A-SIT Technology
E-Government Innovationszentrum

21. Juli 2021 Studie · Web & Cloud · Digitale Identität · 13 Min. Lesezeit

Ressourcen schonende DTLS Handshakes für IoT Sensoren

Archiviert. Dieser Inhalt wird nicht mehr aktualisiert. Er bleibt zur Information verfügbar.

Stand 2021-07-17
Quelle https://technology2.a-sit.at/veroeffentlichungen/ressourcen-schonende-dtls-handshakes-fuer-iot-sensoren/

Von Jahr zu Jahr messen mehr und mehr intelligente Sensoren unser Leben und beeinflussen so viele Teile unseres täglichen Lebens. Die Sensoren arbeiten dabei meist mit recht limitierten Ressourcen, da sie klein und mobil sein müssen. Andererseits sind sie meist Teil des „Internet of Things" (IoT): Intelligente Sensoren übertragen ihre Daten zu Internetdiensten, die zunehmend auf Cloud-orientierten Lösungen setzen. So treffen beide Eigenschaften als Gegensätze aufeinander: Ein möglichst Ressourcen-schonender Betrieb trifft auf die Notwendigkeit einer sicheren Kommunikation über das Internet.

Die traditionelle Zertifikats-basierte Authentifizierung und der Schlüsselaustausch können für solche Ressourcen-limitierten Geräten zur Belastung werden. In diesem Projektbericht beschreiben wir aktuelle Forschung in diesem Bereich und skizzieren die Konzipierung einer eigenen Delegation-basierter Lösung zur beschriebenen Problemstellung.

1. Einleitung

Immer mehr Geräte unseres täglichen Gebrauchs sind smarte Geräte. Sie haben die Fähigkeit, sich problemlos mit anderen Geräten zu verbinden, untereinander zu kommunizieren und von einem hohen Grad an Automatisierung zu profitieren. Speziell im letzten Jahrzehnt hat die Forschung in diesem Gebiet stark angezogen [1]. Doch so komfortabel das Leben in einer intelligenten Umgebung ist, so gefährlich kann sie für die Sicherheit der angesammelten Daten (nötig für das intelligente Auftreten) sein. Da speziell „Internet of Things"(IoT)-Geräte (wie intelligente Sensoren) mit begrenzten Rechenressourcen arbeiten, treten immer wieder Sicherheitsprobleme auf [2]. Oft hat dies mit nicht ausreichender kryptographischer Absicherung der Internetverbindungen zu tun, was zum Verlust sensibler Daten führen kann.

Restuccia et al. [3] zeigen in einer Analyse der nötigen Ressourcen von TLS Handshakes auf IoT-Geräten, dass das Nutzen von Public-Key Kryptographie viel mehr Energie verbraucht als das Nutzen von Pre-Shared Keys (PSK). Darum wurde in der Vergangenheit oft versucht mit lokalen Gateways und Proxys, die

direkt Verbindung von IoT-Geräten mit dem Internet zu umgehen. Hierbei wird das End-to-end Versprechen von TLS aber vernachlässigt und neue Sicherheitsprobleme entstehen.

In den letzten Jahren erfreuen sich Cloud-basierte Lösungen größerer Beliebtheit im IoT-Sektor. Minderaud et al. [4] befanden in einer Studie, dass 34 von 39 IoT Frameworks zentralisiert oder Cloud-basiert sind. Solche Architekturen haben den entscheidenden Vorteil, dass IoT-Geräte oft nicht Standby-Server sind (wie z.B. Sensoren), nur in regelmäßigen Intervallen aus dem Schlaf aufwachen um Daten an die Cloud senden und so Ressourcen zu sparen. Für unsere Forschungszwecke berücksichtigen wir genau so ein Device-to-Cloud Szenario mit periodisch schlafenden IoT-Sensoren (cf. IoT Netzwerk Modelle [5]).

Obwohl Cloud-basierte Lösungen weit verbreitet sind, leiden viele unter unzureichender IT-Sicherheit. Man fand Geräte die ihre Kommunikation weder verschlüsseln noch authentifizieren [6]. Sivanathan et al. [7] befanden in ihrer Analyse, dass 39% der getesteten IoT-Geräte nicht auf TLS oder ähnlichen Protokollen zurückgreifen. Da jedoch solche Geräte oft sensible Nutzerdaten über das Internet übertragen, ist es wichtig, adäquate Sicherheitsmaßnahmen zu ergreifen, trotz der limitierten Energieressourcen.

In diesem Projektbericht versuchen wir, bestehende Lösungen zu dieser Problemstellung auf zu zeigen (siehe Kapitel 3). Auch skizzieren wir eine eigene Lösung (siehe Kapitel 4), welche in einer folgenden wissenschaftlichen Arbeit genauer behandelt wird [8]. Ziel ist es, Ressourcen-limitierten IoT-Geräten einen End-to-end gesicherten Verbindungsaufbau zu ermöglichen. Unser Lösungsansatz beinhaltet die Auslagerung der von Ressourcen-intensiven Operationen des TLS Handshakes an einen Sicherheitsagenten (SA).

2. Hintergrundinformationen

Dieses Kapitel gibt grobe Hintergrundinformationen zu grundlegenden Bausteinen, die in diesem Bericht verwendet werden, um die Forschungsarbeit zu beschreiben.

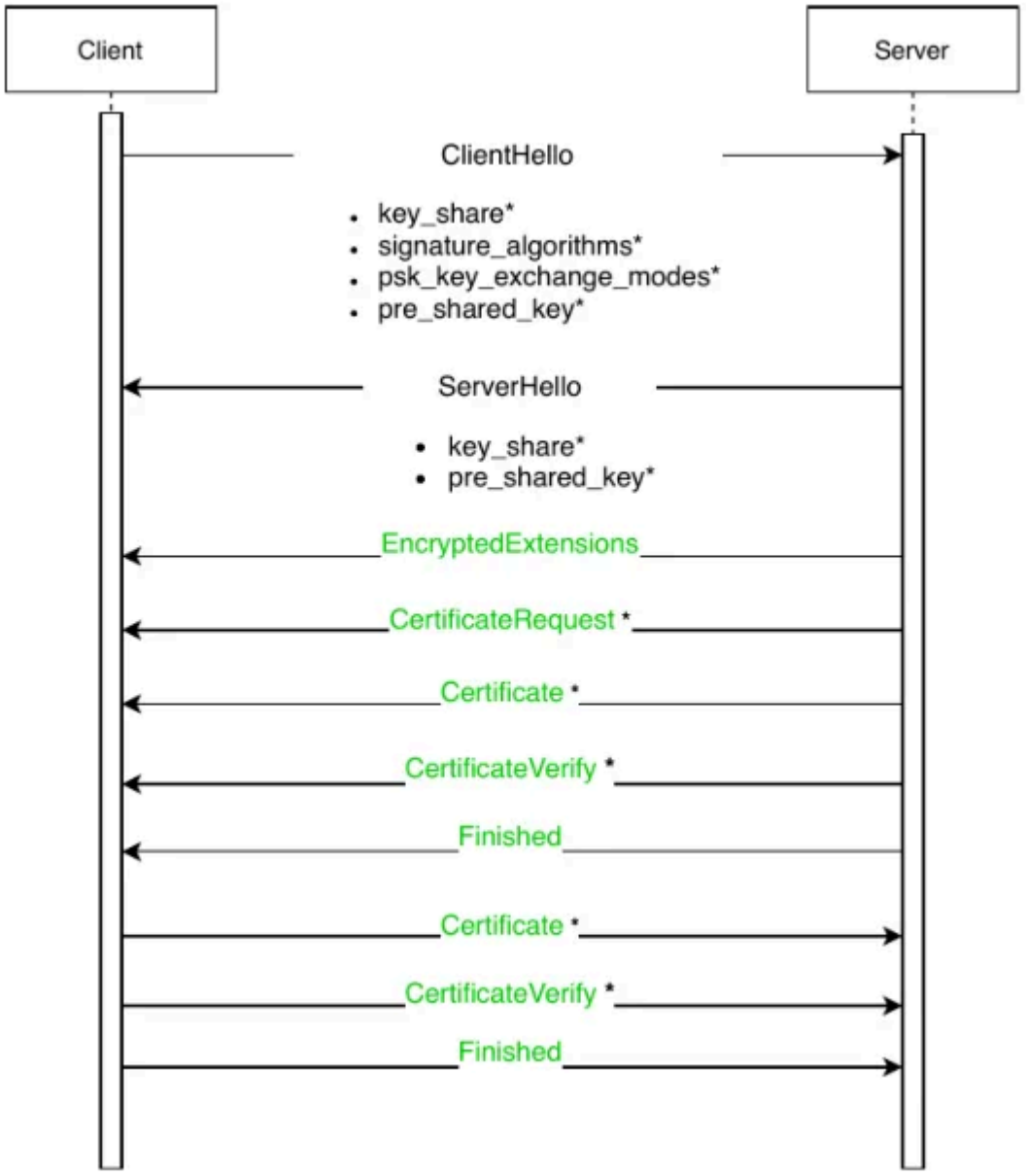
2.1. Transport Layer Security

Transport Layer Security (TLS) ist ein Protokoll, das der Kommunikationssicherheit über einen öffentlichen Kanal wie das Internet dient. TLS wurde von der Internet Engineering Task Force (IETF) standardisiert und gewährleistet eine End-to-end Verschlüsselung, Authentifizierung und Integrität der Kommunikation. In einer ersten Handshake-Phase (siehe Abbildung 1), die aus Authentifizierung und Schlüsselaustausch besteht, wird eine sichere Verbindung eingerichtet. Man kann zwischen zwei Arten des Handshakes unterscheiden: Zum einen gibt es den Zertifikatsbasierten Handshake, der auch als „Full Handshake“ bezeichnet wird. Er verwendet Public-Key Kryptographie zur Authentifizierung der beiden Seiten und einen Diffie-Hellman-Schlüsselaustausch, um einen gemeinsamen, geheimen Schlüssel zwischen den Seiten zu etablieren. Es gibt aber auch den Pre-Shared-Key (PSK) Handshake, der das sogenannte „Session Resumption“ ermöglicht. Hier wird ein bereits vereinbarter symmetrischer Schlüssel genutzt, um die sichere Verbindung erneut aufzubauen [9].

*mit * markiert sind optional oder situationsab- hängig. Nachrichten in grün sind verschlüsselt.*

2.2. Datagramm TLS

Ursprünglich wurde TLS für den Betrieb auf einem Transport-Layer Protokoll mit geordneter und zuverlässiger Zustellung wie TCP entwickelt. Datagramm TLS (DTLS) [10] wurde jedoch eingeführt, um die bekannten TLS-Sicherheitseigenschaften auch auf nicht-verbindungsorientierte Transportprotokolle wie UDP zu bringen. Auf diese Weise wurden mehrere Erweiterungen wie



DDOS-Schutz, Nachrichtenverlust und Duplikationserkennung einbezogen, während bestehende Codebasen wiederverwendet werden konnten.

2.3. Identity-Based Signatures

Shamir [11] stellte 1984 erstmals eine Notation für Identitäts-basierte Signaturen vor. Erste Lösungsansätze wurden von Fiat et al. [12] und Simmons et al. [13] geliefert. Aber erst im Jahr 2001 wurden Weil-Paarungen für die Identitäts-basierte Kryptographie eingesetzt [14]. Im Allgemeinen bestehen Identitäts-basierte

Signaturverfahren (IBS) aus fünf Algorithmen: *Setup*, *Gen*, *Del*, *Sign* und *Verify* . *Setup* initialisiert die öffentlichen Parameter, die die Einstellung des Signiervorgangs festlegen. *Gen* erzeugt das Schlüsselpaar (geheimer Hauptschlüssel und öffentlicher Schlüssel). *Del* leitet einen Delegationsschlüssel aus dem geheimen Master-Schlüssel und einer bestimmten Identität ab. *Sign* erzeugt eine Signatur der bereitgestellten Nachricht. *Verify* schließlich prüft die Gültigkeit der Signatur bezüglich einer bereitgestellten Nachricht unter Berücksichtigung des ursprünglichen öffentlichen Schlüssels und der Identität.

2.4. Time-Bound Identity-Based Signature

Zeitgebundene Identitäts-basierte Signaturen (TBIBS) werden aus Identitäts-basierten Signaturen konstruiert, wie in einer früheren Arbeit von Alber et al. gezeigt wurde [15]. Die Konstruktion ist nachweislich EUF-CMA-sicher in Bezug auf Epochen und Identitäten. Die funktionale Abbildung zwischen TBIBS und einem Identitäts-basierten Signaturschema läuft auf eine Verkettung von Epoche und Identität hinaus, d.h. da eine Epoche die Zeit partitioniert, trägt sie eine zusätzliche Zeitinformation zur Identität bei.

3. Related Work

In diesem Kapitel werden ausgewählte wissenschaftliche Arbeiten vorgestellt, welche zur beschriebenen Problemstellung einen Lösungsansatz bereitstellen.

Li et al. [16] schlagen ein Identity-based Authenticated Key Agreement (IBAKA) für TLS vor. Das iTLS genannte Konzept ermöglicht Zero-Round-Trip-Time und perfekte Forward Secrecy. Die System-weiten Parameter des Identitäts-basierten Algorithmus ermöglichen die Authentifizierung ganz ohne Zertifikate. Eine Implementierung des Konzeptes zeigte schnellere Handshakes dank Zero-Round-Trip-Time, aber auch einen höheren Datenüberhang. Der größte Nachteil ist jedoch, dass eine dritte Partei die System-weiten, öffentlichen Parameter des Identitäts-basierten Algorithmus generieren und verteilen muss, was es für IoT-Systemen unpraktikabel macht.

Einen anderen Weg beschritten Hummen et al. [17] [18] und später Moosavi et al. [19]. Der liegt im Auslagern von Ressourcen-fressenden Operationen zu einem dritten Gerät im selben lokalen Netzwerk. So übernimmt das vertrauenswürdige Drittgerät z.B. den Zertifikats-basierten Handshake und liefert dem IoT-Gerät einen PSK-Schlüssel, welcher zum Wiederaufnehmen der Verbindung verwendet werden kann. Leider missachtet dieses Vorgehen Grundlagen der Public-Key Kryptographie [20]: Das Drittgerät kann nur dann im Namen des IoT-Geräts authentifizieren, wenn das IoT-Gerät dem Drittgerät den privaten Schlüssel liefert. Das führt zu Sicherheitsproblemen, die bis hin zum kompletten Kontrollverlust über die Kommunikation reichen.

Cho et al. [21] prangern diese Schlüsselhinterlegung in ihrer Arbeit an und unterbreiten einen Gegenvorschlag: eine TLS Splitting Technik ähnlich zu KeylessSSL [1] soll verwendet werden. Bei diesem Konzept geht es darum, dass das Drittgerät im lokalen Netzwerk nicht den Schlüssel erhält, jedoch bei der Authentifizierungsoperation angelangt, Rücksprache mit dem IoT-Gerät hält, wobei das IoT-Gerät dann die notwendige Signatur an das Drittgerät liefert. Dieses Konzept hat aber zwei entscheidende Nachteile: Zum einen wird ein neue Umlaufsverzögerung eingeführt, zum anderen wird eine bereits ausgelagerte, kostspielige Operation wieder auf IoT-Gerät rückverlagert.

1 https://www.cloudflare.com/ssl/keyless-ssl/, zugegriffen am 17. Juni 2021 Zur Wiederaufnahme von Verbindungen (Session Resumption) fanden wir besonders folgende Arbeiten für die aufgezeigte Problemstellung von Bedeutung: Aviram et al. [22] präsentierten auf der Eurocrypt 2019 eine foward- und

replay-sichere Möglichkeit den Verbindungswiederaufbau mit rein symmetrischen Schlüsseln zu gestalten. Das hat den Vorteil, dass keine rechnerisch aufwendige, asymmetrische Kryptographie notwendig ist und somit der Verbindungswiederaufbau kostengünstig und sicher auf dem IoT-Gerät oder -Sensor ausführbar ist. Auch Tange et al. [23] zeigten ein symmetrisches Konzept zum Verbindungswiederaufbau, welches forward- und replay-sicher ist. Es basiert auf den Double Ratchet Algorithmus von Perin et al. [24].

4. Konzept

Dieses Kapitel skizziert unser generisches Konzept mit dem Ziel, eine sichere und performante Lösung für IoT-Geräte zur effizienten Kommunikation über das Internet zu erreichen. Wir stellen zunächst die beteiligten Akteure vor (siehe Abschnitt 4.1), dann legen wir die Anforderungen fest (siehe Abschnitt 4.2), die wir mit unserem Ansatz erreichen wollen. Schließlich beschreiben wir in Abschnitt 4.3 die Phasen unseres Konzepts und erläutern in Abschnitt 4.4, wie die Maßnahmen, die unsere Anforderungen erfüllen, aussehen.

4.1. Akteure

- **IoT-Gerät:** Das Ressourcen-limitierte IoT-Gerät ist Teil eines intelligenten Sensornetzwerks, in dem die Geräte nicht ständig eingeschaltet sind, sondern ihren Ruhezustand periodisch unterbrechen, um aktuelle Daten zu versenden.
- **Sicherheitsagent (SA):** Der Sicherheitsagent ist ein vertrauenswürdiges Drittgerät im lokalen Netzwerk des IoT-Geräts, der die zertifikatsbasierten Handshakes für das IoT-Gerät durchführt.
- **Cloud:** Die Cloud ist ein verteilter Rechenzentrumsdienst, der die eingehenden Daten von IoT-Geräten zentral sammelt, Operationen darauf durchführt und autorisierten Clients bzw. Benutzern zur Verfügung stellt.

4.2. Anforderungen

Wir haben folgende Anforderungen identifiziert, die unser Ansatz erfüllen muss, um die Problemstellung zu lösen. Diese Anforderungen ergaben sich aus der Untersuchung von ähnlichen Arbeiten (siehe Abschnitt 3).

- **Problem der Schlüsselhinterlegung:** Das IoT-Gerät sollte seinen privaten Schlüssel nicht an den Sicherheitsagenten übertragen müssen, um einen vollständigen Handshake zu delegieren.
- **Geringere Belastung für das intelligente Gerät:** Der Sicherheitsagent sollte die teuren Teile eines Handshakes übernehmen, damit das IoT-Gerät Energie sparen kann.
- **Widerrufbare Delegation:** Das IoT-Gerät sollte in der Lage sein, die Delegation bei Bedarf zu widerrufen.
- **Intakte End-to-End-Sicherheit von TLS:** Die bewährte Sicherheit von TLS sollte nicht durch Änderungen am Protokoll gebrochen werden.

4.3. Phasen

Unser Konzept lässt sich in 4 verschiede Phasen gliedern: Einer Einrichtungsphase, einer Bereitstellungsphase, einer Handshakephase und einer Wiederaufbauphase. Im Detail ist das Konzept in dem dazugehörigen Paper beschrieben [8].

Einrichtungsphase In der Einrichtungsphase wird die Möglichkeit einer sicheren Verbindung zwischen IoT-Gerät und Sicherheitsagent (SA) geschaffen. Dies kann z.B. über ein NFC-Pairing von Statten gehen. Außerdem stellt eine Challenge-Response sicher, dass der Sicherheitsagent ein zertifiziertes Gerät ist, welches adäquat für den Einsatz ist. Nachdem ein PSK-Schlüssel beim ersten Verbindungsaufbau ausgetauscht wurde, kann die Verbindung zwischen den Geräten immer wieder neu aufgebaut werden.

Bereitstellungsphase In dieser Phase erstellt das IoT-Gerät eine Delegation mit Hilfe von Identitäts-basierter Kryptographie und sendet diese über den zuvor gesicherten Kanal an den Sicherheitsagenten. Der Sicherheitsagent kann dann die Delegation nutzen um im Namen des IoT-Geräts den Handshake zu authentifizieren. Die Delegation an sich hat nur eine begrenzte Lebensdauer und muss dann vom IoT-Gerät erneuert werden. Will das IoT-Gerät seine Zustimmung dem Sicherheitsagenten entziehen, unterlässt es die Delegation zu erneuern.

Handshakephase Sobald das IoT-Gerät das erste Mal mit der Cloud kommunizieren muss, versucht der SA eine Verbindung im Namen des IoT-Geräts aufzubauen. Dazu nutzt er die zuvor erhaltene Delegation. Der SA eröffnet einen Zertifikats-basierten Handshake und nutzt die Delegation zur Authentifizierung. Nach dem Handshake erhält der SA ein Session Ticket von der Cloud, das den PSK-Schlüssel enthält. Das Ticket wird nun an das IoT-Gerät weitergeleitet, welches es in der nächsten Phase verwendet.

Wiederaufbauphase Will das IoT-Gerät nun eine direkte Verbindung zu der Cloud herstellen, kann es das erhaltene Session Ticket dafür nutzen. TLS 1.3 unterstützt forward-sichere Session Resumption, welches wir nutzen, um den SA aus der sichern Verbindung auszuschließen. In Kapitel 3 haben wir bereits von effiziente Session Resumption Methoden gehört, die auf IoT-Geräten angewendet werden können. So kann das IoT-Gerät oder der IoT-Sensor nun zu einem beliebigen Zeitpunkt die Verbindung neu aufbauen ohne groß Ressourcen aufzuwenden. Schlägt der Wiederaufbau fehl, so kehrt das System zur Handshakephase zurück, auf dass der SA erneut ein Session Ticket liefert.

4.4. Maßnahmen

In diesem Unterkapitel werden die Maßnahmen unseres Konzepts dargelegt und ausgeführt wie diese Maßnahmen unsere definierten Anforderungen erfüllen.

- Dank eines zeitgebundenen Identität-basierten Signaturschemas sind wir in der Lage, den Sicherheitsagenten mit einer kurzlebigen Delegation auszustatten. Auf diese Weise **benötigt der Agent nicht den privaten Schlüssel** des Smart Devices, um einen Handshake erfolgreich durchzuführen.
- Außerdem sind **keine Public-Key-Operationen** bei jedem Zertifikats-basierten Verbindungsaufbau auf dem Smart-Gerät erforderlich, sondern nur die anfängliche Authentifizierung des Sicherheitsagenten und die Erstellung der Delegation. Auf diese Weise kann die Belastung des Smart-Geräts verringert werden.

- Des Weiteren ist die **Delegation kurzlebig** . Abhängig von der eingestellten Lebensdauer kann das Smart-Gerät also die Delegation widerrufen, indem es nach Ablauf der Delegation keine Folge-Delegation ausstellt.
- Um den Handshake mit einem TBIBS-Schema zu signieren, müssen keine Änderungen am TLS-Stack vorgenommen werden; lediglich das Signaturschema muss zu den unterstützten hinzugefügt werden. Somit bleiben **die Sicherheitseigenschaften von TLS unangetastet** .

5. Schlussfolgerungen

Dieser Bericht zeigt Möglichkeiten auf, wie man IoT-Geräte oder -Sensoren entlasten und trotzdem sicher über das Internet verbindet. Auch skizzieren wir unser eigenes Konzept, welches im Detail in einer ausführlichen, wissenschaftlichen Arbeit veröffentlicht wird [8]. Im Kern werden alle aufwendigen Operationen des TLS Handshakes auf ein Drittgerät ausgelagert, um Energie am IoT-Gerät selbst zu sparen. Unser Konzept schafft dies ohne Schlüsselweitergabe oder das Zurückverlagern von aufwendigen Operationen auf das IoT-Gerät. Auch nützen wir Session Resumption effizient aus, um erneute Verbindungen zwischen dem IoT-Gerät und der Cloud Ressourcen-schonend wiederaufzubauen.

Beim Abschluss dieses Projekts stellten sich folgende Anschlussfragen: Wie oft werden Verbindungen in IoT-Frameworks wiederaufgebaut? Und was sind die Gründe für den fehlgeschlagenen Wiederaufbau? Weiters sollte auch Post-Quanten Sicherheit für IoT-Sensoren genauer betrachtet werden.

6. Bibliographie

[1] L. D. Xu, W. He und S. Li, „Internet of Things in Industries: A Survey," *{IEEE} Trans. Ind. Informatics,* Bd. 10, pp. 2233-2243, 2014.

[2] Y. Lu und L. D. Xu, „Internet of Things (IoT) Cybersecurity Research: A Review of Current Research Topics," *{IEEE} Internet Things J.,* Bd. 6, pp. 2103-2115, 2019.

[3] G. Restuccia, H. Tschofenig und E. Baccelli, „Low-Power IoT Communication Security: On the Performance of DTLS and TLS 1.3," in *{PEMWN}*, 2020.

[4] J. Mineraud, O. Mazhelis, X. Su und S. Tarkoma, „A gap analysis of Internet-of-Things platforms," *Comput. Commun.,* Bde. %1 von %289-90, pp. 5-16, 2016.

[5] H. Tschofenig, J. Arkko, D. Thaler und D. McPherson, „Architectural Considerations in Smart Object Networking," *{RFC},* Bd. 7452, pp. 1-24, 2015.

[6] M. B. Barcena und C. Wueest, „Insecurity in the Internet of Things," *Security response, Symantec,* 2015.

[7] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath und V.

Sivaraman, „Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *{IEEE} Trans. Mob. Comput.,* Bd. 18, pp. 1745-1759, 2019.

[8] L. Alber, „Towards a Delegation Scheme for IoT Devices using Identity-Based Signatures," in *in submission*, 2021.

[9] E. Rescorla, „The Transport Layer Security (TLS) Protocol Version 1.3,“ *{RFC},* Bd. 8446, pp.

1-160, 2018.

[10] E. Rescorla, H. Tschofenig und N. Modadugu, „The Datagram Transport Layer Security (DTLS) Protocol Version 1.3,“ Internet Engineering Task Force, 2021.

[11] A. Shamir, „Identity-Based Cryptosystems and Signature Schemes,“ in *{CRYPTO}*, 1984.

[12] A. Fiat und A. Shamir, „How to Prove Yourself: Practical Solutions to Identification and Signature Problems,“ in *{CRYPTO}*, 1986.

[13] G. J. Simmons und G. B. Purdy, „Zero-Knowledge Proofs of Identity And Veracity of Transaction Receipts,“ in *{EUROCRYPT}*, 1988.

[14] D. Boneh und M. K. Franklin, „Identity-Based Encryption from the Weil Pairing,“ in *{CRYPTO}*, 2001.

[15] L. Alber, S. More und S. Ramacher, „Short-Lived Forward-Secure Delegation for TLS,“ *CoRR,* Bd. abs/2009.02137, 2020.

[16] P. Li, J. Su und X. Wang, „iTLS: Lightweight Transport-Layer Security Protocol for IoT With Minimal Latency and Perfect Forward Secrecy,“ *{IEEE} Internet Things J.,* Bd. 7, pp. 6828 6841, 2020.

[17] R. Hummen, J. H. Ziegeldorf, H. Shafagh, S. Raza und K. Wehrle, „Towards viable certificate based authentication for the internet of things,“ in *Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy*, 2013.

[18] R. Hummen, H. Shafagh, S. Raza, T. Voigt und K. Wehrle, „Delegation-based authentication and authorization for the IP-based Internet of Things,“ in *{SECON}*, 2014.

[19] S. R. Moosavi, T. N. Gia, E. Nigussie, A.-M. Rahmani, S. Virtanen, H. Tenhunen und J.

Isoaho, „Session Resumption-Based End-to-End Security for Healthcare Internet-of-Things,“ in *CIT/IUCC/DASC/PICom*, 2015.

[20] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley und W. T. Polk, „Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,“ *{RFC},* Bd. 5280, pp. 1-151, 2008.

[21] E. Cho, M. Park, H. Lee, J. Choi und T. T. Kwon, „D2TLS: delegation-based DTLS for cloud based IoT services,“ in *IoTDI*, 2019.

[22] N. Aviram, K. Gellert und T. Jager, „Session Resumption Protocols and Efficient Forward Security for TLS 1.3 0-RTT,“ in *{EUROCRYPT} {(2)}*, 2019.

[23] K. Tange, D. Howard, T. Shanahan, S. Pepe, X. Fafoutis und N. Dragoni, „rTLS: Lightweight TLS Session Resumption for Constrained IoT Devices,“ in *{ICICS}*, 2020.

[24] T. Perrin und M. Marlinspike, „The double ratchet algorithm,“ *GitHub wiki,* 2016.