Kryptographie und Anycast

Erwin Nindl

http://www.anytun.org

Inhaltsverzeichnis

1 Einleitung

Anycast ist ein selten angewendetes Adressierungs Modell in IP Netzwerken. Da dieses Modell noch eher neuartig ist gibt es noch keine kryptographischen Protokolle welche welche Anycast unterstützen. In diesem Text wird die Eignung verschiedener Kryptographischer Protokolle für den Einsatz in Kombination mit Anycast Adressierung untersucht.

1.1 Anycast aus Netzwerk Sicht

Anycast ist ein Service orientiertes Adressierungs Modell bei dem mehrere Hosts mit einer identischen Adresse ausgezeichnet werden. Ein Anycast Packet wird dabei an einen der Hosts mit der Anycast Adresse weitergeleitet. Anycast Adressierung wurde erstmals im RFC1546 [1] definiert und ist erstmals standardisiert in IPV6 [2]. Die Grundidee bei Anycast ist die Services von dem physikalischen Host Equipment zu trennen und einen logischen Host für einen bestimmten Service einzurichten.

Bei Unicast darf jede Adresse nur genau einmal einem Netzwerk Interface zugeordnet werden und ist somit ein eindeutiger Kennzeichner für das jeweilige Netzwerk Interface. Hier werden Packete mit der selben Unicast Adresse auch immer dem gleichen Netzwerkteilnehmer zugesendet.

Im Gegensatz dazu wird bei Anycast eine bestimmte Adresse einer Gruppe von Hosts zugeordnet. Durch die Mechanismen von IP ist es nicht möglich zu bestimmen welches IP Packet zu welchem Anycast Teilnehmer gesendet wird bzw. ob ein IP Packet gleich an mehrere Anycast Teilnehmer weitergeleitet wird. Es können daher nur Server via Anycast adressiert werden.

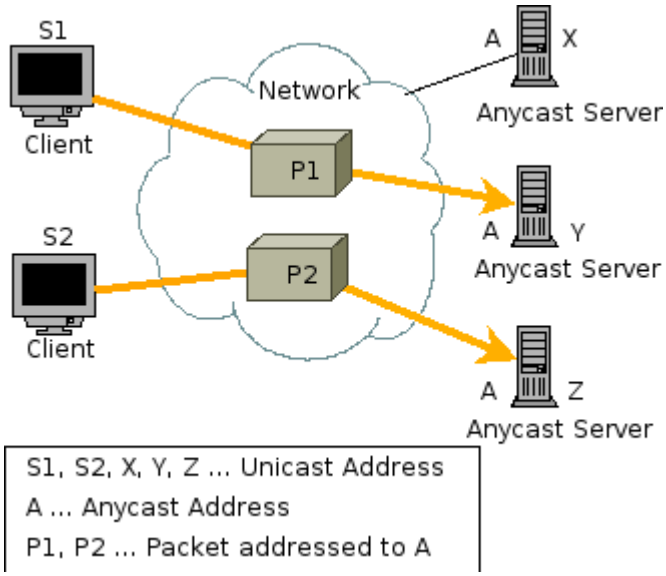

Abb: Anycast Kommunikation

Alle TCP basierenden und manche UDP basierenden Internet Protokolle sind zustandsbehaftet. Da nicht sichergestellt werden kann an welchem Anycast Server die IP Packete ankommen sind für zustandsbehaftete Protokolle weitere Maßnahmen zu ergreifen damit die Kommunikation funktioniert:

Im RFC1546 [1] wird vorgeschlagen bei zustandsbehafteten Internetprotokollen wie z.B. TCP nach dem Empfang des ersten Packets die entsprechende Unicast Adresse des Servers für den Rest der Datenübertragung zu verwenden um sicherzustellen dass auch immer der gleiche Server die Packete erhält.

Im Gegensatz dazu *wird hier die Idee verfolgt* den Kontext der zustandsbehafteten Verbindung zwischen den einzelnen Anycast Hosts zu synchronisieren bzw. wenn das Synchronisieren nicht möglich ist diesen Kontext abzuschätzen. Dadurch entstehen für eine integrierte Kryptographie in Anycast Services neue Anforderungen an die kryptographischen Protokolle, welche nachfolgend weiter beschrieben werden.

1.2 Anycast aus Sicht der Kryptographie

Da es bei Anycast nicht vorhersehbar ist welches verschlüsselte Packet von welchem Anycast Server empfangen wird muss sichergestellt werden dass jedes einzelne, empfangene Packet entschlüsselt werden kann. Damit die senderseitige Verschlüsselung und die empfängerseitige Entschlüsselung funktionieren kann muss auf beiden Seiten eine entsprechende kryptographische Status Information verwaltet werden. Diese kryptographische Status Information wird gleich wie in *Secure Real-time Transport Protocol (SRTP, RFC3711)* [3], Section 3.2 im weiteren Text als *kryptographischer Kontext* bezeichnet.

Der kryptographische Kontext setzt sich im wesentlichen aus verschiedenen Keys, Variablen zur Erzeugung des zur Verschlüsselung notwendigen *Initial Vector (IV)* bzw. dem IV selbst und weiteren Variablen für Schutzmassnahmen wie z.B. Replay Protection zusammen. Je nach dem wie oft sich dieser kryptographische Kontext bzw. Teile davon ändern muss dieser bei der Verwendung einer Kryptographie bei Anycast zwischen den einzelnen Anycast Servern synchronisiert werden. Dies ist einer der wesentlichsten Punkte bei der Wahl der entsprechenden kryptographischen Protokolle.

Besonders deutlich wird dieser Punkt bei der Betrachtung des Worst Case: Das ist der Fall wenn für jedes empfangene Packet an einem Anycast Host der kryptographische Kontext bei allen restlichen Anycast Hosts upgedatet werden muss. Bei n Anycast Hosts und k notwendigen Packeten zum Update der Information auf einem Host würde das einen zusätzlichen Mehraufwand von k*(n-1) Packeten pro empfangenes Packet bedeuten.

Ziel ist es daher ein kryptographisches Protokoll zu finden bei dem der zusätzliche Mehraufwand an Packeten zur Synchronisation des kryptographischen Kontexts so gering als möglich ist.

Im Nachfolgenden werden vorhandene kryptographische Protokolle hingehend auf die Verwendbarkeit mit Anycast untersucht.

2 Secure Real-time Transport Protocol (SRTP)

Das *Secure Real-time Transport Protocol (SRTP)* [3] ist eine Erweiterung des Real-time Transport Protokolls (RTP) und wurde daher ursprünglich zur kontiunierlichen Übertragung von audiovisuellen Datenstreams entwickelt. Es bietet Vertraulichkeit, Nachrichten Authentifizierung, Replay Protection von RTP und RTCP Verkehr.

Wie man aus RFC3711, Section 4.3 entnehmen kann werden für jedes Packet neue Keys berechnet. Aus einem *Master Key* und einem optionalen *Master Salt* werden in Kombination mit dem Packet Index *i* die so genannten *Session Keys* abgeleitet. Mit diesen *Session Keys* wird dann jedes entsprechende Packet verschlüsselt. Die *Salting Keys* werden eingesetzt um die Verschlüsselung gegen Pre-Computation und Time-Memory Tradeoff Attacken zu schützen. Der zur kryptographischen Transformation benötigte Initial Vektor *IV* berechnet sich dabei immer aus den Variablen des kryptographischen Kontexts und Variablen die im SRTP Packet enthalten sind. Die genaue Ableitung von *IV* hängt dann speziell vom eingesetzten Cipher ab.

Ein Key Management für den Master Key ist in SRTP nicht integriert und muss daher zusätzlich gemacht werden.

2.1 Chiffre

Es können mit SRTP beliebige Cipher eingesetzt werden. Diese müssen entsprechend RFC3711 [3], Section 6 definiert werden. Im Standard sind schon zwei verschiedene Verfahren vordefiniert, welche beide den *Advanced Encryption Standard (AES)* [AES] verwenden:

- **Segmented Integer Counter Mode**

  Dieser Modus erlaubt einen beliebigen Zugriff zu den einzelnen Blöcken.

- **f8-mode**

  Dieser Modus wird in 3G UMTS Netzen verwendet.

Zur Nachrichten Authentifizierung wird *HMAC-SHA1* vordefiniert, so wie es in *HMAC: Keyed-Hashing for Message Authentication (RFC2104)* [13] weiter beschrieben ist.

2.2 Packet Index Determination

Für die Erzeugung der Session Keys ist nur der statische Teil des Kryptographischen Kontexts und der Packet Index *i* des entsprechenden SRTP Packets notwendig. Dieser Packet Index ist jedoch nicht direkt im SRTP Packet enthalten, sondern muss zusätlich am Sender und Empfänger berechnet werden:

Variablen:

- **ROC**: (Roll over Counter, max. Grösse: 2^{32}-1)

ROC wird inkrementiert bei einem Überlauf von SEQ
ROC wird sende- und empfangsseitig gespeichert und upgedated.

- **SEQ**: (Sequence Number, max. Grösse: 2^{16}-1)

  SEQ ist im SRTP Packet enthalten.
- **s_l** (höchste empfangene SEQ)

  nur empfangsseitig
- **i**: (Packet Index)

  i wird benötigt zur Ver- und Entschlüsselung des entsprechenden Packets.

Der Packet Index des SRTP Packets wird wie flogt berechnet:

$i = 2^{16} * ROC + SEQ$

Zu Beginn der Kommunikation wird der ROC initialisiert. Empfängerseitig wird i aus ROC und SEQ berechnet. ROC wird inkrementiert sobald SEQ überläuft. Beim Überlauf von ROC müssen neue Master Keys erzeugt werden. Die Berechnung von ROC auf der Empfangsseite ist kritisch wenn SEQ sich nahe an 0 bzw. 2^{16} befindet. Weitere Informationen diesbezüglich findet man in RFC3711: *Sektion 3.3.1. Packet Index Determination, and ROC, s_l Update* [3].

Laut RFC müssen 2^{15} Packete verloren gehen oder 2^{15} Packete aus der Reihung geraten damit die Synchronisation von *ROC* und in weiterer Folge dazu von *i* verloren geht.

2.3 Replay Protection

Es wird im RFC3711 eine Replay Protection für SRTP empfohlen. Diese funktioniert nur wenn eine Integritätsüberprüfung der SRTP Packete vorgenommen wird. Dabei wird eine Liste der empfangenen Packet Indizes geführt und eventuell doppelt ankommende Packete werden verworfen. Zur Realisierung dieser Liste wird ein 'sliding window' Verfahren vorgeschlagen welches in *Security Architecture for the Internet Protocol (RFC4301)* [6] ausführlich beschrieben wird.

2.4 SRTP und Anycast

Packet Index

Bei SRTP wird jedes Packet mit Hilfe des kryptographischen Kontexts und SEQ entschlüsselt. Dieser kryptographische Kontext muss zwischen allen Anycast Servern synchronisiert werden. Bis auf *ROC* und *s_l* bleiben alle Variablen des kryptgraphischen Kontexts bei einer laufenden Session unverändert. Es muss daher also der gesammte kryptographische Kontext nur zu Beginn einer Session abeglichen werden. Im Laufe einer Session reicht es die Variable *ROC* auf den einzelnen Hosts abzugleichen. Da sich *ROC* nur alle 2^{16} Packete ändert kann diese Variable auch sehr leicht auf den einzelnen Anycast Hosts upgedatet werden. Man liegt so beim Aufwand zur Synchronisation auch ca. um den Faktor 2^{-16} unter dem oben genannten Worst Case Szenario.

Replay Protection

Man kann die untere Grenze der Indizes der empfangenen Packete in fixen Intervallen zwischen den einzelnen Anycast Servern synchronisieren. Wenn die restliche Liste der Packete im Window Bereich nicht upgedatet wird kann jedes Packet noch ein mal für eine Replay Attacke pro Anycast Server verwendet werden. Da es aber nicht vorhersehbar ist welcher Anycast Server die Replay Packete erhält ist diese Attacke eher unrealistisch.

SRTP kann daher auch mit Anycast mit relativ geringem Mehraufwand an zusätzlicher Synchronisation eingesetzt werden.

2.5 Verfügbare Implementationen

libSRTP

Die libSRTP [5] wurde von Cisco als Referenzimplementierung zu SRTP zur Verfügung gestellt. Sie ist unter der BSD Lizenz lizenziert und eignet sich in dieser Hinsicht auch zur Verwendung.

Eine grobe Performance Abschätzung von libSRTP kann unter http://srtp.sourceforge.net/faq.html#Q5 abgerufen werden. Testsystem war dabei libSRTP auf einem 1.67 Ghz PowerPC G4 Prozessor unter verwendung der Default Cryptoalgorithmen (AES-128 Counter Mode and HMAC-SHA1). Die Performance bei Authentifikation und Verschlüsselung liegt je nach RTP Payload Länge zwischen 37.6 und 173 Megabit pro Sekunde.

3 IPsec

IPsec (RFC4301) [6] wurde zuerst in *IPV6 (RFC2460)* [2] definiert und für IPV4 'rückportiert'. IPsec bietet Mechanismen zur Überprüfung der Nachrichten Integrität (*Authentication Header (AH)* [7]) und zur Nachrichten Verschlüsselung (*Encapsulating Security Payload (ESP)* [8]).

IPsec kann in zwei verschiedenen Modis betrieben werden:

- Transport Mode
  Wird benutzt zu Host zu Host Kommunikation. Nur der Inhalt des IP Packets wird verschlüsselt, hingegen der IP Header wird in Plaintext übertragen. Die Packete können voll geroutet werden. Bei NAT wird der Hash Wert des Packets verändert und kann daher nicht verwendet werden.
- Tunnel Mode
  Hier wird das ganze Packet verschlüsselt und in ein neues IP Packet gepackt. Dieser Mode wird für Netzwerk zu Netzwerk Verbindungen (Tunnels) verwendet.

In *Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH) (RFC4305)* [11] werden alle Chiffre aufgelistet welche ESP unterstützen muss. Bei ESP wird der IV mit jedem einzelnen Packet übermittelt. Daher ist jedes Packet am Empfänger entschlüsselbar.

Weiters besitzt IPsec noch Mechanismen zur *Replay Protection*. Da die Replay Protection Einfluss auf die Verwendbarkeit von ESP mit IPsec haben kann wird im folgenden noch näher darauf eingegangen:

3.1 Replay Protection

Die *Sequence Number* wird zur Replay-Protection verwendet. Dieser Mechanismus ist bei AH defaultmässig aktiv. Beim Beginn einer Verbindung wird die Sequence Number beim Sender und Empfänger auf 0 gesetzt. Die *Sequence Number* wird mit jedem geschickten Packet erhöht. Wiederholt sich die *Sequence Number* tritt die Replay Protection in Kraft und das entsprechende Packet wird verworfen. Diese empfängerseitige Kontrolle wird mittels 'sliding-window' Verfahren durchgeführt, welches näher in ESP (RFC4303) [8] beschrieben ist.

Replay Protection und Packet Loss in Kombination mit ESN

Die *Extended Sequence Number (ESN)* (RFC4303 [8], Section 2.2.1) besitzt 64bit im Gegensatz zur Sequence Number welche 32bit gross ist. Wenn die *ESN* verwendet wird werden nur die unteren 32bit der ESN mit einem Packet übertragen. Die höherwertigen 32bit werden am Sender und Empfangsseitig entsprechend eines Roll over Counters nachgerechnet. Sobald die unteren 32bit im Packet kleiner sind als die unteren 32bit der ESN beim Empfänger werden die oberen 32bit der ESN beim Empfänger inkrementiert.

Bei einem Packet Loss grösser als 2^{32} oder mehr auf einer *Security Association (SA)* geht die Synchronisation bei den höherwertigen Bits zwischen Sender und Empfänger verloren. Folgende Packete werden dann durch die Replay Protection verworfen.

Nachfolgendes gilt nur für ESN: Zur Resynchronisation werden alle aufeinanderfolgenden Packete welche abgewiesen worden sind gezählt. Übersteigt diese Zahl einen eingestellten Wert so wird die Synchronisation neu gestartet. Sobald ein gültiges Packet empfangen wird wird diese Zahl auf 0 zurückgesetzt. Sobald die Zahl einen Triggerlevel übersteigt wird die resynchronisation eingeleitet. Dabei wird ein verworfenes Packet verwendet und die höheren 32bit der ESN werden sucksesive inkrementiert und jedes mal neu geprüft. Nach einer gewissen Zahl an Versuchen wird das Packet endgültig verworfen.

3.2 IPsec und Anycast

Die in RFC4305 [11] definierten kryptographischen Algorithmen basieren alle auf einen im IPsec Packet enthaltenen IV. Dieser IV wird bei IPsec immer im Packet mit übermittelt. Es kann daher jedes ESP Packet ohne zusätzlichen Synchronisationsaufwand entschlüsselt werden.

Fragmentierung

Kommt ein fragmentiertes IPsec Packet am Empfänger an (mit Offset) wird dies vom Empfänger sofort erkannt und verworfen. Es muss daher sichergestellt werden dass alle fragmentierten IPsec Packete vor der Verarbeitung zu einem IPsec Datagramm defragmentiert werden. Dies ist bei Anycast praktisch nicht möglich, da die fragmentierten IPsec Packete bei Verschiedenen Anycast Hosts ankommen können. Es ist daher darauf zu achten dass die IPsec Datagramme mit der kleinsten MTU am Netzwerk abgesendet werden. Dies kann mit Path MTU (PMTU) discovery [10] herausgefunden werden.

Replay Protektion

Aufgrund der vorhin erwähnten Replay Protection muss das Anti Replay Window auf allen Anycast Empfängern in regelmäßingen Abständen upgedatet werden. Hier gelten die gleichen Erkenntnisse wie bei der Replay Protection bei SRTP.

Zusätzlich müssen bei der Verwendung von *ESN* die höherwertigen 32bit der *ESN* zwischen den einzelnen Anycast Empfängern synchronisiert werden da es sonst bei einem senderseitigen Überlauf der höherwertigen 32bit der *ESN* zu einer Abweisung der nachfolgenden Packete führen kann. Da sich diese nur alle 2^{32} IPsec Packete ändert ist eine Synchronisation der *ESN* leicht zu bewältigen.

Zum Senden wäre eine komplette Synchronisation der Sequence Number notwendig.

3.3 Verfügbare Implementationen

Openswan

Openswan ist eine Open Source Implementierung von IPsec für Linux. Es ist eine Weiterentwicklung FreeS/WAN welches nicht mehr weiter entwickelt wird. Die offizielle Homepage ist unter http://www.openswan.org erreichbar.

4 TLS/SSL

4.1 Allgemeines

Transport Layer Security (TLS) [TLS] oder ***Secure Socket Layer (SSL)*** sind Verschlüsselungsprotokolle für Datenübertragung in IP Netzwerken. TLS ist eine standardisierte Weiterentwicklung von SSL 3.0. TLS/SSL ist im Session Layer des OSI Referenz Modells angesiedelt. TLS unterscheidet sich von SSL nur in wenigen Punkten. Es wird daher im nachfolgenden nur mehr *TLS 1.1 (RFC4346)* [TLS] repräsentativ für TLS und SSL betrachtet.

Der Verbindungsaufbau bei TLS basiert auf einem Handshake Protokoll. RFC4346 [TLS], Section 7.3 gibt einen Überblick über das TLS Handshake Protokoll. Ein TLS Handshake besteht aus mehreren (>8) Schritten und wird zur Authentifizierung, Key Exchange und zum Austausch des benutzten Ciphers verwendet. Es muss für den Aufbau einer neuen Session ein Handshake durchgeführt werden. Zum Fortsetzen einer vorhergegangenen Session kann ein verkürzter Handshake verwendet werden welcher nur mehr sechs Schritten besteht.

4.2 Chiffre

Nach RFC4346, Section 6.2.3. kann bei TLS entweder im NULL-, im Standard-Stream-Cipher- oder im Cipher Block Chaining (CBC) Mode betrieben werden.

Bei einem Stream Cipher werden die Daten mit einem pseudo Zufallszahlen-Strom logisch verknüpft und beim Empfänger mit dem gleichen Zufallszahlen-Strom rekonstruiert.
Beim CBC Mode wird zur Verchlüsselung / Entschlüsselung wird der aktuell verschlüsselte Block immer unter Einbezug des vorher erzeugten Blocks generiert.

4.3 TLS und Anycast

Bei TLS wird die Verbindung via Handshake Protokoll aufgebaut. Damit dieser Handshake auch mit Anycast funktioniert muss zwischen jedem Schritt der Handshake Status zwischen den einzelnen Anycast Hosts abgeglichen werden. Da dies nur für den Verbindungsaufbau benötigt wird könnte man diesen Overhead theoretisch noch in Kauf nehmen.

Hingegen werden die Anwendungsdaten entweder mit einem Stream Cipher oder mit einem CBC Block Cipher verschlüsselt. Hier tritt der unter Punkt 'Anforderungen an die Kryptographie bei Anycast' aufgeführte Worst Case ein. Es müsste für jedes Empfangene Packet der kryptographische Kontext zwischen den einzelnen Anycast Servern abgeglichen werden.

TLS ist daher aus den oben genannten Gründen nicht mit Anycast einsetzbar.

5 Vergleich der Algorithmen

TLS/SSL scheiden wegen des oben schon erwähnten Worst Case Verhalten im Bezug zur Synchronisation des kryptographischen Kontexts an den einzelnen Anycast Hosts für die Verwendung mit Anycast aus und wird daher nicht mehr behandelt.

Da SRTP gleich wie IPsec (IPsec nur unter der Verwendung bestimmter Cipher) Anycast fähig sind werden diese beiden kryptographischen Protokolle noch weiter verglichen:

Beide Protokoll Spezifikationen schreiben keinen bestimmten kryptographischen Algorithmus vor. Jedoch wird bei SRTP schon AES [AES] im Counter Mode als Beispiel mit angegeben. Bei IPsec kann man aus einer Reihe von kryptographischen Algorithmen wählen, welche in RFC4305 näher beschrieben sind.
Der Rechenaufwand für AES ist relativ gering. Er eignet sich daher auch besonders gut für embedded Devices. Es sollte nicht unerwähnt bleiben dass AES zur Zeit als einer der sichersten Verschlüsselungsalgorithmen angenommen werden kann.

Bei der Replay Protection sollte das Anti Replay Window zwischen den einzelnen Anycast Servern upgedatet werden. Hier gelten für beide Protokolle die selben Einschänkungen.

IPsec erfordert ein Rekying nach jedem Überlauf der Sequence Number. Dies geschieht nach 2^{32} Packeten. Bei SRTP muss erst nach 2^{48} SRTP Packeten durchgeführt werden. Es können daher im Vergleich um 2^{16} mehr SRTP Packete als IPsec Packete übertragen werden bevor ein Rekeying notwendig ist.

IPsec besitzt ein komplexeres Protokoll als SRTP. IPsec besitzt im Vergleich zu SRTP einen höheren Overhead pro Packet da der IV bei jedem Packet mitgesendet wird.

Die verfügbare Implementation von SRTP ist um einiges schlanker als jene von IPsec. Sie kann daher leichter um die entsprechenden Anforderungen für ein Tunnel Protokoll erweitert und angepasst werden.

Im vorangegangenen Vergleich im Bezug zu Anycast schneidet SRTP besser ab als IPsec. Es erfordert weniger Synchronisationsaufwand und bringt trotzdem weniger Overhead pro Packet mit sich. Zum Entschlüsseln ist auch nur die unverschlüsselte Sequence Number, welche im Packet vorhanden ist, und der Kryptographische Kontext notwendig. Dieser kann leicht zwischen den einzelnen Hosts synchronisiert werden da er sich nur alle 2^{16} Packete ändert.

6 Quellen

[0] Internet Engineering Task Force, R. Braden, "Requirements for Internet Hosts -- Communication Layers", RFC1122, October 1989.

[1] C. Partridge, T. Mendez, W. Milliken, "Host Anycasting Service", RFC1546, November 1993.

[2] S. Deering, R. Hinden, "Internet Protocol version 6 (IPV6) specification", RFC2460, Dec 1998.

[3] M. Baugher, D. McGrew, M. Naslund, E. Carrara, K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC3711, March 2004.

[AES] Joan Daemen, Vincent Rijmen: "The Design of Rijndael. AES: The Advanced Encryption Standard", Oktober 2000.

[5] "libSRTP", http://srtp.sourceforge.net/srtp.html

[6] Kent, S., "Security Architecture for the Internet Protocol", RFC4301, December 2005.

[7] Kent, S., "IP Authentication Header", RFC4302, December 2005.

[8] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC4303, December 2005.

[9] Jun-ichiro itojun Hagino, K. Ettikan, "An analysis of IPv6 anycast", draft-ietf-ipngwg-ipv6-anycast-analysis-02.txt, June 2003.

[10] J. Mogul, S. Deering, "Path MTU Discovery", RFC1191, November 1990.

[11] D. Eastlake 3rd, "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC4305, December 2005.

[13] H. Krawczyk, M. Bellare, R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC2104, February 1997.

[TLS] T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC4346, April 2006