WebSocket Implementation Test Report

Autobahn|Testsuite

Pass	Test case was executed and passed successfully.
Non-Strict	Test case was executed and passed non-strictly. A non-strict behavior is one that does not adhere to a SHOULD-behavior as described in the protocol specification or a well-defined, canonical behavior that appears to be desirable but left open in the protocol specification. An implementation with non-strict behavior is still conformant to the protocol specification.
Fail	Test case was executed and failed. An implementation which fails a test case - other than a performance/limits related one - is non-conforming to a MUST-behavior as described in the protocol specification.
Info	Informational test case which detects certain implementation behavior left unspecified by the spec but nevertheless potentially interesting to implementors.
Missing	Test case is missing, either because it was skipped via the test suite configuration or deactivated, i.e. because the implementation does not implement the tested feature or breaks during running the test case.

1 Framing	WASDv10.3.0+1.0.7	
1.1 Text Messages		
Case 1.1.1	Pass	None
Case 1.1.2	Pass	None
Case 1.1.3	Pass	None
Case 1.1.4	Pass	None
Case 1.1.5	Pass	None
Case 1.1.6	Pass	None
Case 1.1.7	Pass	None
Case 1.1.8	Pass	None

1 Framing	WASDv10.3.0+1.0.7	
1.2 Binary Messages		
Case 1.2.1	Pass	None
Case 1.2.2	Pass	None
Case 1.2.3	Pass	None
Case 1.2.4	Pass	None
Case 1.2.5	Pass	None
Case 1.2.6	Pass	None
Case 1.2.7	Pass	None
Case 1.2.8	Pass	None

2 Pings/Pongs	WASDv10.3.0+1.0.7	
Case 2.1	Pass	None
Case 2.2	Pass	None
Case 2.3	Pass	None
Case 2.4	Pass	None
Case 2.5	Pass	1002
Case 2.6	Pass	None
Case 2.7	Pass	None
Case 2.8	Pass	None
Case 2.9	Pass	None
Case 2.10	Pass	None
Case 2.11	Pass	None

3 Reserved Bits	WASDv10.3.0+1.0.7	
Case 3.1	Pass	1002
Case 3.2	Pass	1002

Case 3.3	Pass	1002
Case 3.4	Pass	1002
Case 3.5	Pass	1002
Case 3.6	Pass	1002
Case 3.7	Pass	1002
4 Opcodes	WASDv10.3.0+1.0.7	
4.1 Non-control Opcodes		
Case 4.1.1	Pass	1002
Case 4.1.2	Pass	1002
Case 4.1.3	Pass	1002
Case 4.1.4	Pass	1002
Case 4.1.5	Pass	1002
4 Opcodes	WASDv10.3.0+1.0.7	
4.2 Control Opcodes		
Case 4.2.1	Pass	1002
Case 4.2.2	Pass	1002
Case 4.2.3	Pass	1002
Case 4.2.4	Pass	1002
Case 4.2.5	Pass	1002
5 Fragmentation	WASDv10.3.0+1.0.7	
Case 5.1	Pass	1002
Case 5.2	Pass	1002
Case 5.3	Pass	None
Case 5.4	Pass	None
Case 5.5	Pass	None
Case 5.6	Pass	None
Case 5.7	Pass	None
Case 5.8	Pass	None
Case 5.9	Pass	1002
Case 5.10	Pass	1002
Case 5.11	Pass	1002
Case 5.12	Pass	1002
Case 5.13	Pass	1002
Case 5.14	Pass	1002
Case 5.15	Pass	1002
Case 5.16	Pass	1002
Case 5.17	Pass	1002
Case 5.18	Pass	1002
Case 5.19	Pass	None
Case 5.20	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.1 Valid UTF-8 with zero payload fragments		
Case 6.1.1	Pass	None
Case 6.1.2	Pass	None
Case 6.1.3	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.2 Valid UTF-8 unfragmented, fragmented on code-points and within code-points		
Case 6.2.1	Pass	None
Case 6.2.2	Pass	None
Case 6.2.3	Pass	None
Case 6.2.4	Pass	None

6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.3 Invalid UTF-8 differently fragmented		
Case 6.3.1	Pass	1007
Case 6.3.2	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.4 Fail-fast on invalid UTF-8		
Case 6.4.1	Pass	1007
Case 6.4.2	Pass	1007
Case 6.4.3	Pass	1007
Case 6.4.4	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.5 Some valid UTF-8 sequences		
Case 6.5.1	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.6 All prefixes of a valid UTF-8 string that contains multi-byte code points		
Case 6.6.1	Pass	1007
Case 6.6.2	Pass	None
Case 6.6.3	Pass	1007
Case 6.6.4	Pass	1007
Case 6.6.5	Pass	None
Case 6.6.6	Pass	1007
Case 6.6.7	Pass	None
Case 6.6.8	Pass	1007
Case 6.6.9	Pass	None
Case 6.6.10	Pass	1007
Case 6.6.11	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.7 First possible sequence of a certain length		
Case 6.7.1	Pass	None
Case 6.7.2	Pass	None
Case 6.7.3	Pass	None
Case 6.7.4	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.8 First possible sequence length 5/6 (invalid codepoints)		
Case 6.8.1	Pass	1007
Case 6.8.2	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.9 Last possible sequence of a certain length		
Case 6.9.1	Pass	None
Case 6.9.2	Pass	None
Case 6.9.3	Pass	None
Case 6.9.4	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.10 Last possible sequence length 4/5/6 (invalid codepoints)		
Case 6.10.1	Pass	1007
Case 6.10.2	Pass	1007
Case 6.10.3	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.11 Other boundary conditions		
Case 6.11.1	Pass	None
Case 6.11.2	Pass	None

Case 6.11.3	Pass	None
Case 6.11.4	Pass	None
Case 6.11.5	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.12 Unexpected continuation bytes		
Case 6.12.1	Pass	1007
Case 6.12.2	Pass	1007
Case 6.12.3	Pass	1007
Case 6.12.4	Pass	1007
Case 6.12.5	Pass	1007
Case 6.12.6	Pass	1007
Case 6.12.7	Pass	1007
Case 6.12.8	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.13 Lonely start characters		
Case 6.13.1	Pass	1007
Case 6.13.2	Pass	1007
Case 6.13.3	Pass	1007
Case 6.13.4	Pass	1007
Case 6.13.5	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.14 Sequences with last continuation byte missing		
Case 6.14.1	Pass	1007
Case 6.14.2	Pass	1007
Case 6.14.3	Pass	1007
Case 6.14.4	Pass	1007
Case 6.14.5	Pass	1007
Case 6.14.6	Pass	1007
Case 6.14.7	Pass	1007
Case 6.14.8	Pass	1007
Case 6.14.9	Pass	1007
Case 6.14.10	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.15 Concatenation of incomplete sequences		
Case 6.15.1	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.16 Impossible bytes		
Case 6.16.1	Pass	1007
Case 6.16.2	Pass	1007
Case 6.16.3	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.17 Examples of an overlong ASCII character		
Case 6.17.1	Pass	1007
Case 6.17.2	Pass	1007
Case 6.17.3	Pass	1007
Case 6.17.4	Pass	1007
Case 6.17.5	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.18 Maximum overlong sequences		
Case 6.18.1	Pass	1007
Case 6.18.2	Pass	1007

Case 6.18.3	Pass	1007
Case 6.18.4	Pass	1007
Case 6.18.5	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.19 Overlong representation of the NUL character		
Case 6.19.1	Pass	1007
Case 6.19.2	Pass	1007
Case 6.19.3	Pass	1007
Case 6.19.4	Pass	1007
Case 6.19.5	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.20 Single UTF-16 surrogates		
Case 6.20.1	Pass	1007
Case 6.20.2	Pass	1007
Case 6.20.3	Pass	1007
Case 6.20.4	Pass	1007
Case 6.20.5	Pass	1007
Case 6.20.6	Pass	1007
Case 6.20.7	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.21 Paired UTF-16 surrogates		
Case 6.21.1	Pass	1007
Case 6.21.2	Pass	1007
Case 6.21.3	Pass	1007
Case 6.21.4	Pass	1007
Case 6.21.5	Pass	1007
Case 6.21.6	Pass	1007
Case 6.21.7	Pass	1007
Case 6.21.8	Pass	1007
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.22 Non-character code points (valid UTF-8)		
Case 6.22.1	Pass	None
Case 6.22.2	Pass	None
Case 6.22.3	Pass	None
Case 6.22.4	Pass	None
Case 6.22.5	Pass	None
Case 6.22.6	Pass	None
Case 6.22.7	Pass	None
Case 6.22.8	Pass	None
Case 6.22.9	Pass	None
Case 6.22.10	Pass	None
Case 6.22.11	Pass	None
Case 6.22.12	Pass	None
Case 6.22.13	Pass	None
Case 6.22.14	Pass	None
Case 6.22.15	Pass	None
Case 6.22.16	Pass	None
Case 6.22.17	Pass	None
Case 6.22.18	Pass	None
Case 6.22.19	Pass	None
Case 6.22.20	Pass	None

Case 6.22.21	Pass	None
Case 6.22.22	Pass	None
Case 6.22.23	Pass	None
Case 6.22.24	Pass	None
Case 6.22.25	Pass	None
Case 6.22.26	Pass	None
Case 6.22.27	Pass	None
Case 6.22.28	Pass	None
Case 6.22.29	Pass	None
Case 6.22.30	Pass	None
Case 6.22.31	Pass	None
Case 6.22.32	Pass	None
Case 6.22.33	Pass	None
Case 6.22.34	Pass	None
6 UTF-8 Handling	WASDv10.3.0+1.0.7	
6.23 Unicode specials (i.e. replacement char)		
Case 6.23.1	Pass	None
Case 6.23.2	Pass	None
Case 6.23.3	Pass	None
Case 6.23.4	Pass	None
Case 6.23.5	Pass	None
Case 6.23.6	Pass	None
Case 6.23.7	Pass	None
7 Close Handling	WASDv10.3.0+1.0.7	
7.1 Basic close behavior (fuzzer initiated)		
Case 7.1.1	Pass	None
Case 7.1.2	Pass	None
Case 7.1.3	Pass	None
Case 7.1.4	Pass	None
Case 7.1.5	Pass	None
Case 7.1.6	Info	None
7 Close Handling	WASDv10.3.0+1.0.7	
7.3 Close frame structure: payload length (fuzzer initiated)		
Case 7.3.1	Pass	None
Case 7.3.2	Pass	None
Case 7.3.3	Pass	None
Case 7.3.4	Pass	None
Case 7.3.5	Pass	None
Case 7.3.6	Pass	1002
7 Close Handling	WASDv10.3.0+1.0.7	
7.5 Close frame structure: payload value (fuzzer initiated)		
Case 7.5.1	Pass	None
7 Close Handling	WASDv10.3.0+1.0.7	
7.7 Close frame structure: valid close codes (fuzzer initiated)		
Case 7.7.1	Pass	None
Case 7.7.2	Pass	None
Case 7.7.3	Pass	None
Case 7.7.4	Pass	None
Case 7.7.5	Pass	None
Case 7.7.6	Pass	None
Case 7.7.7	Pass	None

Case 7.7.8	Pass	None
Case 7.7.9	Pass	None
Case 7.7.10	Pass	None
Case 7.7.11	Pass	None
Case 7.7.12	Pass	None
Case 7.7.13	Pass	None
7 Close Handling	WASDv10.3.0+1.0.7	
7.9 Close frame structure: invalid close codes (fuzzer initiated)		
Case 7.9.1	Pass	None
Case 7.9.2	Pass	None
Case 7.9.3	Pass	None
Case 7.9.4	Pass	None
Case 7.9.5	Pass	None
Case 7.9.6	Pass	None
Case 7.9.7	Pass	None
Case 7.9.8	Pass	None
Case 7.9.9	Pass	None
Case 7.9.10	Pass	None
Case 7.9.11	Pass	None
Case 7.9.12	Pass	None
Case 7.9.13	Pass	None
7 Close Handling	WASDv10.3.0+1.0.7	
7.13 Informational close information (fuzzer initiated)		
Case 7.13.1	Info	None
Case 7.13.2	Info	None
9 Limits/Performance	WASDv10.3.0+1.0.7	
9.1 Text Message (increasing size)		
Case 9.1.1	Pass 185 ms	None
Case 9.1.2	Pass 4189 ms	None
Case 9.1.3	Pass 4812 ms	None
Case 9.1.4	Pass 16929 ms	None
Case 9.1.5	Pass 41085 ms	None
Case 9.1.6	Pass 98496 ms	None
9 Limits/Performance	WASDv10.3.0+1.0.7	
9.2 Binary Message (increasing size)		
Case 9.2.1	Pass 2003 ms	None
Case 9.2.2	Pass 1597 ms	None
Case 9.2.3	Pass 9477 ms	None
Case 9.2.4	Pass 9924 ms	None
Case 9.2.5	Pass 94844 ms	None
Case 9.2.6	Pass 98701 ms	None
9 Limits/Performance	WASDv10.3.0+1.0.7	
9.3 Fragmented Text Message (fixed size, increasing fragment size)		
Case 9.3.1	Pass	

	19856 ms	None
Case 9.3.2	Pass 37574 ms	None
Case 9.3.3	Pass 17976 ms	None
Case 9.3.4	Pass 23315 ms	None
Case 9.3.5	Pass 40107 ms	None
Case 9.3.6	Pass 34001 ms	None
Case 9.3.7	Pass 33722 ms	None
Case 9.3.8	Pass 42445 ms	None
Case 9.3.9	Pass 88072 ms	None

9 Limits/Performance — WASDv10.3.0+1.0.7

9.4 Fragmented Binary Message (fixed size, increasing fragment size)

Case 9.4.1	Pass 20693 ms	None
Case 9.4.2	Pass 14260 ms	None
Case 9.4.3	Pass 23714 ms	None
Case 9.4.4	Pass 66395 ms	None
Case 9.4.5	Pass 54225 ms	None
Case 9.4.6	Pass 68560 ms	None
Case 9.4.7	Pass 63372 ms	None
Case 9.4.8	Pass 91702 ms	None
Case 9.4.9	Pass 98040 ms	None

9 Limits/Performance — WASDv10.3.0+1.0.7

9.5 Text Message (fixed size, increasing chop size)

Case 9.5.1	Pass 12177 ms	None
Case 9.5.2	Pass 4688 ms	None
Case 9.5.3	Pass 56995 ms	None
Case 9.5.4	Pass 31490 ms	None
Case 9.5.5	Pass 3832 ms	None
Case 9.5.6	Pass 78092 ms	None

9 Limits/Performance — WASDv10.3.0+1.0.7

9.6 Binary Text Message (fixed size, increasing chop size)

Case 9.6.1	Pass 32624 ms	None
Case 9.6.2	Pass 18776 ms	None
Case 9.6.3	Pass 60278 ms	None
Case 9.6.4	Pass 64128 ms	None
Case 9.6.5	Pass 59977 ms	None

Case 9.6.6	Pass 44311 ms	None
9 Limits/Performance	WASDv10.3.0+1.0.7	
9.7 Text Message Roundtrip Time (fixed number, increasing size)		
Case 9.7.1	Pass 3464 ms	None
Case 9.7.2	Pass 3612 ms	None
Case 9.7.3	Pass 3878 ms	None
Case 9.7.4	Pass 7176 ms	None
Case 9.7.5	Pass 9291 ms	None
Case 9.7.6	Pass 21039 ms	None
9 Limits/Performance	WASDv10.3.0+1.0.7	
9.8 Binary Message Roundtrip Time (fixed number, increasing size)		
Case 9.8.1	Pass 3507 ms	None
Case 9.8.2	Pass 3480 ms	None
Case 9.8.3	Pass 3777 ms	None
Case 9.8.4	Pass 7478 ms	None
Case 9.8.5	Pass 8810 ms	None
Case 9.8.6	Pass 17292 ms	None
10 Misc	WASDv10.3.0+1.0.7	
10.1 Auto-Fragmentation		
Case 10.1.1	Pass	None

Case 1.1.1

Case Description   Up

Send text message with payload 0.

Case Expectation

Receive echo'ed text message (with empty payload). Clean close with normal code.

Case 1.1.2

Case Description   Up

Send text message message with payload of length 125.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.3

Case Description

Send text message message with payload of length 126.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.4

Case Description

Send text message message with payload of length 127.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.5

Case Description

Send text message message with payload of length 128.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.6

Case Description

Send text message message with payload of length 65535.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.7

Case Description

Send text message message with payload of length 65536.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.1.8

Case Description

Send text message message with payload of length 65536. Sent out data in chops of 997 octets.

Case Expectation

Receive echo'ed text message (with payload as sent). Clean close with normal code.

Case 1.2.1

Case Description   Up

Send binary message with payload 0.

Case Expectation

Receive echo'ed binary message (with empty payload). Clean close with normal code.

Case 1.2.2

Case Description   Up

Send binary message message with payload of length 125.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.3

Case Description   Up

Send binary message message with payload of length 126.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.4

Case Description   Up

Send binary message message with payload of length 127.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.5

Case Description   Up

Send binary message message with payload of length 128.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.6

Case Description   Up

Send binary message message with payload of length 65535.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.7

Case Description

Send binary message message with payload of length 65536.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 1.2.8

Case Description

Send binary message message with payload of length 65536. Sent out data in chops of 997 octets.

Case Expectation

Receive echo'ed binary message (with payload as sent). Clean close with normal code.

Case 2.1

Case Description

Send ping without payload.

Case Expectation

Pong (with empty payload) is sent in reply to Ping. Clean close with normal code.

Case 2.2

Case Description

Send ping with small text payload.

Case Expectation

Pong with payload echo'ed is sent in reply to Ping. Clean close with normal code.

Case 2.3

Case Description

Send ping with small binary (non UTF-8) payload.

Case Expectation

Pong with payload echo'ed is sent in reply to Ping. Clean close with normal code.

Case 2.4

Case Description

Send ping with binary payload of 125 octets.

Case Expectation

Pong with payload echo'ed is sent in reply to Ping. Clean close with normal code.

Case 2.5

Case Description

Send ping with binary payload of 126 octets.

Case Expectation

Connection is failed immediately (1002/Protocol Error), since control frames are only allowed to have payload up to and including 125 octets..

Case 2.6

Case Description

Send ping with binary payload of 125 octets, send in octet-wise chops.

Case Expectation

Pong with payload echo'ed is sent in reply to Ping. Implementations must be TCP clean. Clean close with normal code.

Case 2.7

Case Description

Send unsolicited pong without payload. Verify nothing is received. Clean close with normal code.

Case Expectation

Nothing.

Case 2.8

Case Description

Send unsolicited pong with payload. Verify nothing is received. Clean close with normal code.

Case Expectation

Nothing.

Case 2.9

Case Description

Send unsolicited pong with payload. Send ping with payload. Verify pong for ping is received.

Case Expectation

Nothing in reply to own Pong, but Pong with payload echo'ed in reply to Ping. Clean close with normal code.

Case 2.10

Case Description

Send 10 Pings with payload.

Case Expectation

Pongs for our Pings with all the payloads. Note: This is not required by the Spec .. but we check for this behaviour anyway. Clean close with normal code.

Case 2.11

Case Description

Send 10 Pings with payload. Send out octets in octet-wise chops.

Case Expectation

Pongs for our Pings with all the payloads. Note: This is not required by the Spec .. but we check for this behaviour anyway. Clean close with normal code.

Case 3.1

Case Description

Send small text message with **RSV = 1**.

Case Expectation

The connection is failed immediately (1002/protocol error), since RSV must be 0, when no extension defining RSV meaning has been negoiated.

Case 3.2

Case Description

Send small text message, then send again with **RSV = 2**, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since RSV must be 0, when no extension defining RSV meaning has been negoiated. The Pong is not received.

Case 3.3

Case Description

Send small text message, then send again with **RSV = 3**, then send Ping. Octets are sent in frame-wise chops. Octets are sent in octet-wise chops.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since RSV must be 0, when no extension defining RSV meaning has been negoiated. The Pong is not received.

Case 3.4

Case Description

Send small text message, then send again with **RSV = 4**, then send Ping. Octets are sent in octet-wise chops.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since RSV must be 0, when no extension defining RSV meaning has been negoiated. The Pong is not received.

Case 3.5

Case Description

Send small binary message with **RSV = 5**.

Case Expectation

The connection is failed immediately, since RSV must be 0.

Case 3.6

Case Description

Send Ping with **RSV = 6**.

Case Expectation

The connection is failed immediately, since RSV must be 0.

Case 3.7

Case Description

Send Close with **RSV = 7**.

Case Expectation

The connection is failed immediately, since RSV must be 0.

Case 4.1.1

Case Description

Send frame with reserved non-control **Opcode = 3**.

Case Expectation

The connection is failed immediately.

Case 4.1.2

Case Description

Send frame with reserved non-control **Opcode = 4** and non-empty payload.

Case Expectation

The connection is failed immediately.

Case 4.1.3

Case Description   Up

Send small text message, then send frame with reserved non-control **Opcode = 5**, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 4.1.4

Case Description   Up

Send small text message, then send frame with reserved non-control **Opcode = 6** and non-empty payload, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 4.1.5

Case Description   Up

Send small text message, then send frame with reserved non-control **Opcode = 7** and non-empty payload, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 4.2.1

Case Description   Up

Send frame with reserved control **Opcode = 11**.

Case Expectation

The connection is failed immediately.

Case 4.2.2

Case Description   Up

Send frame with reserved control **Opcode = 12** and non-empty payload.

Case Expectation

The connection is failed immediately.

Case 4.2.3

Case Description   Up

Send small text message, then send frame with reserved control **Opcode = 13**, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 4.2.4

Case Description Up

Send small text message, then send frame with reserved control **Opcode = 14** and non-empty payload, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 4.2.5

Case Description Up

Send small text message, then send frame with reserved control **Opcode = 15** and non-empty payload, then send Ping.

Case Expectation

Echo for first message is received, but then connection is failed immediately, since reserved opcode frame is used. A Pong is not received.

Case 5.1

Case Description Up

Send Ping fragmented into 2 fragments.

Case Expectation

Connection is failed immediately, since control message MUST NOT be fragmented.

Case 5.2

Case Description Up

Send Pong fragmented into 2 fragments.

Case Expectation

Connection is failed immediately, since control message MUST NOT be fragmented.

Case 5.3

Case Description Up

Send text Message fragmented into 2 fragments.

Case Expectation

Message is processed and echo'ed back to us.

Case 5.4

Case Description

Send text Message fragmented into 2 fragments, octets are sent in frame-wise chops.

Case Expectation

Message is processed and echo'ed back to us.

Case 5.5

Case Description

Send text Message fragmented into 2 fragments, octets are sent in octet-wise chops.

Case Expectation

Message is processed and echo'ed back to us.

Case 5.6

Case Description

Send text Message fragmented into 2 fragments, one ping with payload in-between.

Case Expectation

A pong is received, then the message is echo'ed back to us.

Case 5.7

Case Description

Send text Message fragmented into 2 fragments, one ping with payload in-between. Octets are sent in frame-wise chops.

Case Expectation

A pong is received, then the message is echo'ed back to us.

Case 5.8

Case Description

Send text Message fragmented into 2 fragments, one ping with payload in-between. Octets are sent in octet-wise chops.

Case Expectation

A pong is received, then the message is echo'ed back to us.

Case 5.9

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = true, where there is nothing to continue, sent in one chop.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.10

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = true, where there is nothing to continue, sent in per-frame chops.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.11

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = true, where there is nothing to continue, sent in octet-wise chops.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.12

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = false, where there is nothing to continue, sent in one chop.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.13

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = false, where there is nothing to continue, sent in per-frame chops.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.14

Case Description

Send unfragmented Text Message after Continuation Frame with FIN = false, where there is nothing to continue, sent in octet-wise chops.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.15

Case Description

Send text Message fragmented into 2 fragments, then Continuation Frame with FIN = false where there is nothing to continue, then unfragmented Text Message, all sent in one chop.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.16

Case Description   Up

Repeated 2x: Continuation Frame with FIN = false (where there is nothing to continue), then text Message fragmented into 2 fragments.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.17

Case Description   Up

Repeated 2x: Continuation Frame with FIN = true (where there is nothing to continue), then text Message fragmented into 2 fragments.

Case Expectation

The connection is failed immediately, since there is no message to continue.

Case 5.18

Case Description   Up

Send text Message fragmented into 2 fragments, with both frame opcodes set to text, sent in one chop.

Case Expectation

The connection is failed immediately, since all data frames after the initial data frame must have opcode 0.

Case 5.19

Case Description   Up

A fragmented text message is sent in multiple frames. After sending the first 2 frames of the text message, a Ping is sent. Then we wait 1s, then we send 2 more text fragments, another Ping and then the final text fragment. Everything is legal.

Case Expectation

The peer immediately answers the first Ping before it has received the last text message fragment. The peer pong's back the Ping's payload exactly, and echo's the payload of the fragmented message back to us.

Case 5.20

Case Description   Up

Same as Case 5.19, but send all frames with SYNC = True. Note, this does not change the octets sent in any way, only how the stream is chopped up on the wire.

Case Expectation

Same as Case 5.19. Implementations must be agnostic to how octet stream is chopped up on wire (must be TCP clean).

Case 6.1.1

Case Description

Send text message of length 0.

Case Expectation

A message is echo'ed back to us (with empty payload).

Case 6.1.2

Case Description

Send fragmented text message, 3 fragments each of length 0.

Case Expectation

A message is echo'ed back to us (with empty payload).

Case 6.1.3

Case Description

Send fragmented text message, 3 fragments, first and last of length 0, middle non-empty.

Case Expectation

A message is echo'ed back to us (with payload = payload of middle fragment).

Case 6.2.1

Case Description

Send a valid UTF-8 text message in one fragment.

MESSAGE:
Hello-μ@ßöäüàá-UTF-8!!
48656c6c6f2dc2b540c39fc3b6c3a4c3bcc3a0c3a12d5554462d382121

Case Expectation

The message is echo'ed back to us.

Case 6.2.2

Case Description

Send a valid UTF-8 text message in two fragments, fragmented on UTF-8 code point boundary.

MESSAGE FRAGMENT 1:
Hello-μ@ßöä
48656c6c6f2dc2b540c39fc3b6c3a4

MESSAGE FRAGMENT 2:
üàá-UTF-8!!
c3bcc3a0c3a12d5554462d382121

Case Expectation

The message is echo'ed back to us.

Case 6.2.3

Case Description

Send a valid UTF-8 text message in fragments of 1 octet, resulting in frames ending on positions which are not code point ends.

MESSAGE:
Hello-µ@ßöäüàá-UTF-8!!
48656c6c6f2dc2b540c39fc3b6c3a4c3bcc3a0c3a12d5554462d382121

Case Expectation

The message is echo'ed back to us.

Case 6.2.4

Case Description

Send a valid UTF-8 text message in fragments of 1 octet, resulting in frames ending on positions which are not code point ends.

MESSAGE:
κόσμε
cebae1bdb9cf83cebcceb5

Case Expectation

The message is echo'ed back to us.

Case 6.3.1

Case Description

Send invalid UTF-8 text message unfragmented.

MESSAGE:
cebae1bdb9cf83cebcceb5eda080656469746564

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.3.2

Case Description

Send invalid UTF-8 text message in fragments of 1 octet, resulting in frames ending on positions which are not code point ends.

MESSAGE:
cebae1bdb9cf83cebcceb5eda080656469746564

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.4.1

Case Description

Send invalid UTF-8 text message in 3 fragments (frames). First frame payload is valid, then wait, then 2nd frame which contains the payload making the sequence invalid, then wait, then 3rd frame with rest. Note that PART1 and PART3 are valid UTF-8 in themselves, PART2 is a 0x110000 encoded as in the UTF-8 integer encoding scheme, but the codepoint is invalid (out of range).

MESSAGE PARTS:
PART1 = cebae1bdb9cf83cebcceb5
PART2 = f4908080
PART3 = 656469746564

Case Expectation

The first frame is accepted, we expect to timeout on the first wait. The 2nd frame should be rejected immediately (fail fast on UTF-8). If we timeout, we expect the connection is failed at least then, since the complete message payload is not valid UTF-8.

Case 6.4.2

Case Description

Same as Case 6.4.1, but in 2nd frame, we send only up to and including the octet making the complete payload invalid.

MESSAGE PARTS:
PART1 = cebae1bdb9cf83cebcceb5f4
PART2 = 90
PART3 = 8080656469746564

Case Expectation

The first frame is accepted, we expect to timeout on the first wait. The 2nd frame should be rejected immediately (fail fast on UTF-8). If we timeout, we expect the connection is failed at least then, since the complete message payload is not valid UTF-8.

Case 6.4.3

Case Description

Same as Case 6.4.1, but we send message not in 3 frames, but in 3 chops of the same message frame.

MESSAGE PARTS:
PART1 = cebae1bdb9cf83cebcceb5
PART2 = f4908080
PART3 = 656469746564

Case Expectation

The first chop is accepted, we expect to timeout on the first wait. The 2nd chop should be rejected immediately (fail fast on UTF-8). If we timeout, we expect the connection is failed at least then, since the complete message payload is not valid UTF-8.

Case 6.4.4

Case Description

Same as Case 6.4.2, but we send message not in 3 frames, but in 3 chops of the same message frame.

MESSAGE PARTS:
PART1 = cebae1bdb9cf83cebcceb5f4
PART2 = 90
PART3 =

Case Expectation

The first chop is accepted, we expect to timeout on the first wait. The 2nd chop should be rejected immediately (fail fast on UTF-8). If we timeout, we expect the connection is failed at least then, since the complete message payload is not valid UTF-8.

Case 6.5.1

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83cebcceb5

Case Expectation

The message is echo'ed back to us.

Case 6.6.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xce

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.2

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xceba

Case Expectation

The message is echo'ed back to us.

Case 6.6.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xcebae1

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.4

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xcebae1bd

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.5

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9

Case Expectation

The message is echo'ed back to us.

Case 6.6.6

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.7

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83

Case Expectation

The message is echo'ed back to us.

Case 6.6.8

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83ce

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.9

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83cebc

Case Expectation

The message is echo'ed back to us.

Case 6.6.10

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83cebcce

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.6.11

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xcebae1bdb9cf83cebcceb5

Case Expectation

The message is echo'ed back to us.

Case 6.7.1

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0x00

Case Expectation

The message is echo'ed back to us.

Case 6.7.2

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xc280

Case Expectation

The message is echo'ed back to us.

Case 6.7.3

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xe0a080

Case Expectation

The message is echo'ed back to us.

Case 6.7.4

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf0908080

Case Expectation

The message is echo'ed back to us.

Case 6.8.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf888808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.8.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc8480808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.9.1

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0x7f

Case Expectation

The message is echo'ed back to us.

Case 6.9.2

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xdfbf

Case Expectation

The message is echo'ed back to us.

Case 6.9.3

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.9.4

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf48fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.10.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf7bfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.10.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfbbfbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.10.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfdbfbfbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.11.1

Case Description

Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xed9fbf

Case Expectation

The message is echo'ed back to us.

Case 6.11.2

Case Description

Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xee8080

Case Expectation

The message is echo'ed back to us.

Case 6.11.3

Case Description

Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbd

Case Expectation

The message is echo'ed back to us.

Case 6.11.4

Case Description

Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf48fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.11.5

Case Description

Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf4908080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80bf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.4

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80bf80

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.5

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80bf80bf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.6

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80bf80bf80

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.7

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0x80bf80bf80bf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.12.8

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload:
0x808182838485868788898a8b8c8d8e8f909192939495969798999a9b9c9d9e9fa0a1a2a3a4a5a6a7a8a9aaabacadaeafb0b1b2b3b4b5b6b7b8b9babbbcb

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.13.1

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload:
0xc020c120c220c320c420c520c620c720c820c920ca20cb20cc20cd20ce20cf20d020d120d220d320d420d520d620d720d820d920da20db20dc20dd20de20

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.13.2

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xe020e120e220e320e420e520e620e720e820e920ea20eb20ec20ed20ee20

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.13.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf020f120f220f320f420f520f620

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.13.4

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf820f920fa20

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.13.5

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc20

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xc0

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xe080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf08080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.4

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf8808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.5

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc80808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.6

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xdf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.7

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xefbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.8

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf7bfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.9

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfbbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.14.10

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfdbfbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.15.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xc0e080f08080f8808080fc80808080dfefbff7bfbffbbfbfbffdbfbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.16.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfe

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.16.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xff

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.16.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfefeffff

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.17.1

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xc0af

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.17.2

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xe080af

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.17.3

Case Description

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf08080af

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.17.4

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf8808080af

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.17.5

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc80808080af

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.18.1

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xc1bf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.18.2

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xe09fbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.18.3

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf08fbfbf

Case 6.18.4

Case Description Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf887bfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.18.5

Case Description Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc83bfbfbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.19.1

Case Description Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xc080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.19.2

Case Description Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xe08080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.19.3

Case Description Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf0808080

Case 6.19.4

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xf880808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.19.5

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xfc8080808080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.1

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xeda080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.2

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedadbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.3

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedae80

Case 6.20.4

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedafbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.5

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedb080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.6

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedbe80

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.20.7

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.1

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xeda080edb080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.2

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xeda080edbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.3

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedadbfedb080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.4

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedadbfedbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.5

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedae80edb080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.6

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedae80edbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.7

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedafbfedb080

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.21.8

Case Description   Up

Send a text message with payload which is not valid UTF-8 in one fragment.

Payload: 0xedafbfedbfbf

Case Expectation

The connection is failed immediately, since the payload is not valid UTF-8.

Case 6.22.1

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.2

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.3

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf09fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.4

Case Description Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf09fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.5

Case Description Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf0afbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.6

Case Description Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf0afbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.7

Case Description Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf0bfbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.8

Case Description Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf0bfbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.9

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf18fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.10

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf18fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.11

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf19fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.12

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf19fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.13

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf1afbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.14

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf1afbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.15

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf1bfbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.16

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf1bfbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.17

Case Description   Up

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf28fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.18

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf28fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.19

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf29fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.20

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf29fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.21

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf2afbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.22

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf2afbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.23

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf2bfbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.24

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf2bfbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.25

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf38fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.26

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf38fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.27

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf39fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.28

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf39fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.29

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf3afbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.30

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf3afbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.31

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf3bfbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.32

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf3bfbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.22.33

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf48fbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.22.34

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xf48fbfbf

Case Expectation

The message is echo'ed back to us.

Case 6.23.1

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfb9

Case Expectation

The message is echo'ed back to us.

Case 6.23.2

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfba

Case Expectation

The message is echo'ed back to us.

Case 6.23.3

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbb

Case Expectation

The message is echo'ed back to us.

Case 6.23.4

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbc

Case Expectation

The message is echo'ed back to us.

Case 6.23.5

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbd

Case Expectation

The message is echo'ed back to us.

Case 6.23.6

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbe

Case Expectation

The message is echo'ed back to us.

Case 6.23.7

Case Description

Send a text message with payload which is valid UTF-8 in one fragment.

Payload: 0xefbfbf

Case Expectation

The message is echo'ed back to us.

Case 7.1.1

Case Description

Send a message followed by a close frame

Case Expectation

Echoed message followed by clean close with normal code.

Case 7.1.2

Case Description

Send two close frames

Case Expectation

Clean close with normal code. Second close frame ignored.

Case 7.1.3

Case Description

Send a ping after close message

Case Expectation

Clean close with normal code, no pong.

Case 7.1.4

Case Description

Send text message after sending a close frame.

Case Expectation

Clean close with normal code. Text message ignored.

Case 7.1.5

Case Description

Send message fragment1 followed by close then fragment

Case Expectation

Clean close with normal code.

Case 7.1.6

Case Description

Send 256K message followed by close then a ping

Case Expectation

Case outcome depends on implimentation defined close behavior. Message and close frame are sent back to back. If the close frame is processed before the text message write is complete (as can happen in asyncronous processing models) the close frame is processed first and the text message may not be recieved or may only be partially recieved.

Case 7.3.1

Case Description

Send a close frame with payload length 0 (no close code, no close reason)

Case Expectation

Clean close with normal code.

Case 7.3.2

Case Description   Up

Send a close frame with payload length 1

Case Expectation

Clean close with protocol error or drop TCP.

Case 7.3.3

Case Description   Up

Send a close frame with payload length 2 (regular close with a code)

Case Expectation

Clean close with normal code.

Case 7.3.4

Case Description   Up

Send a close frame with close code and close reason

Case Expectation

Clean close with normal code.

Case 7.3.5

Case Description   Up

Send a close frame with close code and close reason of maximum length (123)

Case Expectation

Clean close with normal code.

Case 7.3.6

Case Description   Up

Send a close frame with close code and close reason which is too long (124) - total frame payload 126 octets

Case Expectation

Clean close with protocol error code or dropped TCP connection.

Case 7.5.1

Case Description   Up

Send a close frame with invalid UTF8 payload

Case Expectation

Clean close with protocol error or invalid utf8 code or dropped TCP.

Case 7.7.1

Case Description   Up

Send close with valid close code 1000

Case Expectation

Clean close with normal or echoed code

Case 7.7.2

Case Description   Up

Send close with valid close code 1001

Case Expectation

Clean close with normal or echoed code

Case 7.7.3

Case Description   Up

Send close with valid close code 1002

Case Expectation

Clean close with normal or echoed code

Case 7.7.4

Case Description   Up

Send close with valid close code 1003

Case Expectation

Clean close with normal or echoed code

Case 7.7.5

Case Description   Up

Send close with valid close code 1007

Case Expectation

Clean close with normal or echoed code

Case 7.7.6

Case Description   Up

Send close with valid close code 1008

Case Expectation

Clean close with normal or echoed code

Case 7.7.7

Case Description   Up

Send close with valid close code 1009

Case Expectation

Clean close with normal or echoed code

Case 7.7.8

Case Description   Up

Send close with valid close code 1010

Case Expectation

Clean close with normal or echoed code

Case 7.7.9

Case Description   Up

Send close with valid close code 1011

Case Expectation

Clean close with normal or echoed code

Case 7.7.10

Case Description   Up

Send close with valid close code 3000

Case Expectation

Clean close with normal or echoed code

Case 7.7.11

Case Description   Up

Send close with valid close code 3999

Case Expectation

Clean close with normal or echoed code

Case 7.7.12

Case Description   Up

Send close with valid close code 4000

Case Expectation

Clean close with normal or echoed code

Case 7.7.13

Case Description   Up

Send close with valid close code 4999

Case Expectation

Clean close with normal or echoed code

Case 7.9.1

Case Description   Up

Send close with invalid close code 0

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.2

Case Description   Up

Send close with invalid close code 999

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.3

Case Description   Up

Send close with invalid close code 1004

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.4

Case Description   Up

Send close with invalid close code 1005

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.5

Case Description

Up

Send close with invalid close code 1006

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.6

Case Description

Up

Send close with invalid close code 1012

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.7

Case Description

Up

Send close with invalid close code 1013

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.8

Case Description

Up

Send close with invalid close code 1014

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.9

Case Description

Up

Send close with invalid close code 1015

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.10

Case Description   Up

Send close with invalid close code 1016

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.11

Case Description   Up

Send close with invalid close code 1100

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.12

Case Description   Up

Send close with invalid close code 2000

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.9.13

Case Description   Up

Send close with invalid close code 2999

Case Expectation

Clean close with protocol error code or drop TCP

Case 7.13.1

Case Description   Up

Send close with close code 5000

Case Expectation

Actual events are undefined by the spec.

Case 7.13.2

Case Description   Up

Send close with close code 65536

Case Expectation

Actual events are undefined by the spec.

Case 9.1.1

Case Description

Send text message message with payload of length 64 * 2**10 (64k).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.1.2

Case Description

Send text message message with payload of length 256 * 2**10 (256k).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.1.3

Case Description

Send text message message with payload of length 1 * 2**20 (1M).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.1.4

Case Description

Send text message message with payload of length 4 * 2**20 (4M).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.1.5

Case Description

Send text message message with payload of length 8 * 2**20 (8M).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.1.6

Case Description

Send text message message with payload of length 16 * 2**20 (16M).

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.2.1

Case Description Up

Send binary message message with payload of length 64 * 2**10 (64k).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.2.2

Case Description Up

Send binary message message with payload of length 256 * 2**10 (256k).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.2.3

Case Description Up

Send binary message message with payload of length 1 * 2**20 (1M).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.2.4

Case Description Up

Send binary message message with payload of length 4 * 2**20 (4M).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.2.5

Case Description Up

Send binary message message with payload of length 8 * 2**20 (16M).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.2.6

Case Description

Send binary message message with payload of length 16 * 2**20 (16M).

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.3.1

Case Description

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 64.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.2

Case Description

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 256.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.3

Case Description

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 1k.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.4

Case Description

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 4k.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.5

Case Description

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 16k.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.6

Case Description   Up

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 64k.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.7

Case Description   Up

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 256k.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.8

Case Description   Up

Send fragmented text message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 1M.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.3.9

Case Description   Up

Send fragmented text message message with message payload of length 4 * 2**20 (8M). Sent out in fragments of 4M.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.4.1

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 64.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.2

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 256.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.3

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 1k.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.4

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 4k.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.5

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 16k.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.6

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 64k.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.7

Case Description   Up

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 256k.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.8

Case Description

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 1M.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.4.9

Case Description

Send fragmented binary message message with message payload of length 4 * 2**20 (4M). Sent out in fragments of 4M.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.5.1

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 64 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.5.2

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 128 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.5.3

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 256 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.5.4

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 512 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.5.5

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 1024 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.5.6

Case Description

Send text message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 2048 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.6.1

Case Description

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 64 octets.

Case Expectation

Receive echo'ed binary message (with payload as sent).

Case 9.6.2

Case Description

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 128 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.6.3

Case Description

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 256 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.6.4

Case Description

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 512 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.6.5

Case Description Up

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 1024 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.6.6

Case Description Up

Send binary message message with payload of length 1 * 2**20 (1M). Sent out data in chops of 2048 octets.

Case Expectation

Receive echo'ed text message (with payload as sent).

Case 9.7.1

Case Description Up

Send 1000 text messages of payload size 0 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 60 secs.

Case 9.7.2

Case Description Up

Send 1000 text messages of payload size 16 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 60 secs.

Case 9.7.3

Case Description Up

Send 1000 text messages of payload size 64 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 60 secs.

Case 9.7.4

Case Description

Send 1000 text messages of payload size 256 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 120 secs.

Case 9.7.5

Case Description

Send 1000 text messages of payload size 1024 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 240 secs.

Case 9.7.6

Case Description

Send 1000 text messages of payload size 4096 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed text messages (with payload as sent). Timeout case after 480 secs.

Case 9.8.1

Case Description

Send 1000 binary messages of payload size 0 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 60 secs.

Case 9.8.2

Case Description

Send 1000 binary messages of payload size 16 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 60 secs.

Case 9.8.3

Case Description

Send 1000 binary messages of payload size 64 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 60 secs.

Case 9.8.4

Case Description   Up

Send 1000 binary messages of payload size 256 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 120 secs.

Case 9.8.5

Case Description   Up

Send 1000 binary messages of payload size 1024 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 240 secs.

Case 9.8.6

Case Description   Up

Send 1000 binary messages of payload size 4096 to measure implementation/network RTT (round trip time) / latency.

Case Expectation

Receive echo'ed binary messages (with payload as sent). Timeout case after 480 secs.

Case 10.1.1

Case Description   Up

Send text message with payload of length 65536 auto-fragmented with **autoFragmentSize = 1300**.

Case Expectation

Receive echo'ed text message (with payload as sent and transmitted frame counts as expected). Clean close with normal code.

Toggle Details