Packet Sender Documentation

Packet Sender is an open source utility to allow sending and receiving TCP, UDP, and SSL (encrypted TCP) packets. The mainline branch officially supports Windows, Mac, and Desktop Linux (with Qt). Other places may recompile and redistribute Packet Sender. Packet Sender is free and licensed GPL v2 or later. It can be used for both commercial and personal use.

Uses

- Controlling network-based devices in ways beyond their original apps
- Test automation (using its command line tool and/or hotkeys)
- Testing network APIs (using the built-in TCP, UDP, SSL clients)
- Malware analysis (using the built-in UDP, TCP, SSL servers)
- Troubleshooting secure connections (using SSL).
- Testing network connectivity/firewalls (by having 2 Packet Senders talk to each other)
- Tech support (by sending customers a portable Packet Sender with pre-defined settings and packets)
- Sharing/Saving/Collaboration using the Packet Sender Cloud (https://cloud.packetsender.com/) service.

Packet Sender Mobile

The mobile apps are open source at this repo: https://github.com/dannagle/Packet_Sender_Mobile (https://github.com/dannagle/Packet_Sender_Mobile)

Packet Sender Mobile binaries for iOS and Android may be purchased for a small fee on the Apple App Store (https://itunes.apple.com/us/app/packet-sender-mobile/id1338199854?ls=1&mt=8) and the Google Play Store (https://play.google.com/store/apps/details?id=com.packetsender.mobile). Packet Sender Mobile contains bare minimum permissions, no ads, no analytics, no "Review me!" nags, no email list sign-ups, no notifications, and no IAPs. It is absolutely nothing but the app. This is old-fashioned software development that respects the user. Please show your support.

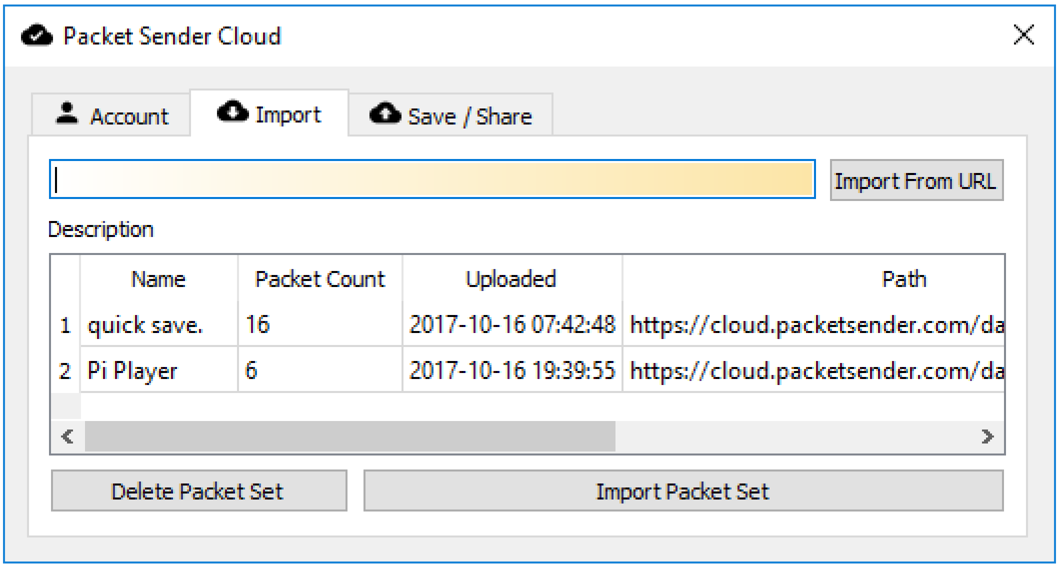
The GitHub project for Packet Sender Mobile is located at https://github.com/dannagle/Packet_Sender_Mobile (https://github.com/dannagle/Packet_Sender_Mobile)

Packet Sender Cloud

Packets sets can be quickly saved/retrieved/shared using the free Packet Sender Cloud (https://cloud.packetsender.com/) service. The cloud may also be used to publicly display and distribute your packets (via an URL) for collaboration, tutorials, end users, etc. Packet Sender may import public packet sets with public URL.

There are various reasons to do this:

- Keeping all your packets ready so you can quickly retrieve them when installing a fresh Packet Sender
- Quickly swapping between packet sets when working on different projects.
- Sharing a login (it is allowed) for collaborative packet set generation
- Having a public page of your packet sets so others can quickly find and import



	Name	Packet Count	Uploaded	Path
1	quick save.	16	2017-10-16 07:42:48	https://cloud.packetsender.com/da
2	Pi Player	6	2017-10-16 19:39:55	https://cloud.packetsender.com/da

If you are publishing a network API, maintaining a public cloud page is significantly easier than painfully detailing (IP, port, type, etc) the packets to your users. Plus, updating that page is easy.

More information about it can be found at https://cloud.packetsender.com/help (https://cloud.packetsender.com/help)

Support

- Twitter: @NagleCode (http://twitter.com/NagleCode)

- You may also track this project on GitHub (http://github.com/dannagle/PacketSender).
- Email: hello AT packetsender DOT com
- Connect with me on LinkedIn (https://www.linkedin.com/in/dannagle/)

Portable Mode

Packet Sender has a "portable" mode. At launch, it will look for `packets.ini` and `ps_settings.ini` in its run-time directory. For the SSL server, it will look for `ps.key` and `ps.pem`.

Windows users, this directory is the same place as the .exe. For Mac users, this run-time directory is at `PacketSender.app/Contents/MacOS`. If INI files are found, it will use them instead of `%APPDATA%` or `Library/Application Support`.

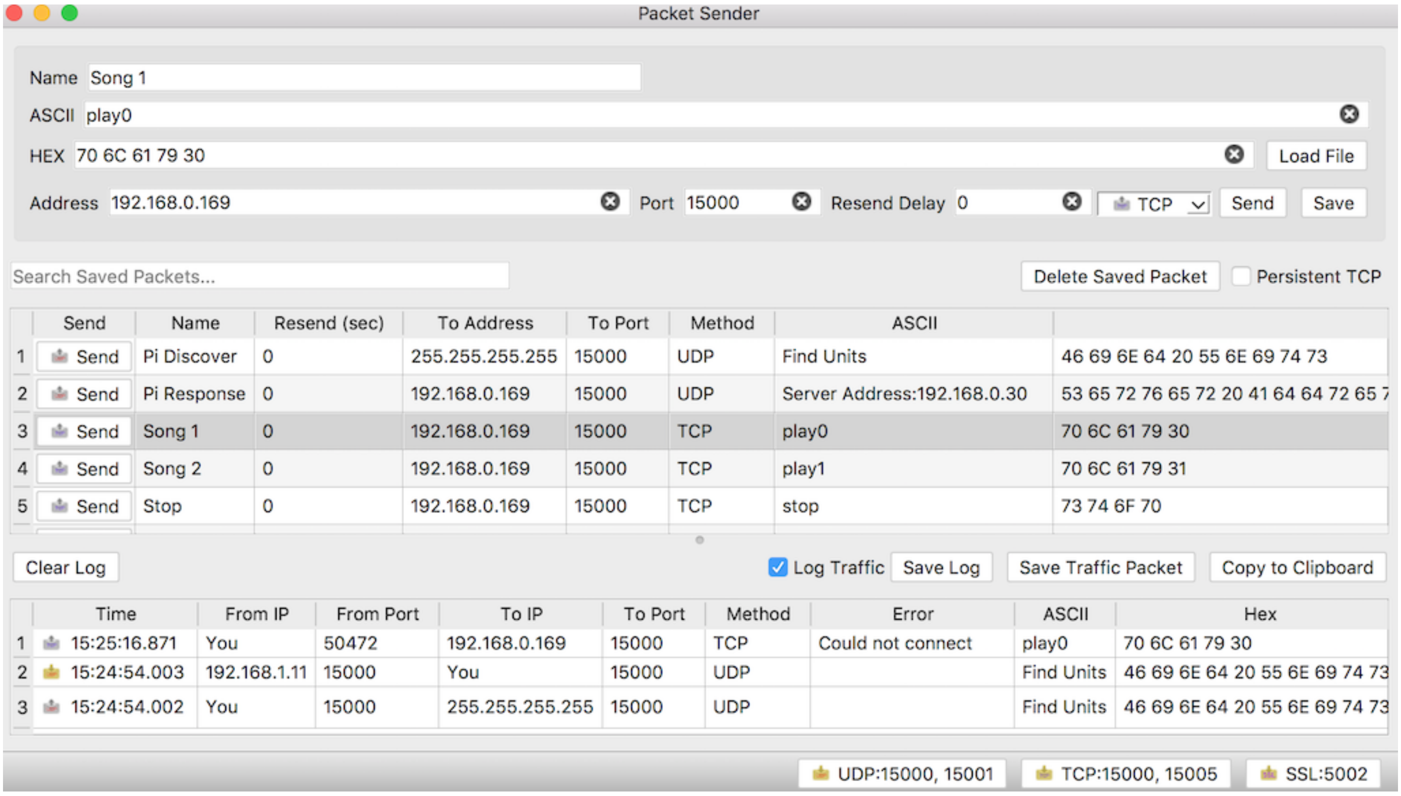
IPv4 and IPv6 Support

Packet Sender's built-in servers are configured to support IPv4 and IPv6. The most recent version of Packet Sender is smart enough to switch between the two modes. Older versions Packet Sender, mostly found on Linux's older Qt SDK libraries, may render a toggle switch between IPv4 and IPv6. Click to switch between the two.

Packet Sender's GUI client sender and CLI will detect the type of IP address being used and bind appropriately to send the packet. Note that you must include a scope ID to send.

Documentation (GUI)

Packet Sender is identical for all the desktop versions. The only difference is its theme to match the operating system.



Packet Sender

Name Song 1

ASCII play0

HEX 70 6C 61 79 30

Address 192.168.0.169 Port 15000 Resend Delay 0 TCP

	Send	Name	Resend (sec)	To Address	To Port	Method	ASCII	
1	Send	Pi Discover	0	255.255.255.255	15000	UDP	Find Units	46 69 6E 64 20 55 6E 69 74 73
2	Send	Pi Response	0	192.168.0.169	15000	UDP	Server Address:192.168.0.30	53 65 72 76 65 72 20 41 64 64 72 65 7
3	Send	Song 1	0	192.168.0.169	15000	TCP	play0	70 6C 61 79 30
4	Send	Song 2	0	192.168.0.169	15000	TCP	play1	70 6C 61 79 31
5	Send	Stop	0	192.168.0.169	15000	TCP	stop	73 74 6F 70

	Time	From IP	From Port	To IP	To Port	Method	Error	ASCII	Hex
1	15:25:16.871	You	50472	192.168.0.169	15000	TCP	Could not connect	play0	70 6C 61 79 30
2	15:24:54.003	192.168.1.11	15000	You	15000	UDP		Find Units	46 69 6E 64 20 55 6E 69 74 73
3	15:24:54.002	You	15000	255.255.255.255	15000	UDP		Find Units	46 69 6E 64 20 55 6E 69 74 73

- In the bottom right, there are UDP, TCP, and SSL server status and port(s). You can click to activate or deactivate these. Packet Sender supports binding to any number of ports.
- During packet resending, there will be a button to cancel all resends.
- Please check your firewall. Windows aggressively blocks TCP-based servers. Packet Sender will still work if the firewall blocks it, but it can't receive unsolicited TCP-based packets.
- In the table, there is a list of saved packets. You can double-click to directly edit fields in this table.
- Select multiple packets to enable "Multi-Send". Selected packets are shown in a quick-send area above the traffic log.
- Fields can be rearranged by drag-and-drop in the settings menu.
- A resend value of "0" means it is a single-shot packet.
- A packet has a name, destination address (domain names are default resolved just before sending), port, and data associated with it.
- Click "Send" to immediately send. Click "Save" to send later.
- For IPv6 sending, you will also need the scope ID.
- Packet Sender supports mixed ASCII and HEX notation:
  - \XX gets translated to XX in hex
  - \n, \r, \t will get translated to 0A, 0D, and 09

  - HEX numbers are space delimited
  - The HEX field will attempt to interpret other common delimiters (such as commas, colons (Wireshark), semicolons, " 0x", etc) and auto-correct.
  - A single stream of HEX is also supported. If the number of bytes is odd, Packet Sender will assume the front byte needs a zero and then auto-correct.
  - Example ASCII: hello world\r
  - Example HEX: 68 65 6c 6c 6f 20 77 6f 72 6c 64 0d
  - You may save a packet directly from the traffic log. You will be prompted for a name, and the source address and port will be switched for your convenience.
  - You may also load a file directly in to the HEX field. The HEX field supports sending up to 10,922 bytes. The theoretical limit for sending via the command line is 200 MB.
- An optional response can be sent. The same response is used for TCP and UDP.

Hotkeys / Keyboard Shortcuts

The fields at the top can be navigated using CTRL+1, CTRL+2, etc, up to CTRL+8 (send button). On Mac, the shortcut key is Command.

Some notes:

- When you navigate to the TCP/UDP/SSL option, you may use up/down or t/u/s characters.
- If you are going to automate with hotkeys (using tools such at AutoHotKey), you may want to turn off "Restore previous session".

SSL Client and Server

Packet Sender supports establishing encrypted connections over SSL. This is supported in the GUI and on the command line. Some notes on this:

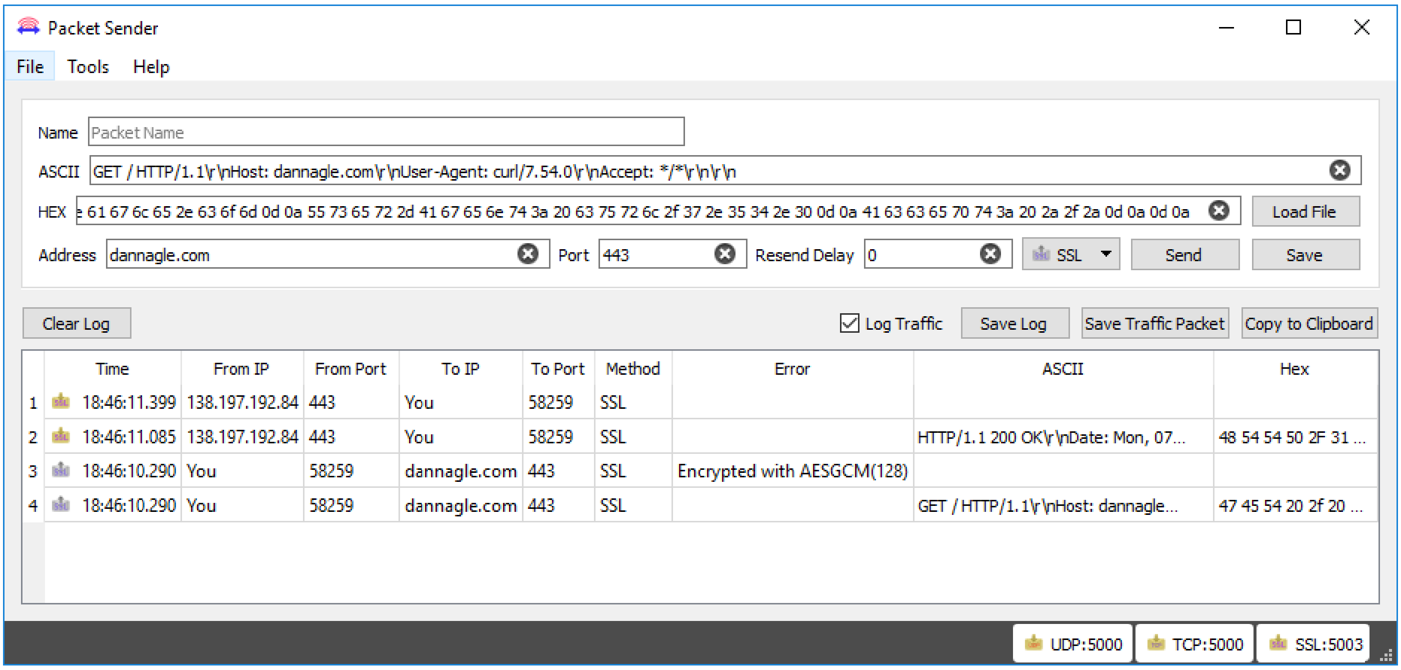
- The certificate negotiation is handled immediately during connection.
- By default, Packet Sender ignores all SSL errors (expired cert, wrong host name, self-signed, etc).
- Packet Sender outputs the cert negotiation progress in to the traffic log.
- Packet Sender outputs the encryption algorithm used in the traffic log (such as AES 128).

Packet Sender bundles an internal "Snake Oil" certificate for use as a server for all supported platforms. The certificate and key is in the same place as packet and settings.

- Overriding the cert locations in Settings also overrides these snake-oil certificate.

Packet Sender bundles OpenSSL for use in Windows. On Mac and Linux, Packet Sender will use the native SSL libraries.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/ (http://www.openssl.org/))



Name: Packet Name

ASCII: GET / HTTP/1.1\r\nHost: dannagle.com\r\nUser-Agent: curl/7.54.0\r\nAccept: */*\r\n\r\n

HEX: e 61 67 6c 65 2e 63 6f 6d 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c 2f 37 2e 35 34 2e 30 0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 0d 0a

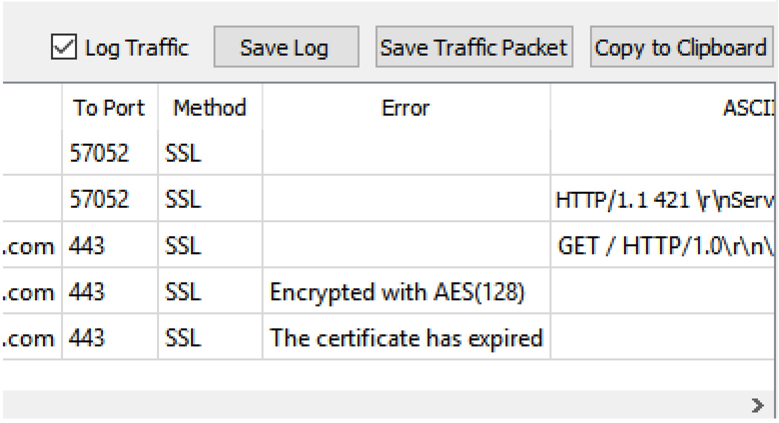
Address: dannagle.com Port: 443 Resend Delay: 0 SSL

☑ Log Traffic

	Time	From IP	From Port	To IP	To Port	Method	Error	ASCII	Hex
1	18:46:11.399	138.197.192.84	443	You	58259	SSL			
2	18:46:11.085	138.197.192.84	443	You	58259	SSL		HTTP/1.1 200 OK\r\nDate: Mon, 07...	48 54 54 50 2F 31 ...
3	18:46:10.290	You	58259	dannagle.com	443	SSL	Encrypted with AESGCM(128)		
4	18:46:10.290	You	58259	dannagle.com	443	SSL		GET / HTTP/1.1\r\nHost: dannagle...	47 45 54 20 2f 20 ...

UDP:5000 TCP:5000 SSL:5003

- If there is an SSL error, Packet Sender will output it to the traffic log. If the setting is to continue anyway (default), it will continue to negotiate encryption. Otherwise, the connection ends with failure to connect.

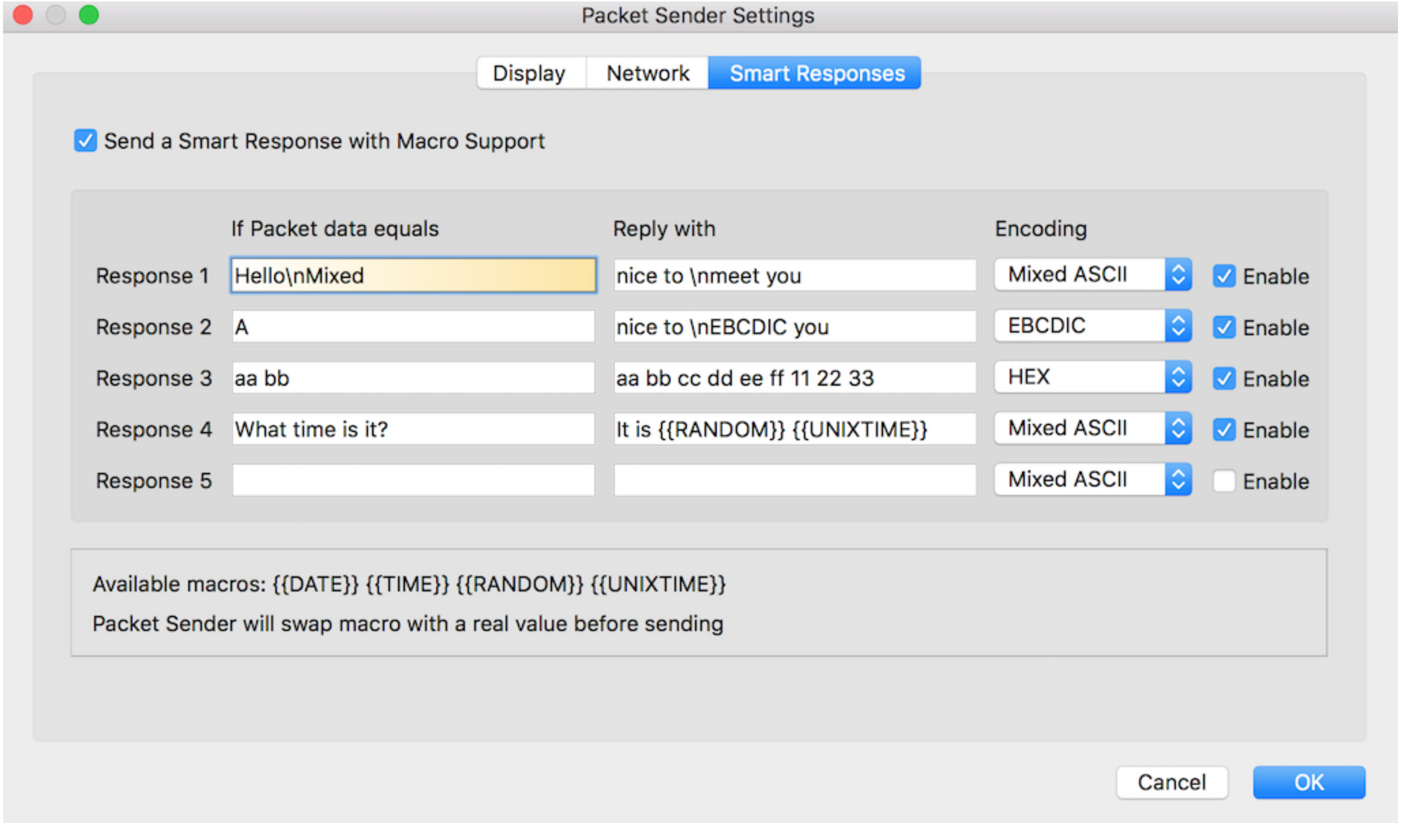


☑ Log Traffic

	To Port	Method	Error	ASCI
	57052	SSL		
	57052	SSL		HTTP/1.1 421 \r\nServ
.com	443	SSL		GET / HTTP/1.0\r\n\
.com	443	SSL	Encrypted with AES(128)	
.com	443	SSL	The certificate has expired	

Smart Responses

Packet Sender supports up to 5 smart responses. To use it, disable normal responses.



☑ Send a Smart Response with Macro Support

	If Packet data equals	Reply with	Encoding	
Response 1	Hello\nMixed	nice to \nmeet you	Mixed ASCII	☑ Enable
Response 2	A	nice to \nEBCDIC you	EBCDIC	☑ Enable
Response 3	aa bb	aa bb cc dd ee ff 11 22 33	HEX	☑ Enable
Response 4	What time is it?	It is {{RANDOM}} {{UNIXTIME}}	Mixed ASCII	☑ Enable
Response 5			Mixed ASCII	□ Enable

Available macros: {{DATE}} {{TIME}} {{RANDOM}} {{UNIXTIME}}
Packet Sender will swap macro with a real value before sending

- Packet Sender will compare the packet within the encoding you choose.
- Packet Sender translates the encoding before sending out the reply.
- The available encodings are:
    - Mixed ASCII -- The standard Packet Sender way of encoding ASCII along with non-printable characters
    - HEX -- Packet Sender's normal HEX encoding
    - EBCDIC (https://en.wikipedia.org/wiki/EBCDIC) -- An encoding used mostly by IBM mainframes. The input field is normal Mixed ASCII and is translated when performing the comparison and sending.

Macros

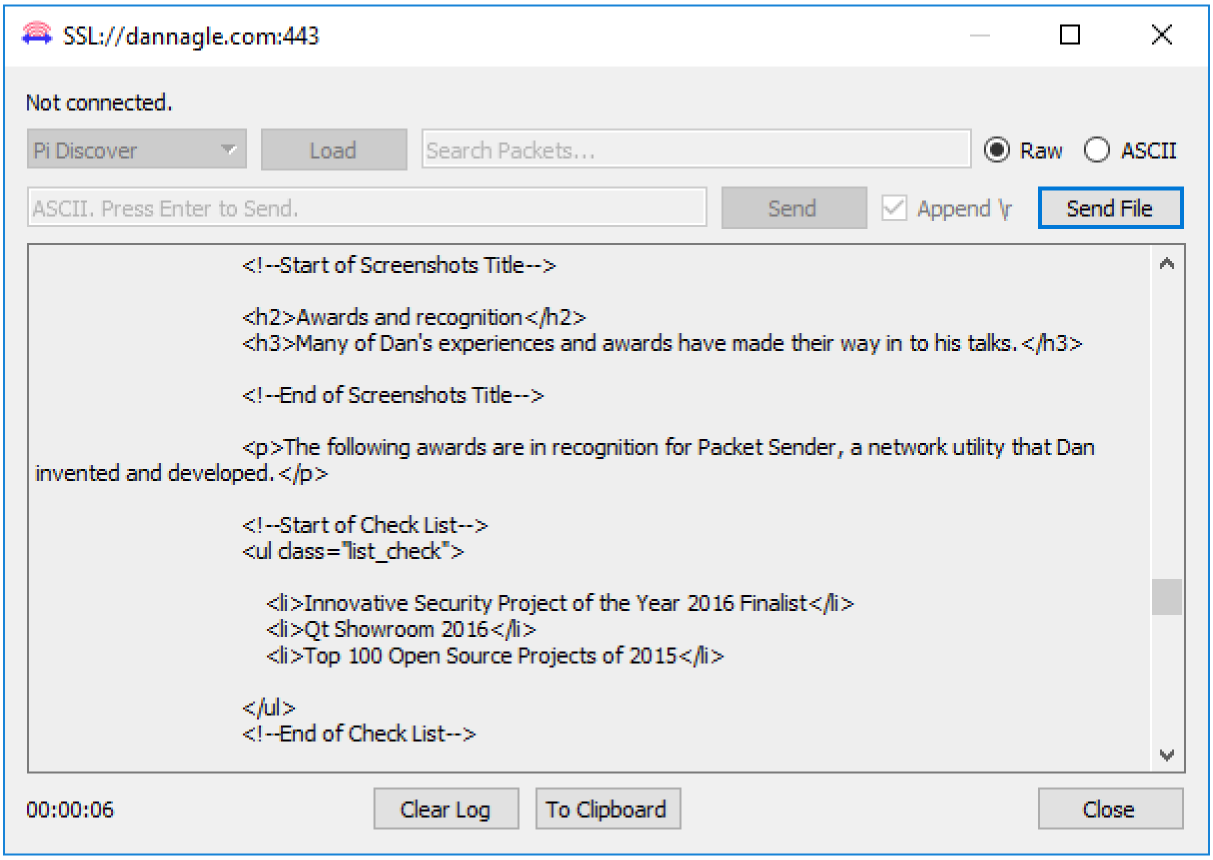
Packet Sender supports these macros when sending responses:

- {{DATE}} -- Sends the current date in "yyy-mm-dd" format.
- {{TIME}} -- Sends the current time in "hh:mm:ss ap" format.
- {{UNIXTIME}} -- Sends the current epoch time stamp.
- {{RANDOM}} -- Sends a random number ranging from either 0 to 32767 or 2147483647, depending on 32-bit or 64-bit (default installer for Windows is 32-bit. Mac is 64-bit).

- {{UNIQUE}} -- Sends a random string. Uses an internal UUID to generate it. Packet Sender will swap the macro with real values before sending.

Persistent TCP and SSL

Packet Sender supports persistent TCP and SSL connections via a separate UI dialog. It is enabled by checkbox on the main window or in the settings dialog.



Not connected.

```
                        <!--Start of Screenshots Title-->

                        <h2>Awards and recognition</h2>
                        <h3>Many of Dan's experiences and awards have made their way in to his talks.</h3>

                        <!--End of Screenshots Title-->

                        <p>The following awards are in recognition for Packet Sender, a network utility that Dan
invented and developed.</p>

                        <!--Start of Check List-->
                        <ul class="list_check">

                           <li>Innovative Security Project of the Year 2016 Finalist</li>
                           <li>Qt Showroom 2016</li>
                           <li>Top 100 Open Source Projects of 2015</li>

                        </ul>
                        <!--End of Check List-->
```

00:00:06

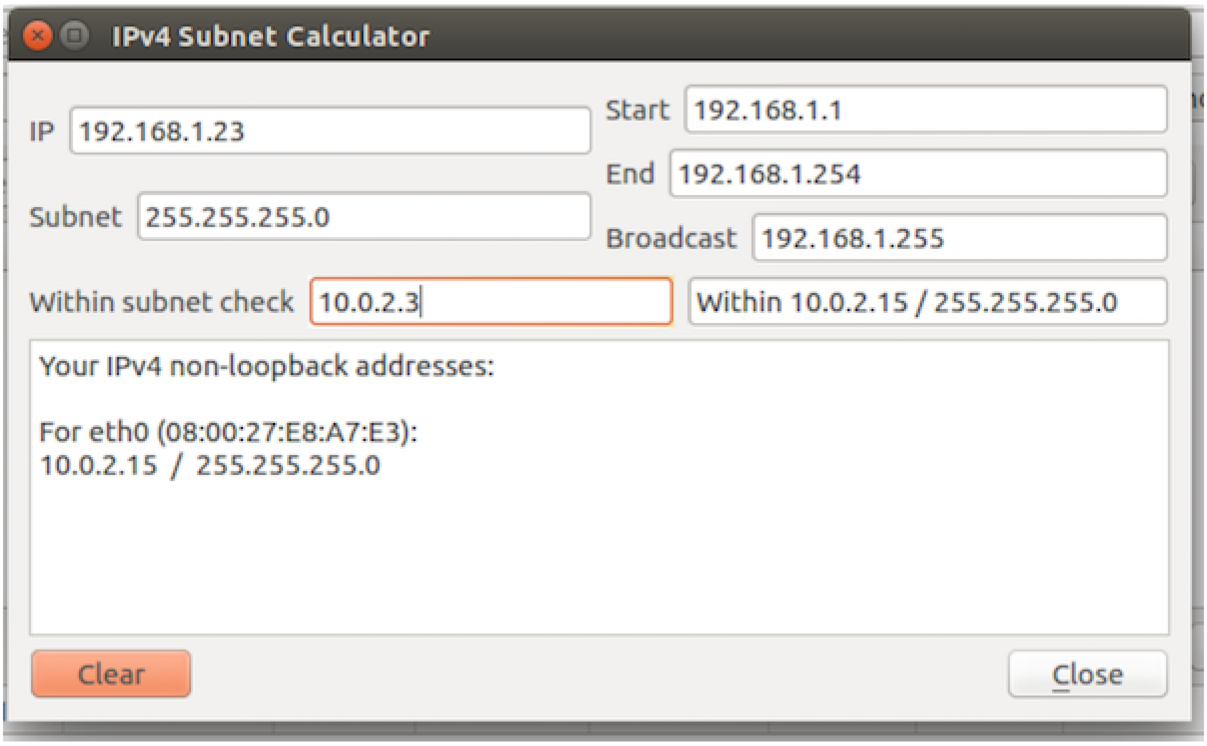
- Any number of persistent connections can be created.
- Previously saved packets can be loaded in the drop-down.
- There is a "raw" view and "ASCII" view. The ASCII view is useful to troubleshoot data that is not printed by the raw view.
- Traffic is also saved in the main window traffic log.
- A file may be uploaded to the persistent connection. You may wish to turn off logging if you use this.
- The timer in the bottom lefts starts as soon as a valid data packet is sent/received. It stops when the connection is closed.
- You may optionally append a carriage return when you quick-send by hitting the return key. This is useful for command-prompt menus over TCP / SSL connections.
- Incoming persistent connections to the server will launch the UI dialog.

- During resend, the persistent connection packet is carried over to the new UI dialog. Clicking "Resending(1)" will cancel it.

Persistent connections are not supported via the command line.

IPv4 Subnet Calculator

Packet Sender has a built-in subnet calculator. It is under the Tools menu.



IP 192.168.1.23

Subnet 255.255.255.0

Start 192.168.1.1

End 192.168.1.254

Broadcast 192.168.1.255

Within subnet check 10.0.2.3

Within 10.0.2.15 / 255.255.255.0

```
Your IPv4 non-loopback addresses:

For eth0 (08:00:27:E8:A7:E3):
10.0.2.15 / 255.255.255.0
```

- The log window will display non-loopback IPv4 and IPv6 addresses found on your computer.
- The top left field allow inputting an IPv4 address.
- The field below is for a subnet, either X.X.X.X or /XX notation.
- The results of the calculation are on the right.
- The field below is a quick check to see if an IPv4 is within one of your subnets.

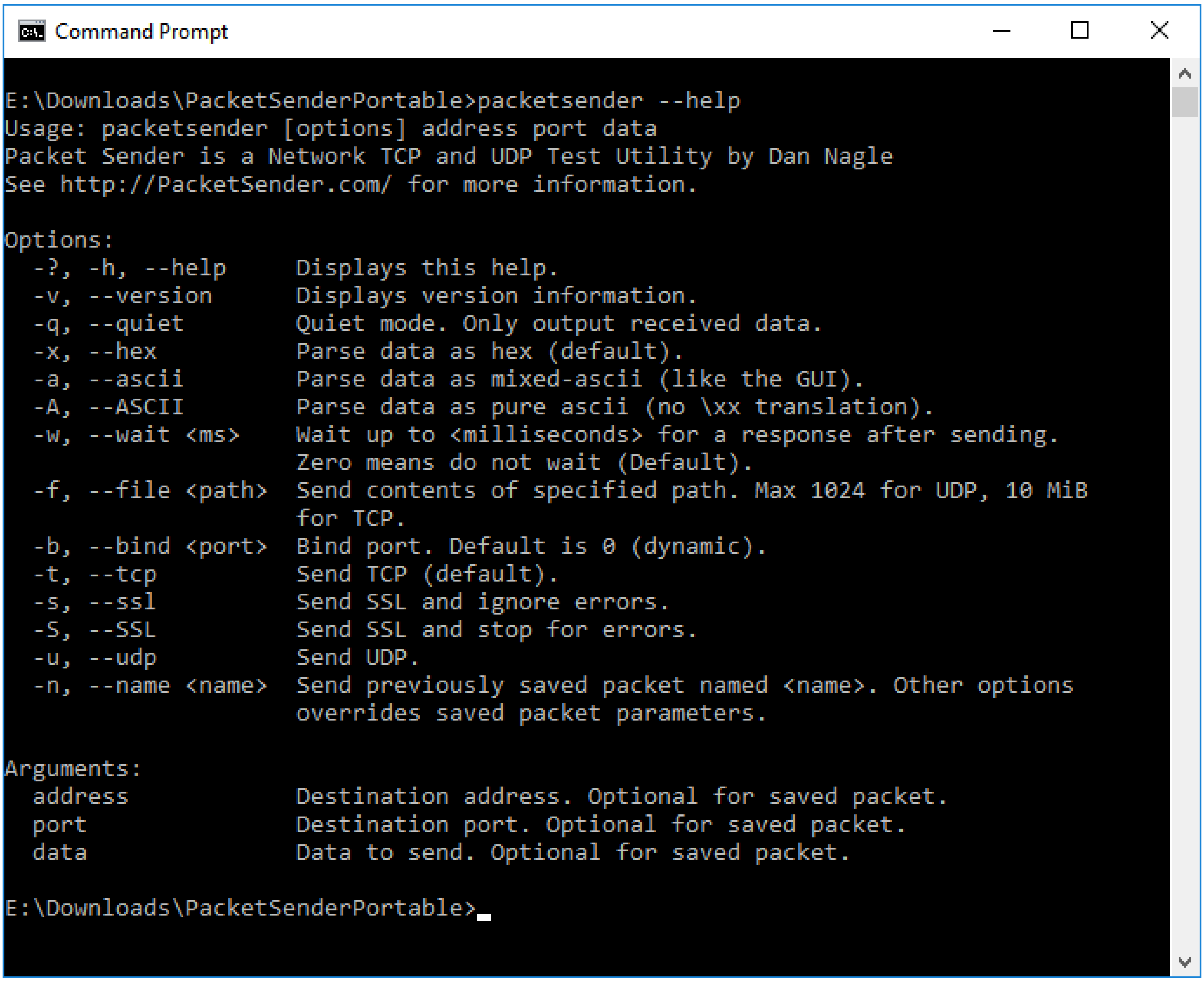
Additional Configuration Options

- IPv4 Mode or IPv6 Mode in the servers. This is identical to the toggle switch.
- The traffic log and packet table is divided by a draggable splitter. This splitter can also be collapsed on either side.
- Copy to the clipboard the raw packet data (default). If your data has a large amount of non-ASCII characters, you may prefer a translated version.
- Resending can be auto-cancelled after X number of packets. Set to 0 to resend forever.

- Traffic log can be set to roll at 100 entries.
- Import/Export of packets is available via menus.
- Attempt receive before send (some servers expect this behavior).
- 500 ms delay before sending data (some servers are slow).
- Enable keyboard shortcut for ASCII --> EBCDIC translation.
- Resolve DNS during input. The default is to resolve DNS just before sending.
- Ignore SSL errors. Packet Sender will note the error and continue with encryption. Otherwise, Packet Sender abandons the connection. The SSL server always ignores errors.

Documentation (Command Line)

The command line extension used in Windows installations is .com. Using .exe will launch the GUI. Leave off the extension and Windows will choose the correct program. The same executable controls the command line and GUI for Mac and Linux operating systems.



```
E:\Downloads\PacketSenderPortable>packetsender --help
Usage: packetsender [options] address port data
Packet Sender is a Network TCP and UDP Test Utility by Dan Nagle
See http://PacketSender.com/ for more information.

Options:
  -?, -h, --help     Displays this help.
  -v, --version      Displays version information.
  -q, --quiet        Quiet mode. Only output received data.
  -x, --hex          Parse data as hex (default).
  -a, --ascii        Parse data as mixed-ascii (like the GUI).
  -A, --ASCII        Parse data as pure ascii (no \xx translation).
  -w, --wait <ms>    Wait up to <milliseconds> for a response after sending.
                     Zero means do not wait (Default).
  -f, --file <path>  Send contents of specified path. Max 1024 for UDP, 10 MiB
                     for TCP.
  -b, --bind <port>  Bind port. Default is 0 (dynamic).
  -t, --tcp          Send TCP (default).
  -s, --ssl          Send SSL and ignore errors.
  -S, --SSL          Send SSL and stop for errors.
  -u, --udp          Send UDP.
  -n, --name <name>  Send previously saved packet named <name>. Other options
                     overrides saved packet parameters.

Arguments:
  address            Destination address. Optional for saved packet.
  port               Destination port. Optional for saved packet.
  data               Data to send. Optional for saved packet.

E:\Downloads\PacketSenderPortable>_
```

The command line system in Packet Sender follows the same pattern as other Linux utilities. It has a long name (such as --version) and a short name (such as -v). These options can be arranged in any order and Packet Sender will parse them correctly. The last 3 options are positional and must appear last. They are IP, port, and data. These last options are optional if using a stored packet.

```
packetsender --help
Usage: packetsender [options] address port data
Packet Sender is a Network UDP/TCP/SSL Test Utility by NagleCode
See https://PacketSender.com/ for more information.

Options:
-?, -h, --help  Displays this help.
-v, --version   Displays version information.
-q, --quiet     Quiet mode. Only output received data.
-x, --hex   Parse data-to-send as hex (default).
-a, --ascii     Parse data-to-send as mixed-ascii (like the GUI).
-A, --ASCII     Parse data-to-send as pure ascii (no \xx translation).
-w, --wait <ms> Wait up to <milliseconds> for a response after sending. Zero means do
 not wait (Default).
-f, --file <path>   Send contents of specified path. Max 1024 for UDP, 100 MiB for TC
P.
-b, --bind <port>   Bind port. Default is dynamic.
-t, --tcp   Send TCP (default).
-s, --ssl   Send SSL and ignore errors.
-S, --SSL   Send SSL and stop for errors.
-u, --udp   Send UDP.
-n, --name <name>   Send previously saved packet named <name>. Other options overrides
 saved packet parameters.

-Arguments:
-address    Destination address. Optional for saved packet.
-port   Destination port. Optional for saved packet.
-data   Data to send. Optional for saved packet.
```

Example CLI

```
packetsender -taw 500 packetsender.com 22 "Hello\nWorld"
TCP (56620)://192.185.38.130 48 65 6c 6c 6f 0a 57 6f 72 6c 64
Response Time:5:51:37.042 pm
Response HEX:53 53 48 2D 32 2E 30 2D 4F 70 65 6E 53 53 48 5F 35 2E 33 70 31 20 44 65 6
2 69 61 6E 2D 33 75 62 75 6E 74 75 33 2E 31 2E 49 53 2E 31 30 2E 30 34 0D 0A
Response ASCII:SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu3.1.IS.10.04\r\n
```

Example CLI using SSL and ignoring errors

The command line has the option to ignore or abandon on SSL errors. The default is to ignore.

```
packetsender -saw 500 expired.packetsender.com 443 "GET / HTTP/1.0\r\n\r\n"
SSL Error: The certificate has expired
SSL (54202)://expired.packetsender.com:443 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 30 0
d 0a 0d 0a
Cipher: Encrypted with AES(128)

Response Time:3:24:55.695 pm
Response HEX:48 54 54 50 2f 31 2e 31 20 34 32 31 20 0d 0a 53 65 72 76 65 72 3a 20 6e 6
7 69 6e 78 2f 31 2e 31 30 2e 30 20 28 55 62 75 6e 74 75 29 0d
Response ASCII:HTTP/1.1 421 \r\nServer: nginx/1.10.0 (Ubuntu)\r
```

Building Packet Sender

The only dependency is Qt SDK. Here is how to build the app.

Build for Windows and Mac

1. Download the Qt installer from http://www.qt.io/download-open-source/ (http://www.qt.io/download-open-source/)
2. Let it install MingGW if you don't have a compiler.
3. Open the project PacketSender.pro
4. Build! Qt is the only dependency!

The Windows and Mac versions were built using Qt 5.10

Build for Linux

Here is the sequence of commands for Ubuntu 15.04. Please adapt to your Linux platform. Packet Sender requires no additional libraries beyond the stock Qt SDK. I have been told there are build issues with stock Fedora. If a Fedora wizard can get it to properly compile, please let me know, and I'll add your instructions.

If you are feeling adventurous, feel free to build from the master branch. It contains the latest stable build. The development branch should probably be avoided.

```
sudo apt-get update
sudo apt-get install qt5-default build-essential
wget https://github.com/dannagle/PacketSender/archive/(Version).tar.gz
tar -xzvf (Version).tar.gz
cd PacketSender-(Version)/src
qmake
make
```

To run use:

```
./PacketSender
```

If it doesn't run, you may need to set it executable

```
chmod a+x PacketSender
```

The Linux release has an Ubuntu build script that will make Packet Sender globally launchable with just `packetsender` .

Enhancements

Missing a feature? You can hire me to add it to Packet Sender (https://packetsender.com/enhancements).

License

GPL v2 or Later. Contact me (https://packetsender.com/contact) if you require a different license.

Copyright

Packet Sender was written by Dan Nagle (https://dannagle.com/) and is published by © NagleCode, LLC - @NagleCode (https://twitter.com/NagleCode) - PacketSender.com (https://PacketSender.com)

LEGAL

Read the License (/LICENSE.txt)

Vector-based Logo (/packetsender.svg)

SUPPORT

Documentation (/documentation)

File a Ticket (https://github.com/dannagle/PacketSender/issues)

Stack Overflow (https://stackoverflow.com/search?q=packet+sender)

FUNDING

Donate (Thank You!) (http://dannagle.com/PayPal)

Hire Me (/enhancements)

Sponsorships (/sponsors)

GUIDES

TCP vs UDP (https://en.wikibooks.org/wiki/Communication_Networks/TCP_and_UDP_Protocols)

The OSI Model (https://en.wikipedia.org/wiki/OSI_model)

Wikipedia (https://en.wikipedia.org/wiki/Packet_Sender)

Home (/) Enhancements (/enhancements) Me (https://dannagle.com/)
GitHub (https://github.com/dannagle/PacketSender) Contact (https://packetsender.com/contact)