NTOP Usage Tracking

Capturing and Reporting Network Usage

By:
Shawn Wall

NTOP Usage Tracking
Capturing and Reporting Network Usage

Overview

The purpose of this white paper is to show how NTOP can be combined with MySQL and PHP to capture and report on customer network usage for the purposes of billing in an ISP environment.

Environment

This project was deployed in a small WISP (wireless ISP) with a subscriber base of approximately 300 residential and commercial users. There are 3 subscriber packages for customers to choose from: 1) Basic – up to 1.6 Mbps symmetrical and 15 GB total/month. 2) Pro - up to 2.6 Mbps symmetrical and 20 GB total/month. 3) Premium – up to 3.0 Mbps and unlimited usage. NTOP was deployed as a way to measure the amount of network traffic each subscriber was accumulating on a monthly basis. Netflows are collected by NTOP from a Cisco 2620 on both the 'inside' and 'outside' interfaces so as to capture the egress and ingress traffic. Each customer is issued a single public IP address via DHCP. IP addresses are reserved by MAC addresses; unknown MAC addresses are not issued IP addresses. There are 2 class B networks which are further subnetted down to class C being monitored. The server platform is FreeBSD 6.0-RELEASE-p4 which is running on 3 servers: 1) DHCP 2) Email 3) Web. Each server is configured with the following software: 1) DHCP: isc-dhcp30-server, NTOP 3.2, MySQL 5.0.18, PHP-5.1.2, PHP5-MySQL, phpmyadmin 2.7.0., Apach2 2.0.55. 2) Email: Postfix 2.2.4, Postfixadmin 2.1.0, MySQL 5.0.18, Apache 2.0.55, PHP-5.1.2, PHP5-Mysql 3) Web: Apache 2.0.55, MySQL 5.0.18, PHP-5.1.2, PHP5-Mysql. See Figure 1.0 for a detailed network topology diagram.

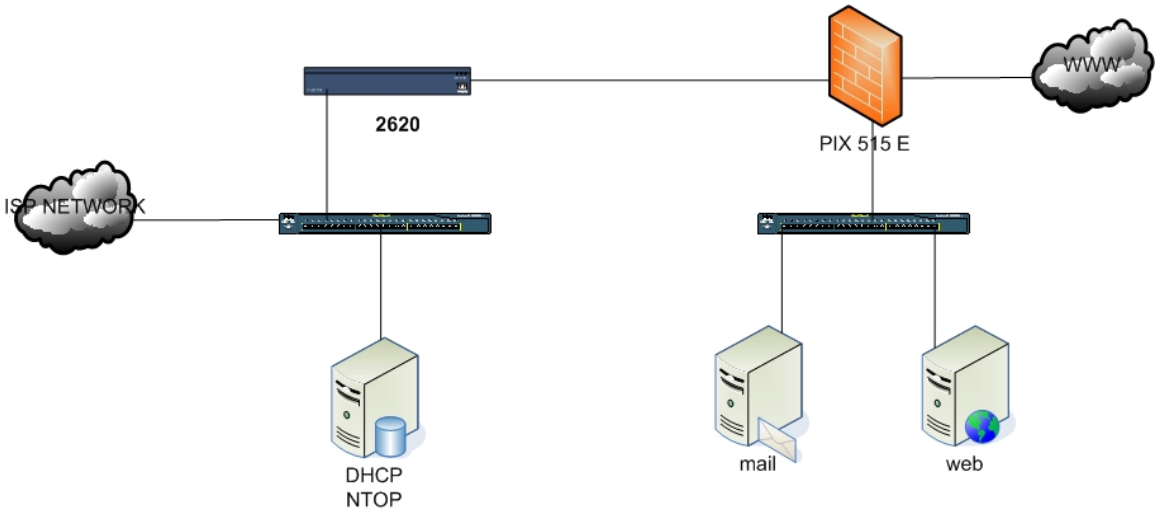
Outline

NTOP and netflow configuration are covered in detail elsewhere and will not be reviewed here.

To capture and report on the total amount of network traffic used by customers is simple: install NTOP is such a way that it has visibility on all incoming and outgoing traffic. However, if the purpose of this is to bill customers on their usage, then a means for the individual customer to view their current usage must be implemented. This is the primary goal of this project.

A secure area must be provided where customers can login and check their current network usage. Since this ISP already has a web presence, the existing web site was modified to provide a secure login area. To make the process as streamlined as possible, the existing email address and password was chosen as the login credential. The process is outlined as follows: User logins into secure area of website providing email address and password. Authentication against email server is done. If successful, user is directed to a new web page. Here the user

Figure 1.0



has several administrative options available, one of which is to check their current network usage. To check their usage, the user enters their email address, which is checked against a database of email addresses and IP addresses. If there is a match, the corresponding IP address is used to query the NTOP traffic database and returns the total amount of traffic in gigabytes.

Configuration

To begin, we must have a method to extract from NTOP the statistics which are presented in the browser. We do this by using wget. Wget can be run from the command line to 'dump' the data collected by NTOP in several formats[1]. For the purposes of this project the following wget command is used:
*/usr/local/bin/wget -O /usr/local/etc/ntopdump/ntopdumptbl --no-check-certificate "https://admin:**password**@**x.x.x.x**:3000/dumpData.html?language=txt&view=short"*

The output from this command will place a text file in the specified directory called **ntopdumptbl**. See example 1.1 for the text file format. To make this file useful, it needs to be imported into a database such as MySQL. A database will be needed to hold the table, for this project the database name **ntop** is used. Create the database however you wish but using phpmyadmin will greatly simplify the process. Once the database is created, import the **ntopdumptbl** file into the database as a new table called **ntopdumptbl**[2]. You will need to modify 2 fields; the first pair of **ipBytesSent** and **ipBytesRcvd** (the import process will automatically rename the second pair of ipBytesSent and ipBytesRcvd to ipBytesSent1 and ipBytesRcvd1 respectively). Change the type type of these fields to **bigint** due to the fact that some of the numeric values stored will be larger than the other data types allow for and we do not need excessive decimal places.[3] All fields should be changed to allow **NULL** also. A user with all access to the NTOP database will need to be created. The import

[1] See the NTOP documentation
[2] The name of the table and the name of the dump file should match for use with mysqlimport.
[3] See MySQL data type overview: http://dev.mysql.com/doc/refman/5.0/en/numeric-type-overview.html

Example 1.1

Ntopdumptbl fields delimited by |

```
key|hostResolvedName|pktSent|pktRcvd|ipBytesSent|ipBytesRcvd|bytesMulti
castSent|pktMulticastSent|bytesMulticastRcvd|pktMulticastRcvd|bytesSent
|bytesRcvd|ipBytesSent|ipBytesRcvd|ipv6Sent|ipv6Rcvd|tcpBytesSent|tcpBy
tesRcvd|udpBytesSent|udpBytesRcvd|icmpSent|icmpRcvd|
```

Host entry example

```
x.x.x.x|x.x.x.x|80|55|107058|2684|0|0|0|0|107058|2684|107058|2684|0|0|1
06992|2583|66|101|0|0|
```

process can also be done using the **mysqlimport** utility. The command is issued as follows:

/usr/local/bin/mysqlimport --user=ntop[4] *--password=xxxxxx --local --verbose --delete --fields-terminated-by="|" --ignore-line=1 ntop /usr/local/etc/ntopdump/ntopdumptbl*

So far two things have been accomplished:
1) Dump of NTOP statistics to a text file.
2) A database and table to store the data from the text file have been created.

The previous two tasks can be automated using cron. This way the data can be extracted and imported at regular intervals for querying and displaying as will be discussed shortly.

Allowing the user to login with their existing email address and password requires some digging around in the postfixadmin[5] directory. Postfixadmin is written in PHP, giving postfix email administrators an easy to use web browser interface to managing the postfix database. If you already have postfixadmin in use, check your **../postfixadmin/conifig.inc.php** file for the **Config** and **Encrypt.** The configuration here will be important when you create the user login area of your website.

In this project, the web site and email system reside on two different servers in the DMZ. When a user logins into the secure area of the web site, the php script will query the postfix database. In order to do so, a MySQL user will have to be created that is allowed to login from the web server. In this case the following user was added:

userlogin@x.x.x.x (x.x.x.x representing the IP address of the web server).

The **Encrypt** section of your **config.inc.php** will determine how you will code your password authentication. If you are using clear text, meaning that the email account password is stored in the postfix database in plain text, you will not need any additional coding other than a normal PHP login session. If **md5cyrpt** is in use, you will need to use some of the posfixadmin php code to read the encrypted password for the postfix database[6]. Begin by examining the **../postfixadmin/users/login.php** file. Note

[4] MySQL account
[5] http://high5.net/postfixadmin/
[6] Thanks to mihau at http://forums.high5.net/index.php for the suggestion.

settings under **Database**
require ("../variables.inc.php");
require ("../config.inc.php");
require ("../functions.inc.php");
include ("../languages/" .
check_language () . ".lang");

These files are required by postfixadmin when a user logins into the user area of postfixadmin. The remainder of the php code defines the templates to use for the html and how to read the encrypted password. To work, one could simply copy this file, the require files and the template files over to the web server, making sure to change the relative location of the require and template files. This, however, will reproduce the postfixadmin user login which will most likely not meet with your web site template. Alternatively, you could copy the php from the **login.php** file into a new web page that you have created. This will allow the functionality to remain intact while providing a customized look and feel. See example 1.2. Be sure to either use the existing form variables or create your own and change them in the code.

The above process is only useful if you are using postfix for email. If you are using some other email system that uses MySQL, something similar should be possible. If you are not using an email system, you can create a database for user authentication in whatever format you like. At the end of the day, it was decided to make use of the existing postfix database to eliminate the need for a second set of login credentials for the user or trying to maintain synchronization between a second set of passwords.

As noted in example 1.2, once successful authentication has occurred, the user is redirected to another web page, in this case the require files at the top of the script: On this page, the user has several administrative options available relative to the ISP. Also, there is a link labeled '**Check Traffic Usage**'. Clicking on this link will take the user to a new php page where the use is prompted to enter their email address into a form text box. When the user hits submit, a query is done against the **ntop** database previously mentioned. We know that there is already one table in the database, **ntopdumptbl,** that is automatically updated with the ntopdump data. A second table has been added to the **ntop** database called **userlookuptbl.** The table schema is as follows:

accountname	varchar(255)	PRI
ipAddress	varchar(255)	
macAddress	varchar(255)	
emailAddress	varchar(255)	

This table is manually updated with the appropriate information unique to each customer. The query is a join between the two tables, **ntopdumptbl** and **userlookuptbl.** Using the **ipAddress** and **emailAddress** fields from the **userlookuptbl** and the **hostResolvedName, ipBytesSent and ipBytesRcvd** we can diplay the total amount of traffic used by our customer.[7] The sql is as follows:

SELECT
(ipBytesSent+ipBytesRcvd)/1073741824
FROM ntopdumptbl as nto inner join
userlookuptbl as ult ON
nto.hostResolvedName = ult.ipAddress
WHERE ult.emailAddress='$emailAddress'

We combine the **ipBytesSent** and **ipBytesRcvd** columns to give us a total on

[7] Thanks to lostboy at http://www.weberforums.com for help with the query.

the **cust_area.php** page.
the amount of network traffic used by the customer[8]. See example 1.3 for a look at the php used in this process.

Note from figure 1.0 that the **ntop** database resides on the dhcp server which is inside the firewall. For the above query to work, a rule is added in the firewall allowing the web server to access the dhcp server on port 3306. From the PIX firewall:

access-list 130 permit tcp host x.x.x.x host y.y.y.y eq 3306

Where x.x.x.x is the web server ip and y.y.y.y is the dhcp server ip.

Example 1.2 Customized login.php

```
// change the relative path for the new location
require ("variables.inc.php");
require ("config.inc.php");
require ("functions.inc.php");
include ("languages/" . check_language () . ".lang");

// add php self action for form processing
$loginFormAction = $_SERVER['PHP_SELF'];

//remove the get method and templates
//if ($_SERVER['REQUEST_METHOD'] == "GET")
//{
//   include ("../templates/header.tpl");
//   include ("../templates/users_login.tpl");
//   include ("../templates/footer.tpl");
//}

if ($_SERVER['REQUEST_METHOD'] == "POST")
{
  $fUsername = escape_string ($_POST['fUsername']);
  $fPassword = escape_string ($_POST['fPassword']);
```

[8] Thanks to Burton Strauss from the NTOP development team for correcting a problem with the ipBytesSent and ipBytesRcvd columns

```
  $result = db_query ("SELECT password FROM mailbox WHERE
username='$fUsername' AND active='1'");
  if ($result['rows'] == 1)
  {
    $row = db_array ($result['result']);
    $password = pacrypt ($fPassword, $row['password']);

    $result = db_query ("SELECT * FROM mailbox WHERE username='$fUsername'
AND password='$password' AND active='1'");
    if ($result['rows'] != 1)
    {
      $error = 1;
      $tMessage = $PALANG['pLogin_password_incorrect'];
      $tUsername = $fUsername;
    }
  }
  else
  {
    $error = 1;
    $tMessage = $PALANG['pLogin_username_incorrect'];
  }

  if ($error != 1)
  {
// remove postfixadmins successful login session
//    session_start();
//    session_register("userid");
//    $_SESSION['userid']['username'] = $fUsername;
// add a new redirection after successful login
    header("Location: cust_area.php");
    exit;
  }
// remove templates
//  include ("../templates/header.tpl");
//  include ("../templates/users_login.tpl");
//  include ("../templates/footer.tpl");
}
?>
```

Example 1.3

Form used to collect user email address:

```
<td valign="top" class="leftnav"><p></p>
                    <p align="center" class="head2">Enter Your Email Address to view
your total accumlated traffic usage to date:</p>
```

```
                    <p align="center" class="head2"></p>
                    <form id="trafficlookup" name="trafficlookup" method="get"
action="traffictotal.php">
                     <p> </p>
                     <p align="center">
                      <label>Email Address:
                      <input type="text" name="emailAddress" id="emailAddress"/>
                      </label></p>
                     <p align="center">
                      <label>
                      <input type="submit" name="Submit" value="Submit" />
                      </label>
                     </p>
```

traffictotal.php

```
<?php
      $emailAddress= $_GET['emailAddress'];
      mysql_connect("x.x.x.x","xxxxxx","yyyyyyy");
      mysql_select_db("ntop");
            $result=mysql_query("SELECT (ipBytesSent+ipBytesRcvd)/1073741824
FROM ntopdumptbl as nto inner join userlooku
ptbl as ult ON nto.hostResolvedName = ult.ipAddress WHERE
ult.emailAddress='$emailAddress'") or die(mysql_error());
                  if(mysql_num_rows($result)){
                        while($row = mysql_fetch_row($result)){
                                    print "<h3>Your total traffic usage for the month is
$row[0] GB</h3>";
                                    } } else { echo 'That email address is incorrect or is not
registered. Please
 contact tech support.'; }
?>
```

Now that the data is in MySQL there are a number of possibilities. Also used in this project is a page for tech staff at the ISP to update the **ntop.userlookuptbl,** display users that are over their subscribed limit and list all users in the the **userlookuptbl.**

Future Development

The following are in progress to improve the project:

- Integrate the ntopdumptbl into the postfix database, eliminating the need for the ntop database on the dhcp server.
- Customize the postfix.mailbox table to include additional fields for ipaddress, macaddress and package.
- Customize the postfixadmin create_mailbox.php to include ipaddress, macaddress and package during mailbox creation.
- Email daily reports to tech staff on current user traffic, showing only

users that have exceeded subscribed limits.

- Allow users to set a threshold which when reached an email will be sent advising them that they are about to reach their monthly limit
- Eliminate the need for the user to enter their email address a second time to display traffic usage.