

python / cpython Public

<> Code Issues 5k+ Pull requests 2.1k Actions Projects Security and q

# Commit 39258d3



**tiran** authored on Apr 17, 2021 Verified

[bpo-43669](#): PEP 644: Require OpenSSL 1.1.1 or newer ([GH-23014](#))

- Remove HAVE\_X509\_VERIFY\_PARAM\_SET1\_HOST check
- Update hashopenssl to require OpenSSL 1.1.1
- multissltests only OpenSSL > 1.1.0
- ALPN is always supported
- SNI is always supported
- Remove deprecated NPN code. Python wrappers are no-op.
- ECDH is always supported
- Remove OPENSSL\_VERSION\_1\_1 macro
- Remove locking callbacks
- Drop PY\_OPENSSL\_1\_1\_API macro
- Drop HAVE\_SSL\_CTX\_CLEAR\_OPTIONS macro
- SSL\_CTRL\_GET\_MAX\_PROTO\_VERSION is always defined now
- security level is always available now
- get\_num\_tickets is available with TLS 1.3
- X509\_V\_ERR\_MISMATCH is always available now
- Always set SSL\_MODE\_RELEASE\_BUFFERS
- X509\_V\_FLAG\_TRUSTED\_FIRST is always available
- get\_ciphers is always supported
- SSL\_CTX\_set\_keylog\_callback is always available
- Update Modules/Setup with static link example
- Mention PEP in whatsnew
- Drop 1.0.2 and 1.1.0 from GHA tests

[main](#) (#23014) · v3.15.0a8 ... v3.10.0b1

1 parent [b467d9a](#) commit 39258d3

**17 files changed**

**+5,310 -8,440**

Top



▼ .github/workflows

- ├─ build.yml
- └─ Doc
  - └─ using
    - ├─ unix.rst
    - └─ whatsnew
      - ├─ 3.10.rst
  - └─ Lib
    - ├─ ssl.py
    - └─ test
      - ├─ test\_ssl.py
  - └─ Misc/NEWS.d/next/Build
    - ├─ 2021-03-30-14-19-39.bpo-43669.IWMUYx.rst
  - └─ Modules
    - ├─ Setup
    - ├─ \_hashopenssl.c
    - ├─ \_ssl.c
    - └─ \_ssl
      - ├─ debughelpers.c
      - └─ clinic
        - ├─ \_hashopenssl.c.h
        - └─ \_ssl.c.h
  - └─ Tools/ssl
    - ├─ multissltests.py
    - ├─ configure.ac
    - ├─ configure
    - ├─ pyconfig.h.in
    - └─ setup.py

1

🔍 Search within code



```

  ▾ .github/workflows/build.yml
  ↑... @@ -177,7 +177,7 @@ jobs:
177 177     strategy:
178 178         fail-fast: false
179 179     matrix:
180 180 -     openssl_ver: [1.0.2u, 1.1.0l, 1.1.1k, 3.0.0-alpha14]
180 180 +     openssl_ver: [1.1.1k, 3.0.0-alpha14]
181 181     env:
182 182         OPENSSL_VER: ${ matrix.openssl_ver }
183 183         MULTISSL_DIR: ${ github.workspace }/multissl
  ↓...

```

```

  ▾ Doc/using/unix.rst
  <> 📄 ...
  ↑... @@ -135,6 +135,7 @@ some Unices may not have the :program:`env` command, so
  you may need to hardcode
135 135
136 136     To use shell commands in your Python scripts, look at the :mod:`subprocess`
  module.
137 137
138 138 + .. _unix_custom_openssl:
138 139
139 140     Custom OpenSSL
140 141     =====
  ↓...

```

```

  ▾ Doc/whatsnew/3.10.rst
  <> 📄 ...
  ↑... @@ -65,6 +65,7 @@ Summary -- Release highlights
65 65
66 66     .. PEP-sized items next.
67 67
68 68 + * :pep:`644`, require OpenSSL 1.1.1 or newer
68 69
69 70
70 71     New Features
  ↓...
  ↑... @@ -1438,6 +1439,10 @@ CPython bytecode changes
1438 1439     Build Changes
1439 1440     =====

```

1440	1441	
1442	1443	+ * :pep:`644`: Python now requires OpenSSL <b>1.1.1</b> or newer. OpenSSL <b>1.0.2</b> is no longer supported.
1444	1445	+ (Contributed by Christian Heimes <a href="#">in :issue:`43669`</a> .)
1441	1442	+ * The C99 functions <code>:c:func:`snprintf`</code> and <code>:c:func:`vsprintf`</code> are now required
1442	1443	to build Python.
1443	1444	(Contributed by Victor Stinner <a href="#">in :issue:`36020`</a> .)
↓		@@ -1483,7 +1488,6 @@ Build Changes
1483	1484	(Contributed by Christian Heimes <a href="#">in :issue:`43466`</a> .)
1484	1485	
1485	1486	
1486	1487	-
1487	1488	C API Changes
1488	1489	=====
1489	1490	
↓		

Lib/ssl.py		...
↑		@@ -909,15 +909,12 @@ def selected_npn_protocol(self):
909	909	"""Return the currently selected NPN protocol as a string, or
		``None``
910	910	if a next protocol was not negotiated or if NPN is not supported by
		one
911	911	of the peers."""
912	912	-     if _ssl.HAS_NPN:
913	913	-         return self._sslobj.selected_npn_protocol()
914	914	
915	915	def selected_alpn_protocol(self):
916	916	"""Return the currently selected ALPN protocol as a string, or
		``None``
917	917	if a next protocol was not negotiated or if ALPN is not supported by
		one
918	918	of the peers."""
919	919	-     if _ssl.HAS_ALPN:
920	920	-         return self._sslobj.selected_alpn_protocol()
	917	+         return self._sslobj.selected_alpn_protocol()
921	921	

```

922     919         def cipher(self):
923     920             """Return the currently selected cipher as a 3-tuple ``(name,
@@ -1126,10 +1123,7 @@ def getpeer_cert(self, binary_form=False):
    ↓
    ↑
1126    1123         @_sslcopydoc
1127    1124         def selected_npn_protocol(self):
1128    1125             self._checkClosed()
1129    -         if self._sslobj is None or not _ssl.HAS_NPN:
1130    -             return None
1131    -         else:
1132    -             return self._sslobj.selected_npn_protocol()
+         return None
1133    1127
1134    1128         @_sslcopydoc
1135    1129         def selected_alpn_protocol(self):
    ↓

```

```

Lib/test/test_ssl.py
    ↑
@@ -40,7 +40,6 @@
40     40     PROTOCOLS = sorted(_ssl._PROTOCOL_NAMES)
41     41     HOST = socket_helper.HOST
42     42     IS_LIBRESSL = _ssl.OPENSSL_VERSION.startswith('LibreSSL')
43     - IS_OPENSSL_1_1_0 = not IS_LIBRESSL and _ssl.OPENSSL_VERSION_INFO >= (1, 1, 0)
44     43     IS_OPENSSL_1_1_1 = not IS_LIBRESSL and _ssl.OPENSSL_VERSION_INFO >= (1, 1, 1)
45     44     IS_OPENSSL_3_0_0 = not IS_LIBRESSL and _ssl.OPENSSL_VERSION_INFO >= (3, 0, 0)
46     45     PY_SSL_DEFAULT_CIPHERS = sysconfig.get_config_var('PY_SSL_DEFAULT_CIPHERS')
    ↓
    ↑
@@ -270,18 +269,6 @@ def handle_error(prefix):
270    269         if support.verbose:
271    270             sys.stdout.write(prefix + exc_format)
272    271
273    - def can_clear_options():
274    -     # 0.9.8m or higher
275    -     return _ssl._OPENSSL_API_VERSION >= (0, 9, 8, 13, 15)
276    -
277    - def no_sslv2_implies_sslv3_hello():
278    -     # 0.9.7h or higher
279    -     return _ssl.OPENSSL_VERSION_INFO >= (0, 9, 7, 8, 15)
280    -
281    - def have_verify_flags():

```

282	-	# 0.9.8 or higher
283	-	return ssl.OPENSSL_VERSION_INFO >= (0, 9, 8, 0, 15)
284	-	
285	272	def _have_secp_curves():
286	273	if not ssl.HAS_ECDH:
287	274	return False
		@@ -372,17 +359,15 @@ def test_constants(self):
372	359	ssl.OP_SINGLE_DH_USE
373	360	if ssl.HAS_ECDH:
374	361	ssl.OP_SINGLE_ECDH_USE
375	-	if ssl.OPENSSL_VERSION_INFO >= (1, 0):
376	-	ssl.OP_NO_COMPRESSION
362	+	ssl.OP_NO_COMPRESSION
377	363	self.assertIn(ssl.HAS_SNI, {True, False})
378	364	self.assertIn(ssl.HAS_ECDH, {True, False})
379	365	ssl.OP_NO_SSLv2
380	366	ssl.OP_NO_SSLv3
381	367	ssl.OP_NO_TLSv1
382	368	ssl.OP_NO_TLSv1_3
383	-	if ssl.OPENSSL_VERSION_INFO >= (1, 0, 1):
384	-	ssl.OP_NO_TLSv1_1
385	-	ssl.OP_NO_TLSv1_2
369	+	ssl.OP_NO_TLSv1_1
370	+	ssl.OP_NO_TLSv1_2
386	371	self.assertEqual(ssl.PROTOCOL_TLS, ssl.PROTOCOL_SSLv23)
387	372	
388	373	def test_private_init(self):
		@@ -1161,7 +1146,6 @@ def test_python_ciphers(self):
1161	1146	self.assertNotIn("RC4", name)
1162	1147	self.assertNotIn("3DES", name)
1163	1148	
1164	-	@unittest.skipIf(ssl.OPENSSL_VERSION_INFO < (1, 0, 2, 0, 0), 'OpenSSL too old')
1165	1149	def test_get_ciphers(self):
1166	1150	ctx = ssl.SSLContext(ssl.PROTOCOL_TLS_CLIENT)
1167	1151	ctx.set_ciphers('AESGCM')
		@@ -1181,15 +1165,11 @@ def test_options(self):
1181	1165	self.assertEqual(default, ctx.options)

```

1182 1166         ctx.options |= ssl.OP_NO_TLSv1
1183 1167         self.assertEqual(default | ssl.OP_NO_TLSv1, ctx.options)
1184 -         if can_clear_options():
1185 -             ctx.options = (ctx.options & ~ssl.OP_NO_TLSv1)
1186 -             self.assertEqual(default, ctx.options)
1187 -             ctx.options = 0
1188 -             # Ubuntu has OP_NO_SSLv3 forced on by default
1189 -             self.assertEqual(0, ctx.options & ~ssl.OP_NO_SSLv3)
1190 -         else:
1191 -             with self.assertRaises(ValueError):
1192 -                 ctx.options = 0
1168 +         ctx.options = (ctx.options & ~ssl.OP_NO_TLSv1)
1169 +         self.assertEqual(default, ctx.options)
1170 +         ctx.options = 0
1171 +         # Ubuntu has OP_NO_SSLv3 forced on by default
1172 +         self.assertEqual(0, ctx.options & ~ssl.OP_NO_SSLv3)
1193 1173
1194 1174         def test_verify_mode_protocol(self):
1195 1175             ctx = ssl.SSLContext(ssl.PROTOCOL_TLS)
1196 1176
1197 1177         @@ -1327,8 +1307,6 @@ def test_security_level(self):
1327 1307             }
1328 1308             self.assertIn(ctx.security_level, security_level_range)
1329 1309
1330 -         @unittest.skipUnless(have_verify_flags(),
1331 -                             "verify_flags need OpenSSL > 0.9.8")
1332 1310         def test_verify_flags(self):
1333 1311             ctx = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
1334 1312             # default value
1335 1313
1336 1314         @@ -1797,7 +1775,6 @@ class MySSLObject(ssl.SSLObject):
1797 1775             obj = ctx.wrap_bio(ssl.MemoryBIO(), ssl.MemoryBIO())
1798 1776             self.assertIsInstance(obj, MySSLObject)
1799 1777
1800 -         @unittest.skipUnless(IS_OPENSSL_1_1_1, "Test requires OpenSSL 1.1.1")
1801 1778         def test_num_tickets(self):
1802 1779             ctx = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
1803 1780             self.assertEqual(ctx.num_tickets, 2)
1804 1781
1805 1782         @@ -2956,8 +2933,6 @@ def test_getpeercert(self):

```

2956	2933	after = ssl.cert_time_to_seconds(cert['notAfter'])
2957	2934	self.assertLess(before, after)
2958	2935	
2959	-	@unittest.skipUnless(have_verify_flags(),
2960	-	"verify_flags need OpenSSL > 0.9.8")
2961	2936	def test_crl_check(self):
2962	2937	if support.verbose:
2963	2938	sys.stdout.write("\n")
		@@ -3859,12 +3834,7 @@ def test_version_basic(self):
3859	3834	self.assertIs(s.version(), None)
3860	3835	self.assertIs(s._sslobj, None)
3861	3836	s.connect((HOST, server.port))
3862	-	if IS_OPENSSL_1_1_1 and has_tls_version('TLSv1.3'):
3863	-	self.assertEqual(s.version(), 'TLSv1.3')
3864	-	elif ssl.OPENSSL_VERSION_INFO >= (1, 0, 2):
3865	-	self.assertEqual(s.version(), 'TLSv1.2')
3866	-	else: # 0.9.8 to 1.0.1
3867	-	self.assertIn(s.version(), ('TLSv1', 'TLSv1.2'))
	3837	+ self.assertEqual(s.version(), 'TLSv1.3')
3868	3838	self.assertIs(s._sslobj, None)
3869	3839	self.assertIs(s.version(), None)
3870	3840	
		@@ -3966,8 +3936,6 @@ def test_default_ecdh_curve(self):
3966	3936	# explicitly using the 'ECCdraft' cipher alias. Otherwise,
3967	3937	# our default cipher list should prefer ECDH-based ciphers
3968	3938	# automatically.
3969	-	if ssl.OPENSSL_VERSION_INFO < (1, 0, 0):
3970	-	context.set_ciphers("ECCdraft:ECDH")
3971	3939	with ThreadedEchoServer(context=context) as server:
3972	3940	with context.wrap_socket(socket.socket()) as s:
3973	3941	s.connect((HOST, server.port))
		@@ -4099,15 +4067,11 @@ def test_ecdh_curve(self):
4099	4067	server_context.set_ciphers("ECDHE:!eNULL:!aNULL")
4100	4068	server_context.options  = ssl.OP_NO_TLSv1   ssl.OP_NO_TLSv1_1
4101	4069	try:
4102	-	stats = server_params_test(client_context, server_context,
4103	-	chatty=True, connectionchatty=True,

```

4104 -             sni_name=hostname)
4105 +         server_params_test(client_context, server_context,
4106 +                             chatty=True, connectionchatty=True,
4107 +                             sni_name=hostname)
4108 -     except ssl.SSLError:
4109 -         pass
4110 -         else:
4111 -             # OpenSSL 1.0.2 does not fail although it should.
4112 -             if IS_OPENSSL_1_1_0:
4113 -                 self.fail("mismatch curve did not fail")
4114 +         self.fail("mismatch curve did not fail")
4115 -
4116 -     def test_selected_alpn_protocol(self):
4117 -         # selected_alpn_protocol() is None unless ALPN is used.
4118 -         @@ -4117,7 +4081,6 @@ def test_selected_alpn_protocol(self):
4119 -             sni_name=hostname)
4120 -             self.assertIs(stats['client_alpn_protocol'], None)
4121 -
4122 -     @unittest.skipUnless(ssl.HAS_ALPN, "ALPN support required")
4123 -     def test_selected_alpn_protocol_if_server_uses_alpn(self):
4124 -         # selected_alpn_protocol() is None unless ALPN is used by the client.
4125 -         client_context, server_context, hostname = testing_context()
4126 -         @@ -4127,7 +4090,6 @@ def
4127 -             test_selected_alpn_protocol_if_server_uses_alpn(self):
4128 -                 sni_name=hostname)
4129 -                 self.assertIs(stats['client_alpn_protocol'], None)
4130 -
4131 -     @unittest.skipUnless(ssl.HAS_ALPN, "ALPN support needed for this test")
4132 -     def test_alpn_protocols(self):
4133 -         server_protocols = ['foo', 'bar', 'milkshake']
4134 -         protocol_tests = [
4135 -             @@ -4150,22 +4112,17 @@ def test_alpn_protocols(self):
4136 -                 except ssl.SSLError as e:
4137 -                     stats = e
4138 -
4139 -                 if (expected is None and IS_OPENSSL_1_1_0
4140 -                     and ssl.OPENSSL_VERSION_INFO < (1, 1, 0, 6)):
4141 -                     # OpenSSL 1.1.0 to 1.1.0e raises handshake error
4142 -                     self.assertIsInstance(stats, ssl.SSLError)
4143 -                 else:

```

```

4158 -         msg = "failed trying %s (s) and %s (c).\n" \
4159 -             "was expecting %s, but got %s from the %s" \
4160 -             % (str(server_protocols), str(client_protocols),
4161 -               str(expected))
4162 -         client_result = stats['client_alpn_protocol']
4163 -         self.assertEqual(client_result, expected,
4164 -                          msg % (client_result, "client"))
4165 -         server_result = stats['server_alpn_protocols'][-1] \
4166 -             if len(stats['server_alpn_protocols']) else 'nothing'
4167 -         self.assertEqual(server_result, expected,
4168 -                          msg % (server_result, "server"))

```

```

4115 +         msg = "failed trying %s (s) and %s (c).\n" \
4116 +             "was expecting %s, but got %s from the %s" \
4117 +             % (str(server_protocols), str(client_protocols),
4118 +               str(expected))
4119 +         client_result = stats['client_alpn_protocol']
4120 +         self.assertEqual(client_result, expected,
4121 +                          msg % (client_result, "client"))
4122 +         server_result = stats['server_alpn_protocols'][-1] \
4123 +             if len(stats['server_alpn_protocols']) else 'nothing'
4124 +         self.assertEqual(server_result, expected,
4125 +                          msg % (server_result, "server"))

```

4169 4126

4170 4127 `def test_selected_npn_protocol(self):`4171 4128 `# selected_npn_protocol() is None unless NPN is used`

```
@@ -4175,31 +4132,8 @@ def test_selected_npn_protocol(self):
```

4175 4132 `sni_name=hostname)`4176 4133 `self.assertIs(stats['client_npn_protocol'], None)`

4177 4134

4178 - `@unittest.skipUnless(ssl.HAS_NPN, "NPN support needed for this test")`4179 4135 `def test_npn_protocols(self):`4180 - `server_protocols = ['http/1.1', 'spdy/2']`4181 - `protocol_tests = [`4182 - `(['http/1.1', 'spdy/2'], 'http/1.1'),`4183 - `(['spdy/2', 'http/1.1'], 'http/1.1'),`4184 - `(['spdy/2', 'test'], 'spdy/2'),`4185 - `(['abc', 'def'], 'abc')`4186 - `]`4187 - `for client_protocols, expected in protocol_tests:`4188 - `client_context, server_context, hostname = testing_context()`

```

4189 - server_context.set_npn_protocols(server_protocols)
4190 - client_context.set_npn_protocols(client_protocols)
4191 - stats = server_params_test(client_context, server_context,
4192 -                             chatty=True, connectionchatty=True,
4193 -                             sni_name=hostname)
4194 - msg = "failed trying %s (s) and %s (c).\n" \
4195 -       "was expecting %s, but got %s from the %s" \
4196 -       % (str(server_protocols), str(client_protocols),
4197 -          str(expected))
4198 - client_result = stats['client_npn_protocol']
4199 - self.assertEqual(client_result, expected, msg % (client_result,
4200 - "client"))
4201 - server_result = stats['server_npn_protocols'][-1] \
4202 -               if len(stats['server_npn_protocols']) else 'nothing'
4203 - self.assertEqual(server_result, expected, msg % (server_result,
4204 - "server"))
4205 + assert not ssl.HAS_NPN
4206
4207 4203
4208 4204
4209 4205
4210 4206
4211 4207
4212 4208
4213 4209
4214 4210
4215 4211
4216 4212
4217 4213
4218 4214
4219 4215
4220 4216
4221 4217
4222 4218
4223 4219
4224 4220
4225 4221
4226 4222
4227 4223
4228 4224
4229 4225
4230 4226
4231 4227
4232 4228
4233 4229
4234 4230
4235 4231
4236 4232
4237 4233
4238 4234
4239 4235
4240 4236
4241 4237
4242 4238
4243 4239
4244 4240
4245 4241
4246 4242
4247 4243
4248 4244
4249 4245
4250 4246
4251 4247
4252 4248
4253 4249
4254 4250
4255 4251
4256 4252
4257 4253
4258 4254
4259 4255
4260 4256
4261 4257
4262 4258
4263 4259
4264 4260
4265 4261
4266 4262
4267 4263
4268 4264
4269 4265
4270 4266
4271 4267
4272 4268
4273 4269
4274 4270
4275 4271
4276 4272
4277 4273
4278 4274
4279 4275
4280 4276
4281 4277
4282 4278
4283 4279
4284 4280
4285 4281
4286 4282
4287 4283
4288 4284
4289 4285
4290 4286
4291 4287
4292 4288
4293 4289
4294 4290
4295 4291
4296 4292
4297 4293
4298 4294
4299 4295
4300 4296
4301 4297
4302 4298
4303 4299
4304 4300
4305 4301
4306 4302
4307 4303
4308 4304
4309 4305
4310 4306
4311 4307
4312 4308
4313 4309
4314 4310
4315 4311
4316 4312
4317 4313
4318 4314
4319 4315
4320 4316
4321 4317
4322 4318
4323 4319
4324 4320
4325 4321
4326 4322
4327 4323
4328 4324
4329 4325
4330 4326
4331 4327
4332 4328
4333 4329
4334 4330
4335 4331
4336 4332
4337 4333
4338 4334
4339 4335
4340 4336
4341 4337
4342 4338
4343 4339
4344 4340
4345 4341
4346 4342
4347 4343
4348 4344
4349 4345
4350 4346
4351 4347
4352 4348
4353 4349
4354 4350
4355 4351
4356 4352
4357 4353
4358 4354
4359 4355
4360 4356
4361 4357
4362 4358
4363 4359
4364 4360
4365 4361
4366 4362
4367 4363
4368 4364
4369 4365
4370 4366
4371 4367
4372 4368
4373 4369
4374 4370
4375 4371
4376 4372
4377 4373
4378 4374
4379 4375
4380 4376
4381 4377
4382 4378
4383 4379
4384 4380
4385 4381
4386 4382
4387 4383
4388 4384
4389 4385
4390 4386
4391 4387
4392 4388
4393 4389
4394 4390
4395 4391
4396 4392
4397 4393
4398 4394
4399 4395
4400 4396
4401 4397
4402 4398
4403 4399
4404 4400
4405 4401
4406 4402
4407 4403
4408 4404
4409 4405
4410 4406
4411 4407
4412 4408
4413 4409
4414 4410
4415 4411
4416 4412
4417 4413
4418 4414
4419 4415
4420 4416
4421 4417
4422 4418
4423 4419
4424 4420
4425 4421
4426 4422
4427 4423
4428 4424
4429 4425
4430 4426
4431 4427
4432 4428
4433 4429
4434 4430
4435 4431
4436 4432
4437 4433
4438 4434
4439 4435
4440 4436
4441 4437
4442 4438
4443 4439
4444 4440
4445 4441
4446 4442
4447 4443
4448 4444
4449 4445
4450 4446
4451 4447
4452 4448
4453 4449
4454 4450
4455 4451
4456 4452
4457 4453
4458 4454
4459 4455
4460 4456
4461 4457
4462 4458
4463 4459
4464 4460
4465 4461
4466 4462
4467 4463
4468 4464
4469 4465
4470 4466
4471 4467
4472 4468
4473 4469
4474 4470
4475 4471
4476 4472
4477 4473
4478 4474
4479 4475
4480 4476
4481 4477
4482 4478
4483 4479
4484 4480
4485 4481
4486 4482
4487 4483
4488 4484
4489 4485
4490 4486
4491 4487
4492 4488
4493 4489
4494 4490
4495 4491
4496 4492
4497 4493
4498 4494
4499 4495
4500 4496
4501 4497
4502 4498
4503 4499
4504 4500
4505 4501
4506 4502
4507 4503
4508 4504
4509 4505
4510 4506
4511 4507
4512 4508
4513 4509
4514 4510
4515 4511
4516 4512
4517 4513
4518 4514
4519 4515
4520 4516
4521 4517
4522 4518
4523 4519
4524 4520
4525 4521
4526 4522
4527 4523
4528 4524
4529 4525
4530 4526
4531 4527
4532 4528
4533 4529
4534 4530
4535 4531
4536 4532
4537 4533
4538 4534
4539 4535
4540 4536
4541 4537
4542 4538
4543 4539
4544 4540
4545 4541
4546 4542
4547 4543
4548 4544
4549 4545
4550 4546
4551 4547
4552 4548
4553 4549
4554 4550
4555 4551
4556 4552
4557 4553
4558 4554
4559 4555
4560 4556
4561 4557
4562 4558
4563 4559
4564 4560
4565 4561
4566 4562
4567 4563
4568 4564
4569 4565
4570 4566
4571 4567
4572 4568
4573 4569
4574 4570
4575 4571
4576 4572
4577 4573
4578 4574
4579 4575
4580 4576
4581 4577
4582 4578
4583 4579
4584 4580
4585 4581
4586 4582
4587 4583
4588 4584
4589 4585
4590 4586
4591 4587
4592 4588
4593 4589
4594 4590
4595 4591
4596 4592
4597 4593
4598 4594
4599 4595
4600 4596
4601 4597
4602 4598
4603 4599
4604 4600
4605 4601
4606 4602
4607 4603
4608 4604
4609 4605
4610 4606
4611 4607
4612 4608
4613 4609
4614 4610
4615 4611
4616 4612
4617 4613
4618 4614
4619 4615
4620 4616
4621 4617
4622 4618
4623 4619
4624 4620
4625 4621
4626 4622
4627 4623
4628 4624
4629 4625
4630 4626
4631 4627
4632 4628
4633 4629
4634 4630
4635 4631
4636 4632
4637 4633
4638 4634
4639 4635
4640 4636
4641 4637
4642 4638
4643 4639
4644 4640
4645 4641
4646 4642
4647 4643
4648 4644
4649 4645
4650 4646
4651 4647
4652 4648
4653 4649
4654 4650
4655 4651
4656 4652
4657 4653
4658 4654
4659 4655
4660 4656
4661 4657
4662 4658
4663 4659
4664 4660
4665 4661
4666 4662
4667 4663
4668 4664
4669 4665
4670 4666
4671 4667
4672 4668
4673 4669
4674 4670
4675 4671
4676 4672
4677 4673
4678 4674
4679 4675
4680 4676
4681 4677
4682 4678
4683 4679
4684 4680
4685 4681
4686 4682
4687 4683
4688 4684
4689 4685
4690 4686
4691 4687
4692 4688
4693 4689
4694 4690
4695 4691
4696 4692
4697 4693
4698 4694
4699 4695
4700 4696
4701 4697
4702 4698
4703 4699
4704 4700
4705 4701
4706 4702
4707 4703
4708 4704
4709 4705
4710 4706
4711 4707
4712 4708
4713 4709
4714 4710
4715 4711
4716 4712
4717 4713
4718 4714
4719 4715
4720 4716
4721 4717
4722 4718
4723 4719
4724 4720
4725 4721
4726 4722
4727 4723
4728 4724
4729 4725
4730 4726
4731 4727
4732 4728
4733 4729
4734 4730
4735 4731
4736 4732
4737 4733
4738 4734
4739 4735
4740 4736
4741 4737
4742 4738
4743 4739
4744 4740
4745 4741
4746 4742
4747 4743
4748 4744
4749 4745
4750 4746
4751 4747
4752 4748
4753 4749
4754 4750
4755 4751
4756 4752
4757 4753
4758 4754
4759 4755
4760 4756
4761 4757
4762 4758
4763 4759
4764 4760
4765 4761
4766 4762
4767 4763
4768 4764
4769 4765
4770 4766
4771 4767
4772 4768
4773 4769
4774 4770
4775 4771
4776 4772
4777 4773
4778 4774
4779 4775
4780 4776
4781 4777
4782 4778
4783 4779
4784 4780
4785 4781
4786 4782
4787 4783
4788 4784
4789 4785
4790 4786
4791 4787
4792 4788
4793 4789
4794 4790
4795 4791
4796 4792
4797 4793
4798 4794
4799 4795
4800 4796
4801 4797
4802 4798
4803 4799
4804 4800
4805 4801
4806 4802
4807 4803
4808 4804
4809 4805
4810 4806
4811 4807
4812 4808
4813 4809
4814 4810
4815 4811
4816 4812
4817 4813
4818 4814
4819 4815
4820 4816
4821 4817
4822 4818
4823 4819
4824 4820
4825 4821
4826 4822
4827 4823
4828 4824
4829 4825
4830 4826
4831 4827
4832 4828
4833 4829
4834 4830
4835 4831
4836 4832
4837 4833
4838 4834
4839 4835
4840 4836
4841 4837
4842 4838
4843 4839
4844 4840
4845 4841
4846 4842
4847 4843
4848 4844
4849 4845
4850 4846
4851 4847
4852 4848
4853 4849
4854 4850
4855 4851
4856 4852
4857 4853
4858 4854
4859 4855
4860 4856
4861 4857
4862 4858
4863 4859
4864 4860
4865 4861
4866 4862
4867 4863
4868 4864
4869 4865
4870 4866
4871 4867
4872 4868
4873 4869
4874 4870
4875 4871
4876 4872
4877 4873
4878 4874
4879 4875
4880 4876
4881 4877
4882 4878
4883 4879
4884 4880
4885 4881
4886 4882
4887 4883
4888 4884
4889 4885
4890 4886
4891 4887
4892 4888
4893 4889
4894 4890
4895 4891
4896 4892
4897 4893
4898 4894
4899 4895
4900 4896
4901 4897
4902 4898
4903 4899
4904 4900
4905 4901
4906 4902
4907 4903
4908 4904
4909 4905
4910 4906
4911 4907
4912 4908
4913 4909
4914 4910
4915 4911
4916 4912
4917 4913
4918 4914
4919 4915
4920 4916
4921 4917
4922 4918
4923 4919
4924 4920
4925 4921
4926 4922
4927 4923
4928 4924
4929 4925
4930 4926
4931 4927
4932 4928
4933 4929
4934 4930
4935 4931
4936 4932
4937 4933
4938 4934
4939 4935
4940 4936
4941 4937
4942 4938
4943 4939
4944 4940
4945 4941
4946 4942
4947 4943
4948 4944
4949 4945
4950 4946
4951 4947
4952 4948
4953 4949
4954 4950
4955 4951
4956 4952
4957 4953
4958 4954
4959 4955
4960 4956
4961 4957
4962 4958
4963 4959
4964 4960
4965 4961
4966 4962
4967 4963
4968 4964
4969 4965
4970 4966
4971 4967
4972 4968
4973 4969
4974 4970
4975 4971
4976 4972
4977 4973
4978 4974
4979 4975
4980 4976
4981 4977
4982 4978
4983 4979
4984 4980
4985 4981
4986 4982
4987 4983
4988 4984
4989 4985
4990 4986
4991 4987
4992 4988
4993 4989
4994 4990
4995 4991
4996 4992
4997 4993
4998 4994
4999 4995
5000 4996
5001 4997
5002 4998
5003 4999
5004 5000
5005 5001
5006 5002
5007 5003
5008 5004
5009 5005
5010 5006
5011 5007
5012 5008
5013 5009
5014 5010
5015 5011
5016 5012
5017 5013
5018 5014
5019 5015
5020 5016
5021 5017
5022 5018
5023 5019
5024 5020
5025 5021
5026 5022
5027 5023
5028 5024
5029 5025
5030 5026
5031 5027
5032 5028
5033 5029
5034 5030
5035 5031
5036 5032
5037 5033
5038 5034
5039 5035
5040 5036
5041 5037
5042 5038
5043 5039
5044 5040
5045 5041
5046 5042
5047 5043
5048 5044
5049 5045
5050 5046
5051 5047
5052 5048
5053 5049
5054 5050
5055 5051
5056 5052
5057 5053
5058 5054
5059 5055
5060 5056
5061 5057
5062 5058
5063 5059
5064 5060
5065 5061
5066 5062
5067 5063
5068 5064
5069 5065
5070 5066
5071 5067
5072 5068
5073 5069
5074 5070
5075 5071
5076 5072
5077 5073
5078 5074
5079 5075
5080 5076
5081 5077
5082 5078
5083 5079
5084 5080
5085 5081
5086 5082
5087 5083
5088 5084
5089 5085
5090 5086
5091 5087
5092 5088
5093 5089
5094 5090
5095 5091
5096 5092
5097 5093
5098 5094
5099 5095
5100 5096
5101 5097
5102 5098
5103 5099
5104 5100
5105 5101
5106 5102
5107 5103
5108 5104
5109 5105
5110 5106
5111 5107
5112 5108
5113 5109
5114 5110
5115 5111
5116 5112
5117 5113
5118 5114
5119 5115
5120 5116
5121 5117
5122 5118
5123 5119
5124 5120
5125 5121
5126 5122
5127 5123
5128 5124
5129 5125
5130 5126
5131 5127
5132 5128
5133 5129
5134 5130
5135 5131
5136 5132
5137 5133
5138 5134
5139 5135
5140 5136
5141 5137
5142 5138
5143 5139
5144 5140
5145 5141
5146 5142
5147 5143
5148 5144
5149 5145
5150 5146
5151 5147
5152 5148
5153 5149
5154 5150
5155 5151
5156 5152
5157 5153
5158 5154
5159 5155
5160 5156
5161 5157
5162 5158
5163 5159
5164 5160
5165 5161
5166 5162
5167 5163
5168 5164
5169 5165
5170 5166
5171 5167
5172 5168
5173 5169
5174 5170
5175 5171
5176 5172
5177 5173
5178 5174
5179 5175
5180 5176
5181 5177
5182 5178
5183 5179
5184 5180
5185 5181
5186 5182
5187 5183
5188 5184
5189 5185
5190 5186
5191 5187
5192 5188
5193 5189
5194 5190
5195 5191
5196 5192
5197 5193
5198 5194
5199 5195
5200 5196
5201 5197
5202 5198
5203 5199
5204 5200
5205 5201
5206 5202
5207 5203
5208 5204
5209 5205
5210 5206
5211 5207
5212 5208
5213 5209
5214 5210
5215 5211
5216 5212
5217 5213
5218 5214
5219 5215
5220 5216
5221 5217
5222 5218
5223 5219
5224 5220
5225 5221
5226 5222
5227 5223
5228 5224
5229 5225
5230 5226
5231 5227
5232 5228
5233 5229
5234 5230
5235 5231
5236 5232
5237 5233
5238 5234
5239 5235
5240 5236
5241 5237
5242 5238
5243 5239
5244 5240
5245 5241
5246 5242
5247 5243
5248 5244
5249 5245
5250 5246
5251 5247
5252 5248
5253 5249
5254 5250
5255 5251
5256 5252
5257 5253
5258 5254
5259 5255
5260 5256
5261 5257
5262 5258
5263 5259
5264 5260
5265 5261
5266 5262
5267 5263
5268 5264
5269 5265
5270 5266
5271 5267
5272 5268
5273 5269
5274 5270
5275 5271
5276 5272
5277 5273
5278 5274
5279 5275
5280 5276
5281 5277
5282 5278
5283 5279
5284 5280
5285 5281
5286 5282
5287 5283
5288 5284
5289 5285
5290 5286
5291 5287
5292 5288
5293 5289
5294 5290
5295 5291
5296 5292
5297 5293
5298 5294
5299 5295
5300 5296
5301 5297
5302 5298
5303 5299
5304 5300
5305 5301
5306 5302
5307 5303
5308 5304
5309 5305
5310 5306
5311 5307
5312 5308
5313 5309
5314 5310
5315 5311
5316 5312
5317 5313
5318 5314
5319 5315
5320 5316
5321 5317
5322 5318
5323 5319
5324 5320
5325 5321
5326 5322
5327 5323
5328 5324
5329 5325
5330 5326
5331 5327
5332 5328
5333 5329
5334 5330
5335 5331
5336 5332
5337 5333
5338 5334
5339 5335
5340 5336
5341 5337
5342 5338
5343 5339
5344 5340
5345 5341
5346 5342
5347 5343
5348 5344
5349 5345
5350 5346
5351 5347
5352 5348
5353 5349
5354 5350
5355 5351
5356 5352
5357 5353
5358 5354
5359 5355
5360 5356
5361 5357
5362 5358
5363 5359
5364 5360
5365 5361
5366 5362
5367 5363
5368 5364
5369 5365
5370 5366
5371 5367
5372 5368
5373 5369
5374 5370
5375 5371
5376 5372
5377 5373
5378 5374
5379 5375
5380 5376
5381 5377
5382 5378
5383 5379
5384 5380
5385 5381
5386 5382
5387 5383
5388 5384
5389 5385
5390 5386
5391 5387
5392 5388
5393 5389
5394 5390
5395 5391
5396 5392
5397 5393
5398 5394
5399 5395
5400 5396
```

```
207 207  #_socket socketmodule.c
208 208
209 209  # Socket module helper for SSL support; you must comment out the other
210 - # socket line above, and possibly edit the SSL variable:
211 - #SSL=/usr/local/ssl
212 - #_ssl _ssl.c \
213 - # -DUSE_SSL -I$(SSL)/include -I$(SSL)/include/openssl \
214 - # -L$(SSL)/lib -lssl -lcrypto

210 + # socket line above, and edit the OPENSSL variable:
211 + # OPENSSL=/path/to/openssl/directory
212 + # _ssl _ssl.c \
213 + # -I$(OPENSSL)/include -L$(OPENSSL)/lib \
214 + # -lssl -lcrypto
215 + #_hashlib _hashopenssl.c \
216 + # -I$(OPENSSL)/include -L$(OPENSSL)/lib \
217 + # -lcrypto
218 +
219 + # To statically link OpenSSL:
220 + # _ssl _ssl.c \
221 + # -I$(OPENSSL)/include -L$(OPENSSL)/lib \
222 + # -l:libssl.a -Wl,--exclude-libs,libssl.a \
223 + # -l:libcrypto.a -Wl,--exclude-libs,libcrypto.a
224 + #_hashlib _hashopenssl.c \
225 + # -I$(OPENSSL)/include -L$(OPENSSL)/lib \
226 + # -l:libcrypto.a -Wl,--exclude-libs,libcrypto.a

215 227
216 228  # The crypt module is now disabled by default because it breaks builds
217 229  # on many systems (where -lcrypt is needed), e.g. Linux (I believe).
```



Comments 0