacks
   supporting external PSKs provide interfaces that applications may use
   when configuring PSKs for individual connections. Details about some
   existing stacks at the time of writing are below.

   * OpenSSL and BoringSSL: Applications can specify support for
      external PSKs via distinct ciphersuites in TLS 1.2 and below.
      Also, they can then configure callbacks that are invoked for PSK
      selection during the handshake. These callbacks must provide a
      PSK identity and key. The exact format of the callback depends on
      the negotiated TLS protocol version, with new callback functions
      added specifically to OpenSSL for TLS 1.3 [RFC8446] PSK support.
      The PSK length is validated to be between 1-256 bytes (inclusive).
      The PSK identity may be up to 128 bytes long.

   * mbedTLS: Client applications configure PSKs before creating a
      connection by providing the PSK identity and value inline.
      Servers must implement callbacks similar to that of OpenSSL. Both
      PSK identity and key lengths may be between 1-16 bytes long
      (inclusive).

   * gnuTLS: Applications configure PSK values as either raw byte
      strings or hexadecimal strings. The PSK identity and key size are
      not validated.

   * wolfSSL: Applications configure PSKs with callbacks similar to
      OpenSSL.

6.1.1. PSK Identity Encoding and Comparison

   Section 5.1 of [RFC4279] mandates that the PSK identity should be
   first converted to a character string and then encoded to octets
   using UTF-8. This was done to avoid interoperability problems
   (especially when the identity is configured by human users). On the
   other hand, [RFC7925] advises implementations against assuming any
   structured format for PSK identities and recommends byte-by-byte
   comparison for any operation. When PSK identities are configured
   manually, it is important to be aware that visually identical strings
   may, in fact, differ due to encoding issues.

   TLS 1.3 [RFC8446] follows the same practice of specifying the PSK
   identity as a sequence of opaque bytes (shown as opaque
   identity<1..2^16-1> in the specification) that thus is compared on a

byte-by-byte basis. [RFC8446] also requires that the PSK identities
are at least 1 byte and at the most 65535 bytes in length. Although
[RFC8446] does not place strict requirements on the format of PSK
identities, note that the format of PSK identities can vary depending
on the deployment:

* The PSK identity MAY be a user-configured string when used in
   protocols like Extensible Authentication Protocol (EAP) [RFC3748].
   For example, gnuTLS treats PSK identities as usernames.

* PSK identities MAY have a domain name suffix for roaming and
   federation. In applications and settings where the domain name
   suffix is privacy sensitive, this practice is NOT RECOMMENDED.

* Deployments should take care that the length of the PSK identity
   is sufficient to avoid collisions.

6.1.2. PSK Identity Collisions

   It is possible, though unlikely, that an external PSK identity may
   clash with a resumption PSK identity. The TLS stack implementation
   and sequencing of PSK callbacks influences the application's behavior
   when identity collisions occur. When a server receives a PSK
   identity in a TLS 1.3 ClientHello, some TLS stacks execute the
   application's registered callback function before checking the
   stack's internal session resumption cache. This means that if a PSK
   identity collision occurs, the application's external PSK usage will
   typically take precedence over the internal session resumption path.

   Because resumption PSK identities are assigned by the TLS stack
   implementation, it is RECOMMENDED that these identifiers be assigned
   in a manner that lets resumption PSKs be distinguished from external
   PSKs to avoid concerns with collisions altogether.

7. Privacy Considerations

   PSK privacy properties are orthogonal to security properties
   described in Section 4. TLS does little to keep PSK identity
   information private. For example, an adversary learns information
   about the external PSK or its identifier by virtue of the identifier
   appearing in cleartext in a ClientHello. As a result, a passive
   adversary can link two or more connections together that use the same
   external PSK on the wire. Depending on the PSK identity, a passive
   attacker may also be able to identify the device, person, or
   enterprise running the TLS client or TLS server. An active attacker
   can also use the PSK identity to suppress handshakes or application
   data from a specific device by blocking, delaying, or rate-limiting
   traffic. Techniques for mitigating these risks require further
   analysis and are out of scope for this document.

   In addition to linkability in the network, external PSKs are
   intrinsically linkable by PSK receivers. Specifically, servers can
   link successive connections that use the same external PSK together.
   Preventing this type of linkability is out of scope.

8. Security Considerations

   Security considerations are provided throughout this document. It
   bears repeating that there are concerns related to the use of
   external PSKs regarding proper identification of TLS 1.3 endpoints
   and additional risks when external PSKs are known to a group.

   It is NOT RECOMMENDED to share the same PSK between more than one
   client and server. However, as discussed in Section 5.1, there are
   application scenarios that may rely on sharing the same PSK among
   multiple nodes. [RFC9258] helps in mitigating rerouting and Selfie-
   style reflection attacks when the PSK is shared among multiple nodes.
   This is achieved by correctly using the node identifiers in the
   ImportedIdentity.context construct specified in [RFC9258]. One
   solution would be for each endpoint to select one globally unique
   identifier to use in all PSK handshakes. The unique identifier can,

for example, be one of its Media Access Control (MAC) addresses, a
32-byte random number, or its Universally Unique IDentifier (UUID)
[RFC4122]. Note that such persistent, global identifiers have
privacy implications; see Section 7.

Each endpoint SHOULD know the identifier of the other endpoint with
which it wants to connect and SHOULD compare it with the other
endpoint's identifier used in ImportedIdentity.context. However, it
is important to remember that endpoints sharing the same group PSK
can always impersonate each other.

Considerations for external PSK usage extend beyond proper
identification. When early data is used with an external PSK, the
random value in the ClientHello is the only source of entropy that
contributes to key diversity between sessions. As a result, when an
external PSK is used more than one time, the random number source on
the client has a significant role in the protection of the early
data.

9. IANA Considerations

This document has no IANA actions.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
           Requirement Levels", BCP 14, RFC 2119,
           DOI 10.17487/RFC2119, March 1997,
           <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
           2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
           May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol
           Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018,
           <https://www.rfc-editor.org/info/rfc8446>.

[RFC9258] Benjamin, D. and C. A. Wood, "Importing External Pre-
           Shared Keys (PSKs) for TLS 1.3", RFC 9258,
           DOI 10.17487/RFC9258, July 2022,
           <https://www.rfc-editor.org/info/rfc9258>.

10.2. Informative References

[AASS19]   Akhmetzyanova, L., Alekseev, E., Smyshlyaeva, E., and A.
           Sokolov, "Continuing to reflect on TLS 1.3 with external
           PSK", April 2019, <https://eprint.iacr.org/2019/421.pdf>.

[CPACE]   Abdalla, M., Haase, B., and J. Hesse, "CPace, a balanced
           composable PAKE", Work in Progress, Internet-Draft, draft-
           irtf-cfrg-cpace-06, 24 July 2022,
           <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-
           cpace-06>.

[CTLS]   Rescorla, E., Barnes, R., Tschofenig, H., and B. M.
           Schwartz, "Compact TLS 1.3", Work in Progress, Internet-
           Draft, draft-ietf-tls-ctls-06, 9 July 2022,
           <https://datatracker.ietf.org/doc/html/draft-ietf-tls-
           ctls-06>.

[EAP-TLS-PSK]
           Mattsson, J. P., Sethi, M., Aura, T., and O. Friel, "EAP-
           TLS with PSK Authentication (EAP-TLS-PSK)", Work in
           Progress, Internet-Draft, draft-mattsson-emu-eap-tls-psk-
           00, 9 March 2020, <https://datatracker.ietf.org/doc/html/
           draft-mattsson-emu-eap-tls-psk-00>.

[GAA]   ETSI, "Digital cellular telecommunications system (Phase

              2+); Universal Mobile Telecommunications System (UMTS);
              LTE; 3G Security; Generic Authentication Architecture
              (GAA); System description", version 12.0.0, ETSI TR 133
              919, October 2014, <https://www.etsi.org/deliver/
              etsi_tr/133900_133999/133919/12.00.00_60/
              tr_133919v120000p.pdf>.

[Krawczyk]   Krawczyk, H., "SIGMA: The 'SIGn-and-MAc' Approach to
              Authenticated Diffie-Hellman and Its Use in the IKE
              Protocols", DOI 10.1007/978-3-540-45146-4_24, 2003,
              <https://link.springer.com/content/
              pdf/10.1007/978-3-540-45146-4_24.pdf>.

[LwM2M]   Open Mobile Alliance, "Lightweight Machine to Machine
              Technical Specification", version 1.0, February 2017,
              <http://www.openmobilealliance.org/release/LightweightM2M/
              V1_0-20170208-A/OMA-TS-LightweightM2M-
              V1_0-20170208-A.pdf>.

[OPAQUE]   Bourdrez, D., Krawczyk, H., Lewi, K., and C. A. Wood, "The
              OPAQUE Asymmetric PAKE Protocol", Work in Progress,
              Internet-Draft, draft-irtf-cfrg-opaque-09, 6 July 2022,
              <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-
              opaque-09>.

[RFC2865]   Rigney, C., Willens, S., Rubens, A., and W. Simpson,
              "Remote Authentication Dial In User Service (RADIUS)",
              RFC 2865, DOI 10.17487/RFC2865, June 2000,
              <https://www.rfc-editor.org/info/rfc2865>.

[RFC3748]   Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H.
              Levkowetz, Ed., "Extensible Authentication Protocol
              (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004,
              <https://www.rfc-editor.org/info/rfc3748>.

[RFC4122]   Leach, P., Mealling, M., and R. Salz, "A Universally
              Unique IDentifier (UUID) URN Namespace", RFC 4122,
              DOI 10.17487/RFC4122, July 2005,
              <https://www.rfc-editor.org/info/rfc4122>.

[RFC4279]   Eronen, P., Ed. and H. Tschofenig, Ed., "Pre-Shared Key
              Ciphersuites for Transport Layer Security (TLS)",
              RFC 4279, DOI 10.17487/RFC4279, December 2005,
              <https://www.rfc-editor.org/info/rfc4279>.

[RFC6066]   Eastlake 3rd, D., "Transport Layer Security (TLS)
              Extensions: Extension Definitions", RFC 6066,
              DOI 10.17487/RFC6066, January 2011,
              <https://www.rfc-editor.org/info/rfc6066>.

[RFC6614]   Winter, S., McCauley, M., Venaas, S., and K. Wierenga,
              "Transport Layer Security (TLS) Encryption for RADIUS",
              RFC 6614, DOI 10.17487/RFC6614, May 2012,
              <https://www.rfc-editor.org/info/rfc6614>.

[RFC7925]   Tschofenig, H., Ed. and T. Fossati, "Transport Layer
              Security (TLS) / Datagram Transport Layer Security (DTLS)
              Profiles for the Internet of Things", RFC 7925,
              DOI 10.17487/RFC7925, July 2016,
              <https://www.rfc-editor.org/info/rfc7925>.

[RFC8773]   Housley, R., "TLS 1.3 Extension for Certificate-Based
              Authentication with an External Pre-Shared Key", RFC 8773,
              DOI 10.17487/RFC8773, March 2020,
              <https://www.rfc-editor.org/info/rfc8773>.

[RFC9147]   Rescorla, E., Tschofenig, H., and N. Modadugu, "The
              Datagram Transport Layer Security (DTLS) Protocol Version
              1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022,
              <https://www.rfc-editor.org/info/rfc9147>.

[Selfie]   Drucker, N. and S. Gueron, "Selfie: reflections on TLS 1.3
             with PSK", DOI 10.1007/s00145-021-09387-y, May 2021,
             <https://eprint.iacr.org/2019/347.pdf>.

[Sethi]   Sethi, M., Peltonen, A., and T. Aura, "Misbinding Attacks
             on Secure Device Pairing and Bootstrapping",
             DOI 10.1145/3321705.3329813, May 2019,
             <https://arxiv.org/pdf/1902.07550>.

[SmartCard]
             Bundesamt für Sicherheit in der Informationstechnik,
             "Technical Guideline TR-03112-7 eCard-API-Framework -
             Protocols", version 1.1.5, April 2015, <https://www.bsi.bu
             nd.de/SharedDocs/Downloads/DE/BSI/Publikationen/
             TechnischeRichtlinien/TR03112/TR-
             03112-api_teil7.pdf?__blob=publicationFile&v=1>.

Acknowledgements

Authors' Addresses

   Russ Housley
   Vigil Security, LLC
   Email: housley@vigilsec.com

   Jonathan Hoyland
   Cloudflare Ltd.
   Email: jonathan.hoyland@gmail.com

   Mohit Sethi
   Aalto University
   Email: mohit@iki.fi

   Christopher A. Wood
   Cloudflare
   Email: caw@heapingbits.net