SiLK Acceptance Tests
(SiLK-2.5.0)

CERT Network Situational Awareness

June 28, 2012

1 Introduction

SiLK, the System for Internet-Level Knowledge, is a collection of traffic analysis tools to facilitate security analysis of large networks. The SiLK tool suite supports the efficient collection, storage and analysis of network flow data, enabling network security analysts to rapidly query large historical traffic data sets.

The tools in SiLK suite can be grouped into two categories:

- The packing tools are responsible for collecting flow records, converting them to the SiLK format, categorizing them, and storing them in the data repository.
- The analysis tools read SiLK flow records from the data repository and can display, sort, or group the flow records by various attributes and compute the flow volume of each group.

This document describes the testing procedures used to verify that the tools in the SiLK suite are implemented correctly and work as advertised.

1.1 Structure of this document

This document begins with a general description of SiLK and of some conventions used in the tests.

The remainder of the document consists of the tests themselves, broken down by functional area. Each test is referenced by the requirement it tests, and is broken down into the following sections:

Prerequisites If the test requires some conditions to be satisfied that are outside the scope of the test, they will be mentioned here. This section may not be present if there are no special prerequisites for running the test.

Preparation Steps to conduct prior to the test. Some of these steps may be unnecessary to repeat between tests. If something goes wrong during this phase, the test is considered impossible to run due to error.

Procedure Steps to conduct during the test. These steps should be performed in order each time the test is run.

Expected results The tester should verify that these items occur at the appropriate points in the test procedure. If they do not, the test is considered a failure.

2 Testing SiLK Analysis Tools

The tests for the SiLK Analysis Tools are included with the SiLK-2.5.0 source distribution. The tests are invoked by typing `make check` in an application build directory or at the top of the build tree to run all application tests.

The data used to test is applications is created by a Perl script that generates text. This text is piped into the `rwtuc` application to create the SiLK flow records that are used for the tests.

The tests for an application invoke the application with various combinations of its options. Some tests are used to confirm that the application properly fails, for example, when incorrect or conflicting options are specified. Other tests confirm that the output of the application is correct. The output is assumed to be correct if the MD5 hash of the output matches an expected value. The expected value is determined or verified either by using unrelated SiLK applications or by directly processing the output produced by the Perl script that creates the text that was piped to `rwtuc`.

The tests for an application do not attempt an exhaustive permutation of all options, as that would require an extraordinary amount of time for the tests to complete. Knowledge of the software's source code is used to select options that exercise the majority of the application's functionality. When possible, unrelated options are used simultaneously to check multiple parts of the source code.

2.1 Prerequisites

The tests of the SiLK Analysis Tools require that Perl be installed on the system, and that the Digest::MD5 Perl module is installed.

2.2 Preparation

The tests assume you have configured and built SiLK. The tests in this section use the application binaries as they exist in the build tree. The tests do not require that you install SiLK prior to running the tests.

2.3 Procedure

1. Go to the top of the directory where you built SiLK.
2. Type `make check`.

The full lists of tests that `make check` runs are listed in Section 8.

2.4 Expected results

The tests will take several minutes to run.

During the tests, you may see the following sorts of output.

- The following indicates a test that successfully passed.

```
PASS: tests/rwstats-version.pl
```

- The following indicates a test has failed. Any text immediately preceding the `FAIL` line may indicate why the test failed.

  ```
  FAIL: tests/rwstats-sip24-top-pkt-p2.pl
  ```

- The following indicates that the test was skipped.

  ```
  SKIP: tests/rwcut-icmp-type.pl
  ```

  A test can be skipped for two reasons.

  1. The test is not applicable. For example, there is no need to test IPv6 functionality if SiLK was not compiled with IPv6 support.
  2. A file or application that the test requires is not present. This can occur if you fail to build the SiLK tools prior to testing, so that `make check` is building the tools and testing them. Some tests use other tools in SiLK suite, and the tests will be skipped if the required tools are not available.

- The following indicates that no tests exist for the applications or libraries in the named directory.

  ```
  Making check in libaddrtype
  make[2]: Nothing to be done for ‘check’.
  ```

Once all processing stops, you should see output similar to the following to indicate that all tests passed, and the return status from `make` should be 0.

```
make[2]: Nothing to be done for ‘check’.
make[2]: Nothing to be done for ‘check-am’.
```

Output such as the following indicates that the tests for an application have failed:

```
==================================
6 of 32 tests failed
Please report to netsa-help@cert.org
==================================
make[3]: *** [check-TESTS] Error 1
make[2]: *** [check-am] Error 2
make[1]: *** [check-recursive] Error 1
make: *** [check-recursive] Error 1
```

3 Testing `rwsender` and `rwreceiver`

`rwsender` is a daemon which transfers files over the network to one or more `rwreceiver` daemons. An `rwreceiver` may accept files from multiple `rwsender`s. Either `rwsender` or `rwreceiver` may act as the server and accept connections from `rwreceiver` or `rwsender` processes acting as clients. The connection between `rwsender` and `rwreceiver` may be encrypted using GnuTLS. `rwsender` and `rwreceiver` do not require the files they transfer to have any particular format; they treat the contents of the files as a stream of bytes.

The tests will determine whether `rwsender` can successfully send files to `rwreceiver` processes, and whether an `rwreceiver` can successfully receive files from `rwsender` processes. If SiLK was configured with GnuTLS support, tests will also be conducted using GnuTLS.

There is a Python script, `sendrcv_tests.py`, included with the SiLK-2.5.0 distribution that will run tests on `rwsender` and `rwreceiver`. To run this script, go into the `silk/src/sendrcv` directory and type `make check`. (If you type `make check` at the top of the build tree, the tests will be invoked as `make` recursively descends into each directory.) The test script will run several scenarios that invoke the daemons, have them connect, send files, and shut down. Some of the tests will involve shutting down one side of the connection during file transfer to verify that the other side handles that situation correctly.

3.1 Prerequisites

The tests of `rwsender` and `rwreceiver` require that Python 2.4 or later and Perl 5.6 or later be installed on the system.

3.2 Preparation

The test script assumes you have configured and built `rwsender`, `rwreceiver`, and all the libraries they require. The script uses the application binaries as they exist in the build tree, and the script does not require that you install SiLK prior to running the tests.

During many of the tests, a temporary directory is created, and files and subdirectories are created in this directory. The directory is created in the location specified by the TMPDIR environment variable, or in `/tmp` when the TMPDIR environment variable is not set.

3.3 Procedure

1. Go to `src/sendrcv` subdirectory in the directory tree where you built SiLK.
2. Type `make check`. This will invoke some basic checks on `rwsender` and `rwreceiver` and then invoke the `sendrcv_tests.py` testing script.

The `sendrcv_tests.py` script tests the following behaviors:

1. **Simple connection.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and shut down cleanly.

2. **GnuTLS connection.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection using GnuTLS, and shut down cleanly. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

3. **Stop `rwreceiver` server.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwreceiver` a SIGTERM, causing it to shut down cleanly. Restart `rwreceiver` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

4. **Stop `rwreceiver` server when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

5. **Stop `rwsender` server.** With `rwsender` acting as a server and `rwreceiver` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwsender` a SIGTERM, causing it to shut down cleanly. Restart `rwsender` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

6. **Stop `rwsender` server when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

7. **Stop `rwreceiver` client.** With `rwsender` acting as a server and `rwreceiver` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwreceiver` a SIGTERM, causing it to shut down cleanly. Restart `rwreceiver` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

8. **Stop `rwreceiver` client using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

9. **Stop `rwsender` client.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwsender` a SIGTERM, causing it to shut down cleanly. Restart `rwsender` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

10. **Stop `rwsender` client when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

11. **Kill `rwreceiver` server.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwreceiver` a SIGKILL, causing it to abruptly shut down. Check whether `rwsender` handles the sudden loss of connectivity. Restart `rwreceiver` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

12. **Kill `rwreceiver` server when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

13. **Kill `rwsender` server.** With `rwsender` acting as a server and `rwreceiver` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwsender` a SIGKILL, causing it to abruptly shut down. Check whether `rwreceiver` handles the sudden loss of connectivity. Restart `rwsender` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

14. **Kill `rwsender` server when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

15. **Kill `rwreceiver` client.** With `rwsender` acting as a server and `rwreceiver` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwreceiver` a SIGKILL, causing it to abruptly shut down. Check whether `rwsender` handles the sudden loss of connectivity. Restart `rwreceiver` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

16. **Kill `rwreceiver` client when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

17. **Kill `rwsender` client.** With `rwreceiver` acting as a server and `rwsender` acting as a client, check whether `rwsender` and `rwreceiver` start correctly, establish a connection, and begin to transfer files. During file transfer, send `rwsender` a SIGKILL, causing it to abruptly shut down. Check whether `rwreceiver` handles the sudden loss of connectivity. Restart `rwsender` and verify that the connection is reestablished and that file transfer resumes. Finally, check whether `rwsender` and `rwreceiver` shut down cleanly.

18. **Kill `rwsender` client when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

19. **Multiple connections.** Start two `rwreceiver` processes acting as clients and two `rwsender` processes acting as servers. Check whether each of the `rwreceiver` clients establish a connection with each of the `rwsender` servers. Verify that files from each `rwsender` are sent to each `rwreceiver`. Check whether all four daemons shut down cleanly.

20. **Multiple connections when using GnuTLS.** This test is similar to the previous, except the connections are made with GnuTLS. This test uses certificates that are included in the SiLK source code. This test is skipped if GnuTLS support is not available.

21. **Filtering.** Start two `rwreceiver` processes acting as clients and a single `rwsender` process acting as a server. Check whether each of the `rwreceiver` clients establish a connection with `rwsender`. Use filtering rules on `rwsender` so that a subset of the files are sent to each `rwreceiver`. Verify that the correct files are sent. Check whether all three daemons shut down cleanly.

22. **Post processing.** Start `rwreceiver` acting as a server and `rwsender` acting as a client. Establish a connection and successfully transfer files. For each file, verify that the command specified `rwreceiver`'s `--post-command` switch is executed. Check whether the daemons shut down cleanly.

3.4 Expected results

The script will take about six minutes to complete all the tests. As it runs, it will print the name of each test and its status: `ok` indicates the test was successsful and `FAIL` indicates a failure.

If all tests in the script complete successfully, you will see the following on your screen as the script exits:

```
Ran 22 tests in 373.152s

OK
PASS: tests/sendrcv-daemons.pl
=============
7 test passed
=============
```

Text similar to the following at the end of the run indicates that one or more tests have failed.

```
======================================================================
FAIL: testPostCommand (__main__.TestSenderReceiver)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "tests/sendrcv_tests.py", line 425, in testPostCommand
    ("Post command: Ident: %(sname)s  "
AssertionError

----------------------------------------------------------------------
Ran 22 tests in 373.080s

FAILED (failures=1)
===================================
1 of 7 tests failed
Please report to netsa-help@cert.org
===================================
```

4 Testing `rwflowappend`

The `rwflowappend` daemon is used to support multiple copies of the data store, or to allow the data to be stored on a machine separate from the machine where `rwflowpack` is running. Typically an `rwsender`–`rwreceiver` pair is used to move the data files from `rwflowpack` to `rwflowappend`. For testing purposes, the method used to inject files into `rwflowappend` is immaterial.

The tests for `rwflowappend` are included with the SiLK-2.5.0 source distribution. The tests are invoked by typing `make check` in the `rwflowpack` directory. (If you type `make check` at the top of the build tree, the `rwflowappend` tests will be invoked as `make` recursively descends into each directory.)

The tests are written in a combination of Perl and Python. The tests will confirm that the `rwflowappend` daemon can start, process files, and terminate cleanly. The tests also confirm that `rwflowappend` handles unusual input files correctly.

4.1 Prerequisites

The tests of `rwflowappend` require that the following tools are installed on the system:

- Python 2.4 or later
- Perl 5.6 or later
- the Perl module Digest::MD5

4.2 Preparation

The test scripts assume you have configured and built `rwflowappend` and all the libraries it requires. The scripts use the application binary as it exists in the build tree, and the scripts do not require that you install SiLK prior to running the tests.

During many of the tests, a temporary directory is created, and files and subdirectories are created in this directory. The directory is created in the location specified by the TMPDIR environment variable, or in `/tmp` when the TMPDIR environment variable is not set.

4.3 Procedure

1. Go to `src/rwflowpack` subdirectory in the directory tree where you built SiLK.
2. Type `make check`. This will invoke the tests that check the behaviors of all the applications in the `src/rwflowpack` directory, including `rwflowappend` (as well as `rwguess`, `rwflowpack`, and `rwpackchecker`).

The tests relevant to `rwflowappend` check the following behaviors:

1. **Append IPv4.** Check whether `rwflowappend` properly handles two files that exist in its incoming directory when `rwflowappend` is invoked. `rwflowappend` will create a new hourly data file, and append the second file to that hourly file. Both input files will be moved to the archive directory. When `rwflowappend` receives a signal, it should shut down cleanly. This test uses input files that contain only IPv4 data.
2. **Append IPv6.** This test is similar to the previous, except it uses a data file that contains IPv6 data. This test is only invoked when SiLK has been compiled with IPv6 support.
3. **Post processing.** Check whether `rwflowappend` properly handles the `--hour-file-command` and `--post-command` switches to notice a new hourly file and to process an incoming file after `rwflowappend` has processed it. This test is similar to the "Append IPv4" test; in addition, the `--hour-file-command` will write the name of the hourly file to a text file, and the `--post-command` will copy the incoming files to a separate location. When `rwflowappend` receives a signal, it should shut down cleanly.
4. **Time window.** Check whether `rwflowappend` properly handles the `--reject-hours-past` and `--reject-hours-future` switches. Files containing records with start times before the `--reject-hours-past` or after the `--reject-hours-future` times are stored in the error directory. All other files should appear in the archive directory and corresponding data files should be created. When `rwflowappend` receives a signal, it should shut down cleanly.

5. **Bad input.** Check whether `rwflowappend` properly handles unusual files in its incoming directory. One file is a SiLK data file that contains no records; `rwflowappend` should move this file to the archive directory and not create an hourly data file. The second unusual file is a file that does not contain the SiLK file header. `rwflowappend` should move this file into its error directory.

4.4 Expected results

The tests may take several minutes to run.

During the tests, you may see the following sorts of output.

- The following indicates a test that successfully passed.

  ```
  PASS: tests/rwflowappend-version.pl
  ```

- The following indicates a test has failed. Any text immediately preceding the `FAIL` line may indicate why the test failed.

  ```
  FAIL: tests/rwflowappend-append-ipv4.pl
  ```

- The following indicates that the test was skipped.

  ```
  SKIP: tests/rwflowappend-append-ipv6.pl
  ```

  A test can be skipped for two reasons.

  1. The test is not applicable. For example, there is no need to test IPv6 functionality if SiLK was not compiled with IPv6 support.
  2. A file or application that the test requires is not present. This can occur if you fail to build the SiLK tools prior to testing, so that `make check` is building the tools and testing them. Some tests use other tools in SiLK suite, and the tests will be skipped if the required tools are not available.

Once all processing stops, you should see output similar to the following to indicate that all tests passed (you may or may not get the message about tests being skipped), and the return status from `make` should be 0.

```
======================
All 43 tests passed
(3 tests were not run)
======================
```

Output such as the following indicates that the tests for an application have failed:

```
==================================
6 of 43 tests failed
Please report to netsa-help@cert.org
==================================
make[1]: *** [check-TESTS] Error 1
make: *** [check-am] Error 2
```

5 Testing flowcap

The flowcap daemon listens on user-specified network ports to collect NetFlow v5 and/or IPFIX flow records that are created by flow generators. Examples of flow generators include routers and software that processes packet capture (`libpcap`) data. flowcap converts the flow records into a SiLK format and stores the records in temporary files. These files are later processed by `rwflowpack`. The typical way to transfer files from flowcap to `rwflowpack` is via an `rwsender`–`rwreceiver` pair, though the administrator is free to use other software (such as `scp` or `rsync`).

The tests for flowcap are included with the SiLK-2.5.0 source distribution. The tests are invoked by typing `make check` in the flowcap directory. (If you type `make check` at the top of the build tree, the flowcap tests will be invoked as `make` recursively descends into each directory.)

The tests are written in a combination of Perl and Python. The tests will confirm that the flowcap daemon can start, read data from the network, write the data into files, and terminate cleanly. Verifying that the files produced by flowcap are consistent is sufficient; it is not necessary in these tests to confirm that `rwflowpack` can process the files.

5.1 Prerequisites

The tests of flowcap require that the following tools are installed on the system:

- Python 2.4 or later
- Perl 5.6 or later
- the Perl module Digest::MD5

5.2 Preparation

The test scripts assume you have configured and built flowcap and all the libraries it requires. The scripts use the application binary as it exists in the build tree, and the scripts do not require that you install SiLK prior to running the tests.

During many of the tests, a temporary directory is created, and files and subdirectories are created in this directory. The directory is created in the location specified by the TMPDIR environment variable, or in `/tmp` when the TMPDIR environment variable is not set.

5.3 Procedure

1. Go to `src/flowcap` subdirectory in the directory tree where you built SiLK.
2. Type `make check`. This will invoke the scripts that check the behavior of flowcap.

The flowcap tests check the following behaviors:

1. **Collect NetFlow v5 records.** Check whether flowcap properly starts, accepts NetFlow v5 UDP packets on a UDP port, converts them to SiLK flow records, and shuts down cleanly.

2. **Collect IPFIX flow records.** Check whether flowcap properly starts, accepts IPFIX packets on a TCP port, converts them to SiLK flow records, and shuts down cleanly.

5.4 Expected results

The tests may take several minutes to run.

During the tests, you may see the following sorts of output.

- The following indicates a test that successfully passed.

  ```
  PASS: tests/flowcap-version.pl
  ```

- The following indicates a test has failed. Any text immediately preceding the `FAIL` line may indicate why the test failed.

  ```
  FAIL: tests/flowcap-append-ipv4.pl
  ```

- The following indicates that the test was skipped.

  ```
  SKIP: tests/flowcap-ipfix.pl
  ```

  A test can be skipped for two reasons.

  1. The test is not applicable. For example, there is no need to test IPFIX functionality if SiLK was not compiled with IPFIX support.
  2. A file or application that the test requires is not present. This can occur if you fail to build the SiLK tools prior to testing, so that `make check` is building the tools and testing them. Some tests use other tools in SiLK suite, and the tests will be skipped if the required tools are not available.

Once all processing stops, you should see output similar to the following to indicate that all tests passed, and the return status from `make` should be 0.

```
==================================================
All 5 tests passed
==================================================
```

Output such as the following indicates that the tests for an application have failed:

```
==================================
1 of 5 tests failed
Please report to netsa-help@cert.org
==================================
make[1]: *** [check-TESTS] Error 1
make: *** [check-am] Error 2
```

6 Testing `rwflowpack`

`rwflowpack` is the heart of the SiLK packing system. It may either collect NetFlow v5 and/or IPFIX flow records itself (similar to flowcap), or it may process the following types of files:

- files created by flowcap
- files containing NetFlow v5 PDUs, such as those created by NetFlow Collector
- files generated by the `yaf` program which contain IPFIX flow records
- files containing SiLK flow records generated by other SiLK applications

`rwflowpack` is responsible for deciding how and where each flow record gets written into the data store. `rwflowpack` splits the flow data by hour and chooses a *flowtype* (also called a *class/type* pair) for the record according to "packing logic". The packing logic normally categorizes data as incoming or outgoing, and it chooses an appropriate file format for the data.

There are four input-modes for `rwflowpack`.

- In "stream" input mode, `rwflowpack` opens an input "stream" for every *probe* listed in the `sensor.conf` file. These streams can be network ports where `rwflowpack` will read NetFlow v5 or IPFIX records, or they can directories that are routinely polled for files containing NetFlow v5 PDUs, IPFIX records, or SiLK files.
- In "fcfiles" input mode, `rwflowpack` polls a directory for files created by flowcap. In this mode, the probe definitions in the `sensor.conf` file are ignored, and instead `rwflowpack` uses the probe name written into each file's header.
- In "pdufile" input mode, `rwflowpack` reads NetFlow v5 PDUs from a single file specified on the command line, then `rwflowpack` exits.
- In "respool" input mode, `rwflowpack` does not recategorize the data; instead, `rwflowpack` reads SiLK flow files and puts each record into a flow file using the sensor and class/type values that already exist on the record.

There are two output-modes for `rwflowpack`. In the first, `rwflowpack` writes the data directly to the data store; this is called "local-storage" mode. In the second (called "sending" mode), `rwflowpack` stores the flow records in temporary files, and an `rwflowappend` process is responsible for writing the flow records into the data store. Typically `rwflowpack` and `rwflowappend` are running on separate machines, and an `rwsender`–`rwreceiver` pair is used to transfer the temporary files between the machines.

The tests for `rwflowpack` are included with the SiLK-2.5.0 source distribution. The tests are invoked by typing `make check` in the `rwflowpack` directory. (If you type `make check` at the top of the build tree, the `rwflowpack` tests will be invoked as `make` recursively descends into each directory.)

The tests are written in a combination of Perl and Python. The tests will confirm that the `rwflowpack` daemon can start, process files, and terminate cleanly. The tests also confirm that `rwflowpack` handles unusual input files correctly.

6.1 Prerequisites

The tests of `rwflowpack` require that the following tools are installed on the system:

- Python 2.4 or later
- Perl 5.6 or later
- the Perl module Digest::MD5

6.2 Preparation

The test scripts assume you have configured and built `rwflowpack` and all the libraries it requires. The scripts use the application binary as it exists in the build tree, and the scripts do not require that you install SiLK prior to running the tests.

During many of the tests, a temporary directory is created, and files and subdirectories are created in this directory. The directory is created in the location specified by the TMPDIR environment variable, or in `/tmp` when the TMPDIR environment variable is not set.

6.3 Procedure

1. Go to `src/rwflowpack` subdirectory in the directory tree where you built SiLK.
2. Type `make check`. This will invoke the tests that check the behaviors of all the applications in the `src/rwflowpack` directory, including `rwflowpack` (as well as `rwguess`, `rwflowappend`, and `rwpackchecker`).

The tests relevant to `rwflowpack` check the following behaviors (unless otherwise stated, `rwflowpack` is running in "stream" input mode and "local-storage" output mode):

1. **Sensor configuration.** Verify that `rwflowpack` correctly parses a sensor configuration file that contains both valid and invalid probe and sensor definitions.
2. **Pack SiLK IPv4 file.** Check whether `rwflowpack` starts, uses its directory poller to a file (that was present when `rwflowpack` was started), reads the SiLK flow records from the file, creates files and directoriess in its data directory, moves the incoming file to its archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data.
3. **Pack SiLK IPv6 file.** This test is similar to the previous, except it uses a data file that contains IPv6 data. This tests is only invoked when SiLK has been compiled with IPv6 support.
4. **Directory polling check.** This test is similar to the previous test, except the file is put into the polling directory after `rwflowpack` has started. This ensures that the directory poller works as expected.
5. **Pack IPFIX IPv4 file.** Check whether `rwflowpack` starts, uses its directory poller to find a file, reads the IPFIX records from the file, creates files and directoriess in its data directory, moves the incoming file to its archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data. This test is only invoked when SiLK has been compiled with IPFIX support.
6. **Pack IPFIX IPv6 file.** This test is similar to the previous, except it uses a data file that contains IPv6 data. This test is only invoked when SiLK has been compiled with both IPFIX and IPv6 support.

7. **Pack IPFIX from network.** Check whether `rwflowpack` starts, reads IPFIX records on a TCP socket, creates files and directoriess in its data directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data. This test is only invoked when SiLK has been compiled with IPFIX support.

8. **Pack NetFlow v5 file.** Check whether `rwflowpack` starts, uses its directory poller to find a file, reads the NetFlow v5 PDU records from the file, creates files and directoriess in its data directory, moves the incoming file to its archive directory, and exits cleanly when it receives a signal.

9. **Run in "sending" output-mode.** Check whether `rwflowpack` starts, uses its directory poller to find a file, reads the SiLK flow records from the file, creates files in its sending directory, moves the incoming file to its archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data.

10. **Run in "sending" output-mode.** Check whether `rwflowpack` starts, uses its directory poller to find incoming files, reads the SiLK flow records from the files, creates files in its sending directory, moves the incoming file to its archive directory, invokes a command on the incoming files after moving them to the archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data.

11. **Run in "fcfiles" input-mode.** Check whether `rwflowpack` starts, finds a file in its incoming directory, reads the probe name and flowcap records from the file, creates files and directories in its data directory, moves the incoming file to its archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data.

12. **Run in "respool" input-mode.** Check whether `rwflowpack` starts, uses its directory poller to find incoming files, reads the SiLK flow records from the files, creates files and directories in its data directory based on the sensor and class/type data that exists on the flow records, moves the input files to the archive directory, and exits cleanly. This test uses an input file that contains only IPv4 data.

13. **Run in "pdufile" input-mode.** Check whether `rwflowpack` starts, reads the NetFlow v5 PDUs from a file specified on the command line, creates files and directories in its data directory, moves the PDU file to its archive directory, and exits cleanly.

14. **Packing multiple streams.** Check whether `rwflowpack` properly starts, polls two directories (containing SiLK flow files) and listens on two network ports (collecting NetFlow v5 PDUs), and exits cleanly. The test creates data files for three sensors, where the first sensor contains the data from one poll directory and one network port, the second sensor contains the remaining poll directory, and the third sensor contains the renaming network port.

15. **Packing multiple streams.** Check whether `rwflowpack` properly starts, polls two directories containing SiLK flow files and two other directories containing files of NetFlow v5 PDUs, and exits cleanly. The test creates data files for three sensors, where the first sensor contains the data from one SiLK directory and one NetFlow v5 directory, the second sensor contains the remaining SiLK directory, and the third sensor contains the remaining NetFlow v5 directory.

16. **Discarding flows matching CIDR block.** Check whether `rwflowpack` properly starts, uses its directory poller to a file, reads the SiLK flow records from the file, discards records that have a source or destination IP in a particular CIDR block, creates files and directoriess in its data directory containing the remaining flows, moves the incoming file to its archive directory, and exits cleanly when it receives a signal. This test uses an input file that contains only IPv4 data.

17. **Discarding flows not matching CIDR block.** This test is similar to the previous, except flow records that do not match the specified CIDR block are discarded.

18. **Bad SiLK input files.** Check whether `rwflowpack` properly handles unusual files in a directory it is polling for SiLK files. One test file is a SiLK data file that contains no records; `rwflowpack` should move this file to the archive directory

and not create any hourly data files. Another file is one that does not contain the SiLK file header. `rwflowpack` should move this file into its error directory.

19. **Bad flowcap input files.** Check whether `rwflowpack`, running in "fcfiles" input mode, properly handles unusual files its incoming directory. The first test file is a flowcap file that contains no records; `rwflowpack` should move this file to the archive directory and not create any hourly data files. Another file is one that does not contain the SiLK file header. `rwflowpack` should move this file into its error directory. The final test file is a SiLK data file that does not contain the proper header; `rwflowpack` should move this file into the error directory.

20. **Bad NetFlow input files.** Check whether `rwflowpack` properly handles unusual files in a directory it is polling for NetFlow v5 files. `rwflowpack` should treat all these files as invalid and move them to the error directory. The checks include (1) a file that has the correct header and is the correct size but contains a record count of zero, (2) a file that has the correct header but it too small, (3) a file that claims it is NetFlow v8, and (4) a file containing plain text.

21. **Bad IPFIX input files.** Check whether `rwflowpack` properly handles unusual files in a directory it is polling for IPFIX files. `rwflowpack` should treat all these files as invalid and move them to the error directory. The checks include a file that has the correct header but contains no records, and a file containing plain text. This test is only invoked when SiLK has been compiled with IPFIX support.

6.4 Expected results

The tests may take several minutes to run.

During the tests, you may see the following sorts of output.

- The following indicates a test that successfully passed.

  ```
  PASS: tests/rwflowpack-version.pl
  ```

- The following indicates a test has failed. Any text immediately preceding the `FAIL` line may indicate why the test failed.

  ```
  FAIL: tests/rwflowpack-pack-silk.pl
  ```

- The following indicates that the test was skipped.

  ```
  SKIP: tests/rwflowpack-pack-silk-ipv6.pl
  ```

  A test can be skipped for two reasons.

  1. The test is not applicable. For example, there is no need to test IPv6 functionality if SiLK was not compiled with IPv6 support.
  2. A file or application that the test requires is not present. This can occur if you fail to build the SiLK tools prior to testing, so that `make check` is building the tools and testing them. Some tests use other tools in SiLK suite, and the tests will be skipped if the required tools are not available.

Once all processing stops, you should see output similar to the following to indicate that all tests passed (you may or may not get the message about tests being skipped), and the return status from `make` should be 0.

```
======================
All 43 tests passed
(3 tests were not run)
======================
```

Output such as the following indicates that the tests for an application have failed:

```
==================================
6 of 43 tests failed
Please report to netsa-help@cert.org
==================================
make[1]: *** [check-TESTS] Error 1
make: *** [check-am] Error 2
```

7 Testing `rwpollexec`

The `rwpollexec` daemon is used to run a user-defined command on files that appear in a directory which `rwpollexec` periodically examines for new files. `rwpollexec` is intended to provide a stand-alone program that operates similarly to the `--post-command` argument available on `rwflowappend`.

The tests for `rwpollexec` are included with the SiLK-2.5.0 source distribution. The tests are invoked by typing `make check` in the `rwpollexec` directory. (If you type `make check` at the top of the build tree, the `rwpollexec` tests will be invoked as `make` recursively descends into each directory.)

The tests are written in a combination of Perl and Python. The tests will confirm that the `rwpollexec` daemon can start, notices files, execute subprocesses on those files, send signals to subprocesses, properly dispose of files, and terminate cleanly.

7.1 Prerequisites

The tests of `rwpollexec` require that the following tools are installed on the system:

- Python 2.4 or later
- Perl 5.6 or later
- the Perl module Digest::MD5

7.2 Preparation

The test scripts assume you have configured and built `rwpollexec` and all the libraries it requires. The scripts use the application binary as it exists in the build tree, and the scripts do not require that you install SiLK prior to running the tests.

During many of the tests, a temporary directory is created, and files and subdirectories are created in this directory. The directory is created in the location specified by the TMPDIR environment variable, or in `/tmp` when the TMPDIR environment variable is not set.

7.3 Procedure

1. Go to `src/rwpollexec` subdirectory in the directory tree where you built SiLK.
2. Type `make check`. This will invoke the scripts that check the behavior of `rwpollexec`.

The `rwpollexec` tests check the following behaviors:

1. **Handle processes that exit successfully.** Check whether `rwpollexec` properly handles the case when it invokes a command that completes successfully (exit status is 0). The command is invoked sequentially on each of the two files that exist in `rwpollexec`'s incoming directory when `rwpollexec` is invoked. `rwpollexec` should move the files to the archive directory once the command is completed. When `rwpollexec` receives a signal, it should shut down cleanly.
2. **Handle processes that exit unsuccessfully.** This test is similar to the previous, except in this test the command does not complete successfully (i.e., exits with a non-zero status). In this case, the files should be put into the error directory. When `rwpollexec` receives a signal, it should shut down cleanly.
3. **Handle processes that exit due to a signal.** This test is similar to the first, except in this test the command is terminated due to a signal. In this case, the files should be put into the error directory. When `rwpollexec` receives a signal, it should shut down cleanly.
4. **Handle "slow" processes.** Check whether `rwpollexec` properly handles subprocesses that do not exit after a period of time, where the subprocesses will exit when they receive a SIGTERM. When `rwpollexec` is invoked, there are two files in its incoming directory. `rwpollexec` invokes a command on one file, but the command does not exit. `rwpollexec` sends a SIGTERM to the command, at which point the command exits with a status of 0. `rwpollexec` repeats the steps for the second file, and the command exits with a status of 1. The first file should appear in the archive directory, and the second in the error directory. When `rwpollexec` receives a signal, it should shut down cleanly.
5. **Handle "hanging" processes.** This test is similar to the previous, except the subprocesses do not exit when they receive a SIGTERM. Once `rwpollexec` sends the SIGTERM and the process fails to exit, `rwpollexec` sends a SIGKILL to terminate the subprocess. Both input files should be moved to the error directory. When `rwpollexec` receives a signal, it should shut down cleanly.
6. **Handle many types of processes sequentially.** This test is a combination of all of the above tests. `rwpollexec` invokes a command on each of the 12 files that exists in its incoming directory. `rwpollexec` does not invoke the command on the next file until the current command terminates. The command either exits on its own, or `rwpollexec` must send a signal to the process to terminate it. The input files will be moved to the archive or error directory as appropriate. When `rwpollexec` receives a signal, it should shut down cleanly.
7. **No archive directory.** This test is identical to the previous test, except the archive directory is not used. For this test, files whose commands exit successfully should be removed from the file system.
8. **Handle many types of processes simultaneously.** This test is similar to the previous test, except `rwpollexec` is allowed to invoke 4 subprocesses simultaneously. The input files will be moved to the archive or error directory as appropriate. When `rwpollexec` receives a signal, it should shut down cleanly.

7.4 Expected results

The tests may take several minutes to run.

During the tests, you may see the following sorts of output.

- The following indicates a test that successfully passed.

  ```
  PASS: tests/rwpollexec-version.pl
  ```

- The following indicates a test has failed. Any text immediately preceding the `FAIL` line may indicate why the test failed.

  ```
  FAIL: tests/rwpollexec-killed.pl
  ```

Once all processing stops, you should see output similar to the following to indicate that all tests passed, and the return status from `make` should be 0.

```
==================
All 10 tests passed
==================
```

Output such as the following indicates that the tests for an application have failed:

```
==================================
1 of 10 tests failed
Please report to netsa-help@cert.org
==================================
make[1]: *** [check-TESTS] Error 1
make: *** [check-am] Error 2
```

8 Detail of Analysis Tool Testing

This section provides a detailed listing of the tests that will be invoked when you follow the instructions in Section refsecAnalysis.

8.1 Simple help check

The following tests verify the `--help` switch works.

```
flowcap --help
```

```
rwgeoip2ccmap --help
```

```
rwip2cc --help

rwpmapcat --help

mapsid --help

num2dot --help

rwaddrcount --help

rwappend --help

rwbag --help

rwbagbuild --help

rwbagcat --help

rwbagtool --help

rwcat --help

rwcompare --help

rwallformats --help

rwrtd2split --help

rwcount --help

rwcut --help

rwfileinfo --help

rwfglob --help

rwfilter --help

rwflowappend --help

rwflowpack --help
```

```
rwguess --help

rwpackchecker --help

rwgroup --help

rwidsquery --help

rwipaexport --help

rwipaimport --help

rwipfix2silk --help

rwp2yaf2silk --help

rwsilk2ipfix --help

rwmatch --help

rwpollexec --help

rwnetmask --help

rwrandomizeip --help

rwresolve --help

rwscan --help

rwscanquery --help

rwset --help

rwsetbuild --help

rwsetcat --help

rwsetmember --help

rwsettool --help
```

```
rwdedupe --help

rwsort --help

rwsplit --help

rwstats --help

rwstats --legacy-help

rwswapbytes --help

rwtotal --help

rwtuc --help

rwuniq --help

rwreceiver --help

rwsender --help
```

8.2 Simple version check

The following tests verify the `--version` switch works.

```
flowcap --version

rwgeoip2ccmap --version

rwip2cc --version

rwpmapcat --version

mapsid --version

num2dot --version

rwaddrcount --version

rwappend --version
```

```
rwbag --version

rwbagbuild --version

rwbagcat --version

rwbagtool --version

rwcat --version

rwcompare --version

rwallformats --version

rwrtd2split --version

rwcount --version

rwcut --version

rwfileinfo --version

rwfglob --version

rwfilter --version

rwflowappend --version

rwflowpack --version

rwguess --version

rwpackchecker --version

rwgroup --version

rwidsquery --version

rwipaexport --version

rwipaimport --version
```

```
rwipfix2silk --version

rwp2yaf2silk --version

rwsilk2ipfix --version

rwmatch --version

rwpollexec --version

rwnetmask --version

rwrandomizeip --version

rwresolve --version

rwscan --version

rwscanquery --version

rwset --version

rwsetbuild --version

rwsetcat --version

rwsetmember --version

rwsettool --version

rwdedupe --version

rwsort --version

rwsplit --version

rwstats --version

rwswapbytes --version

rwtotal --version

rwtuc --version

rwuniq --version

rwreceiver --version

rwsender --version
```

8.3 Command without arguments

The following tests verify the application does not crash when invoked with no switches or arguments. Most of these tests will result in the application exiting with a non-zero exit status.

`flowcap`

`rwgeoip2ccmap`

`rwip2cc`

`rwpmapcat`

`mapsid`

`rwaddrcount`

`rwappend`

`rwbag`

`rwbagbuild`

`rwbagcat`

`rwbagtool`

`rwcat`

`rwcompare`

`rwallformats`

`rwrtd2split`

`rwcount`

`rwcut`

`rwfileinfo`

`rwfglob`

`rwfilter`

`rwflowappend`

`rwflowpack`

`rwguess`

`rwpackchecker`

`rwgroup`

`rwidsquery`

`rwipaexport`

`rwipaimport`

`rwipfix2silk`

`rwp2yaf2silk`

`rwsilk2ipfix`

`rwmatch`

`rwpollexec`

`rwnetmask`

`rwrandomizeip`

`rwscan`

`rwset`

`rwsetbuild`

`rwsetcat`

`rwsetmember`

```
rwsettool

rwdedupe

rwsort

rwsplit

rwstats

rwswapbytes

rwtotal

rwtuc

rwuniq

rwreceiver

rwsender
```

8.4 Command with null input

The following tests verify the application does not crash when invoked with completely empty (null) input. Most of these tests will result in the application exiting with a non-zero exit status.

```
rwpmapcat </dev/null

num2dot </dev/null

rwaddrcount --print-recs </dev/null

cp empty.rwf sk-teststmp-siqNrDIG                                \
&& rwappend --create sk-teststmp-siqNrDIG /dev/null

rwbag --sport-flows=/dev/null </dev/null

rwbagbuild </dev/null

rwbagcat --minkey=50 --maxkey=100 </dev/null
```

```
rwbagtool --minkey=50 --maxkey=100 </dev/null

rwcat </dev/null

rwcompare data.rwf /dev/null

rwcount </dev/null

rwcut </dev/null

rwfilter --input-pipe=/dev/null --all=/dev/null

rwgroup --id-fields=3 /dev/null

rwnetmask --sip=prefix-length=24 </dev/null

rwrandomizeip </dev/null

rwresolve </dev/null

rwscan --scan-mode=2 /dev/null

rwset --sip-file=/dev/null </dev/null

rwsetbuild </dev/null >/dev/null

rwsetcat </dev/null

rwsetmember 10.x.x.x </dev/null

rwsettool </dev/null >/dev/null

rwdedupe --ignore-fields=1 </dev/null

rwsort --fields=1 </dev/null

rwsplit --basename=./sk-teststmp-AWGPVL17                       \
        --flow-limit=100 /dev/null

rwstats --fields=sip --count=10 </dev/null

rwswapbytes --big-endian </dev/null

rwtotal --sport </dev/null

rwtuc /dev/null

rwuniq --fields=1 </dev/null
```

8.5 Command with empty SiLK file

The following tests verify the application works correctly when invoked with a SiLK file that contains no data section.

```
rwaddrcount --print-rec empty.rwf
```

```
rwaddrcount --print-stat empty.rwf
```

```
rwbag --sport-flow=stdout empty.rwf                              \
| rwbagcat --integer-keys
```

```
rwcount --bin-size=3600 empty.rwf
```

```
rwcount empty.rwf
```

```
rwcut --fields=3-8 empty.rwf
```

```
rwfileinfo --fields=7 --no-title empty.rwf
```

```
rwfilter --proto=0- --pass=stdout empty.rwf                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwgroup --id-fields=3 empty.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwsilk2ipfix empty.rwf                                           \
| rwipfix2silk -                                                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwmatch --relate=1,2 empty.rwf empty.rwf -                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwnetmask --nhip-prefix=16 empty.rwf                             \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwrandomizeip --seed=38901 empty.rwf -                           \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwscan --scan-mode=2 empty.rwf
```

```
rwsetbuild /dev/null sk-teststmp-RvR4voMP                       \
&& rwscan --trw-sip-set=./sk-teststmp-RvR4voMP empty.rwf


rwset --sip-file=stdout empty.rwf                               \
| rwsetcat


rwsort --field=9,1 empty.rwf                                    \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


rwsplit --basename=./sk-teststmp-DcNrItKO                       \
        --flow-limit=100 empty.rwf


rwstats --fields=dip --count=10 --top --ipv6-policy=ignore      \
        --presorted-input empty.rwf


rwstats --fields=dip --count=10 --top                           \
        --ipv6-policy=ignore empty.rwf


rwswapbytes --big-endian empty.rwf -                            \
| rwfileinfo --no-title --field=byte-order,count-records -


rwswapbytes --little-endian empty.rwf -                         \
| rwfileinfo --no-title --field=byte-order,count-records -


rwswapbytes --little-endian empty.rwf -                         \
| rwswapbytes --swap-endian - -                                 \
| rwfileinfo --no-title --field=byte-order,count-records -


rwtotal --sport empty.rwf


rwuniq --fields=sport --sort-output empty.rwf
```

8.6 Checking successful exit status

The following tests perform a variety of checks. In all cases, the application should exit with a zero exit status.

```
mapsid 99999


mapsid 9999


mapsid S9999
```

```
rwset --sip-file=stdout data.rwf                                \
| rwsettool --sample --ratio=0.4 --seed=2749473

rwset --sip-file=stdout data.rwf                                \
| rwsettool --sample --size=2000 --seed=2749473

rwdedupe empty.rwf
```

8.7 Checking for non-zero exit status

The following tests perform a variety of checks for error conditions. In all cases, the application should exit with a non-zero exit status.

```
rwaddrcount --print-recs

rwaddrcount empty.rwf

rwappend --create /dev/null empty.rwf

rwappend stdout empty.rwf >/dev/null

rwbag --sport-flows=/dev/null

rwbag empty.rwf

rwbagcat --mincounter=101 --maxcounter=99 /dev/null

rwbagcat --minkey=101 --maxkey=99 /dev/null

rwbagtool --mincounter=101 --maxcounter=99 /dev/null

rwbagtool --minkey=101 --maxkey=99 /dev/null

rwcompare data.rwf data.rwf data.rwf

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:16 --end-date=2009/02/12:14

rwfilter --fail=/dev/null empty.rwf

rwfilter --pass=/dev/null empty.rwf

rwfilter --print-stats empty.rwf
```

```
rwfilter --all=/dev/null

rwfilter --proto=1 empty.rwf

rwidsquery --intype=fast

rwidsquery --intype=rule --dry-run sk-teststmp-gc1aJ9SJ 2>&1

rwipaexport /dev/null

rwipaexport --catalog=my-cat --time=2009/02/14:00:00 /dev/null

rwipaimport /dev/null

rwset --sip=- empty.rwf                                          \
| rwipaimport --catalog=my-cat --description=my-description     \
        --start-time=2009/02/12:00:00                            \
        --end-time=2009/02/14:23:59:59 -

rwnetmask --sip=prefix-length=24

rwnetmask empty.rwf

rwrandomizeip empty.rwf </dev/null

rwset --sip-file=/dev/null

rwset empty.rwf

rwset --sip-file=stdout empty.rwf                                \
| rwsetmember 10.x.x.x

rwset --sip-file=stdout data.rwf                                 \
| rwsettool --sample

rwdedupe --ignore-fields=1

rwsort --fields=1

rwsort empty.rwf

rwsplit --flow-limit=100 empty.rwf
```

```
rwsplit --basename=./sk-teststmp-j2tNVJwR empty.rwf

rwsplit --basename=./sk-teststmp-hlvSe9Gd --ip-limit=200        \
        --flow-limit=900 empty.rwf

rwstats --fields=sip --count=10

rwstats empty.rwf

rwswapbytes --big-endian empty.rwf

rwswapbytes --big-endian

rwswapbytes empty.rwf /dev/null

rwtotal --sport --dport empty.rwf

rwtotal --sport

rwtotal empty.rwf

rwuniq --fields=1

rwuniq empty.rwf
```

8.8 Perform a checksum of the output–success

The following tests perform a variety of checks. The output of the command is gathered and compared to a known good checksum (MD5). In all cases, the application should exit with a zero exit status.

```
rwip2cc --map-file=fake-cc.pmap --print-ips=0                   \
        --address=10.10.10.10

rwip2cc --map-file=fake-cc.pmap --print-ips=1                   \
        --address=10.10.10.10

rwip2cc --map-file=fake-cc.pmap --address=10.10.10.10

rwcut --fields=sip --ipv6-policy=ignore --no-title              \
        --delimited data.rwf                                    \
| rwip2cc --input-file=-

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=- --delimited=,
```

```
echo 10.10.10.10                                                \
| rwip2cc --input-file=-

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=-                \
        --integer-ips --column-separator=/

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=- --no-columns

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=- --print-ips=0

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=- --print-ips=1

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=-                \
        --zero-pad-ips --no-final-delimiter

echo 10.10.10.10                                                \
| rwip2cc --map-file=fake-cc.pmap --input-file=-

rwpmapcat --no-cidr fake-cc.pmap

rwpmapcat --delimited=, --no-cidr --map-file ip-map.pmap

rwpmapcat --ip-label-to-ignore=0.0.0.0 ip-map.pmap

rwpmapcat --ignore-label=external ip-map.pmap

rwpmapcat --integer-ips --no-columns --map-file ip-map.pmap

rwpmapcat --output-type=labels --map-file ip-map.pmap

rwpmapcat --left-justify-labels ip-map.pmap

rwpmapcat --output-type=mapname --map-file ip-map.pmap

rwpmapcat --no-cidr-blocks --map-file ip-map.pmap

rwpmapcat --output-type=type --no-titles ip-map.pmap

rwpmapcat --zero-pad-ips --output-type=ranges ip-map.pmap
```

```
rwpmapcat ip-map.pmap

rwpmapcat --ignore-label=unknown proto-port-map.pmap

rwpmapcat --output-type=labels --no-title proto-port-map.pmap

rwpmapcat --output-type=mapname proto-port-map.pmap

rwpmapcat --no-titles proto-port-map.pmap

rwpmapcat --output-type=type,mapname                            \
        --map-file=proto-port-map.pmap

rwpmapcat --column-sep=, --map-file=proto-port-map.pmap

cat ip-map.pmap                                                 \
| rwpmapcat --map-file=- --no-cidr

cat ip-map.pmap                                                 \
| rwpmapcat --no-cidr

mapsid S9 8 S11 10 S7

mapsid --print-classes

rwcut --fields=1,3,2,4,5 --no-title --ipv6-policy=ignore        \
        --integer-ips data.rwf                                  \
| num2dot --ip-fields=1,3

rwcut --fields=1,2 --no-title --ipv6-policy=ignore              \
        --integer-ips --no-final-delimiter data.rwf             \
| num2dot --ip-fields=2,1

rwcut --fields=1,3,2,4,5 --no-title                             \
        --ipv6-policy=ignore data.rwf

rwaddrcount --print-rec --sort-ips --column-separator=/         \
        --no-final-delimiter data.rwf

rwaddrcount --print-stat --output-path=/dev/null                \
        --copy-input=stdout data.rwf                            \
| rwaddrcount --print-stat

rwaddrcount --print-rec --sort-ips --delimited=, data.rwf
```

```
rwaddrcount --use-dest --print-rec --sort-ips data.rwf

rwaddrcount --use-dest --print-stat data.rwf

rwaddrcount --print-rec --sort-ips --integer-ips                \
        --max-byte=2000 data.rwf

rwaddrcount --use-dest --print-rec --sort-ips                   \
        --max-packet=20 data.rwf

rwaddrcount --print-ips --sort-ips --zero-pad-ips               \
        --max-record=10 data.rwf

rwaddrcount --print-rec --sort-ips --integer-ips                \
        --min-byte=2000 data.rwf

rwaddrcount --use-dest --print-rec --sort-ips                   \
        --min-packet=20 data.rwf

rwaddrcount --print-ips --sort-ips --zero-pad-ips               \
        --min-record=10 data.rwf

rwaddrcount --print-rec --use-dest --sort-ips data.rwf data.rwf

rwaddrcount --print-rec --sort-ips --no-columns                 \
        --no-title data.rwf

rwaddrcount --print-ips --sort-ips --no-title data.rwf

rwaddrcount --print-rec data.rwf                                \
| sort

rwaddrcount --set-file=stdout data.rwf                          \
| rwsetcat

rwaddrcount --print-stat data.rwf

cat data.rwf                                                    \
| rwaddrcount --print-rec --use-dest --sort-ips

cp data.rwf sk-teststmp-cIapniTR                                \
&& rwappend --create sk-teststmp-cIapniTR empty.rwf empty.rwf   \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output sk-teststmp-cIapniTR
```

```
rwappend --create=data.rwf sk-teststmp-XHXkey6F data.rwf        \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output sk-teststmp-XHXkey6F


rwappend --create sk-teststmp-ZZVlLAf3 empty.rwf data.rwf       \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output sk-teststmp-ZZVlLAf3


rwcat --byte-order=little empty.rwf > sk-teststmp-nMsjxdhj      \
&& rwappend sk-teststmp-nMsjxdhj empty.rwf data.rwf empty.rwf   \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output sk-teststmp-nMsjxdhj


rwcat --byte-order=big empty.rwf > sk-teststmp-AJEVqTQz         \
&& rwappend sk-teststmp-AJEVqTQz data.rwf                       \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output sk-teststmp-AJEVqTQz


rwbag --sport-flows=/dev/null --copy-input=stdout data.rwf      \
| rwbag --sport-flows=-                                        \
| rwbagcat --integer-keys


rwbag --dip-bytes=stdout data.rwf                               \
| rwbagcat


rwbag --dip-flows=stdout data.rwf                               \
| rwbagcat --zero-pad-ips


rwbag --dip-packets=stdout data.rwf                             \
| rwbagcat --integer-keys


rwbag --dport-bytes=- data.rwf                                  \
| rwbagcat --integer-keys --no-final-delimiter -


rwbag --dport-flow=stdout data.rwf                              \
| rwbagcat --integer-keys --delimited


rwbag --dport-flow=stdout data.rwf                              \
| rwbagcat --integer-keys


rwbag --dport-packets=stdout data.rwf                           \
| rwbagcat --integer-keys --no-columns


rwbag --sport-flow=stdout empty.rwf data-v6.rwf empty.rwf data.rwf \
| rwbagcat --integer-keys
```

```
rwbag --sport-flow=stdout empty.rwf data-v6.rwf data-v6.rwf     \
| rwbagcat --integer-keys

rwbag --sport-flow=stdout data.rwf empty.rwf data.rwf           \
| rwbagcat --integer-keys

rwbag --proto-bytes=- data.rwf                                  \
| rwbagcat --integer-keys --minkey=1 --maxkey=20 --zero-counts

rwbag --proto-flow=stdout data.rwf                              \
| rwbagcat --integer-keys --minkey=1

rwbag --proto-packets=stdout data.rwf                           \
| rwbagcat --integer-keys --maxkey=17

rwbag --sip-bytes=stdout data.rwf                               \
| rwbagcat

rwbag --sip-flows=/dev/null --sip-packets=stdout data.rwf       \
| rwbagcat

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat

rwbag --sip-packets=stdout --sip-bytes=/dev/null data.rwf       \
| rwbagcat

rwbag --sip-packets=stdout data.rwf                             \
| rwbagcat

rwbag --sport-bytes=- data.rwf                                  \
| rwbagcat --integer-keys --delimited=, -

rwbag --sport-flow=stdout data.rwf                              \
| rwbagcat --integer-keys --column-separator=,

rwbag --sport-packets=stdout data.rwf                           \
| rwbagcat --integer-keys --column-separator=, --no-final-delim

cat data.rwf                                                    \
| rwbag --sport-flows=stdout                                    \
| rwbagcat --integer-keys

rwuniq --fields=sport --flows --no-title                        \
        --delimited=, data.rwf                                  \
| rwbagbuild --bag-input=stdin --delimiter=,                    \
| rwbagcat --integer-keys
```

```
rwuniq --fields=sport --flows --no-title data.rwf               \
| rwbagbuild --bag-input=stdin                                  \
| rwbagcat --integer-keys

rwset --sip-file=stdout data.rwf                                \
| rwbagbuild --set-input=stdin --output-path=stdout             \
| rwbagcat

rwset --sip-file=stdout data.rwf                                \
| rwbagbuild --set-input=stdin --default-count=200              \
| rwbagcat

rwset --sip-file=stdout data.rwf                                \
| rwbagbuild --set-input=stdin                                  \
| rwbagcat

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --integer-keys --bin-ips=binary

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --integer-keys --bin-ips=decimal

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --integer-keys --bin-ips

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --network-structure=12TS,12

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --network-structure=ATS

rwbag --sip-flows=stdout data.rwf                               \
| rwbagcat --network-structure

rwbag --sport-bytes=stdout data.rwf                             \
| rwbagcat --integer-keys --mincounter=2000

rwbag --sport-flows=stdout data.rwf                             \
| rwbagcat --integer-keys --mincounter=10

rwbag --sport-packets=stdout data.rwf                           \
| rwbagcat --integer-keys --mincounter=20

rwbag --sport-bytes=stdout data.rwf                             \
| rwbagcat --integer-keys --maxcounter=2000
```

```
rwbag --sport-flows=stdout data.rwf                              \
| rwbagcat --integer-keys --maxcounter=10

rwbag --sport-packets=stdout data.rwf                            \
| rwbagcat --integer-keys --maxcounter=20

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --add stdin                                         \
| rwbagcat --integer-keys

rwbag --sip-flows=stdout data.rwf                                \
| rwbagtool --coverset                                          \
| rwsetcat

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --maxcounter=10                                     \
| rwbagcat --integer-keys

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --maxkey=1024                                       \
| rwbagcat --integer-keys

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --mincounter=10                                     \
| rwbagcat --integer-keys

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --minkey=1024                                       \
| rwbagcat --integer-keys

rwbag --sport-flows=stdout data.rwf                              \
| rwbagtool --add --output-path=stdout                          \
| rwbagcat --integer-keys

rwcat --byte-order=big data-v6.rwf                               \
| rwcut --fields=1-15,20,21,26-29 --epoch-time --delimited

rwcat --byte-order=big data.rwf                                  \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited

rwcat --byte-order=little data-v6.rwf                            \
| rwcut --fields=1-15,20,21,26-29 --epoch-time --delimited

rwcat --byte-order=little data.rwf                               \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited
```

```
rwcat empty.rwf data.rwf empty.rwf                              \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


cat data.rwf                                                    \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


rwcat --note-add='my command line note' empty.rwf               \
| rwfileinfo --fields=7,14 -


echo 'my stdin note'                                            \
| rwcat --note-file-add=- empty.rwf                            \
| rwfileinfo --fields=7,14 -


rwcat data-v6.rwf                                               \
| rwcut --fields=1-15,20,21,26-29 --epoch-time --delimited


rwcat data.rwf                                                  \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


cat data.rwf                                                    \
| rwcat                                                        \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


ls -1 empty.rwf data.rwf empty.rwf                              \
| rwcat --xargs=stdin                                          \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


ls -1 empty.rwf data.rwf empty.rwf                              \
| rwcat --xargs                                                \
| rwcut --fields=1-15,20,21,26-29 --ipv6-policy=ignore          \
        --epoch-time --integer-ips --delimited


rwcat --byte-order=big data.rwf                                 \
| rwcompare data.rwf -


rwcat --byte-order=big                                          \
        --output-path=./sk-teststmp-oJp2k19U data.rwf           \
&& rwcompare data.rwf sk-teststmp-oJp2k19U


rwcat --byte-order=little data.rwf                              \
| rwcompare - data.rwf
```

```
rwfilter --stime=2009/02/13:20:00-2009/02/13:20 --sensor=S2     \
        --proto=6 --aport=80,8080,443 --pass=stdout data.rwf    \
| rwallformats --basename=sk-teststmp                           \
&& md5 sk-teststmp*

rwcount --bin-size=1 --load-scheme=1 data.rwf

rwcount --bin-size=30 --load-scheme=2 data.rwf

rwcount --bin-size=3600 --load-scheme=2 --bin-slots data.rwf

rwcount --bin-size=3600 --no-title data.rwf

rwcount --bin-size=86400 --load-scheme=1 --epoch-slots data.rwf

rwcount --bin-size=900 --load-scheme=3 data.rwf

rwcount --bin-size=3600 --load-scheme=1 --column-separator=/    \
        --no-final-delimiter data.rwf

rwcount --bin-size=3600 --load-scheme=1                         \
        --output-path=/dev/null --copy-input=stdout data.rwf    \
| rwcount --bin-size=86400 --load-scheme=1 --epoch-slots

rwcount --bin-size=3600 --load-scheme=1 --delimited=, data.rwf

rwcount --bin-size=3600 --load-scheme=0                         \
        --end-epoch=2009/02/14T20:30:00 data.rwf

rwcount --bin-size=3600 --load-scheme=1                         \
        --legacy-timestamps=0 data.rwf

rwcount --bin-size=3600 --load-scheme=1                         \
        --legacy-timestamps=1 data.rwf

rwcount --bin-size=0.500 --skip-zero --load-scheme=1            \
        --start-epoch=2009/02/14T20:00:00 data.rwf

rwcount --bin-size=0.1 --load-scheme=2 data.rwf

rwcount --bin-size=3600                                         \
        --load-scheme=1 empty.rwf data.rwf data-v6.rwf empty.rwf

rwcount --bin-size=3600                                         \
        --load-scheme=1 data-v6.rwf empty.rwf data-v6.rwf empty.rwf
```

```
rwcount --bin-size=3600                                          \
        --load-scheme=1 empty.rwf data.rwf empty.rwf data.rwf

rwcount --bin-size=3600 --load-scheme=1 --no-columns             \
        --no-title data.rwf

rwcount data.rwf

rwsort --fields=stime --reverse data.rwf                         \
| rwcount --load-scheme=1

rwcount --bin-size=3600 --load-scheme=0                          \
        --start-epoch=2009/02/12T20:00:00                        \
        --end-epoch=2009/02/13T20:00:00 data.rwf

rwcount --bin-size=3600 --load-scheme=0 --skip-zero              \
        --start-epoch=2009/02/11T20:30:00 data.rwf

rwcount --bin-size=604800 --load-scheme=0                        \
        --start-epoch=2009/02/10:00:00:00 data.rwf

rwcount --bin-size=3600 --load-scheme=0                          \
        --start-epoch=2009/02/12T20:30:00 data.rwf

cat data.rwf                                                     \
| rwcount --bin-size=3600 --load-scheme=1

rwcut --fields=1-5 --ipv6-policy=ignore data.rwf

rwcut --fields=1-5 --ipv6-policy=force data.rwf

rwcut --fields=1-5 data-v6.rwf

rwcut --fields=stype,sip,dtype,dip,dtype --delimited             \
        --num-recs=10000 data.rwf

rwcut --all-fields --delimited data-v6.rwf

rwcut --all-fields --delimited data.rwf

rwcut --fields=7,6 --column-separator=/ data.rwf

rwcut --fields=5,4,3 --column-separator=, --no-columns data.rwf
```

```
rwcut --fields=5 --output-path=/dev/null                        \
        --copy-input=stdout data.rwf                            \
| rwcut --fields=5
```

```
rwcut --fields=sip,scc,dip,dcc --ipv6=ignore data.rwf
```

```
rwcut --delimited data.rwf
```

```
rwcut --fields=2 --delimited --zero-pad-ips data.rwf
```

```
rwcut --dry-run --ipv6-policy=ignore data.rwf
```

```
rwcut --fields=8,initialFlags,sessionFlags data.rwf
```

```
rwcut --plugin=flowrate.so                                      \
        --fields=bytes,packets,dur,pckts/sec,bytes/sec,bytes/packet,payload-bytes,payload-rate data.rwf
```

```
rwfilter --proto=58 --pass=- data-v6.rwf                        \
| rwcut --fields=4,5 --icmp-type-and-code
```

```
rwfilter --proto=1 --pass=- data.rwf                            \
| rwcut --fields=4,5 --icmp-type-and-code
```

```
rwfilter --proto=58 --pass=- data-v6.rwf                        \
| rwcut --fields=icmpTypeCode
```

```
rwfilter --proto=1 --pass=- data.rwf                            \
| rwcut --fields=icmpTypeCode
```

```
rwcut --fields=9,11 --legacy-time=0 data.rwf
```

```
rwcut --fields=9,11 --legacy-time=1 data.rwf
```

```
rwcut --fields=attributes,application data.rwf
```

```
rwcut --fields=5 --delimited data-v6.rwf data.rwf
```

```
rwcut --fields=5 --delimited data.rwf data.rwf
```

```
rwcut --fields=5,4,3 --no-columns data.rwf
```

```
rwcut --fields=5,4,3 --no-final-delimiter data.rwf
```

```
rwcut --fields=5,4,3 --no-title < data.rwf

rwcut --pmap-file=servhost:ip-map.pmap                          \
        --fields=dst-servhost data.rwf

rwcut --pmap-file=service-port:proto-port-map.pmap              \
        --pmap-file=ip-map.pmap                                 \
        --fields=src-service-host,src-service-port,src-service-host,src-service-port data.rwf

rwcut --pmap-file=proto-port-map.pmap                           \
        --fields=sval,dval data.rwf

rwcut --pmap-file=ip-map.pmap                                   \
        --fields=src-service-host data.rwf

rwcut --python-file=pysilk-plugin.py                            \
        --fields=scc,py-scc,dcc,py-dcc                          \
        --num-recs=10000 data.rwf

rwcut --python-file=pysilk-plugin.py --fields=3-5,lower_port    \
        --num-recs=10000 data.rwf

rwcut --python-file=pysilk-plugin.py                            \
        --fields=lower_port,lower_port data.rwf

rwcut --python-file=pysilk-plugin.py                            \
        --fields=sip,dip,sport,dport,server_ipv6                \
        --num-recs=10000 data-v6.rwf

rwcut --python-file=pysilk-plugin.py                            \
        --fields=sip,dip,server_ip,sport,dport,lower_port_simple,protocol,proto_name \
        --num-recs=10000 --delimited=, data.rwf

rwcut --fields=3-5 --num-recs=3000 data.rwf

rwcut --fields=9,10 --epoch-time --num-recs=3000                \
        --start-rec-num=2000 data.rwf

rwcut --fields=12 --integer-sensor --num-recs=3000              \
        --end-rec-num=2000 data.rwf

rwcut --fields=sip,dip --delimited=, --num-recs=3000            \
        --end-rec-num=20000 data.rwf

rwcut --fields=sport,dport --start-rec-num=30000                \
        --end-rec-num=40000 data.rwf
```

```
rwcut --fields=1 --delimited --integer-ips data.rwf

rwcut --fields=sensor,class,type data.rwf

rwcut --plugin=skplugin-test.so --ipv6-policy=ignore            \
        --no-columns                                            \
        --fields=bytes,copy-bytes,text-bytes,quant-bytes,sip,copy-sipv4,copy-sip data.rwf

cat data.rwf                                                    \
| rwcut --fields=3-8

rwcut --fields=9 --epoch-time --no-final-delimiter data.rwf

rwcut --fields=9-11 data.rwf

rwfileinfo --fields=1,5-6 --no-title data.rwf

rwfileinfo --fields=count-records data.rwf

cat data.rwf                                                    \
| rwfileinfo --fields=count-records -

rwfileinfo --fields=command-lines,version data.rwf

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14     \
        --class=all 2>&1

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14     \
        --flowtypes=all/in,all/outweb 2>&1

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14     \
        --sensors=4,6-8,10 2>&1

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14     \
        --sensors=S4,S6,S7,S8,S10 2>&1

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/13 --no-summary 2>&1

rwfglob --data-rootdir=. --print-missing                        \
        --start-date=2009/02/12:12                              \
        --end-date=2009/02/12:14 2>&1
```

```
rwfglob --data-rootdir=. --print-missing                          \
        --start-date=2009/02/12:12 2>&1

rwfglob --data-rootdir=. --print-missing                          \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14       \
        --type=inweb --sensor=S12 2>&1

rwfglob --data-rootdir=. --print-missing                          \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14       \
        --type=out 2>&1

rwfilter --active-time=2009/02/13:00:00-2009/02/13:00:05          \
        --pass=stdout data.rwf                                    \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output

rwfilter --any-address=2001:db8:c0:a8::x:c1-ff,c0:x               \
        --fail=stdout data-v6.rwf                                 \
| rwcat --compression-method=none --byte-order=little

rwfilter --any-address=192.168.192-255.x                          \
        --fail=stdout data.rwf                                    \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output

rwfilter --any-cidr=2001:db8:c0:a8::c0:0/107,2001:db8:c0:a8::e0:0/107 \
        --fail=stdout data-v6.rwf                                 \
| rwcat --compression-method=none --byte-order=little

rwfilter --any-cidr=192.168.192.0/18 --fail=stdout data.rwf       \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output

rwfilter --pmap-file=ip-map.pmap --pmap-any-service-host=dhcp     \
        --pass=stdout data.rwf                                    \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output

echo 192.168.192-255.x                                            \
| rwsetbuild - -                                                 \
| rwfilter --anyset=- --fail=stdout data.rwf                      \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output

rwfilter --aport=25 --proto=6 --pass=stdout data.rwf              \
| rwcat --compression-method=none --byte-order=little            \
        --ipv4-output
```

```
rwfilter --application=80 --pass=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --attributes=T/T --pass=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --bytes-per-packet=39-60 --pass=stdout data.rwf        \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --bytes=1-100 --pass=stdout data.rwf                   \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --dcc=xg,xj,xq --pass=stdout data.rwf                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --pmap-file=ip-map.pmap                                \
        --pmap-dst-service-host='internal,internal services'    \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --dport=25 --pass=stdout data.rwf                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --pmap-file=service:proto-port-map.pmap                \
        --pmap-dst-service=TCP/HTTP,TCP/HTTPS                   \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --dtype=2 --pass=stdout data.rwf                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --duration=1-5 --pass=stdout data.rwf                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --etime=2009/02/13:00:00-2009/02/13:00:05              \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwfilter --data-rootdir=. --print-missing                       \
        --start-date=2009/02/12:12 --end-date=2009/02/12:14     \
        --sensors=S4,S6,S7,S8,S10 --type=in,outweb              \
        --all=/dev/null 2>&1

rwfilter --flags-all=R/R --pass=stdout data.rwf                 \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --flags-init=S/SA --pass=stdout data.rwf               \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --flags-session=/F,C/C --pass=stdout data.rwf          \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --plugin=flowrate.so --bytes-per-second=100-           \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --plugin=flowrate.so --proto=17                        \
        --print-volume-statistics=stdout data.rwf

rwfilter --plugin=flowrate.so --packets-per-second=100-1000     \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --icmp-code=3 --pass=stdout data.rwf                   \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --icmp-type=3 --pass=stdout data.rwf                   \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --input-index=10 --pass=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --proto=17 --print-volume-statistics=stdout            \
        data.rwf

rwfilter --ip-version=4 --pass=stdout data.rwf                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwfilter --proto=17 --max-fail=200 --fail=stdout data.rwf       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --proto=17 --max-pass=100 --pass=stdout data.rwf       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --proto=17 --max-pass=100 --max-fail=200               \
        --pass=./sk-teststmp-PCBydKrj                           \
        --fail=./sk-teststmp-eH6FUiRi data.rwf                  \
&& rwcut --fields=1-10                                          \
        --ipv6-policy=ignore sk-teststmp-PCBydKrj sk-teststmp-eH6FUiRi


rwfilter --proto=17 --pass=stdout data.rwf data.rwf data.rwf    \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-titles


rwfilter --not-any-address=2001:db8:c0:a8:x:x:c0-ff:x           \
        --pass=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little


rwfilter --not-any-address=192.168.255,192-254.x                \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --not-any-cidr=2001:db8:c0:a8::c0:0/106                \
        --pass=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little


rwfilter --not-any-cidr=192.168.192.0/19,192.168.224.0/20,192.168.240.0/21,192.168.248.0/22,192.168.252.0/23,192.168.254.0/2
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


echo 192.168.255,192-254.x                                      \
| rwsetbuild - -                                               \
| rwfilter --not-anyset=- --pass=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --not-saddr=x:x:a:fc-ff::0-ffff:0,1-fab,fad-ffff,fac   \
        --pass=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little
```

```
rwfilter --not-saddr=10.252-255.0-255.0,1-100,102-255,101       \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --not-scidr=2001:db8:a:fc::/62                         \
        --pass=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little


rwfilter --not-scidr=10.254.0.0/16,10.252.0.0/16,10.255.0.0/16,10.253.0.0/16 \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


echo 10.252-255.x.x                                             \
| rwsetbuild - -                                               \
| rwfilter --not-sipset=- --pass=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --output-index=10 --pass=stdout data.rwf               \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --packets=1-50 --pass=stdout data.rwf                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --plugin=flowrate.so --payload-bytes=0-1000            \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --plugin=flowrate.so --payload-rate=1000.4-2000.9      \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --proto=17 --print-statistics --print-filenames        \
        --pass=/dev/null data.rwf 2>&1


rwfilter --proto=17                                             \
        --print-volume-statistics=stdout data-v6.rwf


rwfilter --proto=17 --print-volume-statistics=stdout data.rwf
```

```
rwfilter --proto=17 --pass=stdout data.rwf                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --python-expr='rec.sport==rec.dport'                   \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --python-file=pysilk-plugin.py                         \
        --print-volume data.rwf 2>&1


rwfilter --proto=17 --print-volume-statistics=stdout            \
        data.rwf


rwfilter --saddress=2001:db8:a:fc-ff::x:x                       \
        --fail=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little


rwfilter --saddress=10.252-255.x.x --fail=stdout data.rwf       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --scc=xz --dcc=xz --pass=stdout data.rwf               \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --scc=xa,xb,xc --pass=stdout data.rwf                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --scidr=2001:db8:a:fc::/63,2001:db8:a:fe::/63          \
        --fail=stdout data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little


rwfilter --scidr=10.252.0.0/15,10.254.0.0/15                    \
        --fail=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --pmap-file=ip-map.pmap --pmap-src-service-host=ntp    \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
echo 10.252-255.x.x                                             \
| rwsetbuild - -                                               \
| rwfilter --sipset=- --fail=stdout data.rwf                    \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --sport=25 --dport=25 --pass=stdout data.rwf           \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --pmap-file=proto-port-map.pmap                        \
        --pmap-sport-proto=UDP/DHCP --pass=stdout data.rwf      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --sport=25 --pass=stdout data.rwf                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter data.rwf data.rwf data.rwf --all-dest=stdout           \
| rwfilter --input-pipe=- --proto=17 --pass=stdout              \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-titles


rwfilter --stime=2009/02/13:00:00-2009/02/13:00:05              \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --stype=1 --pass=stdout data.rwf                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --threads=4 --proto=17                                 \
        --pass=stdout data.rwf data.rwf data.rwf                \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-titles


echo 25,6                                                       \
| rwfilter --tuple-file=- --tuple-delim=,                       \
        --tuple-fields=sport,proto --tuple-direction=both       \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
echo 25,6                                                       \
| rwfilter --tuple-file=- --tuple-delim=,                       \
        --tuple-fields=sport,proto --pass=stdout data.rwf       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


echo 25,6                                                       \
| rwfilter --tuple-file=- --tuple-delim=,                       \
        --tuple-fields=sport,proto --tuple-direction=reverse    \
        --pass=stdout data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --pmap-file=service:proto-port-map.pmap                \
        --pmap-file=ip-map.pmap --pmap-any-service=UDP/NTP      \
        --pmap-any-service-host=ntp --pass=stdout data.rwf      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwfilter --type=in --pass=stdout data.rwf                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


ls -1 data.rwf data.rwf data.rwf                                \
| rwfilter --xargs=stdin --proto=17 --pass=stdout               \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-titles


rwguess --print-all small.pdu


rwguess small.pdu


rwguess --top=2 small.pdu


rwpackchecker --print-all data.rwf empty.rwf


rwpackchecker --value max-tcp-bpp=5000                          \
        --allowable-count max-tcp-bpp=2 data.rwf


rwsort --fields=dtype data.rwf                                  \
| rwgroup --id-fields=dtype                                     \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwsort --fields=stype data.rwf                                  \
| rwgroup --id-fields=stype                                     \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=dcc data.rwf                                    \
| rwgroup --id-fields=dcc                                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=scc data.rwf                                    \
| rwgroup --id-fields=scc                                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --plugin=flowrate.so --fields=bytes/sec data.rwf         \
| rwgroup --plugin=flowrate.so --id-fields=bytes/sec            \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --plugin=flowrate.so --fields=payload-bytes data.rwf     \
| rwgroup --plugin=flowrate.so --id-fields=payload-bytes        \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --plugin=flowrate.so --fields=pckts/sec data.rwf         \
| rwgroup --plugin=flowrate.so --id-fields=pckts/sec            \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=5,1,3,2,4 data.rwf                              \
| rwgroup --id-fields=5,1,3,2,4                                 \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwsort --fields=1 data-v6.rwf                                   \
| rwgroup --delta-field=1 --delta-value=64                      \
| rwcat --compression-method=none --byte-order=little

rwsort --fields=1 data.rwf                                      \
| rwgroup --delta-field=1 --delta-value=16                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=1,2,9 data-v6.rwf                               \
| rwgroup --id-fields=1,2 --delta-field=9 --delta-value=15      \
        --summarize --rec-threshold=5                           \
| rwcat --compression-method=none --byte-order=little
```

```
rwsort --fields=1,2,9 data.rwf                                  \
| rwgroup --id-fields=1,2 --delta-field=9 --delta-value=15      \
        --summarize --rec-threshold=5                           \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=1,2,9 data.rwf                                  \
| rwgroup --id-fields=1,2 --delta-field=9 --delta-value=15      \
        --summarize                                             \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=1,2,9 data.rwf                                  \
| rwgroup --id-fields=1,2 --delta-field=9 --delta-value=15      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=1,2,9 data-v6.rwf                               \
| rwgroup --id-fields=1,2                                       \
| rwcat --compression-method=none --byte-order=little

rwsort --fields=1,2,9 data.rwf                                  \
| rwgroup --id-fields=1,2                                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=servhost:ip-map.pmap                       \
        --fields=dst-servhost                                   \
| rwgroup --pmap-file=servhost:ip-map.pmap                      \
        --id-fields=dst-servhost                                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=service-port:proto-port-map.pmap           \
        --pmap-file=ip-map.pmap                                 \
        --fields=src-service-host,src-service-port              \
| rwgroup --pmap-file=service-port:proto-port-map.pmap          \
        --pmap-file=ip-map.pmap                                 \
        --id-fields=src-service-host,src-service-port           \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=proto-port-map.pmap --fields=sval          \
| rwgroup --pmap-file=proto-port-map.pmap --id-fields=sval      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=ip-map.pmap --fields=src-service-host       \
| rwgroup --pmap-file=ip-map.pmap                                \
        --id-fields=src-service-host                            \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=3,4,9 data.rwf                                  \
| rwgroup --id-fields=3,4 --delta-field=9 --delta-value=15       \
        --objective                                             \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --python-file=pysilk-plugin.py                           \
        --fields=lower_port data.rwf                            \
| rwgroup --python-file=pysilk-plugin.py                        \
        --id-fields=lower_port                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

cat data.rwf                                                    \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwsort --fields=3 data.rwf                                      \
| rwgroup --id-fields=3 --rec-threshold=20                      \
        --group-offset=0.1.0.0                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=3 data.rwf                                      \
| rwgroup --id-fields=3                                         \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwidsquery --intype=fast --year=2009                            \
        --dry-run sk-teststmp-5VryFELy 2>&1

rwidsquery --intype=full --year=2009                            \
        --dry-run sk-teststmp-uQF5FsDM 2>&1

rwidsquery --intype=rule --start-date=2009/02/11:10             \
        --end-date=2009/02/11:12                                \
        --dry-run sk-teststmp-pMGw7bWu 2>&1

rwsilk2ipfix data-v6.rwf                                        \
| rwipfix2silk --silk-output=/dev/null --print-stat 2>&1
```

```
rwsilk2ipfix data.rwf                                          \
| rwipfix2silk --silk-output=/dev/null --print-stat 2>&1


rwsilk2ipfix data-v6.rwf --ipfix-output=/dev/null              \
        --print-stat 2>&1


rwsilk2ipfix data.rwf --ipfix-output=/dev/null                 \
        --print-stat 2>&1


rwsilk2ipfix data-v6.rwf                                       \
| rwipfix2silk --silk-output=stdout                            \
| rwcat --compression-method=none --byte-order=little


rwsilk2ipfix data.rwf                                          \
| rwipfix2silk --silk-output=stdout                            \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


rwsilk2ipfix empty.rwf data.rwf empty.rwf                      \
| rwipfix2silk                                                 \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


cat data.rwf                                                   \
| rwsilk2ipfix --ipfix-output=stdout                           \
| rwipfix2silk                                                 \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


rwfilter --daddr=192.168.x.x --sport=0-1024                    \
        --pass=stdout data.rwf                                 \
| rwsort --fields=1,4,2,3,5,9 --output=./sk-teststmp-SwpwLuen  \
&& rwfilter --saddr=192.168.x.x --dport=0-1024                 \
        --pass=stdout data.rwf                                 \
| rwsort --fields=2,3,1,4,5,9 --output=./sk-teststmp-DMIINO23  \
&& rwmatch --time-delta=2.5 --symmetric-del --relative-del     \
        --relate=5,5 --relate=1,2 --relate=3,4 --relate=2,1    \
        --relate=4,3 sk-teststmp-DMIINO23 sk-teststmp-SwpwLuen - \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


rwfilter --daddr=192.168.x.x --dport=0-1024                    \
        --pass=stdout data.rwf                                 \
| rwsort --fields=1,4,2,3,5,9 --output=./sk-teststmp-PXDr1uGA  \
&& rwfilter --saddr=192.168.x.x --sport=0-1024                 \
        --pass=stdout data.rwf                                 \
| rwsort --fields=2,3,1,4,5,9 --output=./sk-teststmp-UG2H2ebA  \
```

```
&& rwmatch --time-delta=2.5 --symmetric-del --relative-del      \
        --relate=5,5 --relate=1,2 --relate=3,4 --relate=2,1     \
        --relate=4,3 sk-teststmp-PXDr1uGA sk-teststmp-UG2H2ebA - \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwnetmask --6dip-prefix=64 --6sip-prefix=120 data-v6.rwf        \
| rwcat --compression-method=none --byte-order=little

rwnetmask --6sip-prefix-length=120 data-v6.rwf                  \
| rwcat --compression-method=none --byte-order=little

rwnetmask --dip-prefix=16 --sip-prefix=24 data.rwf              \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwnetmask --sip-prefix-length=24 data.rwf                       \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

cat data.rwf                                                    \
| rwnetmask --sip-prefix-length=24                              \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwrandomizeip --seed=38901 --consistent data.rwf -              \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwrandomizeip --seed=38901                                      \
        --save-table=stdout data.rwf /dev/null                  \
| rwrandomizeip --load-table=stdin data.rwf -                    \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwrandomizeip --seed=38901 data.rwf stdout                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

cat data.rwf                                                    \
| rwrandomizeip --seed=38901 --consistent - -                   \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

echo '0.0.0.0|0.0.0.0|'                                         \
| rwresolve --ip-fields=4,8 --column-width=20
```

```
echo '0.0.0.0|0.0.0.0|'                                          \
| rwresolve --column-width=20

echo '0.0.0.0,0.0.0.0'                                           \
| rwresolve --delimiter=, --column-width=20

echo '0.0.0.0|0.0.0.0'                                           \
| rwresolve --column-width=20

echo '0.0.0.0|0.0.0.0|'                                          \
| rwresolve --ip-fields=1 --column-width=20

echo '0.0.0.0|0.0.0.0|'                                          \
| rwresolve --ip-fields=1,4 --column-width=20

rwfilter --daddr=192.168.0.0/16 --pass=stdout data.rwf           \
| rwsort --fields=sip,proto,dip - scandata.rwf                   \
| rwscan --scan-mode=2

rwfilter --daddr=192.168.0.0/16                                  \
        --pass=./sk-teststmp-n4NBXRQs data.rwf                   \
&& rwset --dip=./sk-teststmp-L9eojb9S sk-teststmp-n4NBXRQs       \
&& rwsort --fields=sip,proto,dip sk-teststmp-n4NBXRQs scandata.rwf \
| rwscan --trw-sip-set=./sk-teststmp-L9eojb9S

rwfilter --daddr=192.168.0.0/16                                  \
        --pass=./sk-teststmp-MSmN4m31 data.rwf                   \
&& rwset --dip=./sk-teststmp-v6JizDuE sk-teststmp-MSmN4m31       \
&& rwsort --fields=sip,proto,dip sk-teststmp-MSmN4m31 scandata.rwf \
| rwscan --scan-mode=1 --trw-sip-set=./sk-teststmp-v6JizDuE

rwset --sip-file=/dev/null --copy-input=stdout data.rwf          \
| rwset --sip-file=-                                            \
| rwsetcat --print-ips

rwset --dip-file=stdout data.rwf                                 \
| rwsetcat

rwset --sip-file=stdout empty.rwf data.rwf empty.rwf             \
| rwsetcat

rwset --nhip-file=stdout data.rwf                                \
| rwsetcat

rwset --sip=stdout --dip=/dev/null data.rwf                      \
| rwsetcat
```

```
rwset --sip=/dev/null --dip=stdout data.rwf                    \
| rwsetcat

rwset --sip-file=stdout data.rwf                               \
| rwsetcat

cat data.rwf                                                   \
| rwset --sip-file=stdout                                      \
| rwsetcat

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --cidr-blocks                                       \
| rwsetbuild                                                   \
| rwsetcat

rwset --sip-file=stdout data.rwf                               \
| rwsetcat                                                     \
| rwsetbuild stdin                                             \
| rwsetcat

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --ip-ranges --delim=,                               \
| cut -d, -f2,3                                                \
| rwsetbuild --ip-ranges=, - -                                 \
| rwsetcat

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --cidr-blocks

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --count-ips

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --integer-ips

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --ip-ranges

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --network-structure=12TS,12

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --network-structure=ATS

rwset --sip-file=stdout data.rwf                               \
| rwsetcat --network-structure
```

```
rwset --sip-file=stdout data.rwf                                \
| rwsetcat --zero-pad-ips stdin

rwset --sip-file=stdout data.rwf                                \
| rwsetmember --count 192.168.x.x -

rwset --sip-file=stdout data.rwf                                \
| rwsetmember 192.168.x.x stdin

rwset --sip-file=stdout data.rwf                                \
| rwsettool --union --output-path=stdout                       \
| rwsetcat

rwset --sip-file=stdout data.rwf                                \
| rwsettool --union stdin                                      \
| rwsetcat

rwdedupe --buffer-size=10m data.rwf                             \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwdedupe --ignore-fields=stime data.rwf                         \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwdedupe data-v6.rwf                                            \
| rwuniq --fields=1-5 --ipv6-policy=force --epoch-time          \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwdedupe data.rwf                                               \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwcat data-v6.rwf data-v6.rwf                                   \
| rwdedupe                                                      \
| rwuniq --fields=1-5 --ipv6-policy=force --epoch-time          \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwcat data.rwf data.rwf                                         \
| rwdedupe                                                      \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title
```

```
rwsort --fields=dtype data.rwf                                  \
| rwuniq --fields=dtype --values=dip-distinct --delimited       \
        --ipv6=ignore --presorted-input


rwsort --fields=stype data.rwf                                  \
| rwuniq --fields=stype --values=sip-distinct --delimited       \
        --ipv6=ignore --presorted-input


rwsort --fields=bytes data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwsort --fields=class,type,sensor data.rwf                      \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwsort --fields=dcc data.rwf                                    \
| rwuniq --fields=dcc --values=dip-distinct --ipv6=ignore       \
        --presorted-input


rwsort --fields=scc data.rwf                                    \
| rwuniq --fields=scc --values=sip-distinct --ipv6=ignore       \
        --presorted-input


rwsort --fields=dip data-v6.rwf                                 \
| rwcat --compression-method=none --byte-order=little


rwsort --fields=dip data.rwf                                    \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwsort --fields=10 data.rwf                                     \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output


rwsort --plugin=flowrate.so --fields=bytes/sec data.rwf         \
| rwuniq --plugin=flowrate.so --fields=bytes/sec                \
        --values=bytes --presorted-input


rwsort --plugin=flowrate.so --fields=payload-bytes data.rwf     \
| rwuniq --plugin=flowrate.so --fields=payload-bytes            \
        --values=bytes,packets,records --presorted-input


rwsort --plugin=flowrate.so --fields=pckts/sec data.rwf         \
| rwuniq --plugin=flowrate.so --fields=pckts/sec                \
        --values=packets --presorted-input
```

```
cat data.rwf                                                    \
| rwsort --field=9,1 --input-pipe=stdin                         \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=5,1,3,2,4 data.rwf                              \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time         \
        --values=bytes,packets,records,stime,etime              \
        --sort-output --delimited --no-title

rwsort --field=9,1 data.rwf data.rwf                            \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --field=9,1 --output-path=stdout data.rwf                \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=servhost:ip-map.pmap                       \
        --fields=dst-servhost                                   \
| rwuniq --pmap-file=servhost:ip-map.pmap                       \
        --fields=dst-servhost --presorted-input

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=service-port:proto-port-map.pmap           \
        --pmap-file=ip-map.pmap                                 \
        --fields=src-service-host,src-service-port              \
| rwuniq --pmap-file=service-port:proto-port-map.pmap           \
        --pmap-file=ip-map.pmap                                 \
        --fields=src-service-host,src-service-port              \
        --presorted-input

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=proto-port-map.pmap --fields=sval          \
| rwuniq --pmap-file=proto-port-map.pmap --fields=sval          \
        --presorted-input

rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --pmap-file=ip-map.pmap --fields=src-service-host      \
| rwuniq --pmap-file=ip-map.pmap --fields=src-service-host      \
        --presorted-input

rwsort --fields=5,3-4 data-v6.rwf                               \
| rwcat --compression-method=none --byte-order=little

rwsort --fields=5,3-4 data.rwf                                  \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output
```

```
rwsort --python-file=pysilk-plugin.py                          \
        --fields=lower_port data.rwf                           \
| rwuniq --python-file=pysilk-plugin.py --fields=lower_port     \
        --values=bytes --presorted-input


rwsort --python-file=pysilk-plugin.py                          \
        --fields=proto_name data.rwf                           \
| rwuniq --python-file=pysilk-plugin.py --fields=proto_name     \
        --values=bytes --presorted-input


rwsort --python-file=pysilk-plugin.py                          \
        --fields=lower_port_simple data.rwf                    \
| rwuniq --python-file=pysilk-plugin.py                        \
        --fields=lower_port_simple --values=bytes              \
        --presorted-input


rwsort --python-file=pysilk-plugin.py                          \
        --fields=server_ip data.rwf                            \
| rwuniq --python-file=pysilk-plugin.py --fields=server_ip      \
        --values=bytes --presorted-input


rwsort --python-file=pysilk-plugin.py                          \
        --fields=server_ipv6 data-v6.rwf                       \
| rwuniq --python-file=pysilk-plugin.py --fields=server_ipv6    \
        --values=bytes --presorted-input


rwsort --fields=6 --reverse data.rwf                           \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


cat data.rwf                                                   \
| rwuniq --fields=1-5 --ipv6-policy=ignore --epoch-time        \
        --values=bytes,packets,records,stime,etime             \
        --sort-output --delimited --no-title


rwsort --fields=1 data-v6.rwf                                  \
| rwcat --compression-method=none --byte-order=little


rwsort --fields=1 data.rwf                                     \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output


rwsort --plugin=skplugin-test.so --fields=copy-bytes data.rwf  \
| rwuniq --plugin=skplugin-test.so --ipv6-policy=ignore        \
        --fields=copy-bytes --values=bytes,packets,records     \
        --presorted-input
```

```
rwsort --field=9,1 --sort-buffer-size=10M data.rwf              \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

cat data.rwf                                                    \
| rwsort --field=9,1                                           \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsort --fields=9,1 data.rwf                                    \
| rwcat --compression-method=none --byte-order=little           \
        --ipv4-output

rwsplit --basename=$temp --byte-limit=10000000 --seed=737292    \
        --file-ratio=800 data.rwf                               \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --byte-limit=10000000                  \
        --max-outputs=4 data.rwf                                \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --byte-limit=10000000 --seed=737292    \
        --sample-ratio=1000 data.rwf                            \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --flow-limit=10000 data.rwf            \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --ip-limit=5000 data.rwf               \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --packet-limit=10000000 data.rwf       \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwsplit --basename=$temp --packet-limit=50000 --seed=737292     \
        --sample-ratio=20 --file-ratio=10 data.rwf              \
&& rwcat --compression-method=none --byte-order=little          \
        --ipv4-output $temp*

rwstats --fields=dtype --values=dip-distinct --delimited        \
        --ipv6=ignore --count=2 --no-percent data.rwf
```

```
rwstats --fields=stype --values=sip-distinct --delimited       \
        --ipv6=ignore --count=2 --no-percent data.rwf


rwstats --count=10 --fields=dip --column-sep=/ --top           \
        --ipv6-policy=ignore data.rwf


rwstats --fields=sip --top --count=10 --output-path=/dev/null  \
        --copy-input=stdout data.rwf                           \
| rwstats --fields=dip --count=10 --ipv6-policy=ignore


rwstats --fields=dcc --values=dip-distinct --ipv6=ignore       \
        --count=10 --no-percent data.rwf


rwstats --fields=scc --values=sip-distinct --ipv6=ignore       \
        --count=10 --no-percent data.rwf


rwstats --fields=dip --count=10 --delimited=, --top            \
        --ipv6-policy=ignore data.rwf


rwstats --fields=dip --values=records --percentage=4           \
        --ipv6-policy=ignore data.rwf


rwstats --fields=dip --values=packets --threshold=25000        \
        --top data-v6.rwf


rwstats --fields=dip --values=packets --threshold=25000 --top  \
        --ipv6-policy=ignore data.rwf


rwstats --dip=16 --values=bytes --count=10 --bottom            \
        --ipv6-policy=ignore data.rwf


rwfilter --dport=0-66,69-1023,8080 --pass=- data.rwf           \
| rwstats --fields=dport --bottom --values=bytes --count=20


rwstats --fields=dport --values=dip-distinct,records           \
        --threshold=5000 --no-percent data.rwf


rwstats --fields=dport --threshold=8000 --top data.rwf


rwstats --plugin=flowrate.so --fields=bytes/sec                \
        --values=bytes --count=10 data.rwf


rwstats --plugin=flowrate.so --fields=payload-bytes            \
        --values=bytes,packets,records --count=10 data.rwf
```

```
rwstats --plugin=flowrate.so --fields=pckts/sec                \
        --values=packets --count=10 data.rwf

rwfilter --proto=1 --pass=- data.rwf                           \
| rwstats --icmp --byte --percentage=5

rwstats --fields=sip,dip --values=bytes --count=8 --top        \
        --integer-ips --ipv6-policy=ignore data.rwf

rwstats --fields=dport --values=bytes --count=20               \
        --top empty.rwf data-v6.rwf empty.rwf data.rwf

rwstats --fields=dport --values=bytes --count=20               \
        --top data-v6.rwf data-v6.rwf empty.rwf

rwstats --fields=dport --values=bytes --count=20               \
        --top data.rwf empty.rwf data.rwf

rwstats --fields=dip --count=10 --top --no-column              \
        --column-sep=, --ipv6-policy=ignore data.rwf

rwstats --fields=dip --count=10 --top --no-titles              \
        --ipv6-policy=ignore data.rwf

rwstats --overall-stats data.rwf

rwfilter --type=in,inweb --pass=stdout data.rwf                \
| rwstats --pmap-file=servhost:ip-map.pmap                     \
        --fields=dst-servhost --count=10

rwfilter --type=in,inweb --pass=stdout data.rwf                \
| rwstats --pmap-file=service-port:proto-port-map.pmap         \
        --pmap-file=ip-map.pmap                                \
        --fields=src-service-host,src-service-port --count=10

rwfilter --type=in,inweb --pass=stdout data.rwf                \
| rwstats --pmap-file=proto-port-map.pmap --fields=sval        \
        --bottom --count=10

rwfilter --type=in,inweb --pass=stdout data.rwf                \
| rwstats --pmap-file=ip-map.pmap --fields=src-service-host     \
        --count=10

rwstats --fields=protocol --values=packets --count=15          \
        --bottom data.rwf
```

```
rwstats --fields=proto --values=sip-distinct,dip-distinct       \
        --count=5 --ipv6=ignore --no-percent data.rwf


rwstats --detail-proto-stats=1 data.rwf


rwstats --fields=protocol --values=packets --count=15 data.rwf


rwstats --python-file=pysilk-plugin.py --fields=lower_port      \
        --values=max_bytes --count=10 --no-percent data.rwf


rwstats --python-file=pysilk-plugin.py --fields=lower_port      \
        --value=bytes --count=10 --no-percent data.rwf


rwstats --python-file=pysilk-plugin.py                          \
        --fields=lower_port_simple                              \
        --values=large_packet_flows,largest_packets,smallest_packets \
        --count=5 --no-percent data.rwf


rwstats --python-file=pysilk-plugin.py --fields=sip             \
        --values=max_bytes --ipv6=ignore --count=10             \
        --no-percent data.rwf


rwstats --fields=sip,dip --values=bytes --count=8               \
        --top data-v6.rwf


rwstats --fields=sip,dip --values=bytes --count=8 --top         \
        --ipv6-policy=ignore data.rwf


rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwstats --fields=sport,dport --count=5


rwstats --fields=sip --values=bytes --count=100 --top           \
        --ipv6-policy=ignore data.rwf


rwstats --fields=sip --percentage=4 --top data-v6.rwf


rwstats --fields=sip --percentage=4 --top                       \
        --ipv6-policy=ignore data.rwf


rwstats --sip=24 --values=packets --percentage=1 --top          \
        --ipv6-policy=ignore data.rwf


rwstats --sip=24 --values=packets --percentage=2 --top          \
        --ipv6-policy=ignore data.rwf
```

```
rwstats --plugin=skplugin-test.so --fields=copy-bytes          \
        --values=bytes,packets,records --count=10 data.rwf


rwfilter --sport=0-66,69-1023,8080 --pass=- data.rwf           \
| rwstats --fields=sport --values=records --bottom --count=4


rwstats --fields=sport --values=sip-distinct --threshold=5000  \
        --no-percent data.rwf


rwstats --fields=sport,sip --values=packets,bytes --count=10   \
        --ipv6-policy=ignore data.rwf


rwstats --fields=sport --percentage=5 data.rwf


cat data.rwf                                                   \
| rwstats --fields=dip --top --count=10 --ipv6-policy=ignore


rwswapbytes --big-endian data-v6.rwf stdout                    \
| rwcut --fields=1-15,26-29 --epoch-time


rwswapbytes --big-endian data.rwf stdout                       \
| rwcut --fields=1-15,26-29 --integer-ips --epoch-time         \
        --ipv6-policy=ignore


rwswapbytes --little-endian data-v6.rwf -                      \
| rwcut --fields=1-15,26-29 --epoch-time


rwswapbytes --little-endian data.rwf -                         \
| rwcut --fields=1-15,26-29 --integer-ips --epoch-time         \
        --ipv6-policy=ignore


cat data.rwf                                                   \
| rwswapbytes --big - -                                        \
| rwcut --fields=1-15,26-29 --integer-ips --epoch-time         \
        --ipv6-policy=ignore


rwswapbytes --swap-endian data.rwf stdout                      \
| rwcut --fields=1-15,26-29 --integer-ips --epoch-time         \
        --ipv6-policy=ignore


rwtotal --bytes --skip-zero data.rwf


rwtotal --sport --output-path=/dev/null                        \
        --copy-input=stdout data.rwf                           \
| rwtotal --sport --skip-zero
```

```
rwtotal --sport --delimited --skip-zero data.rwf

rwtotal --dip-first-16 --skip-zero data.rwf

rwtotal --dip-first-24 --skip-zero data.rwf

rwtotal --dip-first-8 data.rwf

rwtotal --dip-last-16 --skip-zero data.rwf

rwtotal --dip-last-8 data.rwf

rwtotal --dport data.rwf

rwtotal --duration --skip-zero data.rwf

rwfilter --proto=1 --pass=- data.rwf                            \
| rwtotal --icmp-code

rwtotal --sport                                                 \
        --skip-zero empty.rwf data.rwf data-v6.rwf empty.rwf

rwtotal --sport --skip-zero data-v6.rwf empty.rwf data-v6.rwf

rwtotal --sport --skip-zero data.rwf empty.rwf data.rwf

rwtotal --sport --no-column --column-sep=, data.rwf

rwtotal --sport --no-titles data.rwf

rwtotal --packets --skip-zero data.rwf

rwtotal --proto data.rwf

rwtotal --sip-first-16 --skip-zero data.rwf

rwtotal --sip-first-24 --skip-zero data.rwf

rwtotal --sip-first-8 data.rwf

rwtotal --sip-last-16 --skip-zero data.rwf

rwtotal --sip-last-8 data.rwf
```

```
rwtotal --sport --min-byte=2000 data.rwf

rwtotal --sport --min-packet=20 data.rwf

rwtotal --sport --min-record=10 data.rwf

rwtotal --sport --max-byte=2000 --skip-zero data.rwf

rwtotal --sport --max-packet=20 --skip-zero data.rwf

rwtotal --sport --max-record=10 --skip-zero data.rwf

cat data.rwf                                                    \
| rwtotal --sport --skip-zero

rwtotal --sport --summation --skip-zero data.rwf

rwcut --fields=sip,dip,sport,dport,proto,packets,bytes,stime,dur,sensor,class,type,in,out,application,initialflags,sessionfl
| rwtuc                                                         \
| rwcat --compression-method=none --byte-order=little

rwcut --fields=sip,dip,sport,dport,proto,packets,bytes,stime,dur,sensor,class,type,in,out,application,initialflags,sessionfl
| rwtuc                                                         \
| rwcat --compression-method=none --byte-order=little          \
        --ipv4-output

rwuniq --fields=stype,proto --values=packets                    \
    --sort-output data.rwf

rwuniq --fields=dtype --values=dip-distinct --delimited         \
        --ipv6=ignore --sort-output data.rwf

rwuniq --fields=stype --values=sip-distinct --delimited         \
        --ipv6=ignore --sort-output data.rwf

rwuniq --fields=sensor,class,type --sort-output data.rwf

rwuniq --fields=sport --output-path=/dev/null                   \
        --copy-input=stdout data.rwf                            \
| rwuniq --fields=sport --sort-output

rwuniq --fields=dcc --values=dip-distinct --ipv6=ignore         \
        --sort-output data.rwf
```

```
rwuniq --fields=scc --values=sip-distinct --ipv6=ignore         \
        --sort-output data.rwf


rwuniq --fields=sport --delimited --sort-output data.rwf


rwuniq --fields=2 --ipv6-policy=ignore --integer-ips --bytes    \
        --sort-output data.rwf


rwuniq --fields=2 --values=packets --ipv6-policy=force          \
        --sort-output data-v6.rwf


rwuniq --fields=dip --ipv6-policy=ignore --zero-pad-ips         \
        --packets --sort-output data.rwf


rwuniq --fields=dport --all-counts --sort-output data.rwf


rwuniq --fields=dur --bytes --sort-output data.rwf


rwuniq --fields=etime --epoch-time --sort-output data.rwf


rwuniq --plugin=flowrate.so --fields=bytes/sec --values=bytes   \
        --sort-output data.rwf


rwuniq --plugin=flowrate.so --fields=payload-bytes              \
        --values=bytes,packets,records --sort-output data.rwf


rwuniq --plugin=flowrate.so --fields=pckts/sec                  \
        --values=packets --sort-output data.rwf


rwfilter --proto=1 --pass=- data.rwf                            \
| rwuniq --fields=icmpTypeCode --sort-output


rwuniq --fields=9,11 --legacy-time=0 --sort-output data.rwf


rwuniq --fields=9,11 --legacy-time=1 --sort-output data.rwf


rwuniq --fields=sport                                           \
        --sort-output empty.rwf data.rwf empty.rwf


rwuniq --fields=sport --no-column --column-sep=,                \
        --sort-output data.rwf


rwuniq --fields=sport --no-titles --sort-output data.rwf
```

```
rwuniq --pmap-file=servhost:ip-map.pmap --fields=dst-servhost   \
        --sort-output data.rwf


rwuniq --pmap-file=service-port:proto-port-map.pmap             \
        --pmap-file=ip-map.pmap                                 \
        --fields=src-service-host,src-service-port              \
        --sort-output data.rwf


rwuniq --pmap-file=proto-port-map.pmap --fields=sval            \
        --sort-output data.rwf


rwuniq --pmap-file=ip-map.pmap --fields=src-service-host        \
        --sort-output data.rwf


rwfilter --type=in,inweb --pass=stdout data.rwf                 \
| rwsort --fields=3-5 --output-path=./sk-teststmp-0NQ5n7oZ      \
&& rwfilter --type=in,inweb --fail=stdout data.rwf              \
| rwsort --fields=3-5 --output-path=./sk-teststmp-SoUHr78K      \
&& rwuniq --fields=3-5 --presorted-input                        \
        --no-title sk-teststmp-0NQ5n7oZ sk-teststmp-SoUHr78K


rwsort --fields=3-5 data.rwf                                    \
| rwuniq --fields=3-5 --presorted-input --no-title


rwuniq --fields=sport,dport,proto --no-title                    \
        --sort-output data-v6.rwf


rwuniq --fields=sport,dport,proto --no-title                    \
        --sort-output data.rwf


rwuniq --fields=3-5 --no-title data.rwf                         \
| sort


rwuniq --fields=proto --sort-output data.rwf


rwuniq --python-file=pysilk-plugin.py --fields=lower_port       \
        --values=max_bytes --sort-output data.rwf


rwuniq --python-file=pysilk-plugin.py --fields=lower_port       \
        --value=bytes --sort-output data.rwf


rwuniq --python-file=pysilk-plugin.py                           \
        --fields=lower_port_simple                              \
        --values=large_packet_flows,largest_packets,smallest_packets \
        --sort-output data.rwf
```

```
rwuniq --python-file=pysilk-plugin.py --fields=sip              \
        --values=max_bytes --ipv6=ignore --sort-output data.rwf

rwuniq --fields=sip --values=bytes --sort-output data-v6.rwf

rwuniq --fields=sip --bytes --ipv6-policy=ignore                \
        --sort-output data.rwf

rwuniq --plugin=skplugin-test.so --ipv6-policy=ignore           \
        --no-column --fields=protocol                           \
        --values=bytes,sum-bytes,min-bytes,max-bytes,weird-bytes \
        --sort-output data.rwf

rwsort --fields=sport data.rwf                                  \
| rwuniq --fields=sport --sip-distinct --dip-distinct           \
        --presorted-input --ipv6-policy=ignore

rwuniq --fields=sport --sip-distinct --dip-distinct             \
        --ipv6-policy=ignore --sort-output data.rwf

rwsort --fields=sport data.rwf                                  \
| rwuniq --fields=sport --sip-distinct --presorted-input        \
        --ipv6-policy=ignore

rwuniq --fields=sport --sip-distinct --sort-output data-v6.rwf

rwuniq --fields=sport --sip-distinct --sort-output data.rwf

rwuniq --fields=sport --bytes=2000 --sort-output data.rwf

rwuniq --fields=sport --packets=20 --sort-output data.rwf

rwuniq --fields=sport --flows=10 --sort-output data.rwf

rwuniq --fields=sport --bytes=0-2000 --sort-output data.rwf

rwuniq --fields=sport --packets=0-20 --sort-output data.rwf

rwuniq --fields=sport --flows=0-10 --sort-output data.rwf

cat data.rwf                                                    \
| rwuniq --fields=sport --sort-output

rwuniq --fields=stime --bin-time=3600 --sort-output data.rwf

rwuniq --fields=stime --packets --flows --sort-output data.rwf

rwuniq --fields=stime,proto --bin-time=86400                    \
    --sort-output data.rwf
```

8.9 Perform a checksum of the output–failure

The following tests perform a variety of checks for error conditions. The output of the command is gathered and compared to a known checksum (MD5). In all cases, the application should exit with a non-zero exit status.

```
rwfilter --proto=0- --max-pass=10000 --pass=- data.rwf          \
| rwcompare data.rwf - 2>&1

rwcompare --quiet empty.rwf data.rwf

rwpackchecker --value max-tcp-bpp=5000                          \
        --allowable-count max-tcp-bpp=1 data.rwf

rwpackchecker --value match-sport=123                           \
        --value match-dport=123 data.rwf

echo 172.16-31.x.x                                              \
| rwsetbuild - -                                               \
| rwpackchecker --value match-sip=- data.rwf

rwgroup --id-field=3 --delta-value=10 empty.rwf 2>&1

rwgroup --delta-field=9 empty.rwf 2>&1

rwgroup --id-fields=3 data.rwf empty.rwf 2>&1

rwset --sip=- empty.rwf                                         \
| rwipaimport --catalog=my-cat --description=my-description     \
        --start-time=2009/02/12:00:00 - 2>&1

rwset --sip=- empty.rwf                                         \
| rwipaimport --catalog=my-cat --description=my-description     \
        --end-time=2009/02/14:23:59:59 - 2>&1

rwmatch --relate=1,2 data.rwf 2>&1

rwmatch --relate=1,2 data.rwf data.rwf 2>&1

rwscan empty.rwf 2>&1

rwuniq --fields=sport --sip-distinct --dip-distinct             \
        --sort-output data-v6.rwf 2>&1

cat data-v6.rwf                                                 \
| rwuniq --fields=sport --sip-distinct --presorted-input 2>&1
```

8.10 Comparing checksums

The following tests perform a variety of checks. Multiple commands are run and the output of those commands are gathered. The checksum (MD5) of the outputs are compared to ensure the outputs are identical.

```
rwcat --byte-order=little empty.rwf                              \
| rwfileinfo --fields=byte-order --no-title -

rwfileinfo --fields=3 --no-title empty.rwf


rwcat --compression-method=none empty.rwf                        \
| rwfileinfo --fields=compression --no-title -

rwfileinfo --fields=4 --no-title empty.rwf
```