Design Document for IP Fabrics

Author: May06-15 (Network Forensic UI)

Andy Heintz (Communication Leader)

Abraham Devine (Webmaster)

Altay Ozen (Team Leader and Team Key Concept Holder)

Dr. Joseph Zambreno (Adviser)

Curt Schwaderer (Client)

Version	Date	Author	Change
1.0	10/26	AH	Created initial version of design document
2.0	11/23	AH	Created final version of design document
3.0	3/12	AH	Updated design document
4.0	4/26	AH	Added manual and reorganized document

Table of Contents

1 Problem Statement

Currently, DeepSweep, a device/program made by the client, emits traffic as a stream of data and does not provide a graphical user interface for viewing the output. While there are tools for processing traffic, many of them require input as pcap files or some format other than the raw stream of data. Therefore, our project's goal is to develop an interface between an existing open source application for processing internet traffic and DeepSweep. Once the interface was developed, we extended the other program to add additional functionality for our client.

2 Project Design

2.1 FUNCTIONAL REQUIREMENTS

Initial Version

1. Xplico (modified): Starts collector (ASN or Raw) based on user's selection
2. Raw Collector (new): Collects raw traffic from DeepSweep and creates pcaps
3. Xplico Pusher (existing): Decodes pcap files and stores their data
4. Xplico (existing): Displays data from uploaded pcaps in its web GUI

(The primary purpose is to connect all components of the system.)

Final Version

1. Xplico (modified): Starts collector (ASN or Raw) based on user's selection
2. Raw Collector (new): Collects raw traffic from DeepSweep and creates pcaps
3. ASN Collector (new): Collects ASN.1 traffic from DeepSweep and creates pcaps
5. Xplico Pusher (existing): Decodes pcap files and stores their data
4. Xplico (existing): Displays data from uploaded pcaps in its web GUI
5. Xplico (modified): Provides options for generating reports of uploaded traffic

2.2 GLOSSARY

Term	Definition
Xplico	Existing open source program for viewing network traffic
DeepSweep	Existing device for capturing network traffic
Case	Stores all data from a particular endpoint (e.g. DeepSweep port)
Session	Contains data from a set time period (Case contains Sessions)
Raw	Networking data in pcap format (for this project)
ASN / ASN.1	Abstract Syntax Notation – Format for representing data in networking
Pcap	Packet Capture file – Contains captured network data
Decoding	Converting pcap into human-readable data

Table 1: Glossary of Terms

2.3 NON-FUNCTIONAL REQUIREMENTS

1. Reliability: Application should return information consistently and recover from errors.

2. Maintainability: Documentation should be up-to-date and accurate; application should be easy to upgrade to the latest version.

3. Security: Application should protect users' passwords and data from unauthorized access.

4. Extensibility: Future developers should be able to easily add additional features using the existing architecture.

5. Response Time: Application should respond promptly to users' queries and requests.

2.4 FUNCTIONAL DECOMPOSITION

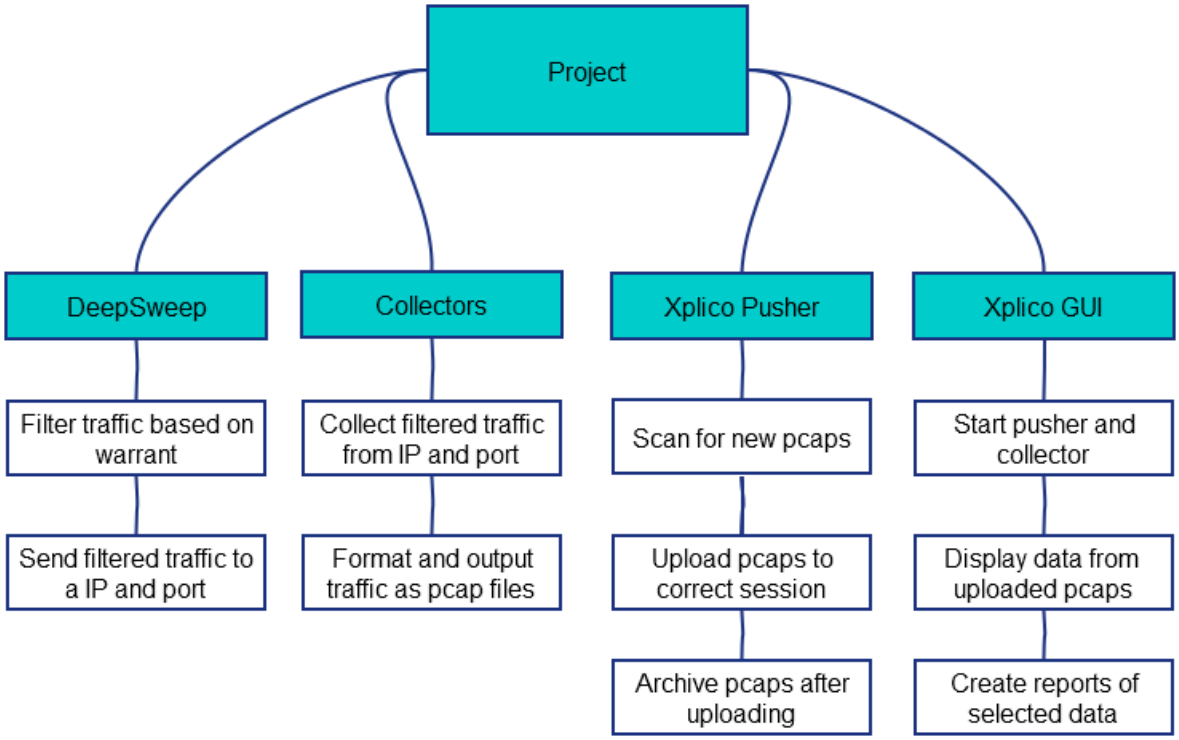

Figure 1: Functional Decomposition Diagram

2.5 SOFTWARE DESIGN

As our project is entirely software, it has no hardware or electrical CAD drawings.

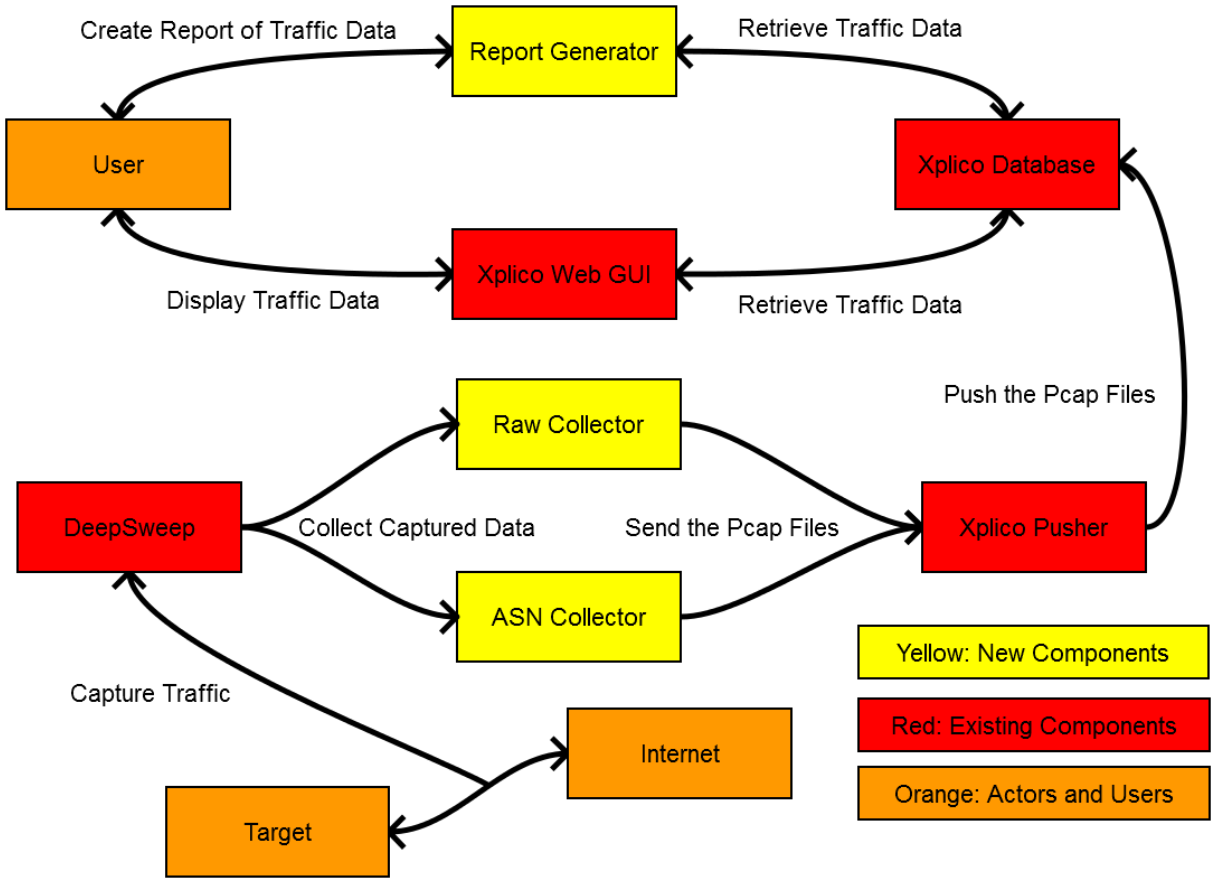

Figure 2: System Diagram

A user configures the DeepSweep device, setting a filter that will capture only traffic from the desired target. The user will also set whether its output is ASN or Raw. Once configured, they start the corresponding collector from a session in Xplico.

The collector listens for traffic from Deepsweep and creates pcap files containing this traffic. Xplico's pusher scans its session's directory for new pcap files and decodes them. Once the files are decoded, it puts the data into Xplico's database.

At this point, the user can view the pcap data in Xplico's GUI. Once uploaded, the pcaps are moved to another directory, so the user can retrieve the original pcaps if necessary. In addition to viewing the traffic in Xplico's web GUI, users also can create PDF reports of the traffic.

3 Implementation Details

3.1 XPLICO GUI

Xplico provides a user interface for viewing decoded pcaps. This user interface is a web based UI developed in PHP, using a framework called CakePHP.

Our project modified Xplico's existing GUI and added additional functionality requested by the client. The below images show what the screens look like after modification.

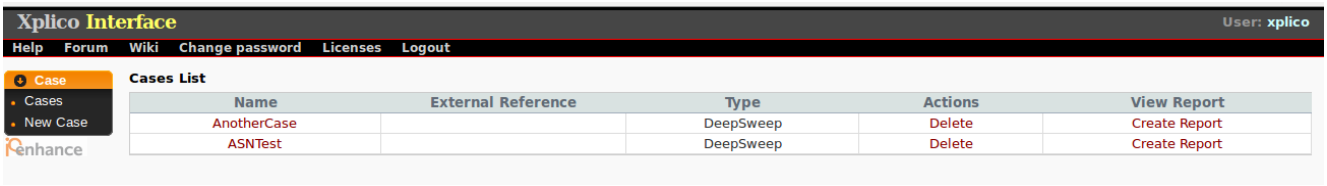

Figure 3: Case Index Page

This page lists all cases created by the user, which contain sessions that store uploaded pcap data. This page was modified to add an option for generating a report for a case.

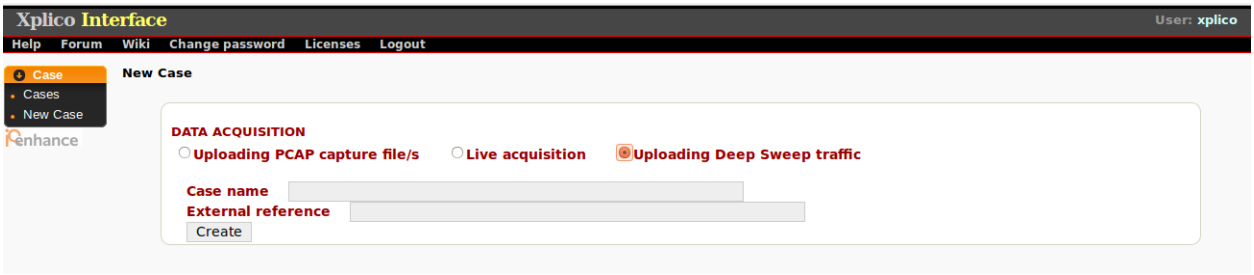

Figure 4: New Case Page

This page provides options for creating new cases. This page was modified to add an option for creating a case that uploads traffic from DeepSweep.

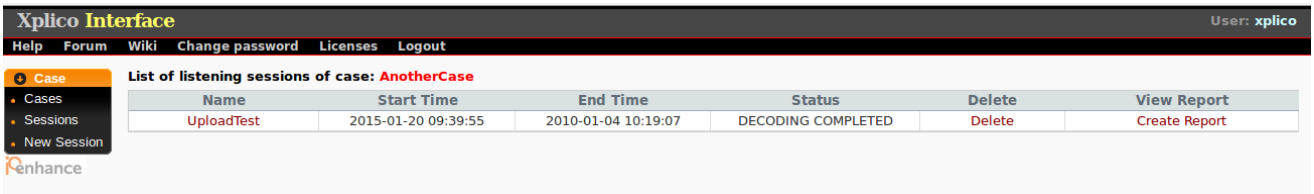

Figure 5: Session Index Page

This page lists all sessions created by the user, which contains data from uploaded pcaps. This page was modified to add an option for generating a report for a session.

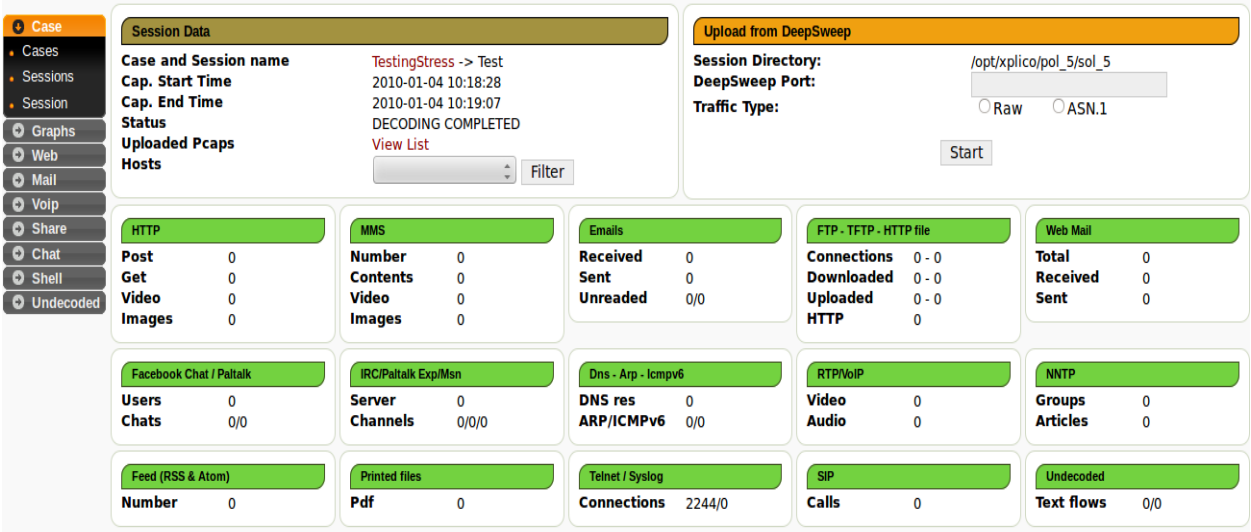

Figure 6: Session View Page

This page lists a summary of the data uploaded for one session. On the left, a user can choose to view specific types of data, e.g. web, mail, chat.

This page was modified to provide an option for starting/stopping a collector. The user inputs the port number used by DeepSweep, selects whether traffic from DeepSweep is Raw or ASN, and presses the Start button.

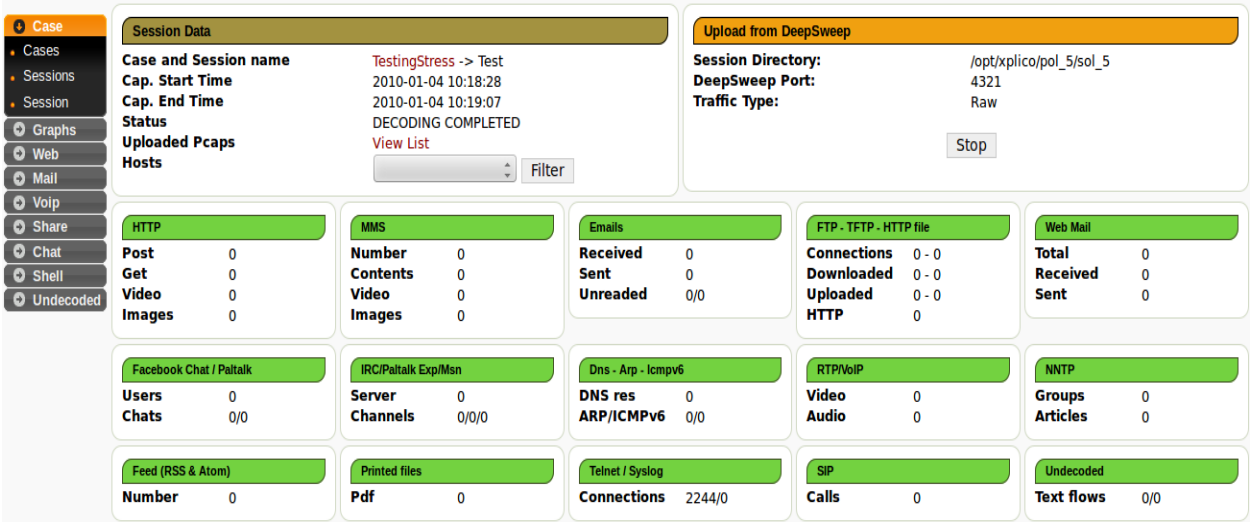

Figure 7: Session View Page

Once the collector is started, it uploads data until the user presses the Stop button or until the collector terminates due to an error.

3.2 STANDARDS

ASN.1 (Abstract Syntax Notation)

This standard is used by DeepSweep for formatting traffic. ASN.1 data can be encoded as either BER (Basic Encoding Rules) or DER (Distinguished Encoding Rules). DeepSweep can send ASN.1 data using either encoding rule. Both of these formats are handled by the ASN Collector.

Details can be found here: http://luca.ntop.org/Teaching/Appunti/asn1.html

Pcap Format

This standard is also used by DeepSweep for formatting traffic. This format is handled by the Raw Collector. Both collectors format their input as pcaps and output this data into pcap files.

Details can be found here: https://wiki.wireshark.org/Development/LibpcapFileFormat

3.3 COLLECTORS

The collectors provide an interface between DeepSweep and Xplico. Both collectors are written in Java and run as command line programs. The ASN Collector makes use of a library called Cream, which converts ASN traffic into actual raw traffic.

Each collector listens on a port for traffic from DeepSweep. As a collector receives data, it parses the data based on its format. The resulting data is split up into pcap files, which are placed in a directory unique for each session directory.

This directory is created by Xplico when the user creates a session. Its pusher monitors this directory and decodes any pcaps in that directory. The data extracted from the pcaps is placed into Xplico's database, and the original pcaps are moved to another directory.

Xplico also uses this directory for signaling when a collector should start or stop. When the start button is pressed, Xplico creates a file called collector_running in this session directory. The collectors will run as long as this file exists. When the stop button is pressed, Xplico deletes this file and the collector for that session will stop.

3.4 INPUT / OUTPUT

1. DeepSweep takes internet traffic and filters it based on settings by the user.
   The traffic is either formatted using raw pcap or ASN.1 standard.

2. The Collectors takes this traffic and output it as pcap files.
   The Raw Collector handles raw traffic; the ASN Collector handles ASN.1 traffic.

3. The Pusher decodes these pcaps and loads the data into the database.

4. Users can either view the data in a web GUI or export the data as a PDF report.

Module	Input	Output
DeepSweep	Unfiltered traffic	Filtered traffic, in raw or ASN format
Raw Collector	Filtered traffic, in raw format	Pcap files
ASN Collector	Filtered traffic, in ASN format	Pcap files
Pusher	Pcap files	Data in database
Xplico	Data in database	Pcap contents and PDF reports

Table 2: Inputs/Outputs for Components

4 Testing

4.1 PROTOTYPES

Initial Version: For the initial prototype, we wrote the Raw Collector. Xplico was modified to start and stop the Raw Collector. We modified the session page, so users can start the collector.

Final Version: For the final prototype, we wrote the ASN Collector. Xplico was modified to start and stop the ASN Collector. We modified the session page, adding an option for selecting either the Raw or ASN Collector. Xplico was also extended to add options for generating a PDF report consisting of all data for each session or case.

4.2 TEST PROCESS

Several different kinds of tests were run for our project:

Component Tests: Once we completed a component of our project, we performed black box tests of the component. These tests verified that the component generates the correct output when given valid input and that it handles error conditions properly.

Integration Tests: As we completed components, we performed black box tests of the entire system whenever it was feasible. These tests ensured that all components of the system worked properly with each other.

Stress Tests: Some stress tests were also performed if possible, so we could verify the system handles heavy loads of traffic. Due to limitations in resources, we only ran stress tests for around 6 hours instead of over a couple of days.

Benchmark Tests: Benchmark tests were performed, so we could determine what limitations the system has. In particular, we tested how quickly the Collectors could create pcaps and Xplico could upload the newly created pcaps.

4.3 TEST RESULTS

Component/Integration Tests

1. Verify that the Raw Collector starts correctly when started via the GUI
2. Test that the Raw Collector collects raw traffic and outputs pcaps
3. Verify that data from pcaps created by the Raw Collector uploads to Xplico
4. Verify that the ASN Collector starts correctly when started via the GUI
5. Test that the ASN Collector collects ASN.1 traffic and outputs pcaps
6. Verify that data from pcaps created by the ASN Collector uploads to Xplico
7. Verify creating a report includes all the data selected by the user

These tests were run manually, as part of each test involved testing Xplico's GUI. For each component, the final tests passed successfully.

Benchmark/Stress Tests

Below are the results from our benchmark tests. For these tests, a script sent the same pcap to Xplico multiple times for decoding. A command line argument for the script determined how many times that it would send the pcap.

Number of Pcaps	Time	Number of Pcaps Decoded
100	1 min 43 sec	100
200	3 min 35 sec	200
300	5 min 20 sec	300
500	8 min 55 sec	500

Table 3: Benchmark Test Results

Below is a test of large packet (around 19 MB) comparing the time it takes for two versions of the collector. The first splits the large pcap and sends smaller pcaps to Xplico. The second sends the entire large pcap to Xplico.

Collector	Time for Sending	Time for Decoding
Split	< 1 second	40 seconds
Combined	2 minutes, 14 seconds	23 seconds

Table 4: Comparison of Test Results for Collectors

We also ran a stress test overnight from about 5:40 AM to 11:50 AM. During that time, Xplico decoded roughly 2240 pcaps successfully.

5 Conclusion

Our project created an interface between Xplico and DeepSweep. Since DeepSweep can emit traffic as raw or in ASN.1 format, we wrote a program for each format. These collectors converted the traffic into pcap files, since Xplico only imports traffic stored in pcap files.

Once a prototype of the interface was developed, we modified the program's existing graphical user interface. This existing GUI displays traffic sorted by its type (e.g. pictures, emails, http, ftp, etc.) and provides a summary of the total traffic uploaded. Our modifications allow a user to select a case and session for uploading traffic and then start the collector. Users can also generate reports consisting of some or all of the uploaded traffic in a session or case.

6 Appendix I – Manual

6.1 INSTALLATION INSTRUCTIONS

Xplico is an existing program for Linux, which will run on most versions of Linux. Xplico is easiest to install on Ubuntu 11.04. Below are instructions for installing it on Ubuntu:

1. Run the following commands:

```
sudo bash -c 'echo "deb http://repo.xplico.org/ $(lsb_release -s -c) main" >>
/etc/apt/sources.list'
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 791C25CE
sudo apt-get update
sudo apt-get install xplico
```

This installs the base Xplico system without any of our modifications.

2. Unzip the archive containing modified Xplico code.

3. Open a terminal window and cd to the directory with the unzipped code.

4. Run the following command for installing the modifications: `sudo perl XplicoInstall.pl`

This command builds and copies the new and modified files into Xplico's directory.

6.2 RUNNING XPLICO

Xplico runs a daemon in the background, which includes the pusher.
Xplico's GUI is a web interface that requires apache.

1. Run the following command to start apache: `sudo /etc/init.d/apache2 start`

2. Run the following command to start Xplico daemon: `sudo /etc/init.d/xplico start`

3. Open a web browser and point it to the following URL: `http://localhost:9876`
You should see a logon screen, with a prompt for username and password.

4. Default logins for admin and regular users:

Admin Username: Admin Password: xplico
Regular Username: Xplico Password: xplico

User accounts can be added or removed when logged in as an admin.

6.3 CREATING CASES AND SESSIONS

1. Login to Xplico with a regular account.
Xplico loads the Case Index page once logged in. (Figure 4)

2. Click New Case on the left. (Figure 1)

3. Select Uploading Deep Sweep Traffic. (Figure 3)

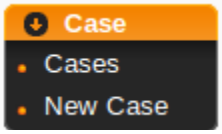

Figure 8 - Case Menu

4. Input a name for it and press the Create button. (Figure 3)
Xplico will redirect back to the Case Index page. (Figure 4)

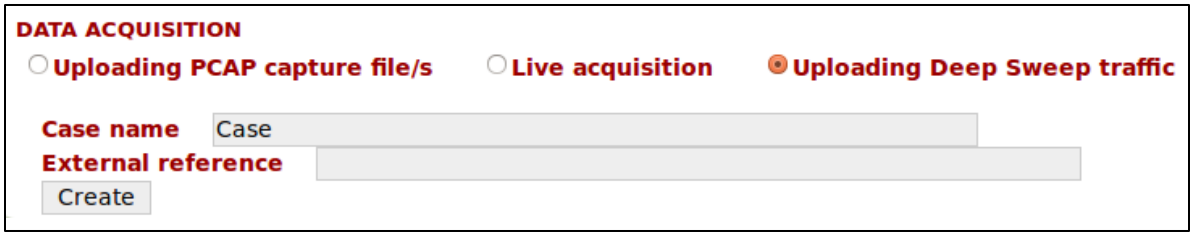

Figure 9 - New Case Page

Figure 10 - Case Index Page

5. Click on the newly created case's name. (Figure 4)
Xplico will redirect to the Session Index page. (Figure 5)

6. Click New Session on the left. (Figure 2)

7. Input a name for it and press the create button. (Figure 5)
Xplico will redirect back to the Session page. (Figure 6)

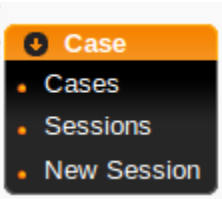

Figure 11 - Session

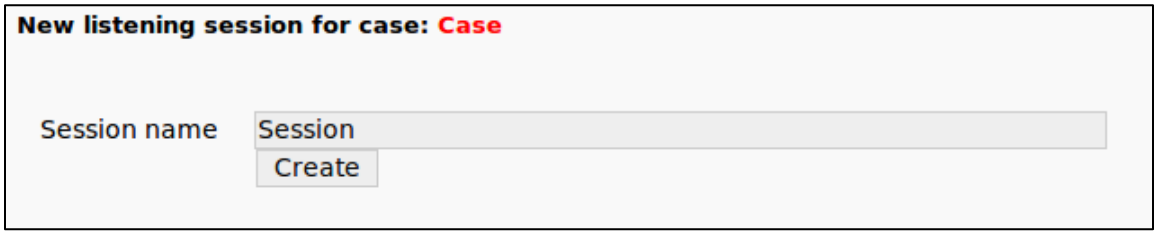

Figure 12 - New Session Page

Figure 13 - Session Index Page

6.4 UPLOADING DEEPSWEEP DATA

1. Click on the newly created session's name. (Figure 6)
Xplico will redirect to the Session View page. (Figure 7)

2. Input the port that DeepSweep is running on. (Figure 8)

3. Select format/type of traffic – Raw or ASN.1. (Figure 8)

4. Click the Start button. (Figure 8)
Xplico will listen for traffic and upload it automatically.

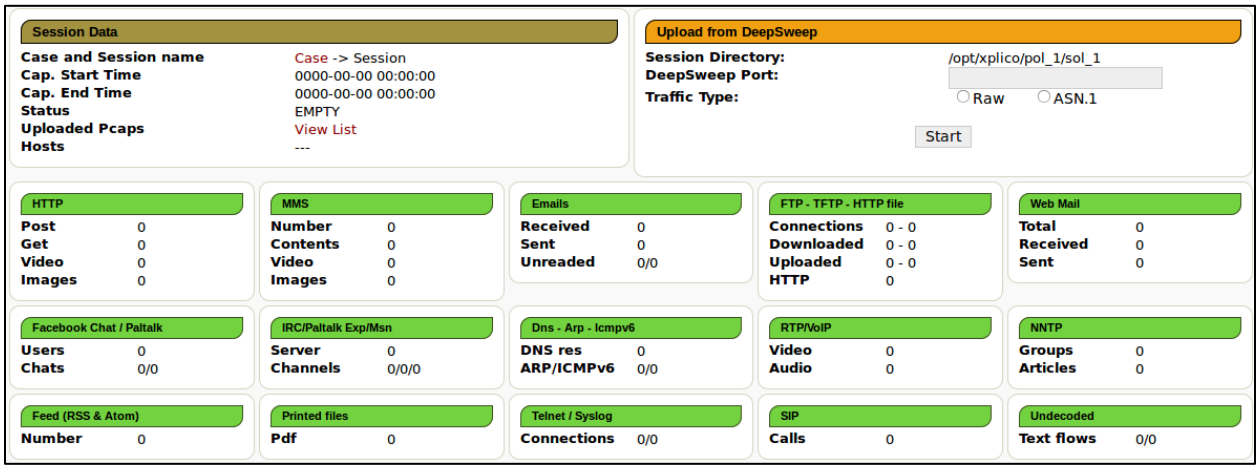

Figure 14 - Session View Page

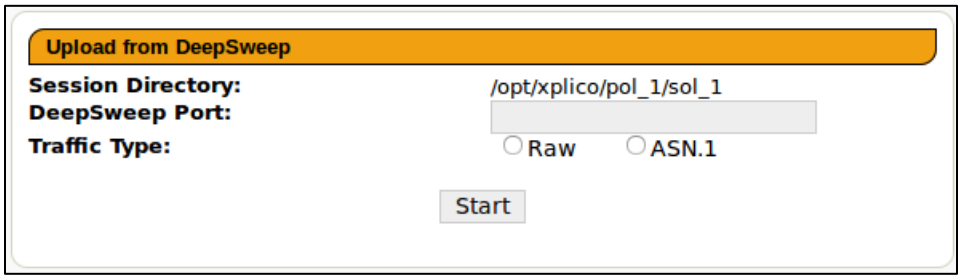

Figure 15 - Close-up of Upload Section

7 Appendix II – Alternative Designs

7.1 LANGUAGE CHOICES

Originally, we planned on using a different collector for ASN. This collector was written in Go and developed by the client's company. However, the client preferred that we use the Cream libraries in Java, which were also developed by his company. Using these libraries reduced the project's complexity and increased maintainability, as it eliminated an extra language that we had not worked with previously.

Initially, we wrote the collector for Raw in C. We felt it would be easier to handle bytes and other low level tasks in C instead of Java. However, we switched to Java for many of the reason why we switched collectors for ASN. Using the same language for both collectors also simplified our project's build/install scripts.

7.2 COLLECTOR DESIGN

The original raw collector simply forwarded the raw pcap traffic to Xplico via a TCP connection. This connection needed closing for each pcap, or Xplico would not decode the traffic. Closing and opening the connection was fairly slow and wasted system resources. Instead, the collector outputs the pcaps as files, and Xplico's pusher retrieves and decodes these files. This approach is much faster and results in fewer bottlenecks.

Both collectors originally output pcaps containing the entire data captured. For large pieces of data, though, Xplico took a long time to decode the pcap files. Furthermore, if the system crashed, large amounts of data would be lost. Therefore, the collectors now split the data up into smaller chunks, creating several smaller pcaps instead of one large pcap. Xplico processes pcaps in parallel, so the overall decoding time is much less.

8 Appendix III – Project Schedule

Task	Days	Start Date	End Date
Project Research	15	09/12/14	10/02/14
Investigate alternatives to Xplico	10	09/12/14	09/25/14
Study Xplico code and docs	5	09/12/14	09/18/14
Study DeepSweep code and docs	5	09/19/14	09/25/14
Documentation and Requirements	59	09/19/14	12/10/14
Project Plan	55	09/19/14	12/04/14
Create 1st version of plan	10	09/19/14	10/02/14
Create 2nd version of plan	10	10/24/14	11/06/14
Create final version of plan	10	11/21/14	12/04/14
Design Document	40	10/10/14	12/04/14
Create initial design	10	10/10/14	10/23/14
Create final design	15	11/14/14	12/04/14
Web Site	11	09/26/14	10/10/14
Build initial/basic web site	6	09/26/14	10/03/14
Build final web site	6	10/03/14	10/10/14
Poster	10	04/09/15	04/22/15
Final Report	13	04/09/15	04/27/15
Development	151	09/05/14	04/03/15
Phase 1	101	09/05/14	01/23/15
Install Xplico on Linux VM	5	09/05/14	09/11/14
Investigate Pusher options	25	10/17/14	11/20/14
Write Raw Collector in C	20	11/07/14	12/04/14
Create option for DeepSweep case	9	01/01/15	01/13/15
Start/Stop Raw Collector from Xplico	8	01/14/15	01/23/15
Phase 2	50	01/26/15	04/03/15
Write ASN Collector in Java	17	01/26/15	02/17/15
Convert Raw Collector from C to Java	8	02/18/15	02/27/15
Start/Stop ASN Collector from Xplico	8	02/18/15	02/27/15
Generate PDF containing raw data	7	03/02/15	03/10/15
Create script for installing program	25	03/02/15	04/03/15
Testing	126	11/02/14	04/24/15
Component Tests	86	11/02/14	02/27/15
Integration Tests	58	01/14/15	04/03/15
Stress Tests	10	04/13/15	04/24/15
Benchmark Tests	10	04/13/15	04/24/15

Table 5: Project Schedule

(Time spent fixing bugs is included in Component and Integration Tests.)