Test Vectors for Session Traversal Utilities for NAT (STUN)

Abstract

The Session Traversal Utilities for NAT (STUN) protocol defines several STUN attributes. The content of some of these -- FINGERPRINT, MESSAGE-INTEGRITY, and XOR-MAPPED-ADDRESS -- involve binary-logical operations (hashing, xor). This document provides test vectors for those attributes.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5769.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
Table of Contents

1. Introduction ........................................... 3
2. Test Vectors ............................................ 3
   2.1. Sample Request .................................... 4
   2.2. Sample IPv4 Response ............................... 5
   2.3. Sample IPv6 Response ............................... 6
   2.4. Sample Request with Long-Term Authentication ... 8
3. Security Considerations ................................. 8
4. Acknowledgments ........................................ 8
5. References ............................................. 8
   5.1. Normative References ............................... 8
   5.2. Informative References ............................. 8
Appendix A. Source Code for Test Vectors ................. 9
1. Introduction

The Session Traversal Utilities for NAT (STUN)[RFC5389] protocol defines two different hashes that may be included in messages exchanged by peers implementing that protocol:

FINGERPRINT attribute: a 32-bit Cyclic Redundancy Check.


This document provides samples of properly formatted STUN messages including these hashes, for the sake of testing implementations of the STUN protocol.

2. Test Vectors

All included vectors are represented as a series of hexadecimal values in network byte order. Each pair of hexadecimal digits represents one byte.

Messages follow the Interactive Connectivity Establishment (ICE) Connectivity Checks use case of STUN (see [RFC5245]). These messages include FINGERPRINT, MESSAGE-INTEGRITY, and XOR-MAPPED-ADDRESS STUN attributes. These attributes are considered to be most prone to implementation errors. An additional message is provided to test STUN authentication with long-term credentials (which is not used by ICE).

In the following sample messages, two types of plain UTF-8 text attributes are included. The values of certain of these attributes were purposely sized to require padding. Non-ASCII characters are represented as <U+xxxx> where xxxx is the hexadecimal number of their Unicode code point.

In this document, ASCII white spaces (U+0020) are used for padding within the first three messages - this is arbitrary. Similarly, the last message uses nul bytes for padding. As per [RFC5389], padding bytes may take any value.
2.1. Sample Request

This request uses the following parameters:

Software name: "STUN test client" (without quotes)

Username: "evtj:h6vY" (without quotes)

Password: "VOkJxbRl1RmTxUk/WvJxBt" (without quotes)

```
00 01 00 58     Request type and message length
21 12 a4 42     Magic cookie
b7 e7 a7 01 }  Transaction ID
fa 87 df ae }  
80 22 00 10     SOFTWARE attribute header
53 54 55 4e }  User-agent...
74 20 63 6c }   ...name
69 65 6e 74 }  
00 24 00 04     PRIORITY attribute header
6e 00 01 ff     ICE priority value
80 29 00 08     ICE-CONTROLLED attribute header
93 2f f9 b1 }   Pseudo-random tie breaker...
51 26 3b 36 }   ...for ICE control
00 06 00 09     USERNAME attribute header
65 76 74 6a }  Username (9 bytes) and padding (3 bytes)
3a 68 36 76 }  
59 20 20 20 }  
00 08 00 14     MESSAGE-INTEGRITY attribute header
9a ea a7 0c }  
bf d8 cb 56 }  HMAC-SHA1 fingerprint
78 1e f2 b5 }  CRC32 fingerprint
```
2.2. Sample IPv4 Response

This response uses the following parameter:

Password: "VOkJxbRl1RmTxUk/WvJxBt" (without quotes)

Software name: "test vector" (without quotes)

Mapped address: 192.0.2.1 port 32853

01 01 00 3c   Response type and message length
21 12 a4 42   Magic cookie
b7 e7 a7 01   }  Transaction ID
bc 34 d6 86   }  XOR-MAPPED-ADDRESS attribute header
fa 87 df ae   }
80 22 00 0b   SOFTWARE attribute header
74 65 73 74   }
20 76 65 63   UTF-8 server name
74 6f 72 20   }
00 20 00 08   MESSAGE-INTEGRITY attribute header
00 01 a1 47   Address family (IPv4) and xor’d mapped port number
e1 12 a6 43   Xor’d mapped IPv4 address
00 08 00 14   FINGERPRINT attribute header
2b 91 f5 99   CRC32 fingerprint
fd 9e 90 c3   }
8c 74 89 f9   HMAC-SHA1 fingerprint
2a f9 ba 53   }
f0 6b e7 d7   }
80 28 00 04   FINGERPRINT attribute header
c0 7d 4c 96   CRC32 fingerprint
2.3. Sample IPv6 Response

This response uses the following parameter:

Password: "VOkJxbRl1RmTxUk/WvJxBt" (without quotes)

Software name: "test vector" (without quotes)


01 01 00 48     Response type and message length
21 12 a4 42     Magic cookie
b7 e7 a7 01 }   Transaction ID
fa 87 df ae }   SOFTWARE attribute header
80 22 00 0b     XOR-MAPPED-ADDRESS attribute header
74 65 73 74 }   Address family (IPv6) and xor’d mapped port number
01 13 a9 fa }   Xor’d mapped IPv6 address
bc 25 fb b5 }   UTF-8 server name
be d2 b9 d9 }   XOR-MAPPED-ADDRESS attribute header
00 08 00 14     MESSAGE-INTEGRITY attribute header
a3 82 95 4e }   HMAC-SHA1 fingerprint
4b e6 7b f1 }   FINGERPRINT attribute header
17 84 c9 7c }   CRC32 fingerprint
2.4. Sample Request with Long-Term Authentication

This request uses the following parameters:

Username: "<U+30DE><U+30C8><U+30EA><U+30C3><U+30AF><U+30B9>" (without quotes) unaffected by SASLprep [RFC4013] processing

Password: "The<U+00AD>M<U+00AA>tr<U+2168>" and "TheMatrIX" (without quotes) respectively before and after SASLprep processing

Nonce: "f///499k954d6OL34oL9FSTvy64sA" (without quotes)

Realm: "example.org" (without quotes)

00 01 00 60  Request type and message length
21 12 a4 42  Magic cookie
78 ad 34 33  )  Transaction ID
00 06 00 12  USERNAME attribute header
d3 83 83 e3  }  Username value (18 bytes) and padding (2 bytes)
e3 82 af e3  )
e3 82 b9 00 00  )
00 15 00 1c  NONCE attribute header
66 2f 2f 34  )
39 39 6b 39  )
35 34 63 36  )
4f 4c 33 34  )  Nonce value
6f 4c 39 46  )
53 54 76 79  )
36 34 73 41  )
00 14 00 0b  REALM attribute header
65 78 61 6d  )
7b 6c 65 2e  )  Realm value (11 bytes) and padding (1 byte)
6f 72 67 00  )
00 08 00 14  MESSAGE-INTEGRITY attribute header
f6 70 24 65  )
6d d6 4a 3e  )
02 b8 e0 71  )  HMAC-SHA1 fingerprint
2e 85 c9 a2  )
8c a8 96 66  )
3. Security Considerations

There are no security considerations.

4. Acknowledgments

The author would like to thank Marc Petit-Huguenin, Philip Matthews and Dan Wing for their inputs, and Brian Korver, Alfred E. Heggestad and Gustavo Garcia for their reviews.

5. References

5.1. Normative References


5.2. Informative References


Appendix A. Source Code for Test Vectors

```c
const unsigned char req[] =
  "\x00\x01\x00\x58"
  "\x21\x12\xa4\x42"
  "\xb7\xe7\xa7\x01\xbc\x34\xd6\x86\xda\x87\xda\xae"
  "\x80\x22\x00\x10"
  "STUN test client"
  "\x00\x24\x00\x04"
  "\x6e\x00\x01\xff"
  "\x80\x29\x00\x08"
  "\x93\x2f\xf9\xb1\x51\x26\x3b\x36"
  "\x00\x06\x00\x09"
  "\x65\x76\x74\x6a\x3a\x68\x36\x76\x59\x20\x20\x20"
  "\x80\x08\x00\x14"
  "\x9a\xae\xa7\x0c\xfb\xd8\xcb\x56\x78\x1e\xf2\xb5"
  "\xe2\x0d\x3f\x49\xc1\x4b\x71\xa2"
  "\x80\x28\x00\x04"
  "\xe5\x7a\x3b\xcf";
```

Request message
const unsigned char respv4[] =
  ":01\x01\x00\x3c
  \x21\x12\xa4\x42
  \xb7\xe7\xa7\x01\xbc\x34\xd6\x86\xfa\x87\xdf\xae
  \x80\x22\x00\x0b
  \x74\x65\x73\x74\x20\x76\x65\x63\x74\x6f\x72
  \x00\x20\x00\x08
  \x00\xa1\x47\xe1\x12\xa6\x43
  \x00\x08\x00\x14
  \x2b\x91\xf5\x99\xfd\x9e\x90\xc3\x8c\x74\x89\xf9
  \x2a\xf9\xba\x53\xf0\x6b\xe7\xd7
  \x80\x28\x00\x04
  \xc0\x7d\x4c\x96";

IPv4 response message

const unsigned char respv6[] =
  ":01\x01\x00\x48
  \x21\x12\xa4\x42
  \xb7\xe7\xa7\x01\xbc\x34\xd6\x86\xfa\x87\xdf\xae
  \x80\x22\x00\x0b
  \x74\x65\x73\x74\x20\x76\x65\x63\x74\x6f\x72
  \x00\x20\x00\x14
  \x01\x13\xa9\xf9\x90\xc3\x8c\x74\x89\xf9
  \x2a\xf9\xba\x53\xf0\x6b\xe7\xd7
  \x80\x28\x00\x04
  \xc0\x7d\x4c\x96";

IPv6 response message
const unsigned char reqltc[] =
  "\00\x01\x00\x60"
  "\x21\x12\xa4\x42"
  "\x78\xad\x34\x33\xc6\xad\x72\xc0\x29\xda\x41\xe2"
  "\x00\x06\x00\x12"
  "\xe3\x83\x9e\xe3\x83\xaa\xe3\x83\x83"
  "\xe3\x82\xaf\xe3\x83\xb9\x00\x00"
  "\x00\x15\x00\x1c"
  "\xe6\x2f\x2f\x34\x39\x39\x6b\x39\x35\x34\x64\x36"
  "\x4f\x4c\x34\x39\x39\x4f\x4c\x39\x39\x4f\x4c\x39\x39\x4f\x4c\x39\x39\x4f\x4c\x39\x39\x4f\x4c\x39\x39"
  "\x36\x34\x73\xe1"
  "\xe6\x78\x61\x6d\x70\x65\x65\x6e\x72\x67\x00"
  "\xe6\x78\x61\x6d\x70\x65\x65\x6e\x72\x67\x00"
  "\xe6\x70\x24\x65\x6d\xa4\x3e\x02\xe8\xe0\x71"
  "\xe2\x85\xc9\x2a\x8c\xe8\xa8\x96\x66";

Request with long-term credentials

Author’s Address

Remi Denis-Courmont
Nokia Corporation
P.O. Box 407
NOKIA GROUP 00045
FI

Phone: +358 50 487 6315
EMail: remi.denis-courmont@nokia.com