Forschungsbericht WS 2016/2017

Network Time Security (NTS) – Unicast-Modus über NTP (Teil 2)

Prof. Dr.-Ing. Rainer Bermbach

Einleitung

Wie schon im Bericht über die Forschungsaktivitäten im Sommersemester 2016 erläutert [1], sind die üblichen Standards zur Zeitübertragung wie NTP und PTP [2, 3] im Allgemeinen nicht ausreichend gegen Angriffe und Manipulationen geschützt. Daher benötigen sicherheitskritische Anwendungen, wie z.B. im Smart Metering [4, 5], zusätzliche Schutzmechanismen. Bisherige Verfahren waren nicht ausreichend sicher [6] oder erfordern einen zu hohen, in der Praxis nicht sinnvollen Aufwand.

Deshalb schlugen Sibold (PTB), Röttger (Google) und Teichel (PTB) das Protokoll Network Time Security (NTS) vor, das als IETF Draft [7] in der Diskussion und Weiterentwicklung ist. Ziel ist es, eine sichere Authentifizierung des Zeitdienstes sowie eine Überprüfung der Integrität der Zeitinformation zu erreichen.

Im Rahmen der Analysen des Unicast-Modus des NTS-Protokolls im SS2016 entstanden detaillierte Sequenz- und Aktivitätsdiagramme sowie Übersichten zu sämtlichen ASN.1-Strukturen, die in NTS zur Anwendung kommen. Letztere umfassen die kompletten Beschreibungen der NTS-Nachrichtenobjekte, der verwendeten CMS-Strukturen und der X.509-Zertifikate. Einen Überblick über den Aufbau und die Abläufe in NTS gibt der letzte Bericht [1]. Aufwändig waren auch die Nachverfolgung und das Einpflegen der vielen Änderungen in den Drafts. Außerdem fanden zahlreiche Diskussionen mit den an der Spezifikationsarbeit Beteiligten der PTB statt.

Durch die entsprechenden Vorarbeiten konnte sogar schon mit der Planung der Implementation begonnen werden. Gleichzeitig starteten auch Arbeiten an einer spezifischen NTP-Implementierung mit einer Schnittstelle zu NTS, damit der tatsächliche Betrieb bei Vorliegen einer NTS-Implementierung in vollem Umfang getestet werden kann.

Für den zweiten Teil des Projektes im zurückliegenden WS2016/2017 waren eine Implementierung und ein entsprechender Test einer realen Client/Server-Kommunikation vorgesehen — wenn möglich auch „gegen“ eine Implementierung der ebenfalls an einer NTS-Version arbeitenden Network Time Foundation (NTF). Außerdem sollte die spezielle NTP-Implementierung mit NTS-Interface realisiert werden.

Softwarearchitektur

Basierend auf den Vorarbeiten entstand die Umsetzung [8] des Konzeptes als hierarchisch gegliedertes Programmsystem in C++ unter Linux (s. Abb. 1). Ein übergeordnetes Hauptprogramm bindet die speziell entwickelte NTP-Implementierung ein, die über das Internet mit Clients (NTP-Modus 3) und/oder Zeitservern (NTP-Modus 4) kommuniziert und über eine definierte Schnittstelle zu NTS verfügt. (Die NTP-Realisierung kann so konfiguriert werden, dass sie auch ohne die Auswertung von NTS-Nachrichten im herkömmlichen Modus betrieben werden kann.)

Der eigentliche NTS-Teil gliedert sich in den NTS-Unicast-Service, das Client-Unicast- und

das Server-Unicast-Modul sowie das ASN.1- und das Kryptografiemodul. Letzteres greift auf die externe Bibliothek OpenSSL [9] zu, während das ASN.1-Modul die Bibliothek asn1c [10,11] direkt einbindet.

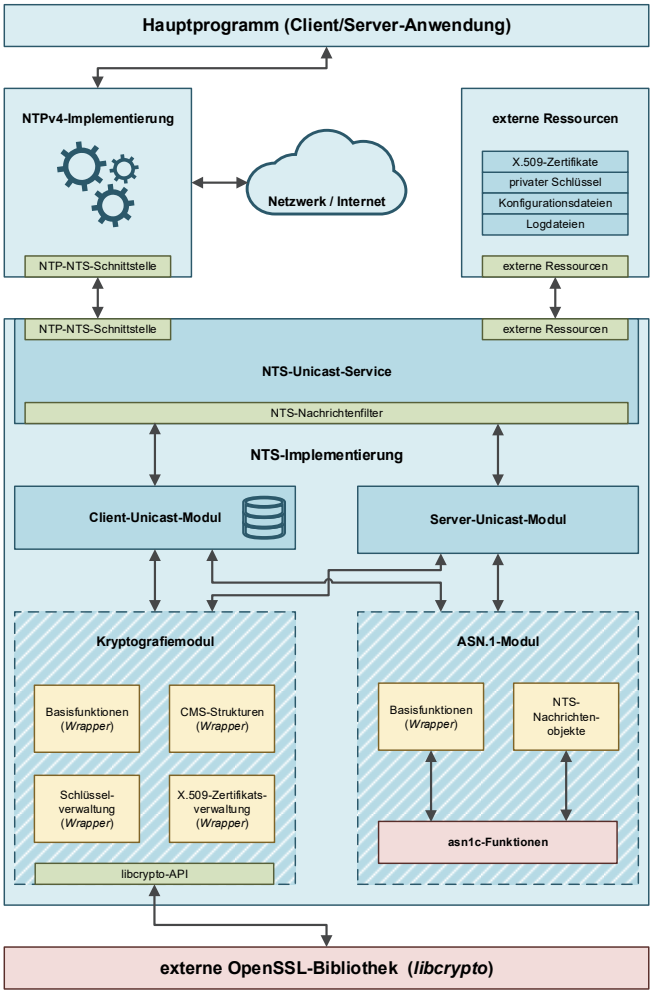

Abb. 1 Architektur der Softwareimplementierung [8]

Der NTS-Unicast-Service analysiert die in der NTP-Nachricht vorhandenen Erweiterungsfelder auf NTS-Protokollteile und leitet sie nach ersten Prüfungen entweder an das Client- oder das Server-Modul weiter. Der Unicast-Service dient auch dazu, auf externe Ressourcen zuzugreifen, die beispielswiese über Konfigurationsdateien die Arbeitsweise des Programms steuern. Weiterhin legt es wichtige, allgemeine Daten ab, wie Zertifikate und Schlüssel. Auch die Logging-Funktionen des Programms realisiert diese Schnittstelle.

Das Client-Unicast-Modul erzeugt Client-Anfragen und verarbeitet die zugehörigen Server-Antworten. Ein intern verwalteter Sitzungsspeicher erlaubt die gleichzeitige Kommunikation mit mehreren Zeitservern. Für Sonderfälle lassen sich auch mehrere Client-Instanzen einsetzen.

Im Server-Betrieb verarbeitet das Server-Unicast-Modul die Client-Anfragen und generiert die entsprechenden Server-Antworten. Der grundsätzliche Aufbau ist vergleichbar zum Client-Modul, wobei aber keine Datenhaltung erfolgt, da der Server ja zustandslos agiert. Er kann eine praktisch beliebige Anzahl von Clients bedienen, da er im Wesentlichen ja nur auf deren Anfragen reagiert und alle benötigten Informationen aus den Nachrichten entnehmen kann.

Das Kryptografie- und das ASN.1-Modul sind keine geschlossenen Programmteile, sondern vielmehr jeweils eine Ansammlung von diversen Routinen, wovon die meisten als Wrapper einheitliche Schnittstellen zwischen Programm und Bibliotheken herstellen.

Die grundsätzlichen Abläufe im NTS-Programm zeigt Abb. 2. Eingehende Nachrichten werden unterschieden in dem Client- bzw. dem Server-Modul zugehörig und nach erster Analyse und Überprüfung diesen zugeordnet. Das Client-Unicast-Modul erkennt anhand der Nachrichteninhalte, von welchem Server die Nachricht kommt, und lädt die entsprechenden Informationen sowie den Zustand der bisherigen Kommunikation mit ihm aus dem Sitzungsspeicher. Unterstützt vom ASN.1-Modul beginnt dann die Auswertung und Verarbeitung der Message. Gewonnene Informationen speichert der Sitzungsspeicher. Nun erfolgt die Erzeugung der nächsten NTS-Nachricht, wobei das Kryptografiemodul benötigte Funktionen bereitstellt. Anschließend übernimmt der Sitzungsspeicher den aktualisierten Zustand der Kommunikation, bevor das Antwortpaket über den NTS-Unicast-Service

an NTP (in die entsprechenden Erweiterungsfelder [12]) übergeben wird.

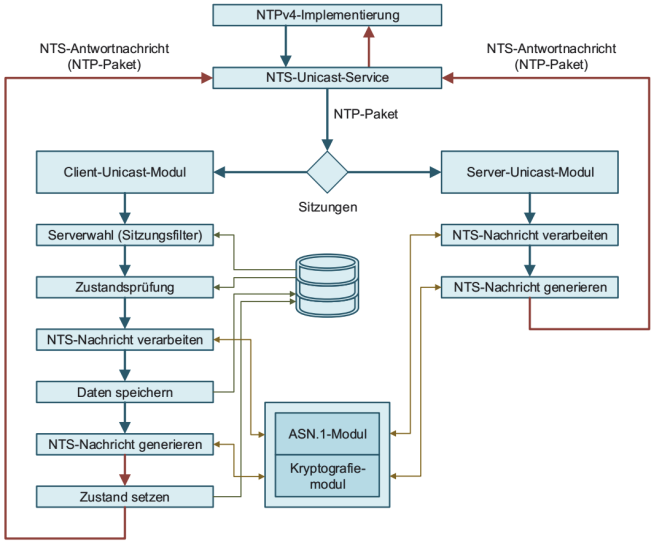

Abb. 2 Nachrichtenfluss in der NTS-Implementierung [8])

Nachrichten an das Server-Modul verarbeitet dieses mit Unterstützung des ASN.1-Moduls und generiert anschließend die Antwort, die es mithilfe des Kryptografiemoduls sichert und verschlüsselt. Auch hier übernimmt der NTS-Unicast-Service das neue Paket und übergibt es an NTP.

Bevor das Service-Modul arbeiten kann, muss die Initialisierung sämtlicher Module durchlaufen sein, die u.a. aus den Konfigurationsdateien gespeist wird. Für den grundsätzlichen Kommunikationsablauf des NTS-Protokolls sei auf den letzten Bericht [1] verwiesen.

Status und Tests

Im Rahmen des FuE-Projektes im WS16/17 konnte die Implementierung des NTS-Programms weitgehend fertiggestellt und geprüft werden. Im SS17 (ohne Forschungsbefreiung) erfolgten bisher weitere, verfeinernde Arbeiten so dass die grundsätzliche Implementierung als vorläufig abgeschlossen angesehen werden kann. Dabei sind aber dennoch etliche Punkte offen, die für einen generellen Einsatz im Feld ergänzt werden müssen. Beispielsweise ist das Zertifikatshandling erweiterungsbedürftig, um über OCSP Zertifikate online überprüfen oder Zertifikatssperrlisten verarbeiten zu können etc.

Auch die NTP-Implementierung erreichte im SS17 ein beinahe finales Stadium [13], was Feldtests von NTP und NTS über NTP ermöglichte. Beide Programme laufen derzeit auf x86/x64-Plattformen sowie auf ARM-Systemen jeweils unter Linux. (Eine Anpassung für den Betrieb unter Windows-Betriebssystemen ist in Planung.) Praktische Tests erfolgten im Client- und/oder Server-Betrieb mit verschiedenen PCs und Laptops sowie diversen Varianten von Raspberry Pis, um auch Aussagen über die Performance auf kleineren Systemen zu erhalten. Insbesondere die fehlende Hardware-Unterstützung für die Kryptografie spielt hier eine Rolle. Weiterhin kamen verschiedene Übertragungswege zum Einsatz, um die Auswirkungen der Kanalübertragungseigenschaften auf erreichbare Genauigkeiten und Latenzen zu analysieren: Local Host, Hochschulnetz, Forschungsnetz (DFN), DSL- und Kabelnetzzugänge, LTE sowie eine Internetanbindung über Richtfunk.

Local-Host-Messungen zeigen dabei die Unterschiede zwischen NTP ohne und mit NTS ohne den Einfluss des Übertragungswegs. Auf dem Beispielsystem ergaben sich hiermit ca. 40µs Delay ohne und ca. 90µs mit NTS, was in diesem speziellen Fall der reinen Verarbeitungszeit in der Software entspricht. NTS benötigt länger, da es das finale NTP-Paket noch signieren muss, bevor NTS es versendet, der Zeitstempel aber davor in die Nachricht eingebracht wird.

Bei den anderen Messungen zeigte sich, dass zwar Unterschiede bestehen zwischen den verschiedenen Übertragungswegen und die Variante mit NTS immer ungenauer ist als ohne, aber dennoch selbst im ungünstigsten Fall Genauigkeiten im mittleren zweistelligen Millisekundenbereich erzielt wurden. Spezielle Ausreißer zu manchen Zeitpunkten auf dem LTE-Weg müs-

sen noch untersucht und interpretiert werden. Zu keinem Zeitpunkt gab es Schwierigkeiten mit IP-Fragmentierung und anderen befürchteten Effekten.

Die bisherigen Implementierungen konnten zeigen, dass das Konzept von NTS grundsätzlich geeignet ist, Protokolle wie NTP abzusichern, ohne dass die erreichbaren Genauigkeiten darunter über Gebühr leiden müssen. Das Protokoll scheint zuverlässig implementiert und funktioniert im Dauerbetrieb. Ein Test gegen die Version der Network Time Foundation erfolgte nicht, da noch keine entsprechende, lauffähige Implementierung vorliegt.

Ausblick

Wie schon im letzten Bericht erläutert, basiert die NTS-Implementierung auf dem Draft-Stand vom September 2016 [7]. Der Draft ging mittlerweile in den sog. *Last Call* vor dem Übergang in den RFC-Status. Hier gab es unvermutet etliche Kritikpunkte am grundsätzlichen Konzept von NTS seitens der IETF Security Group (die man sich natürlich zu einem früheren Zeitpunkt gewünscht hätte). Diese Kritik führte kürzlich zu einem überarbeiteten Protokollvorschlag [14], der mehr Standardprotokolle bei der Schlüsselaushandlung und der anschließenden Zeitpaketübertragung vorsieht. Die Grundüberlegung war, dass bewährte Standardverfahren bereits vielfach geprüft und realisiert seien und daher vermutlich weniger Fehler und Angriffspunkte böten, als eine komplette Neuentwicklung aller Teile eines Sicherungsprotokolls. Zudem wurde eine Reihe weiterer Argumente angeführt (z.B. dass die IP-Fragmentierung unsicher sei), die nicht immer nachvollziehbar waren und u.U. eher politisch oder wirtschaftlich motiviert schienen.

Der neue Entwurf macht leider eine fast vollständige Neuimplementierung notwendig, will man die Brauchbarkeit des Konzeptes beweisen. Allerdings besteht Hoffnung auf eine etwas einfachere Realisierung durch die weitgehende Verwendung von Standardverfahren. Da die neuen Vorschläge seitens der Security Group der IETF eingebracht wurden, besteht auch Hoffnung auf eine schnelle Konvergenz des Draft und entsprechenden Übergang zum RFC-Status.

Literatur

[1] Bermbach, R.: *Network Time Security (NTS) – Unicast-Modus über NTP (Teil 1)*. Forschungsbericht SS 2016. Wolfenbüttel: Ostfalia Hochschule für angewandte Wissenschaften, 2016.

[2] Mills, D., Martin, J., Ed., Burbank, J., Kasch, W.: *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905, DOI 10.17487/RFC5905, 2010.

[3] IEEE: *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. IEEE Std. 1588–2008. New York, 2008.

[4] BSI: *Technische Richtlinie BSI TR-03109-1: Anforderungen an die Interoperabilität der Kommunikationseinheit eines intelligenten Messsystems*. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[5] PTB: *PTB-A 50.8 – Smart Meter-Gateway*. Physikalisch-Technische Bundesanstalt, Braunschweig, 2013.

[6] Röttger, S.: *Analyse des NTP-Autokey-Verfahrens.* Projektarbeit. Braunschweig: Technische Universität Braunschweig, Physikalisch-Technische Bundesanstalt, 2011.

[7] Sibold, D., Röttger, S., Teichel, K.: *Network Time Security*. draft-ietf-ntp-network-time-security-15, IETF, September 2016.

[8] Langer, M.: *Implementierung des Network-Time-Security-Protokolls für den Unicast-Betrieb*. Masterarbeit. Wolfenbüttel: Ostfalia Hochschule für angewandte Wissenschaften, 2016.

[9] OpenSSL Software Foundation: *Welcome to OpenSSL!* URL: https://www.openssl.org

[10] International Organization for Standardization: *Abstract Syntax Notation One (ASN.1) – Specification of basic notation*. Genf: ISO/IEC 8824-1:2015.

[11] Walkin, L.: *asn1c Documentation*. URL: http://lionet.info/asn1c/documentation.html

[12] Mizrahi, T., Mayer, D.: *Network Time Protocol Version 4 (NTPv4) Extension Fields*. RFC 7822, März 2016.

[13] Häußler, S., Jütte, C., Kompa, T., Tuschik, T., König, S.: *Entwicklung einer NTPv4-Implementation zur Unterstützung von NTS*. Teamprojekt. Wolfenbüttel: Ostfalia Hochschule für angewandte Wissenschaften, noch unveröffentlicht, 2017.

[14] Franke, D., Sibold, D., Teichel, K.: *Network Time Security for the Network Time Protocol*. draft-ietf-ntp-using-nts-for-ntp-08, IETF, März 2017.

Kontaktdaten

Ostfalia Hochschule für angewandte Wissenschaften
Fakultät Elektrotechnik
Prof. Dr.-Ing. Rainer Bermbach
Salzdahlumer Straße 46/48
38302 Wolfenbüttel
Telefon: +49 (0)5331 939 42620
E-Mail: r.bermbach@ostfalia.de
Internet: www.ostfalia.de/pws/bermbach