

Unittest for socket_protocol

December 27, 2019

Contents

1	Test Information	4
1.1	Test Candidate Information	4
1.2	Unittest Information	4
1.3	Test System Information	4
2	Statistic	4
2.1	Test-Statistic for testrun with python 2.7.17 (final)	4
2.2	Test-Statistic for testrun with python 3.6.9 (final)	5
2.3	Coverage Statistic	5
3	Testcases with no corresponding Requirement	6
3.1	Summary for testrun with python 2.7.17 (final)	6
3.1.1	socket_protocol.pure_json_protocol: Authentication processed without secret.	6
3.1.2	socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.	6
3.1.3	socket_protocol.pure_json_protocol: Checksum corruption while sending.	6
3.1.4	socket_protocol.pure_json_protocol: Incompatible Callback return value(s).	7
3.1.5	socket_protocol.pure_json_protocol: No Callback at response instance for the request.	7
3.1.6	socket_protocol.pure_json_protocol: Register a Callback which is already defined.	8
3.1.7	socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.	8
3.1.8	socket_protocol.pure_json_protocol: Send and receive check including authentication.	9
3.1.9	socket_protocol.pure_json_protocol: Send and receive check.	9
3.1.10	socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.	10
3.1.11	socket_protocol.pure_json_protocol: Wildcard Callback registration for data_id.	10
3.1.12	socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.	11
3.1.13	socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.	11
3.1.14	socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialisation).	12
3.1.15	socket_protocol.struct_json_protocol: Send and receive check.	13
3.2	Summary for testrun with python 3.6.9 (final)	13
3.2.1	socket_protocol.pure_json_protocol: Authentication processed without secret.	13

3.2.2	<code>socket_protocol.pure_json_protocol</code> : Authentication required, but not processed/correctly processed.	13
3.2.3	<code>socket_protocol.pure_json_protocol</code> : Checksum corruption while sending.	14
3.2.4	<code>socket_protocol.pure_json_protocol</code> : Incompatible Callback return value(s).	14
3.2.5	<code>socket_protocol.pure_json_protocol</code> : No Callback at response instance for the request.	15
3.2.6	<code>socket_protocol.pure_json_protocol</code> : Register a Callback which is already defined.	15
3.2.7	<code>socket_protocol.pure_json_protocol</code> : Register a second Callback with the same <code>service_id</code>	16
3.2.8	<code>socket_protocol.pure_json_protocol</code> : Send and receive check including authentication.	16
3.2.9	<code>socket_protocol.pure_json_protocol</code> : Send and receive check.	17
3.2.10	<code>socket_protocol.pure_json_protocol</code> : Timeout measurement for authentication and send method.	17
3.2.11	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>data_id</code>	18
3.2.12	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>service_id</code> and <code>data_id</code>	18
3.2.13	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>service_id</code>	19
3.2.14	<code>socket_protocol.struct_json_protocol</code> : Send and receive check (Twice for coverage of buffer initialisation).	20
3.2.15	<code>socket_protocol.struct_json_protocol</code> : Send and receive check.	20

A Trace for testrun with python 2.7.17 (final) 22

A.1	Tests with status Info (15)	22
A.1.1	<code>socket_protocol.struct_json_protocol</code> : Send and receive check.	22
A.1.2	<code>socket_protocol.pure_json_protocol</code> : Send and receive check.	23
A.1.3	<code>socket_protocol.pure_json_protocol</code> : Send and receive check including authentication.	25
A.1.4	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>service_id</code> and <code>data_id</code>	28
A.1.5	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>service_id</code>	30
A.1.6	<code>socket_protocol.pure_json_protocol</code> : Wildcard Callback registration for <code>data_id</code>	31
A.1.7	<code>socket_protocol.pure_json_protocol</code> : Register a second Callback with the same <code>service_id</code>	33
A.1.8	<code>socket_protocol.struct_json_protocol</code> : Send and receive check (Twice for coverage of buffer initialisation).	35
A.1.9	<code>socket_protocol.pure_json_protocol</code> : Checksum corruption while sending.	37
A.1.10	<code>socket_protocol.pure_json_protocol</code> : Timeout measurement for authentication and send method.	39
A.1.11	<code>socket_protocol.pure_json_protocol</code> : No Callback at response instance for the request.	40
A.1.12	<code>socket_protocol.pure_json_protocol</code> : Authentication required, but not processed/correctly processed.	41
A.1.13	<code>socket_protocol.pure_json_protocol</code> : Register a Callback which is already defined.	43
A.1.14	<code>socket_protocol.pure_json_protocol</code> : Authentication processed without secret.	44
A.1.15	<code>socket_protocol.pure_json_protocol</code> : Incompatible Callback return value(s).	44

B	Trace for testrun with python 3.6.9 (final)	46
B.1	Tests with status Info (15)	46
B.1.1	socket_protocol.struct_json_protocol: Send and receive check.	46
B.1.2	socket_protocol.pure_json_protocol: Send and receive check.	48
B.1.3	socket_protocol.pure_json_protocol: Send and receive check including authentication.	50
B.1.4	socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.	53
B.1.5	socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.	55
B.1.6	socket_protocol.pure_json_protocol: Wildcard Callback registration for data_id.	56
B.1.7	socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.	58
B.1.8	socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialisation).	60
B.1.9	socket_protocol.pure_json_protocol: Checksum corruption while sending.	62
B.1.10	socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.	64
B.1.11	socket_protocol.pure_json_protocol: No Callback at response instance for the request.	65
B.1.12	socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.	66
B.1.13	socket_protocol.pure_json_protocol: Register a Callback which is already defined.	68
B.1.14	socket_protocol.pure_json_protocol: Authentication processed without secret.	69
B.1.15	socket_protocol.pure_json_protocol: Incompatible Callback return value(s).	69
C	Test-Coverage	71
C.1	socket_protocol	71
C.1.1	socket_protocol.__init__.py	72

1 Test Information

1.1 Test Candidate Information

The Module `socket_protocol` is designed to pack and unpack data for serial transportation. For more Information read the sphinx documentation.

Library Information	
Name	socket_protocol
State	Released
Supported Interpreters	python2, python3
Version	44bfc23658f5a000bcabcf2a34875620

Dependencies	
stringtools	77981dbdc5e8fd54960c4f914c083602

1.2 Unittest Information

Unittest Information	
Version	cd82b7d4eb571a53181f2cb9c7b37417
Testruns with	python 2.7.17 (final), python 3.6.9 (final)

1.3 Test System Information

System Information	
Architecture	64bit
Distribution	LinuxMint 19.3 tricia
Hostname	ahorn
Kernel	5.0.0-37-generic (#40 18.04.1-Ubuntu SMP Thu Nov 14 12:06:39 UTC 2019)
Machine	x86_64
Path	/user_data/data/dirk/prj/modules/socket_protocol/unittest
System	Linux
Username	dirk

2 Statistic

2.1 Test-Statistic for testrun with python 2.7.17 (final)

Number of tests	15
Number of successfull tests	15
Number of possibly failed tests	0
Number of failed tests	0

Executionlevel	Full Test (all defined tests)
Time consumption	11.323s

2.2 Test-Statistic for testrun with python 3.6.9 (final)

Number of tests	15
Number of successfull tests	15
Number of possibly failed tests	0
Number of failed tests	0

Executionlevel	Full Test (all defined tests)
Time consumption	11.309s

2.3 Coverage Statistic

Module- or Filename	Line-Coverage	Branch-Coverage
socket_protocol	100.0%	100.0%
socket_protocol.__init__.py	100.0%	

3 Testcases with no corresponding Requirement

3.1 Summary for testrun with python 2.7.17 (final)

3.1.1 socket_protocol.pure_json_protocol: Authentication processed without secret.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.14!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (44)
Start-Time:	2019-12-27 17:11:10,755
Finished-Time:	2019-12-27 17:11:10,757
Time-Consumption	0.002s

Testsummary:

Info	Authentication with no secret definition (pure_json_protocol).
Success	Return value of authentication is correct (Content False and Type is <type 'bool'>).

3.1.2 socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.12!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (42)
Start-Time:	2019-12-27 17:11:09,341
Finished-Time:	2019-12-27 17:11:10,752
Time-Consumption	1.411s

Testsummary:

Info	Authentication with different secrets for request and response instance (pure_json_protocol).
Success	Return value of authentication is correct (Content False and Type is <type 'bool'>).
Success	Return value of send method is correct (Content True and Type is <type 'bool'>).
Success	Response Status (Authentication required) transfered via pure_json_protocol is correct (Content 2 and Type is <type 'int'>).
Success	Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <type 'NoneType'>).
Success	Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success	Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.1.3 socket_protocol.pure_json_protocol: Checksum corruption while sending.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.9!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (36)
 Start-Time: 2019-12-27 17:11:06,715
 Finished-Time: 2019-12-27 17:11:07,221
 Time-Consumption 0.507s

Testsummary:

Info Send data with wrong checksum by pure_json_protocol.
Success Return value of send method is correct (Content True and Type is <type 'bool'>).
Success Callback executed variable is correct (Content False and Type is <type 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.1.4 socket_protocol.pure_json_protocol: Incompatible Callback return value(s).

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.15!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (45)
 Start-Time: 2019-12-27 17:11:10,757
 Finished-Time: 2019-12-27 17:11:11,167
 Time-Consumption 0.410s

Testsummary:

Info Send and received data with incompatible callback (pure_json_protocol).
Success Exception (TypeError) detection variable is correct (Content True and Type is <type 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success Exception (TypeError) detection variable is correct (Content True and Type is <type 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

3.1.5 socket_protocol.pure_json_protocol: No Callback at response instance for the request.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.11!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (41)
 Start-Time: 2019-12-27 17:11:08,632

Finished-Time: 2019-12-27 17:11:09,341
 Time-Consumption 0.708s

Testsummary:

Info Send data, but no callback registered (pure_json_protocol).
Success Return value of send method is correct (Content True and Type is <type 'bool'>).
Success Response Status (Request has no callback. Data buffered.) transfered via pure_json_protocol is correct (Content 1 and Type is <type 'int'>).
Success Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.1.6 socket_protocol.pure_json_protocol: Register a Callback which is already defined.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.13!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (43)
 Start-Time: 2019-12-27 17:11:10,753
 Finished-Time: 2019-12-27 17:11:10,754
 Time-Consumption 0.002s

Testsummary:

Info Registering two callbacks which overlap for at least one message (pure_json_protocol).
Success Exception (RegistrationError) detection variable is correct (Content True and Type is <type 'bool'>).

3.1.7 socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.7!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (31)
 Start-Time: 2019-12-27 17:11:04,795
 Finished-Time: 2019-12-27 17:11:05,503
 Time-Consumption 0.708s

Testsummary:

Info Send and received data by pure_json_protocol.
Success Return value of send method is correct (Content True and Type is <type 'bool'>).
Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
Success Request Data transfered via pure_json_protocol is correct (Content {'u'test': 'u'test'} and Type is <type 'dict'>).

- Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).
 - Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).
 - Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
 - Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
-

3.1.8 socket_protocol.pure_json_protocol: Send and receive check including authentication.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.3!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (27)
Start-Time:	2019-12-27 17:11:01,252
Finished-Time:	2019-12-27 17:11:02,664
Time-Consumption	1.412s

Testsummary:

- Info** Send and received data by pure_json_protocol.
 - Success** Return value of authentication is correct (Content True and Type is <type 'bool'>).
 - Success** Return value of send method is correct (Content True and Type is <type 'bool'>).
 - Success** Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
 - Success** Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).
 - Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).
 - Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).
 - Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
 - Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
-

3.1.9 socket_protocol.pure_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.2!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (26)
Start-Time:	2019-12-27 17:11:00,544
Finished-Time:	2019-12-27 17:11:01,252
Time-Consumption	0.708s

Testsummary:

Info	Send and received data by pure_json_protocol.
Success	Return value of send method is correct (Content True and Type is <type 'bool'>).
Success	Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
Success	Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).
Success	Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).
Success	Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).
Success	Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success	Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.1.10 socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.10!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (37)
Start-Time:	2019-12-27 17:11:07,222
Finished-Time:	2019-12-27 17:11:08,631
Time-Consumption	1.409s

Testsummary:

Success	Timeout for authentication is correct (Content 0.20129704475402832 in [0.2 ... 0.22000000000000003] and Type is <type 'float'>).
Success	Timeout for authentication is correct (Content 0.5018231868743896 in [0.5 ... 0.55] and Type is <type 'float'>).
Success	Timeout for send method is correct (Content 0.20068693161010742 in [0.2 ... 0.22000000000000003] and Type is <type 'float'>).
Success	Timeout for send method is correct (Content 0.5013918876647949 in [0.5 ... 0.55] and Type is <type 'float'>).

3.1.11 socket_protocol.pure_json_protocol: Wildcard Callback registration for data_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.6!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (30)
Start-Time:	2019-12-27 17:11:04,086
Finished-Time:	2019-12-27 17:11:04,795
Time-Consumption	0.709s

Testsummary:

Info	Send and received data by pure_json_protocol. Wildcard callback registered for .
-------------	--

- Success** Return value of send method is correct (Content True and Type is <type 'bool'>).
 - Success** Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
 - Success** Request Data transfered via pure_json_protocol is correct (Content {'u'test': u'test'} and Type is <type 'dict'>).
 - Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).
 - Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).
 - Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).
 - Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).
-

3.1.12 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.4!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (28)
Start-Time:	2019-12-27 17:11:02,665
Finished-Time:	2019-12-27 17:11:03,374
Time-Consumption	0.709s

Testsummary:

- | | |
|----------------|---|
| Info | Send and received data by pure_json_protocol. Wildcard callback registerd for service_id and data_id. |
| Success | Return value of send method is correct (Content True and Type is <type 'bool'>). |
| Success | Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>). |
| Success | Request Data transfered via pure_json_protocol is correct (Content {'u'test': u'test'} and Type is <type 'dict'>). |
| Success | Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>). |
| Success | Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>). |
| Success | Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>). |
| Success | Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>). |
-

3.1.13 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.5!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (29)

Start-Time: 2019-12-27 17:11:03,374
 Finished-Time: 2019-12-27 17:11:04,085
 Time-Consumption 0.711s

Testsummary:

Info Send and received data by pure_json_protocol. Wildcard callback registerd for service_id.
Success Return value of send method is correct (Content True and Type is <type 'bool'>).
Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'}) and Type is <type 'dict'>).
Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).
Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's']) and Type is <type 'list'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

3.1.14 socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialization).

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.8!

Testrun: python 2.7.17 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (32)
 Start-Time: 2019-12-27 17:11:05,503
 Finished-Time: 2019-12-27 17:11:06,714
 Time-Consumption 1.211s

Testsummary:

Info Send and received data by struct_json_protocol.
Success Return value of send method is correct (Content True and Type is <type 'bool'>).
Success Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <type 'int'>).
Success Request Data transfered via struct_json_protocol is correct (Content {'test': 'test'}) and Type is <type 'dict'>).
Success Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <type 'int'>).
Success Response Data transfered via struct_json_protocol is correct (Content [1, 3, 's']) and Type is <type 'list'>).
Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.1.15 socket_protocol.struct_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.1!

Testrun:	python 2.7.17 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (25)
Start-Time:	2019-12-27 17:10:59,836
Finished-Time:	2019-12-27 17:11:00,543
Time-Consumption	0.707s

Testsummary:

Info	Send and received data by struct_json_protocol.
Success	Return value of send method is correct (Content True and Type is <type 'bool'>).
Success	Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <type 'int'>).
Success	Request Data transfered via struct_json_protocol is correct (Content {u'test': u'test'}) and Type is <type 'dict'>).
Success	Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <type 'int'>).
Success	Response Data transfered via struct_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).
Success	Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).
Success	Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

3.2 Summary for testrun with python 3.6.9 (final)

3.2.1 socket_protocol.pure_json_protocol: Authentication processed without secret.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.14!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (44)
Start-Time:	2019-12-27 17:11:22,529
Finished-Time:	2019-12-27 17:11:22,531
Time-Consumption	0.002s

Testsummary:

Info	Authentication with no secret definition (pure_json_protocol).
Success	Return value of authentication is correct (Content False and Type is <class 'bool'>).

3.2.2 socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.12!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (42)
 Start-Time: 2019-12-27 17:11:21,119
 Finished-Time: 2019-12-27 17:11:22,527
 Time-Consumption 1.408s

Testsummary:

Info Authentication with different secrets for request and response instance (pure_json_protocol).
Success Return value of authentication is correct (Content False and Type is <class 'bool'>).
Success Return value of send method is correct (Content True and Type is <class 'bool'>).
Success Response Status (Authentication required) transfered via pure_json_protocol is correct (Content 2 and Type is <class 'int'>).
Success Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

3.2.3 socket_protocol.pure_json_protocol: Checksum corruption while sending.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.9!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (36)
 Start-Time: 2019-12-27 17:11:18,497
 Finished-Time: 2019-12-27 17:11:19,004
 Time-Consumption 0.506s

Testsummary:

Info Send data with wrong checksum by pure_json_protocol.
Success Return value of send method is correct (Content True and Type is <class 'bool'>).
Success Callback executed variable is correct (Content False and Type is <class 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

3.2.4 socket_protocol.pure_json_protocol: Incompatible Callback return value(s).

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.15!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (45)
 Start-Time: 2019-12-27 17:11:22,531

Finished-Time: 2019-12-27 17:11:22,940
 Time-Consumption 0.410s

Testsummary:

Info Send and received data with incompatible callback (pure_json_protocol).
Success Exception (TypeError) detection variable is correct (Content True and Type is <class 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Exception (TypeError) detection variable is correct (Content True and Type is <class 'bool'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

3.2.5 socket_protocol.pure_json_protocol: No Callback at response instance for the request.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.11!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (41)
 Start-Time: 2019-12-27 17:11:20,413
 Finished-Time: 2019-12-27 17:11:21,119
 Time-Consumption 0.707s

Testsummary:

Info Send data, but no callback registered (pure_json_protocol).
Success Return value of send method is correct (Content True and Type is <class 'bool'>).
Success Response Status (Request has no callback. Data buffered.) transfered via pure_json_protocol is correct (Content 1 and Type is <class 'int'>).
Success Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

3.2.6 socket_protocol.pure_json_protocol: Register a Callback which is already defined.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.13!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (43)
 Start-Time: 2019-12-27 17:11:22,528
 Finished-Time: 2019-12-27 17:11:22,529

Time-Consumption 0.001s

Testsummary:

Info Registering two callbacks which overlap for at least one message (pure_json_protocol).
Success Exception (RegistrationError) detection variable is correct (Content True and Type is <class 'bool'>).

3.2.7 socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.7!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (31)
 Start-Time: 2019-12-27 17:11:16,578
 Finished-Time: 2019-12-27 17:11:17,286
 Time-Consumption 0.708s

Testsummary:

Info Send and received data by pure_json_protocol.
Success Return value of send method is correct (Content True and Type is <class 'bool'>).
Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

3.2.8 socket_protocol.pure_json_protocol: Send and receive check including authentication.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.3!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (27)
 Start-Time: 2019-12-27 17:11:13,042
 Finished-Time: 2019-12-27 17:11:14,453
 Time-Consumption 1.411s

Testsummary:

Info Send and received data by pure_json_protocol.
Success Return value of authentication is correct (Content True and Type is <class 'bool'>).

- Success** Return value of send method is correct (Content True and Type is <class 'bool'>).
 - Success** Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
 - Success** Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
 - Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
 - Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
 - Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
 - Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
-

3.2.9 socket_protocol.pure_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.2!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (26)
Start-Time:	2019-12-27 17:11:12,334
Finished-Time:	2019-12-27 17:11:13,042
Time-Consumption	0.708s

Testsummary:

- Info** Send and received data by pure_json_protocol.
 - Success** Return value of send method is correct (Content True and Type is <class 'bool'>).
 - Success** Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
 - Success** Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
 - Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
 - Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
 - Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
 - Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
-

3.2.10 socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.10!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (37)

Start-Time: 2019-12-27 17:11:19,004
 Finished-Time: 2019-12-27 17:11:20,412
 Time-Consumption 1.408s

Testsummary:

- Success** Timeout for authentication is correct (Content 0.20095229148864746 in [0.2 ... 0.220000000000000003] and Type is <class 'float'>).
- Success** Timeout for authentication is correct (Content 0.5016295909881592 in [0.5 ... 0.55] and Type is <class 'float'>).
- Success** Timeout for send method is correct (Content 0.2006547451019287 in [0.2 ... 0.220000000000000003] and Type is <class 'float'>).
- Success** Timeout for send method is correct (Content 0.5013296604156494 in [0.5 ... 0.55] and Type is <class 'float'>).

3.2.11 socket_protocol.pure_json_protocol: Wildcard Callback registration for data_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.6!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (30)
 Start-Time: 2019-12-27 17:11:15,870
 Finished-Time: 2019-12-27 17:11:16,578
 Time-Consumption 0.708s

Testsummary:

- Info** Send and received data by pure_json_protocol. Wildcard callback registered for .
- Success** Return value of send method is correct (Content True and Type is <class 'bool'>).
- Success** Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
- Success** Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
- Success** Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
- Success** Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
- Success** Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).
- Success** Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

3.2.12 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.4!

Testrun: python 3.6.9 (final)
 Caller: /user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (28)
 Start-Time: 2019-12-27 17:11:14,453

Finished-Time: 2019-12-27 17:11:15,161
 Time-Consumption 0.708s

Testsummary:

Info	Send and received data by pure_json_protocol. Wildcard callback registered for service_id and data_id.
Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
Success	Request Status (Okay) transferred via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
Success	Request Data transferred via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
Success	Response Status (Operation not permitted) transferred via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
Success	Response Data transferred via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
Success	Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).
Success	Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

3.2.13 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.5!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/_init_.py (29)
Start-Time:	2019-12-27 17:11:15,162
Finished-Time:	2019-12-27 17:11:15,869
Time-Consumption	0.708s

Testsummary:

Info	Send and received data by pure_json_protocol. Wildcard callback registered for service_id.
Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
Success	Request Status (Okay) transferred via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
Success	Request Data transferred via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
Success	Response Status (Operation not permitted) transferred via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).
Success	Response Data transferred via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
Success	Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).
Success	Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

3.2.14 socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialisation).

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.8!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/...init...py (32)
Start-Time:	2019-12-27 17:11:17,287
Finished-Time:	2019-12-27 17:11:18,497
Time-Consumption	1.210s

Testsummary:

Info	Send and received data by struct_json_protocol.
Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
Success	Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <class 'int'>).
Success	Request Data transfered via struct_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
Success	Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <class 'int'>).
Success	Response Data transfered via struct_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).
Success	Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).
Success	Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

3.2.15 socket_protocol.struct_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**. See also full trace in section B.1.1!

Testrun:	python 3.6.9 (final)
Caller:	/user_data/data/dirk/prj/modules/socket_protocol/unittest/src/tests/...init...py (25)
Start-Time:	2019-12-27 17:11:11,626
Finished-Time:	2019-12-27 17:11:12,333
Time-Consumption	0.707s

Testsummary:

Info	Send and received data by struct_json_protocol.
Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
Success	Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <class 'int'>).
Success	Request Data transfered via struct_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).
Success	Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <class 'int'>).
Success	Response Data transfered via struct_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

Unittest for socket_protocol

Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

A Trace for testrun with python 2.7.17 (final)

A.1 Tests with status Info (15)

A.1.1 socket_protocol.struct_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by struct_json_protocol.

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
Send data: (29): 00 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↔ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↔ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method
```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```
Result (Return value of send method): True (<type 'bool'>)
```

```
Expectation (Return value of send method): result = True (<type 'bool'>)
```

Success Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <type 'int'>).

```
Result (Request Status (Okay) transfered via struct_json_protocol): 0 (<type 'int'>)
```

```
Expectation (Request Status (Okay) transfered via struct_json_protocol): result = 0 (<type
↔ 'int'>)
```

Success Request Data transfered via struct_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

Result (Request Data transfered via struct_json_protocol): { u'test': u'test' } (<type 'dict'>)

Expectation (Request Data transfered via struct_json_protocol): result = { u'test': u'test' }
 ↪ (<type 'dict'>)

Success Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <type 'int'>).

Result (Response Status (Operation not permitted) transfered via struct_json_protocol): 5
 ↪ (<type 'int'>)

Expectation (Response Status (Operation not permitted) transfered via struct_json_protocol):
 ↪ result = 5 (<type 'int'>)

Success Response Data transfered via struct_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

Result (Response Data transfered via struct_json_protocol): [1, 3, u's'] (<type 'list'>)

Expectation (Response Data transfered via struct_json_protocol): result = [1, 3, u's']
 ↪ (<type 'list'>)

Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for struct_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for struct_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for struct_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for struct_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

A.1.2 socket_protocol.pure_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.


```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
Send data: (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 85 0b b7 34
Receive data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 85 0b b7 34
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```

Result (Return value of send method): True (<type 'bool'>)
Expectation (Return value of send method): result = True (<type 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' }
↪ (<type 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↪ (<type 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↪ result = 5 (<type 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, u's' ] (<type 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, u's' ] (<type  
↪ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): None (<type 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): None (<type 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↪ data_id 0xaffe): result = None (<type 'NoneType'>)
```

A.1.3 socket_protocol.pure_json_protocol: Send and receive check including authentication.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.

Unittest for socket_protocol

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↳ 20 30 7d 2c 2d 2e 5d
Receive data (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↳ 20 30 7d 2c 2d 2e 5d
SJP: RX <- status: 0, service_id: 1, data_id: 0, data: "None"
SJP: Executing callback __authenticate_create_seed__ to process received data
SJP: Got seed request, sending seed for authentication
SJP: TX -> status: 0, service_id: 2, data_id: 0, data:
↳ "'ad050df2d8e7bde79418fe785c0475de33791e0f9c9159474215506f18832e28'"
Send data: (124): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 32 2c 20 22 64 61 74 61 22 3a 20 22 61 64 30 35 30 64 66 32 64 38 65 37 62 64 65
↳ 37 39 34 31 38 66 65 37 38 35 63 30 34 37 35 64 65 33 33 37 39 31 65 30 66 39 63 39 31 35
↳ 39 34 37 34 32 31 35 35 30 36 66 31 38 38 33 32 65 32 38 22 2c 20 22 64 61 74 61 5f 69 64
↳ 22 3a 20 30 7d d4 1c fc c9
Receive data (124): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 32 2c 20 22 64 61 74 61 22 3a 20 22 61 64 30 35 30 64 66 32 64 38 65 37 62 64
↳ 65 37 39 34 31 38 66 65 37 38 35 63 30 34 37 35 64 65 33 33 37 39 31 65 30 66 39 63 39 31
↳ 35 39 34 37 34 32 31 35 35 30 36 66 31 38 38 33 32 65 32 38 22 2c 20 22 64 61 74 61 5f 69
↳ 64 22 3a 20 30 7d d4 1c fc c9
SJP: RX <- status: 0, service_id: 2, data_id: 0, data:
↳ "u'ad050df2d8e7bde79418fe785c0475de33791e0f9c9159474215506f18832e28'"
SJP: Executing callback __authenticate_create_key__ to process received data
SJP: Got seed, sending key for authentication
SJP: TX -> status: 0, service_id: 3, data_id: 0, data:
↳ "'34c6bf8d3d3ef7a8ffa33cda3905e65e3dbe476e96cad2fb1597e5eaa9d65606e2a8c74ec11df267df68193_
↳ ee77e79e948ab6e18b53f2e259a38a5b9a694a227'"
Send data: (188): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 33 2c 20 22 64 61 74 61 22 3a 20 22 33 34 63 36 62 66 38 64 33 64 33 65 66 37 61
↳ 38 66 66 61 33 33 63 64 61 33 39 30 35 65 36 35 65 33 64 62 65 34 37 36 65 39 36 63 61 64
↳ 32 66 62 31 35 39 37 65 35 65 61 61 39 64 36 35 36 30 36 65 32 61 38 63 37 34 65 63 31 31
↳ 64 66 32 36 37 64 66 36 38 31 39 33 65 65 37 37 65 37 39 65 39 34 38 61 62 36 65 31 38 62
↳ 35 33 66 32 65 32 35 39 61 33 38 61 35 62 39 61 36 39 34 61 32 32 37 22 2c 20 22 64 61 74
↳ 61 5f 69 64 22 3a 20 30 7d f4 b0 ce 74
Receive data (188): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 33 2c 20 22 64 61 74 61 22 3a 20 22 33 34 63 36 62 66 38 64 33 64 33 65 66 37
↳ 61 38 66 66 61 33 33 63 64 61 33 39 30 35 65 36 35 65 33 64 62 65 34 37 36 65 39 36 63 61
↳ 64 32 66 62 31 35 39 37 65 35 65 61 61 39 64 36 35 36 30 36 65 32 61 38 63 37 34 65 63 31
↳ 31 64 66 32 36 37 64 66 36 38 31 39 33 65 65 37 37 65 37 39 65 39 34 38 61 62 36 65 31 38
↳ 62 35 33 66 32 65 32 35 39 61 33 38 61 35 62 39 61 36 39 34 61 32 32 37 22 2c 20 22 64 61
↳ 74 61 5f 69 64 22 3a 20 30 7d f4 b0 ce 74
SJP: RX <- status: 0, service_id: 3, data_id: 0, data:
↳ "u'34c6bf8d3d3ef7a8ffa33cda3905e65e3dbe476e96cad2fb1597e5eaa9d65606e2a8c74ec11df267df6819
↳ 3ee77e79e948ab6e18b53f2e259a38a5b9a694a227'"
SJP: Executing callback authenticate check key to process received data
```

Result (Return value of authentication): True (<type 'bool'>)

Expectation (Return value of authentication): result = True (<type 'bool'>)

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

Result (Return value of send method): True (<type 'bool'>)

Expectation (Return value of send method): result = True (<type 'bool'>)

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)

Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type 'int'>)

Success Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)

Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' } (<type 'dict'>)

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5 (<type 'int'>)

Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol): result = 5 (<type 'int'>)

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

Result (Response Data transfered via pure_json_protocol): [1, 3, u's'] (<type 'list'>)

Expectation (Response Data transfered via pure_json_protocol): result = [1, 3, u's'] (<type 'list'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): result = None (<type 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

A.1.4 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol. Wildcard callback registered for service_id and data_id.

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"

Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59

Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59

SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"

SJP: Executing callback response_data_method to process received data

SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's]"

Send data: (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
 ↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49

Receive data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
 ↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49

SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's]"

SJP: Received message has a peculiar status: Operation not permitted

SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

Result (Return value of send method): True (<type 'bool'>)

Expectation (Return value of send method): result = True (<type 'bool'>)

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)

Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type 'int'>)
 ↪ 'int'>)

Success Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)

Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' }
 ↪ (<type 'dict'>)

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5
 ↪ (<type 'int'>)

Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):
 ↪ result = 5 (<type 'int'>)

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

Result (Response Data transfered via pure_json_protocol): [1, 3, u's'] (<type 'list'>)

Expectation (Response Data transfered via pure_json_protocol): result = [1, 3, u's'] (<type 'list'>)
 ↪ 'list'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): result = None (<type 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xbeef): result = None (<type 'NoneType'>)

A.1.5 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.

Testresult

This test was passed with the state: **Success**.

Info	Send and received data by pure_json_protocol. Wildcard callback registerd for service_id.
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"	
Send data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59	
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59	
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"	
SJP: Executing callback response_data_method to process received data	
SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's']"	
Send data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64 ↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49	
Receive data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64 ↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49	
SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's']"	
SJP: Received message has a peculiar status: Operation not permitted	
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method	
Success	Return value of send method is correct (Content True and Type is <type 'bool'>).
Result (Return value of send method): True (<type 'bool'>)	
Expectation (Return value of send method): result = True (<type 'bool'>)	
Success	Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).
Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)	
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type 'int'>)	
Success	Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

```
Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' }
↳ (<type 'dict'>)
```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5
↳ (<type 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):
↳ result = 5 (<type 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, u's' ] (<type 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, u's' ] (<type
↳ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<type 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<type 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↳ data_id 0xbeef): result = None (<type 'NoneType'>)
```

A.1.6 socket_protocol.pure_json_protocol: Wildcard Callback registration for data.id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol. Wildcard callback registered for .

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"
Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d e8 e0 82 59
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's']"
Send data: (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49
Receive data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 7d 0e 05 f1 49
SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```

Result (Return value of send method): True (<type 'bool'>)
Expectation (Return value of send method): result = True (<type 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' }
↪ (<type 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↪ (<type 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↪ result = 5 (<type 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, u's' ] (<type 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, u's' ] (<type  
↪ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): None (<type 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): None (<type 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↪ data_id 0xbeef): result = None (<type 'NoneType'>)
```

A.1.7 socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method_2 to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
Send data: (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 85 0b b7 34
Receive data (74): 7b 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 2c 20 22 64
↪ 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 85 0b b7 34
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```

Result (Return value of send method): True (<type 'bool'>)
Expectation (Return value of send method): result = True (<type 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <type 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<type 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<type
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { u'test': u'test' } (<type 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { u'test': u'test' }
↪ (<type 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <type 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↳ (<type 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↳ result = 5 (<type 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, u's' ] (<type 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, u's' ] (<type  
↳ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): None (<type 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): None (<type 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↳ data_id 0xaffe): result = None (<type 'NoneType'>)
```

A.1.8 socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialisation).

Testresult

This test was passed with the state: **Success**.

Info Send and received data by struct_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
Send data: (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
Send data: (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, u's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```
Result (Return value of send method): True (<type 'bool'>)
```

```
Expectation (Return value of send method): result = True (<type 'bool'>)
```

Success Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <type 'int'>).

```
Result (Request Status (Okay) transfered via struct_json_protocol): 0 (<type 'int'>)
```

```
Expectation (Request Status (Okay) transfered via struct_json_protocol): result = 0 (<type
↳ 'int'>)
```

Success Request Data transfered via struct_json_protocol is correct (Content {u'test': u'test'} and Type is <type 'dict'>).

Result (Request Data transfered via struct_json_protocol): { u'test': u'test' } (<type 'dict'>)

Expectation (Request Data transfered via struct_json_protocol): result = { u'test': u'test' }
 ↪ (<type 'dict'>)

Success Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <type 'int'>).

Result (Response Status (Operation not permitted) transfered via struct_json_protocol): 5
 ↪ (<type 'int'>)

Expectation (Response Status (Operation not permitted) transfered via struct_json_protocol):
 ↪ result = 5 (<type 'int'>)

Success Response Data transfered via struct_json_protocol is correct (Content [1, 3, u's'] and Type is <type 'list'>).

Result (Response Data transfered via struct_json_protocol): [1, 3, u's'] (<type 'list'>)

Expectation (Response Data transfered via struct_json_protocol): result = [1, 3, u's']
 ↪ (<type 'list'>)

Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for struct_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for struct_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for struct_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for struct_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

A.1.9 socket_protocol.pure_json_protocol: Checksum corruption while sending.

Testresult

This test was passed with the state: **Success**.

Info Send data with wrong checksum by pure_json_protocol.

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"u'test': u'test'}"
Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22
↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 25
SJP: Received message has a wrong checksum and will be ignored: (79): 7b 22 73 74 61 74 75 73
↪ 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 64 61 74 61 22
↪ 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a
↪ 20 34 35 30 35 34 7d 63 ee c4 25.
```

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

```
Result (Return value of send method): True (<type 'bool'>)
Expectation (Return value of send method): result = True (<type 'bool'>)
```

Success Callback executed variable is correct (Content False and Type is <type 'bool'>).

```
Result (Callback executed variable): False (<type 'bool'>)
Expectation (Callback executed variable): result = False (<type 'bool'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): None (<type 'NoneType'>)
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): None (<type 'NoneType'>)
Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↪ data_id 0xaffe): result = None (<type 'NoneType'>)
```

A.1.10 socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.

Testresult

This test was passed with the state: **Success**.

Success Timeout for authentication is correct (Content 0.20129704475402832 in [0.2 ... 0.220000000000000003] and Type is <type 'float'>).

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↪ 20 30 7d 2c 2d 2e 5d
Result (Timeout for authentication): 0.20129704475402832 (<type 'float'>)
Expectation (Timeout for authentication): 0.2 <= result <= 0.220000000000000003
```

Success Timeout for authentication is correct (Content 0.5018231868743896 in [0.5 ... 0.55] and Type is <type 'float'>).

```
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↪ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↪ 20 30 7d 2c 2d 2e 5d
Result (Timeout for authentication): 0.5018231868743896 (<type 'float'>)
Expectation (Timeout for authentication): 0.5 <= result <= 0.55
```

Success Timeout for send method is correct (Content 0.20068693161010742 in [0.2 ... 0.220000000000000003] and Type is <type 'float'>).

```
SJP: TIMEOUT (0.2s): Requested data (service_id: 30; data_id: 0) not in buffer.
Result (Timeout for send method): 0.20068693161010742 (<type 'float'>)
Expectation (Timeout for send method): 0.2 <= result <= 0.220000000000000003
```

Success Timeout for send method is correct (Content 0.5013918876647949 in [0.5 ... 0.55] and Type is <type 'float'>).

```
SJP: TIMEOUT (0.5s): Requested data (service_id: 30; data_id: 0) not in buffer.
Result (Timeout for send method): 0.5013918876647949 (<type 'float'>)
Expectation (Timeout for send method): 0.5 <= result <= 0.55
```

A.1.11 socket_protocol.pure_json_protocol: No Callback at response instance for the request.

Testresult

This test was passed with the state: **Success**.

Info	Send data, but no callback registered (pure_json_protocol).
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"	
Send data: (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24	
Receive data (79): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 ↪ 7d 2c 20 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 7d 63 ee c4 24	
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "{u'test': u'test'}"	
SJP: Received message with no registered callback. Sending negative response.	
SJP: TX -> status: 1, service_id: 11, data_id: 45054, data: "None"	
Send data: (67): 7b 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 ↪ 3a 20 34 35 30 35 34 7d b1 70 10 64	
Receive data (67): 7b 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 73 65 72 76 69 63 65 5f 69 64 ↪ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 ↪ 3a 20 34 35 30 35 34 7d b1 70 10 64	
SJP: RX <- status: 1, service_id: 11, data_id: 45054, data: "None"	
SJP: Received message has a peculiar status: Request has no callback. Data buffered.	
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method	
Success	Return value of send method is correct (Content True and Type is <type 'bool'>).
Result (Return value of send method): True (<type 'bool'>)	
Expectation (Return value of send method): result = True (<type 'bool'>)	
Success	Response Status (Request has no callback. Data buffered.) transfered via pure_json_protocol is correct (Content 1 and Type is <type 'int'>).
Result (Response Status (Request has no callback. Data buffered.) transfered via ↪ pure_json_protocol): 1 (<type 'int'>)	
Expectation (Response Status (Request has no callback. Data buffered.) transfered via ↪ pure_json_protocol): result = 1 (<type 'int'>)	
Success	Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <type 'NoneType'>).

Result (Response Data (no data) transfered via pure_json_protocol): None (<type 'NoneType'>)

Expectation (Response Data (no data) transfered via pure_json_protocol): result = None (<type 'NoneType'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): result = None (<type 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe): result = None (<type 'NoneType'>)

A.1.12 socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.

Testresult

This test was passed with the state: **Success**.

Info Authentication with different secrets for request and response instance (pure_json_protocol).

Unittest for socket_protocol

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↳ 20 30 7d 2c 2d 2e 5d
Receive data (62): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22 3a
↳ 20 30 7d 2c 2d 2e 5d
SJP: RX <- status: 0, service_id: 1, data_id: 0, data: "None"
SJP: Executing callback __authenticate_create_seed__ to process received data
SJP: Got seed request, sending seed for authentication
SJP: TX -> status: 0, service_id: 2, data_id: 0, data:
↳ "'d383fc2f52e7de35b3ea80303a8677f3c5457f274bfca7a6c67c0c012b8d9fa4'"
Send data: (124): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 32 2c 20 22 64 61 74 61 22 3a 20 22 64 33 38 33 66 63 32 66 35 32 65 37 64 65 33
↳ 35 62 33 65 61 38 30 33 30 33 61 38 36 37 37 66 33 63 35 34 35 37 66 32 37 34 62 66 63 61
↳ 37 61 36 63 36 37 63 30 63 30 31 32 62 38 64 39 66 61 34 22 2c 20 22 64 61 74 61 5f 69 64
↳ 22 3a 20 30 7d ce 59 18 b4
Receive data (124): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 32 2c 20 22 64 61 74 61 22 3a 20 22 64 33 38 33 66 63 32 66 35 32 65 37 64 65
↳ 33 35 62 33 65 61 38 30 33 30 33 61 38 36 37 37 66 33 63 35 34 35 37 66 32 37 34 62 66 63
↳ 61 37 61 36 63 36 37 63 30 63 30 31 32 62 38 64 39 66 61 34 22 2c 20 22 64 61 74 61 5f 69
↳ 64 22 3a 20 30 7d ce 59 18 b4
SJP: RX <- status: 0, service_id: 2, data_id: 0, data:
↳ "'u'd383fc2f52e7de35b3ea80303a8677f3c5457f274bfca7a6c67c0c012b8d9fa4'"
SJP: Executing callback __authenticate_create_key__ to process received data
SJP: Got seed, sending key for authentication
SJP: TX -> status: 0, service_id: 3, data_id: 0, data:
↳ "'ae191cbab2cd7606a1055763310a33cc792f5b31b8d0a8710b437f5ba6eb781b435fcf4a2961d5cd048daf5_
↳ 2f1d359361cca53dc227b46a83150e86753235aab'"
Send data: (188): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 33 2c 20 22 64 61 74 61 22 3a 20 22 61 65 31 39 31 63 62 61 62 32 63 64 37 36 30
↳ 36 61 31 30 35 35 37 36 33 33 31 30 61 33 33 63 63 37 39 32 66 35 62 33 31 62 38 64 30 61
↳ 38 37 31 30 62 34 33 37 66 35 62 61 36 65 62 37 38 31 62 34 33 35 66 63 66 34 61 32 39 36
↳ 31 64 35 63 64 30 34 38 64 61 66 35 32 66 31 64 33 35 39 33 36 31 63 63 61 35 33 64 63 32
↳ 32 37 62 34 36 61 38 33 31 35 30 65 38 36 37 35 33 32 33 35 61 61 62 22 2c 20 22 64 61 74
↳ 61 5f 69 64 22 3a 20 30 7d f2 86 8f b8
Receive data (188): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 33 2c 20 22 64 61 74 61 22 3a 20 22 61 65 31 39 31 63 62 61 62 32 63 64 37 36
↳ 30 36 61 31 30 35 35 37 36 33 33 31 30 61 33 33 63 63 37 39 32 66 35 62 33 31 62 38 64 30
↳ 61 38 37 31 30 62 34 33 37 66 35 62 61 36 65 62 37 38 31 62 34 33 35 66 63 66 34 61 32 39
↳ 36 31 64 35 63 64 30 34 38 64 61 66 35 32 66 31 64 33 35 39 33 36 31 63 63 61 35 33 64 63
↳ 32 32 37 62 34 36 61 38 33 31 35 30 65 38 36 37 35 33 32 33 35 61 61 62 22 2c 20 22 64 61
↳ 74 61 5f 69 64 22 3a 20 30 7d f2 86 8f b8
SJP: RX <- status: 0, service_id: 3, data_id: 0, data:
↳ "'u'ae191cbab2cd7606a1055763310a33cc792f5b31b8d0a8710b437f5ba6eb781b435fcf4a2961d5cd048daf
↳ 52f1d359361cca53dc227b46a83150e86753235aab'"
SJP: Executing callback authenticate check key to process received data
```

Result (Return value of authentication): False (<type 'bool'>)

Expectation (Return value of authentication): result = False (<type 'bool'>)

Success Return value of send method is correct (Content True and Type is <type 'bool'>).

Result (Return value of send method): True (<type 'bool'>)

Expectation (Return value of send method): result = True (<type 'bool'>)

Success Response Status (Authentication required) transfered via pure_json_protocol is correct (Content 2 and Type is <type 'int'>).

Result (Response Status (Authentication required) transfered via pure_json_protocol): 2
 ↪ (<type 'int'>)

Expectation (Response Status (Authentication required) transfered via pure_json_protocol):
 ↪ result = 2 (<type 'int'>)

Success Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <type 'NoneType'>).

Result (Response Data (no data) transfered via pure_json_protocol): None (<type 'NoneType'>)

Expectation (Response Data (no data) transfered via pure_json_protocol): result = None (<type
 ↪ 'NoneType'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): result = None (<type 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<type 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<type 'NoneType'>)

A.1.13 socket_protocol.pure_json_protocol: Register a Callback which is already defined.

Testresult

This test was passed with the state: **Success**.

Info Registering two callbacks which overlap for at least one message (pure_json_protocol).

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

Success Exception (RegistrationError) detection variable is correct (Content True and Type is <type 'bool'>).

Result (Exception (RegistrationError) detection variable): True (<type 'bool'>)

Expectation (Exception (RegistrationError) detection variable): result = True (<type 'bool'>)

A.1.14 socket_protocol.pure_json_protocol: Authentication processed without secret.

Testresult

This test was passed with the state: **Success**.

Info Authentication with no secret definition (pure_json_protocol).

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

Success Return value of authentication is correct (Content False and Type is <type 'bool'>).

Result (Return value of authentication): False (<type 'bool'>)

Expectation (Return value of authentication): result = False (<type 'bool'>)

A.1.15 socket_protocol.pure_json_protocol: Incompatible Callback return value(s).

Testresult

This test was passed with the state: **Success**.

Info Send and received data with incompatible callback (pure_json_protocol).

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "None"
Send data: (67): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 35 30 35 34 7d fc 3e bd 5f
Receive data (67): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 35 30 35 34 7d fc 3e bd 5f
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "None"
SJP: Executing callback response_data_method_fail to process received data
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "None"
Send data: (67): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 38 38 37 39 7d 77 30 fb 22
Receive data (67): 7b 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 30 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 38 38 37 39 7d 77 30 fb 22
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "None"
SJP: Received message with no registered callback. Sending negative response.
SJP: TX -> status: 1, service_id: 11, data_id: 48879, data: "None"
Send data: (67): 7b 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 38 38 37 39 7d 3a 7e 56 19
Receive data (67): 7b 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 73 65 72 76 69 63 65 5f 69 64
↳ 22 3a 20 31 31 2c 20 22 64 61 74 61 22 3a 20 6e 75 6c 6c 2c 20 22 64 61 74 61 5f 69 64 22
↳ 3a 20 34 38 38 37 39 7d 3a 7e 56 19
SJP: RX <- status: 1, service_id: 11, data_id: 48879, data: "None"
SJP: Received message has a peculiar status: Request has no callback. Data buffered.
SJP: Executing callback response_data_method_fail to process received data

```

Success Exception (TypeError) detection variable is correct (Content True and Type is <type 'bool'>).

Result (Exception (TypeError) detection variable): True (<type 'bool'>)

Expectation (Exception (TypeError) detection variable): result = True (<type 'bool'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xaffe): None (<type 'NoneType'>)
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xaffe): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↳ 0xaffe): None (<type 'NoneType'>)
Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↳ data_id 0xaffe): result = None (<type 'NoneType'>)
```

Success Exception (TypeError) detection variable is correct (Content True and Type is <type 'bool'>).

```
Result (Exception (TypeError) detection variable): True (<type 'bool'>)
Expectation (Exception (TypeError) detection variable): result = True (<type 'bool'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<type 'NoneType'>)
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): result = None (<type 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <type 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<type 'NoneType'>)
Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↳ data_id 0xbeef): result = None (<type 'NoneType'>)
```

B Trace for testrun with python 3.6.9 (final)

B.1 Tests with status Info (15)

B.1.1 socket_protocol.struct_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by struct_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (29): 00 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↳ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

```

Result (Return value of send method): True (<class 'bool'>)
Expectation (Return value of send method): result = True (<class 'bool'>)

```

Success Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <class 'int'>).

```

Result (Request Status (Okay) transfered via struct_json_protocol): 0 (<class 'int'>)
Expectation (Request Status (Okay) transfered via struct_json_protocol): result = 0 (<class
↳ 'int'>)

```

Success Request Data transfered via struct_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

```

Result (Request Data transfered via struct_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via struct_json_protocol): result = { 'test': 'test' }
↳ (<class 'dict'>)

```

Success Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <class 'int'>).

```

Result (Response Status (Operation not permitted) transfered via struct_json_protocol): 5
↳ (<class 'int'>)
Expectation (Response Status (Operation not permitted) transfered via struct_json_protocol):
↳ result = 5 (<class 'int'>)

```

Success Response Data transfered via struct_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

Result (Response Data transfered via struct_json_protocol): [1, 3, 's'] (<class 'list'>)

Expectation (Response Data transfered via struct_json_protocol): result = [1, 3, 's']
↪ (<class 'list'>)

Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for struct_json_protocol.receive with data data_id
↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for struct_json_protocol.receive with data
↪ data_id 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for struct_json_protocol.receive with data data_id
↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for struct_json_protocol.receive with data
↪ data_id 0xaffe): result = None (<class 'NoneType'>)

B.1.2 socket_protocol.pure_json_protocol: Send and receive check.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
Send data: (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 60 f8 dc 89
Receive data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 60 f8 dc 89
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

```

Result (Return value of send method): True (<class 'bool'>)
Expectation (Return value of send method): result = True (<class 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' }
↪ (<class 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↪ (<class 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↪ result = 5 (<class 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, 's' ] (<class 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, 's' ] (<class  
↪ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): result = None (<class 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↪ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↪ data_id 0xaffe): result = None (<class 'NoneType'>)
```

B.1.3 socket_protocol.pure_json_protocol: Send and receive check including authentication.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.

Unittest for socket_protocol

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↳ 6c 6c 7d 9e 85 7b 8d
Receive data (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↳ 6c 6c 7d 9e 85 7b 8d
SJP: RX <- status: 0, service_id: 1, data_id: 0, data: "None"
SJP: Executing callback __authenticate_create_seed__ to process received data
SJP: Got seed request, sending seed for authentication
SJP: TX -> status: 0, service_id: 2, data_id: 0, data:
↳ "'7dcdf29db95b5992d9325321e9f43670d90b97d8a3f27d2b2fa20af2c78f2cb2'"
Send data: (124): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 32 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22 37
↳ 64 63 64 66 32 39 64 62 39 35 62 35 39 39 32 64 39 33 32 35 33 32 31 65 39 66 34 33 36 37
↳ 30 64 39 30 62 39 37 64 38 61 33 66 32 37 64 32 62 32 66 61 32 30 61 66 32 63 37 38 66 32
↳ 63 62 32 22 7d d2 3c f2 d5
Receive data (124): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f
↳ 69 64 22 3a 20 32 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22
↳ 37 64 63 64 66 32 39 64 62 39 35 62 35 39 39 32 64 39 33 32 35 33 32 31 65 39 66 34 33 36
↳ 37 30 64 39 30 62 39 37 64 38 61 33 66 32 37 64 32 62 32 66 61 32 30 61 66 32 63 37 38 66
↳ 32 63 62 32 22 7d d2 3c f2 d5
SJP: RX <- status: 0, service_id: 2, data_id: 0, data:
↳ "'7dcdf29db95b5992d9325321e9f43670d90b97d8a3f27d2b2fa20af2c78f2cb2'"
SJP: Executing callback __authenticate_create_key__ to process received data
SJP: Got seed, sending key for authentication
SJP: TX -> status: 0, service_id: 3, data_id: 0, data:
↳ "'7fbd38486012f39f9ec215124f6c47655586cd4a0a24636dfd04dbcb2a449c17f48921bea79711e3f07bc1b
↳ 6390e9f12bc5a4708f193e40a29d64aceb0d5b10e'"
Send data: (188): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 33 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22 37
↳ 66 62 64 33 38 34 38 36 30 31 32 66 33 39 66 39 65 63 32 31 35 31 32 34 66 36 63 34 37 36
↳ 35 35 35 38 36 63 64 34 61 30 61 32 34 36 33 36 64 66 64 30 34 64 62 63 62 32 61 34 34 39
↳ 63 31 37 66 34 38 39 32 31 62 65 61 37 39 37 31 31 65 33 66 30 37 62 63 31 62 36 33 39 30
↳ 65 39 66 31 32 62 63 35 61 34 37 30 38 66 31 39 33 65 34 30 61 32 39 64 36 34 61 63 65 62
↳ 30 64 35 62 31 30 65 22 7d e7 65 12 d8
Receive data (188): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f
↳ 69 64 22 3a 20 33 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22
↳ 37 66 62 64 33 38 34 38 36 30 31 32 66 33 39 66 39 65 63 32 31 35 31 32 34 66 36 63 34 37
↳ 36 35 35 35 38 36 63 64 34 61 30 61 32 34 36 33 36 64 66 64 30 34 64 62 63 62 32 61 34 34
↳ 39 63 31 37 66 34 38 39 32 31 62 65 61 37 39 37 31 31 65 33 66 30 37 62 63 31 62 36 33 39
↳ 30 65 39 66 31 32 62 63 35 61 34 37 30 38 66 31 39 33 65 34 30 61 32 39 64 36 34 61 63 65
↳ 62 30 64 35 62 31 30 65 22 7d e7 65 12 d8
SJP: RX <- status: 0, service_id: 3, data_id: 0, data:
↳ "'7fbd38486012f39f9ec215124f6c47655586cd4a0a24636dfd04dbcb2a449c17f48921bea79711e3f07bc1b
↳ 6390e9f12bc5a4708f193e40a29d64aceb0d5b10e'"
SJP: Executing callback authenticate check key to process received data
```

Result (Return value of authentication): True (<class 'bool'>)

Expectation (Return value of authentication): result = True (<class 'bool'>)

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

Result (Return value of send method): True (<class 'bool'>)

Expectation (Return value of send method): result = True (<class 'bool'>)

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)

Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class 'int'>)

Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)

Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' } (<class 'dict'>)

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5 (<class 'int'>)

Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol): result = 5 (<class 'int'>)

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

Result (Response Data transfered via pure_json_protocol): [1, 3, 's'] (<class 'list'>)

Expectation (Response Data transfered via pure_json_protocol): result = [1, 3, 's'] (<class 'list'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ Oxaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id Oxaffe): result = None (<class 'NoneType'>)

B.1.4 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id and data_id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol. Wildcard callback registered for service_id and data_id.

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "{\"test\": 'test'}"

Send data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6

Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6

SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "{\"test\": 'test'}"

SJP: Executing callback response_data_method to process received data

SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"

Send data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
 ↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b

Receive data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
 ↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b

SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"

SJP: Received message has a peculiar status: Operation not permitted

SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

Result (Return value of send method): True (<class 'bool'>)

Expectation (Return value of send method): result = True (<class 'bool'>)

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)

Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class 'int'>)

Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)

Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' } (<class 'dict'>)

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5 (<class 'int'>)

Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol): result = 5 (<class 'int'>)

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

Result (Response Data transfered via pure_json_protocol): [1, 3, 's'] (<class 'list'>)

Expectation (Response Data transfered via pure_json_protocol): result = [1, 3, 's'] (<class 'list'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef): result = None (<class 'NoneType'>)

B.1.5 socket_protocol.pure_json_protocol: Wildcard Callback registration for service_id.

Testresult

This test was passed with the state: **Success.**

Info	Send and received data by pure_json_protocol. Wildcard callback registerd for service_id.
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: Cleaning up receive-buffer	
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT	
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: '{"test': 'test'}"	
Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6	
Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6	
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: '{"test': 'test'}"	
SJP: Executing callback response_data_method to process received data	
SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"	
Send data: (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69 ↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61 ↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b	
Receive data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69 ↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61 ↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b	
SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"	
SJP: Received message has a peculiar status: Operation not permitted	
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method	
Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
Result (Return value of send method): True (<class 'bool'>)	
Expectation (Return value of send method): result = True (<class 'bool'>)	
Success	Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).
Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)	
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class 'int'>)	
Success	Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).


```
Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' }
↳ (<class 'dict'>)
```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5
↳ (<class 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):
↳ result = 5 (<class 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, 's' ] (<class 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, 's' ] (<class
↳ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<class 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): result = None (<class 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↳ 0xbeef): None (<class 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↳ data_id 0xbeef): result = None (<class 'NoneType'>)
```

B.1.6 socket_protocol.pure_json_protocol: Wildcard Callback registration for data.id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol. Wildcard callback registered for .

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: '{"test': 'test'}"
Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6
Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d cc a2 b9 e6
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: '{"test': 'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"
Send data: (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b
Receive data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 7d 1e 6e 1b
SJP: RX <- status: 5, service_id: 11, data_id: 48879, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

```

Result (Return value of send method): True (<class 'bool'>)
Expectation (Return value of send method): result = True (<class 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' }
↪ (<class 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↪ (<class 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↪ result = 5 (<class 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, 's' ] (<class 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, 's' ] (<class  
↪ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): None (<class 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): result = None (<class 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↪ 0xbeef): None (<class 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↪ data_id 0xbeef): result = None (<class 'NoneType'>)
```

B.1.7 socket_protocol.pure_json_protocol: Register a second Callback with the same service_id.

Testresult

This test was passed with the state: **Success**.

Info Send and received data by pure_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Executing callback response_data_method_2 to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
Send data: (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 60 f8 dc 89
Receive data (74): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 35 2c 20 22 64 61 74 61
↪ 22 3a 20 5b 31 2c 20 33 2c 20 22 73 22 5d 7d 60 f8 dc 89
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

```

Result (Return value of send method): True (<class 'bool'>)
Expectation (Return value of send method): result = True (<class 'bool'>)

```

Success Request Status (Okay) transfered via pure_json_protocol is correct (Content 0 and Type is <class 'int'>).

```

Result (Request Status (Okay) transfered via pure_json_protocol): 0 (<class 'int'>)
Expectation (Request Status (Okay) transfered via pure_json_protocol): result = 0 (<class
↪ 'int'>)

```

Success Request Data transfered via pure_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

```

Result (Request Data transfered via pure_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via pure_json_protocol): result = { 'test': 'test' }
↪ (<class 'dict'>)

```

Success Response Status (Operation not permitted) transfered via pure_json_protocol is correct (Content 5 and Type is <class 'int'>).

```
Result (Response Status (Operation not permitted) transfered via pure_json_protocol): 5  
↳ (<class 'int'>)
```

```
Expectation (Response Status (Operation not permitted) transfered via pure_json_protocol):  
↳ result = 5 (<class 'int'>)
```

Success Response Data transfered via pure_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

```
Result (Response Data transfered via pure_json_protocol): [ 1, 3, 's' ] (<class 'list'>)
```

```
Expectation (Response Data transfered via pure_json_protocol): result = [ 1, 3, 's' ] (<class  
↳ 'list'>)
```

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.
```

```
Result (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): result = None (<class 'NoneType'>)
```

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

```
SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.
```

```
Result (Return Value (response instance) for pure_json_protocol.receive with data data_id  
↳ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (response instance) for pure_json_protocol.receive with data  
↳ data_id 0xaffe): result = None (<class 'NoneType'>)
```

B.1.8 socket_protocol.struct_json_protocol: Send and receive check (Twice for coverage of buffer initialisation).

Testresult

This test was passed with the state: **Success**.

Info Send and received data by struct_json_protocol.

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↪ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↪ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↪ 74 22 7d 47
Receive data (29): 00 00 00 00 00 00 00 0a 00 00 af fe 7b 22 74 65 73 74 22 3a 20 22 74 65 73
↪ 74 22 7d 47
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Executing callback response_data_method to process received data
SJP: TX -> status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
Send data: (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
Receive data (24): 00 00 00 05 00 00 00 0b 00 00 af fe 5b 31 2c 20 33 2c 20 22 73 22 5d 28
SJP: RX <- status: 5, service_id: 11, data_id: 45054, data: "[1, 3, 's']"
SJP: Received message has a peculiar status: Operation not permitted
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method

```

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

```
Result (Return value of send method): True (<class 'bool'>)
```

```
Expectation (Return value of send method): result = True (<class 'bool'>)
```

Success Request Status (Okay) transfered via struct_json_protocol is correct (Content 0 and Type is <class 'int'>).

```
Result (Request Status (Okay) transfered via struct_json_protocol): 0 (<class 'int'>)
```

```
Expectation (Request Status (Okay) transfered via struct_json_protocol): result = 0 (<class
↪ 'int'>)
```

Success Request Data transfered via struct_json_protocol is correct (Content {'test': 'test'} and Type is <class 'dict'>).

```
Result (Request Data transfered via struct_json_protocol): { 'test': 'test' } (<class 'dict'>)
Expectation (Request Data transfered via struct_json_protocol): result = { 'test': 'test' }
↳ (<class 'dict'>)
```

Success Response Status (Operation not permitted) transfered via struct_json_protocol is correct (Content 5 and Type is <class 'int'>).

```
Result (Response Status (Operation not permitted) transfered via struct_json_protocol): 5
↳ (<class 'int'>)
Expectation (Response Status (Operation not permitted) transfered via struct_json_protocol):
↳ result = 5 (<class 'int'>)
```

Success Response Data transfered via struct_json_protocol is correct (Content [1, 3, 's'] and Type is <class 'list'>).

```
Result (Response Data transfered via struct_json_protocol): [ 1, 3, 's' ] (<class 'list'>)
Expectation (Response Data transfered via struct_json_protocol): result = [ 1, 3, 's' ]
↳ (<class 'list'>)
```

Success Return Value (request instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

```
Result (Return Value (request instance) for struct_json_protocol.receive with data data_id
↳ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (request instance) for struct_json_protocol.receive with data
↳ data_id 0xaffe): result = None (<class 'NoneType'>)
```

Success Return Value (response instance) for struct_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

```
Result (Return Value (response instance) for struct_json_protocol.receive with data data_id
↳ 0xaffe): None (<class 'NoneType'>)
```

```
Expectation (Return Value (response instance) for struct_json_protocol.receive with data
↳ data_id 0xaffe): result = None (<class 'NoneType'>)
```

B.1.9 socket_protocol.pure_json_protocol: Checksum corruption while sending.

Testresult

This test was passed with the state: **Success**.

Info Send data with wrong checksum by pure_json_protocol.

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "{\"test\": 'test'}"

Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89

Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
 ↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
 ↪ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 8a

SJP: Received message has a wrong checksum and will be ignored: (79): 7b 22 64 61 74 61 5f 69
 ↪ 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69 63 65 5f 69 64 22 3a 20 31 30 2c 20 22
 ↪ 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 7b 22 74 65 73 74 22 3a 20 22
 ↪ 74 65 73 74 22 7d 7d 82 1c 8b 8a.

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

Result (Return value of send method): True (<class 'bool'>)

Expectation (Return value of send method): result = True (<class 'bool'>)

Success Callback executed variable is correct (Content False and Type is <class 'bool'>).

Result (Callback executed variable): False (<class 'bool'>)

Expectation (Callback executed variable): result = False (<class 'bool'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<class 'NoneType'>)

B.1.10 socket_protocol.pure_json_protocol: Timeout measurement for authentication and send method.

Testresult

This test was passed with the state: **Success**.

Success Timeout for authentication is correct (Content 0.20095229148864746 in [0.2 ... 0.220000000000000003] and Type is <class 'float'>).

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↪ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↪ 6c 6c 7d 9e 85 7b 8d
Result (Timeout for authentication): 0.20095229148864746 (<class 'float'>)
Expectation (Timeout for authentication): 0.2 <= result <= 0.220000000000000003
```

Success Timeout for authentication is correct (Content 0.5016295909881592 in [0.5 ... 0.55] and Type is <class 'float'>).

```
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↪ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↪ 6c 6c 7d 9e 85 7b 8d
Result (Timeout for authentication): 0.5016295909881592 (<class 'float'>)
Expectation (Timeout for authentication): 0.5 <= result <= 0.55
```

Success Timeout for send method is correct (Content 0.2006547451019287 in [0.2 ... 0.220000000000000003] and Type is <class 'float'>).

```
SJP: TIMEOUT (0.2s): Requested data (service_id: 30; data_id: 0) not in buffer.
Result (Timeout for send method): 0.2006547451019287 (<class 'float'>)
Expectation (Timeout for send method): 0.2 <= result <= 0.220000000000000003
```

Success Timeout for send method is correct (Content 0.5013296604156494 in [0.5 ... 0.55] and Type is <class 'float'>).

```
SJP: TIMEOUT (0.5s): Requested data (service_id: 30; data_id: 0) not in buffer.
Result (Timeout for send method): 0.5013296604156494 (<class 'float'>)
Expectation (Timeout for send method): 0.5 <= result <= 0.55
```

B.1.11 socket_protocol.pure_json_protocol: No Callback at response instance for the request.

Testresult

This test was passed with the state: **Success**.

Info	Send data, but no callback registered (pure_json_protocol).
-------------	---

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
Send data: (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↳ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↳ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
Receive data (79): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↳ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↳ 22 3a 20 7b 22 74 65 73 74 22 3a 20 22 74 65 73 74 22 7d 7d 82 1c 8b 89
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: '{"test': 'test'}"
SJP: Received message with no registered callback. Sending negative response.
SJP: TX -> status: 1, service_id: 11, data_id: 45054, data: "None"
Send data: (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↳ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 64 61 74 61
↳ 22 3a 20 6e 75 6c 6c 7d 24 86 3f 75
Receive data (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↳ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 64 61 74 61
↳ 22 3a 20 6e 75 6c 6c 7d 24 86 3f 75
SJP: RX <- status: 1, service_id: 11, data_id: 45054, data: "None"
SJP: Received message has a peculiar status: Request has no callback. Data buffered.
SJP: Message data is stored in buffer and is now ready to be retrieved by receive method
    
```

Success	Return value of send method is correct (Content True and Type is <class 'bool'>).
----------------	---

```

Result (Return value of send method): True (<class 'bool'>)
Expectation (Return value of send method): result = True (<class 'bool'>)
    
```

Success	Response Status (Request has no callback. Data buffered.) transfered via pure_json_protocol is correct (Content 1 and Type is <class 'int'>).
----------------	---

```

Result (Response Status (Request has no callback. Data buffered.) transfered via
↳ pure_json_protocol): 1 (<class 'int'>)
Expectation (Response Status (Request has no callback. Data buffered.) transfered via
↳ pure_json_protocol): result = 1 (<class 'int'>)
    
```

Success	Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <class 'NoneType'>).
----------------	---

Result (Response Data (no data) transfered via pure_json_protocol): None (<class 'NoneType'>)

Expectation (Response Data (no data) transfered via pure_json_protocol): result = None
↪ (<class 'NoneType'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
↪ data_id 0xaffe): result = None (<class 'NoneType'>)

B.1.12 socket_protocol.pure_json_protocol: Authentication required, but not processed/correctly processed.

Testresult

This test was passed with the state: **Success**.

Info Authentication with different secrets for request and response instance (pure_json_protocol).

Unittest for socket_protocol

```
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Requesting seed for authentication
SJP: TX -> status: 0, service_id: 1, data_id: 0, data: "None"
Send data: (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↳ 6c 6c 7d 9e 85 7b 8d
Receive data (62): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 31 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 6e 75
↳ 6c 6c 7d 9e 85 7b 8d
SJP: RX <- status: 0, service_id: 1, data_id: 0, data: "None"
SJP: Executing callback __authenticate_create_seed__ to process received data
SJP: Got seed request, sending seed for authentication
SJP: TX -> status: 0, service_id: 2, data_id: 0, data:
↳ "'d463e5de27d998291c8e281fb4865db47579f1fe08f59f013e39520d8d44efd8'"
Send data: (124): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 32 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22 64
↳ 34 36 33 65 35 64 65 32 37 64 39 39 38 32 39 31 63 38 65 32 38 31 66 62 34 38 36 35 64 62
↳ 34 37 35 37 39 66 31 66 65 30 38 66 35 39 66 30 31 33 65 33 39 35 32 30 64 38 64 34 34 65
↳ 66 64 38 22 7d 1e 9d 18 d9
Receive data (124): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f
↳ 69 64 22 3a 20 32 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22
↳ 64 34 36 33 65 35 64 65 32 37 64 39 39 38 32 39 31 63 38 65 32 38 31 66 62 34 38 36 35 64
↳ 62 34 37 35 37 39 66 31 66 65 30 38 66 35 39 66 30 31 33 65 33 39 35 32 30 64 38 64 34 34
↳ 65 66 64 38 22 7d 1e 9d 18 d9
SJP: RX <- status: 0, service_id: 2, data_id: 0, data:
↳ "'d463e5de27d998291c8e281fb4865db47579f1fe08f59f013e39520d8d44efd8'"
SJP: Executing callback __authenticate_create_key__ to process received data
SJP: Got seed, sending key for authentication
SJP: TX -> status: 0, service_id: 3, data_id: 0, data:
↳ "'49ffbf9abbfca3e72720defed607774f05b2d0d1806ef70d86d721bc3e20f9716bd1efe6c4326be0f210e6c
↳ 9f947238b79473147ae3f7a8cf20e611a90ffaae8'"
Send data: (188): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f 69
↳ 64 22 3a 20 33 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22 34
↳ 39 66 66 62 66 39 61 62 62 66 63 61 33 65 37 32 37 32 30 64 65 66 65 64 36 30 37 37 37 34
↳ 66 30 35 62 32 64 30 64 31 38 30 36 65 66 37 30 64 38 36 64 37 32 31 62 63 33 65 32 30 66
↳ 39 37 31 36 62 64 31 65 66 65 36 63 34 33 32 36 62 65 30 66 32 31 30 65 36 63 39 66 39 34
↳ 37 32 33 38 62 37 39 34 37 33 31 34 37 61 65 33 66 37 61 38 63 66 32 30 65 36 31 31 61 39
↳ 30 66 66 61 61 65 38 22 7d d6 94 5c a5
Receive data (188): 7b 22 64 61 74 61 5f 69 64 22 3a 20 30 2c 20 22 73 65 72 76 69 63 65 5f
↳ 69 64 22 3a 20 33 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61 22 3a 20 22
↳ 34 39 66 66 62 66 39 61 62 62 66 63 61 33 65 37 32 37 32 30 64 65 66 65 64 36 30 37 37 37
↳ 34 66 30 35 62 32 64 30 64 31 38 30 36 65 66 37 30 64 38 36 64 37 32 31 62 63 33 65 32 30
↳ 66 39 37 31 36 62 64 31 65 66 65 36 63 34 33 32 36 62 65 30 66 32 31 30 65 36 63 39 66 39
↳ 34 37 32 33 38 62 37 39 34 37 33 31 34 37 61 65 33 66 37 61 38 63 66 32 30 65 36 31 31 61
↳ 39 30 66 66 61 61 65 38 22 7d d6 94 5c a5
SJP: RX <- status: 0, service_id: 3, data_id: 0, data:
↳ "'49ffbf9abbfca3e72720defed607774f05b2d0d1806ef70d86d721bc3e20f9716bd1efe6c4326be0f210e6c
↳ 9f947238b79473147ae3f7a8cf20e611a90ffaae8'"
SJP: Executing callback authenticate check key to process received data
```

Result (Return value of authentication): False (<class 'bool'>)

Expectation (Return value of authentication): result = False (<class 'bool'>)

Success Return value of send method is correct (Content True and Type is <class 'bool'>).

Result (Return value of send method): True (<class 'bool'>)

Expectation (Return value of send method): result = True (<class 'bool'>)

Success Response Status (Authentication required) transfered via pure_json_protocol is correct (Content 2 and Type is <class 'int'>).

Result (Response Status (Authentication required) transfered via pure_json_protocol): 2
 ↪ (<class 'int'>)

Expectation (Response Status (Authentication required) transfered via pure_json_protocol):
 ↪ result = 2 (<class 'int'>)

Success Response Data (no data) transfered via pure_json_protocol is correct (Content None and Type is <class 'NoneType'>).

Result (Response Data (no data) transfered via pure_json_protocol): None (<class 'NoneType'>)

Expectation (Response Data (no data) transfered via pure_json_protocol): result = None
 ↪ (<class 'NoneType'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<class 'NoneType'>)

B.1.13 socket_protocol.pure_json_protocol: Register a Callback which is already defined.

Testresult

This test was passed with the state: **Success**.

Info Registering two callbacks which overlap for at least one message (pure_json_protocol).

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

Success Exception (RegistrationError) detection variable is correct (Content True and Type is <class 'bool'>).

Result (Exception (RegistrationError) detection variable): True (<class 'bool'>)

Expectation (Exception (RegistrationError) detection variable): result = True (<class 'bool'>)

B.1.14 socket_protocol.pure_json_protocol: Authentication processed without secret.

Testresult

This test was passed with the state: **Success**.

Info Authentication with no secret definition (pure_json_protocol).

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

SJP: Cleaning up receive-buffer

SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT

Success Return value of authentication is correct (Content False and Type is <class 'bool'>).

Result (Return value of authentication): False (<class 'bool'>)

Expectation (Return value of authentication): result = False (<class 'bool'>)

B.1.15 socket_protocol.pure_json_protocol: Incompatible Callback return value(s).

Testresult

This test was passed with the state: **Success**.

Info Send and received data with incompatible callback (pure_json_protocol).

```

SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: Cleaning up receive-buffer
SJP: Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT
SJP: TX -> status: 0, service_id: 10, data_id: 45054, data: "None"
Send data: (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d e9 e9 77 c0
Receive data (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 35 30 35 34 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d e9 e9 77 c0
SJP: RX <- status: 0, service_id: 10, data_id: 45054, data: "None"
SJP: Executing callback response_data_method_fail to process received data
SJP: TX -> status: 0, service_id: 10, data_id: 48879, data: "None"
Send data: (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d da 63 1a 84
Receive data (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 30 2c 20 22 73 74 61 74 75 73 22 3a 20 30 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d da 63 1a 84
SJP: RX <- status: 0, service_id: 10, data_id: 48879, data: "None"
SJP: Received message with no registered callback. Sending negative response.
SJP: TX -> status: 1, service_id: 11, data_id: 48879, data: "None"
Send data: (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d 17 0c 52 31
Receive data (67): 7b 22 64 61 74 61 5f 69 64 22 3a 20 34 38 38 37 39 2c 20 22 73 65 72 76 69
↪ 63 65 5f 69 64 22 3a 20 31 31 2c 20 22 73 74 61 74 75 73 22 3a 20 31 2c 20 22 64 61 74 61
↪ 22 3a 20 6e 75 6c 6c 7d 17 0c 52 31
SJP: RX <- status: 1, service_id: 11, data_id: 48879, data: "None"
SJP: Received message has a peculiar status: Request has no callback. Data buffered.
SJP: Executing callback response_data_method_fail to process received data

```

Success Exception (TypeError) detection variable is correct (Content True and Type is <class 'bool'>).

Result (Exception (TypeError) detection variable): True (<class 'bool'>)

Expectation (Exception (TypeError) detection variable): result = True (<class 'bool'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 45054) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xaffe is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 45054) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xaffe): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xaffe): result = None (<class 'NoneType'>)

Success Exception (TypeError) detection variable is correct (Content True and Type is <class 'bool'>).

Result (Exception (TypeError) detection variable): True (<class 'bool'>)

Expectation (Exception (TypeError) detection variable): result = True (<class 'bool'>)

Success Return Value (request instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 11; data_id: 48879) not in buffer.

Result (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): None (<class 'NoneType'>)

Expectation (Return Value (request instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): result = None (<class 'NoneType'>)

Success Return Value (response instance) for pure_json_protocol.receive with data data_id 0xbeef is correct (Content None and Type is <class 'NoneType'>).

SJP: TIMEOUT (0.1s): Requested data (service_id: 10; data_id: 48879) not in buffer.

Result (Return Value (response instance) for pure_json_protocol.receive with data data_id
 ↪ 0xbeef): None (<class 'NoneType'>)

Expectation (Return Value (response instance) for pure_json_protocol.receive with data
 ↪ data_id 0xbeef): result = None (<class 'NoneType'>)

C Test-Coverage

C.1 socket_protocol

The line coverage for socket_protocol was 100.0%

The branch coverage for socket_protocol was 100.0%

C.1.1 socket_protocol.__init__.py

The line coverage for socket_protocol.__init__.py was 100.0%

The branch coverage for socket_protocol.__init__.py was 100.0%

```

1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 #
4 """
5 socket_protocol (Socket Protocol)
6 =====
7
8 **Author:**
9
10 * Dirk Alders <sudo-dirk@mount-mockery.de>
11
12 **Description:**
13
14     This Module supports point to point communication for client-server issues.
15
16 **Submodules:**
17
18 * :class:`socket_protocol.struct_json_protocol`
19 * :class:`socket_protocol.pure_json_protocol`
20
21 **Unittest:**
22
23     See also the :download:`unittest <../../socket_protocol/_testresults_/unittest.pdf>`
24     documentation.
25 """
26
27 __DEPENDENCIES__ = ['stringtools']
28
29 import stringtools
30
31 import binascii
32 import hashlib
33 import json
34 import logging
35 import os
36 import struct
37 import sys
38 import time
39
40 logger_name = 'SOCKET_PROTOCOL'
41 logger = logging.getLogger(logger_name)
42
43 __DESCRIPTION__ = """The Module {\\tt %s} is designed to pack and unpack data for serial
44     transportation.
45 For more Information read the sphinx documentation.""" % __name__.replace('-', '\\-')
46 """The Module Description"""
47 __INTERPRETER__ = (2, 3)
48 """The Tested Interpreter-Versions"""
49
50 class RegistrationError(BaseException):
51     pass
52
53

```

```

54 class callback_storage(dict):
55     def __init__(self):
56         dict.__init__(self)
57
58     def get(self, service_id, data_id):
59         if service_id is not None and data_id is not None:
60             try:
61                 return self[service_id][data_id]
62             except KeyError:
63                 pass # nothing to append
64         if data_id is not None:
65             try:
66                 return self[None][data_id]
67             except KeyError:
68                 pass # nothing to append
69         if service_id is not None:
70             try:
71                 return self[service_id][None]
72             except KeyError:
73                 pass # nothing to append
74         try:
75             return self[None][None]
76         except KeyError:
77             pass # nothing to append
78         return None
79
80     def add(self, service_id, data_id, callback):
81         if self.get(service_id, data_id) is not None:
82             raise RegistrationError("Callback for service_id (%s) and data_id (%s) already exists
83 " % (repr(service_id), repr(data_id)))
84         if service_id not in self:
85             self[service_id] = {}
86             self[service_id][data_id] = callback
87
88 class data_storage(dict):
89     KEY_STATUS = 'status'
90     KEY_SERVICE_ID = 'service_id'
91     KEY_DATA_ID = 'data_id'
92     KEY_DATA = 'data'
93
94     def __init__(self, *args, **kwargs):
95         dict.__init__(self, *args, **kwargs)
96
97     def get_status(self, default=None):
98         return self.get(self.KEY_STATUS, default)
99
100     def get_service_id(self, default=None):
101         return self.get(self.KEY_SERVICE_ID, default)
102
103     def get_data_id(self, default=None):
104         return self.get(self.KEY_DATA_ID, default)
105
106     def get_data(self, default=None):
107         return self.get(self.KEY_DATA, default)
108
109
110 class struct_json_protocol(object):

```

Unittest for socket_protocol

```
111 """
112 :param comm_instance: a communication instance supportin at least these functions: :func:`
register_callback`, :func:`register_disconnect_callback`, :func:`send`.
113 :type comm_instance: instance
114 :param secret: A secret (e.g. created by ``binascii.hexlify(os.urandom(24))``).
115 :type secret: str
116
117 This communication protocol supports to transfer a Service-ID, Data-ID and Data. The
transmitted data is shorter than :class:`pure_json_protocol`.
118
119 .. note::
120     This class is here for compatibility reasons. Usage of :class:`pure_json_protocol` is
recommended.
121
122 **Example:**
123
124 Server:
125
126 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__struct_json_protocol_server.py
127
128 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__struct_json_protocol_server.log
129
130
131 Client:
132
133 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__struct_json_protocol_client.py
134
135 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__struct_json_protocol_client.log
136 """
137 LOG_PREFIX = 'SJP: '
138
139 SID_AUTH_SEED_REQUEST = 1
140 SID_AUTH_KEY_REQUEST = 2
141 SID_AUTH_KEY_CHECK_REQUEST = 3
142 SID_AUTH_KEY_CHECK_RESPONSE = 4
143 SID_READ_REQUEST = 10
144 SID_READ_RESPONSE = 11
145 SID_WRITE_REQUEST = 20
146 SID_WRITE_RESPONSE = 21
147 SID_EXECUTE_REQUEST = 30
148 SID_EXECUTE_RESPONSE = 31
149
150 SID_RESPONSE_DICT = {SID_AUTH_SEED_REQUEST: SID_AUTH_KEY_REQUEST,
151                      SID_AUTH_KEY_REQUEST: SID_AUTH_KEY_CHECK_REQUEST,
152                      SID_AUTH_KEY_CHECK_REQUEST: SID_AUTH_KEY_CHECK_RESPONSE,
153                      SID_READ_REQUEST: SID_READ_RESPONSE,
154                      SID_WRITE_REQUEST: SID_WRITE_RESPONSE,
155                      SID_EXECUTE_REQUEST: SID_EXECUTE_RESPONSE}
156
157 SID_AUTH_LIST = [SID_AUTH_SEED_REQUEST, SID_AUTH_KEY_REQUEST, SID_AUTH_KEY_CHECK_REQUEST,
158                 SID_AUTH_KEY_CHECK_RESPONSE]
159
160 STATUS_OKAY = 0
161 STATUS_BUFFERING_UNHANDLED_REQUEST = 1
162 STATUS_AUTH_REQUIRED = 2
163 STATUS_SERVICE_OR_DATA_UNKNOWN = 3
164 STATUS_CHECKSUM_ERROR = 4
165 STATUS_OPERATION_NOT_PERMITTED = 5
166 STATUS_NAMES = {STATUS_OKAY: 'Okay',
```

Unittest for socket_protocol

```

166         STATUS_BUFFERING_UNHANDLED_REQUEST: 'Request has no callback. Data buffered.'
167
168         STATUS_AUTH_REQUIRED: 'Authentication required',
169         STATUS_SERVICE_OR_DATA_UNKNOWN: 'Service or Data unknown',
170         STATUS_CHECKSUM_ERROR: 'Checksum Error',
171         STATUS_OPERATION_NOT_PERMITTED: 'Operation not permitted'}
172
173     AUTH_STATE_UNKNOWN_CLIENT = 0
174     AUTH_STATE_SEED_REQUESTED = 1
175     AUTH_STATE_SEED_TRANSFERRED = 2
176     AUTH_STATE_KEY_TRANSFERRED = 3
177     AUTH_STATE_TRUSTED_CLIENT = 4
178     AUTH_STATUS_NAMES = {AUTH_STATE_UNKNOWN_CLIENT: 'Unknown Client',
179                         AUTH_STATE_SEED_REQUESTED: 'Seed was requested',
180                         AUTH_STATE_SEED_TRANSFERRED: 'Seed has been sent',
181                         AUTH_STATE_KEY_TRANSFERRED: 'Key has been sent',
182                         AUTH_STATE_TRUSTED_CLIENT: 'Trusted Client'}
183
184     def __init__(self, comm_instance, secret=None):
185         self.__secret__ = secret
186         self.__clean_receive_buffer__()
187         self.__callbacks__ = callback_storage()
188         self.__callbacks__.add(self.SID_AUTH_SEED_REQUEST, 0, self.__authenticate_create_seed__
189                                )
190         self.__callbacks__.add(self.SID_AUTH_KEY_REQUEST, 0, self.__authenticate_create_key__
191                                )
192         self.__callbacks__.add(self.SID_AUTH_KEY_CHECK_REQUEST, 0, self.
193                                __authenticate_check_key__
194                                )
195         self.__callbacks__.add(self.SID_AUTH_KEY_CHECK_RESPONSE, 0, self.
196                                __authenticate_process_feedback__
197                                )
198         self.__authentication_state_reset__()
199         self.__seed__ = None
200         self.__comm_inst__ = comm_instance
201         self.__comm_inst__.register_callback(self.__data_available_callback__)
202         self.__comm_inst__.register_connect_callback(self.__clean_receive_buffer__)
203         self.__comm_inst__.register_disconnect_callback(self.__authentication_state_reset__)
204
205     def __authentication_state_reset__(self):
206         logger.info("%s Resetting authentication state to AUTH_STATE_UNKNOWN_CLIENT", self.
207                    LOG_PREFIX)
208         self.__authentication_state__ = self.AUTH_STATE_UNKNOWN_CLIENT
209
210     def __analyse_frame__(self, frame):
211         status, service_id, data_id = struct.unpack('>III', frame[0:12])
212         if sys.version_info >= (3, 0):
213             data = json.loads(frame[12:-1].decode('utf-8'))
214         else:
215             data = json.loads(frame[12:-1])
216         return self.__mk_msg__(status, service_id, data_id, data)
217
218     def __build_frame__(self, service_id, data_id, data, status=STATUS_OKAY):
219         frame = struct.pack('>III', status, service_id, data_id)
220         if sys.version_info >= (3, 0):
221             frame += bytes(json.dumps(data), 'utf-8')
222             frame += self.__calc_chksum__(frame)
223         else:
224             frame += json.dumps(data)
225             frame += self.__calc_chksum__(frame)
226         return frame
227
228     def __calc_chksum__(self, raw_data):
229         checksum = 0
230         for b in raw_data:
231             if sys.version_info >= (3, 0):
232                 checksum ^= b

```

Unittest for socket_protocol

```

225         else:
226             checksum ^= ord(b)
227         if sys.version_info >= (3, 0):
228             return bytes([checksum])
229         else:
230             return chr(checksum)
231
232     def __check_frame_checksum__(self, frame):
233         return self.__calc_chksum__(frame[:-1]) == frame[-1:]
234
235     def __data_available_callback__(self, comm_inst):
236         frame = comm_inst.receive()
237         if not self.__check_frame_checksum__(frame):
238             logger.warning("%s Received message has a wrong checksum and will be ignored: %s.",
239                             self.LOG_PREFIX, stringtools.hexlify(frame))
240         else:
241             msg = self.__analyse_frame__(frame)
242             logger.info(
243                 "%s RX <- status: %s, service_id: %s, data_id: %s, data: \"%s\"!",
244                 self.LOG_PREFIX,
245                 repr(msg.get_status()),
246                 repr(msg.get_service_id()),
247                 repr(msg.get_data_id()),
248                 repr(msg.get_data())
249             )
250             callback = self.__callbacks__.get(msg.get_service_id(), msg.get_data_id())
251             if msg.get_service_id() in self.SID_RESPONSE_DICT.keys():
252                 #
253                 # REQUEST RECEIVED
254                 #
255                 if self.__secret__ is not None and not self.check_authentication_state() and
256                 msg.get_service_id() not in self.SID_AUTH_LIST:
257                     status = self.STATUS_AUTH_REQUIRED
258                     data = None
259                     logger.warning("%s Received message needs authentication: %s. Sending
260                                     negative response.", self.LOG_PREFIX, self.AUTH_STATUS_NAMES.get(self.
261                                     __authentication_state__, 'Unknown authentication status!'))
262                 elif callback is None:
263                     logger.warning("%s Received message with no registered callback. Sending
264                                     negative response.", self.LOG_PREFIX)
265                     status = self.STATUS_BUFFERING_UNHANDLED_REQUEST
266                     data = None
267                 else:
268                     try:
269                         logger.debug("%s Executing callback %s to process received data", self.
270                                     LOG_PREFIX, callback.__name__)
271                         status, data = callback(msg)
272                     except TypeError:
273                         raise TypeError('Check return value of callback function {callback_name}
274                                     for service_id {service_id} and data_id {data_id}'.format(callback_name=callback.__name__,
275                                     service_id=repr(msg.get_service_id()), data_id=repr(msg.get_data_id())))
276                         self.send(self.SID_RESPONSE_DICT[msg.get_service_id()], msg.get_data_id(), data,
277                                     status=status)
278                 else:
279                     #
280                     # RESPONSE RECEIVED
281                     #
282                     if msg.get_status() not in [self.STATUS_OKAY]:
283                         logger.warning("%s Received message has a peculiar status: %s", self.
284                                     LOG_PREFIX, self.STATUS_NAMES.get(msg.get_status(), 'Unknown status response!'))
285                     if callback is None:
286                         status = self.STATUS_OKAY
287                         data = None
288                         self.__buffer_received_data__(msg)

```

Unittest for socket_protocol

```

279         else:
280             try:
281                 logger.debug("%s Executing callback %s to process received data", self.
LOG_PREFIX, callback.__name__)
282                 status, data = callback(msg)
283             except TypeError:
284                 raise TypeError('Check return value of callback function {callback_name}
for service_id {service_id} and data_id {data_id}'.format(callback_name=callback.__name__,
service_id=repr(msg.get_service_id()), data_id=repr(msg.get_data_id())))
285
286     def __buffer_received_data__(self, msg):
287         if not msg.get_service_id() in self.__msg_buffer__:
288             self.__msg_buffer__[msg.get_service_id()] = {}
289         if not msg.get_data_id() in self.__msg_buffer__[msg.get_service_id()]:
290             self.__msg_buffer__[msg.get_service_id()][msg.get_data_id()] = []
291         self.__msg_buffer__[msg.get_service_id()][msg.get_data_id()].append(msg)
292         logger.debug("%s Message data is stored in buffer and is now ready to be retrieved by
receive method", self.LOG_PREFIX)
293
294     def __clean_receive_buffer__(self):
295         logger.debug("%s Cleaning up receive-buffer", self.LOG_PREFIX)
296         self.__msg_buffer__ = {}
297
298     def receive(self, service_id, data_id, timeout=1):
299         data = None
300         cnt = 0
301         while data is None and cnt < timeout * 10:
302             try:
303                 data = self.__msg_buffer__.get(service_id, {}).get(data_id, []).pop(0)
304             except IndexError:
305                 data = None
306             cnt += 1
307             time.sleep(0.1)
308         if data is None and cnt >= timeout * 10:
309             logger.warning('%s TIMEOUT (%ss): Requested data (service_id: %s; data_id: %s) not in
buffer.', self.LOG_PREFIX, repr(timeout), repr(service_id), repr(data_id))
310         return data
311
312     def __mk_msg__(self, status, service_id, data_id, data):
313         return data_storage({data_storage.KEY_DATA.ID: data_id, data_storage.KEY_SERVICE.ID:
service_id, data_storage.KEY_STATUS: status, data_storage.KEY_DATA: data})
314
315     def send(self, service_id, data_id, data, status=STATUS_OKAY, timeout=2, log_lvl=logging.INFO
):
316         """
317         :param service_id: The Service-ID for the message. See class definitions starting with ``
SERVICE``.
318         :type service_id: int
319         :param data_id: The Data-ID for the message.
320         :type data_id: int
321         :param data: The data to be transfered. The data needs to be json compatible.
322         :type data: str
323         :param status: The Status for the message. All requests should have ``STATUS_OKAY``.
324         :type status: int
325         :param timeout: The timeout for sending data (e.g. time to establish new connection).
326         :type timeout: float
327         :param rx_log_lvl: The log level to log outgoing TX-data
328         :type rx_log_lvl: int
329         :return: True if data had been sent, otherwise False.
330         :rtype: bool
331
332         This methods sends out a message with the given content.
333         """

```

```

334     logger.log(log_lvl, '%s TX -> status: %d, service_id: %d, data_id: %d, data: "%s"', self.
LOG_PREFIX, status, service_id, data_id, repr(data))
335     return self.__comm_inst__.send(self.__build_frame__(service_id, data_id, data, status),
timeout=timeout, log_lvl=logging.DEBUG)
336
337
338 def register_callback(self, service_id, data_id, callback):
339     """
340     :param service_id: The Service-ID for the message. See class definitions starting with ``
SID_``.
341     :type service_id: int
342     :param data_id: The Data-ID for the message.
343     :type data_id: int
344     :returns: True, if registration was successfull; False, if registration failed (e.g.
existence of a callback for this configuration)
345     :rtype: bool
346
347     This method registers a callback for the given parameters. Givin ``None`` means, that all
Service-IDs or all Data-IDs are used.
348     If a message hitting these parameters has been received, the callback will be executed.
349
350     .. note:: The :func:`callback` is prioritised in the following order:
351
352     * Callbacks with defined Service-ID and Data-ID.
353     * Callbacks with a defined Data-ID.
354     * Callbacks with a defined Service-ID.
355     * Unspecific Callbacks
356
357     .. note:: The :func:`callback` is executed with these arguments:
358
359     :param msg: A :class:`dict` containing all message information.
360     :returns: status (see class definition starting with ``STATUS``), response_data (
JSON compatible object)
361     """
362     self.__callbacks__.add(service_id, data_id, callback)
363
364 def authenticate(self, timeout=2):
365     """
366     :param timeout: The timeout for the authentication (requesting seed, sending key and
getting authentication_feedback).
367     :type timeout: float
368     :returns: True, if authentication was successfull; False, if not.
369     :rtype: bool
370
371     This method autheticates the client at the server.
372
373     .. note:: An authentication will only processed, if a secret had been given on
initialisation.
374
375     .. note:: Client and Server needs to use the same secret.
376     """
377     if self.__secret__ is not None:
378         self.__authentication_state__ = self.AUTH.STATE.SEED.REQUESTED
379         logger.info("%s Requesting seed for authentication", self.LOG_PREFIX)
380         self.send(self.SID_AUTH_SEED_REQUEST, 0, None)
381         cnt = 0
382         while cnt < timeout * 10:
383             time.sleep(0.1)
384             if self.__authentication_state__ == self.AUTH.STATE.TRUSTED_CLIENT:
385                 return True
386             elif self.__authentication_state__ == self.AUTH.STATE.UNKNOWN_CLIENT:
387                 break
388             cnt += 1
389         return False

```

Unittest for socket_protocol

```

389
390 def check_authentication_state(self):
391     """
392     :return: True, if authentication state is okay, otherwise False
393     :rtype: bool
394     """
395     return self.__authentication_state__ == self.AUTH_STATE_TRUSTED_CLIENT
396
397 def __authenticate_salt_and_hash__(self, seed):
398     if sys.version_info >= (3, 0):
399         return hashlib.sha512(bytes(seed, 'utf-8') + self.__secret__).hexdigest()
400     else:
401         return hashlib.sha512(seed.encode('utf-8') + self.__secret__.encode('utf-8')).
402             hexdigest()
403
404 def __authenticate_create_seed__(self, msg):
405     logger.info("%s Got seed request, sending seed for authentication", self.LOG_PREFIX)
406     self.__authentication_state__ = self.AUTH_STATE_SEED_TRANSFERRED
407     if sys.version_info >= (3, 0):
408         self.__seed__ = binascii.hexlify(os.urandom(32)).decode('utf-8')
409     else:
410         self.__seed__ = binascii.hexlify(os.urandom(32))
411     return self.STATUS_OKAY, self.__seed__
412
413 def __authenticate_create_key__(self, msg):
414     logger.info("%s Got seed, sending key for authentication", self.LOG_PREFIX)
415     self.__authentication_state__ = self.AUTH_STATE_KEY_TRANSFERRED
416     seed = msg.get_data()
417     key = self.__authenticate_salt_and_hash__(seed)
418     return self.STATUS_OKAY, key
419
420 def __authenticate_check_key__(self, msg):
421     key = msg.get_data()
422     if key == self.__authenticate_salt_and_hash__(self.__seed__):
423         self.__authentication_state__ = self.AUTH_STATE_TRUSTED_CLIENT
424         logger.info("%s Got correct key, sending positive authentication feedback", self.
425             LOG_PREFIX)
426         return self.STATUS_OKAY, True
427     else:
428         self.__authentication_state__ = self.AUTH_STATE_UNKNOWN_CLIENT
429         logger.info("%s Got incorrect key, sending negative authentication feedback", self.
430             LOG_PREFIX)
431         return self.STATUS_OKAY, False
432
433 def __authenticate_process_feedback__(self, msg):
434     feedback = msg.get_data()
435     if feedback:
436         self.__authentication_state__ = self.AUTH_STATE_TRUSTED_CLIENT
437         logger.info("%s Got positive authentication feedback", self.LOG_PREFIX)
438     else:
439         self.__authentication_state__ = self.AUTH_STATE_UNKNOWN_CLIENT
440         logger.warning("%s Got negative authentication feedback", self.LOG_PREFIX)
441     return self.STATUS_OKAY, None
442
443 class pure_json_protocol(struct_json_protocol):

```


Unittest for socket_protocol

```
442 """
443 :param comm_instance: a communication instance supportin at least these functions: :func:`
register_callback`, :func:`register_disconnect_callback`, :func:`send`.
444 :type comm_instance: instance
445 :param secret: A secret (e.g. created by ``binascii.hexlify(os.urandom(24))``).
446 :type secret: str
447
448 This communication protocol supports to transfer a Service-ID, Data-ID and Data.
449
450 **Example:**
451
452 Server:
453
454 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__pure_json_protocol_server.py
455
456 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__pure_json_protocol_server.log
457
458
459 Client:
460
461 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__pure_json_protocol_client.py
462
463 .. literalinclude:: ../../socket_protocol/_examples_/
socket_protocol__pure_json_protocol_client.log
464 """
465 def __init__(self, comm_instance, secret=None):
466     struct_json_protocol.__init__(self, comm_instance, secret)
467
468 def __build_frame__(self, service_id, data_id, data, status=struct_json_protocol.STATUS_OKAY)
469 :
470     data_frame = json.dumps(self.__mk_msg__(status, service_id, data_id, data))
471     if sys.version_info >= (3, 0):
472         data_frame = bytes(data_frame, 'utf-8')
473     checksum = self.__calc_chksum__(data_frame)
474     return data_frame + checksum
475
476 def __analyse_frame__(self, frame):
477     if sys.version_info >= (3, 0):
478         return data_storage(json.loads(frame[:-4].decode('utf-8')))
479     else:
480         return data_storage(json.loads(frame[:-4]))
481
482 def __calc_chksum__(self, raw_data):
483     return struct.pack('>I', binascii.crc32(raw_data) & 0xffffffff)
484
485 def __check_frame_checksum__(self, frame):
486     return self.__calc_chksum__(frame[:-4]) == frame[-4:]
```