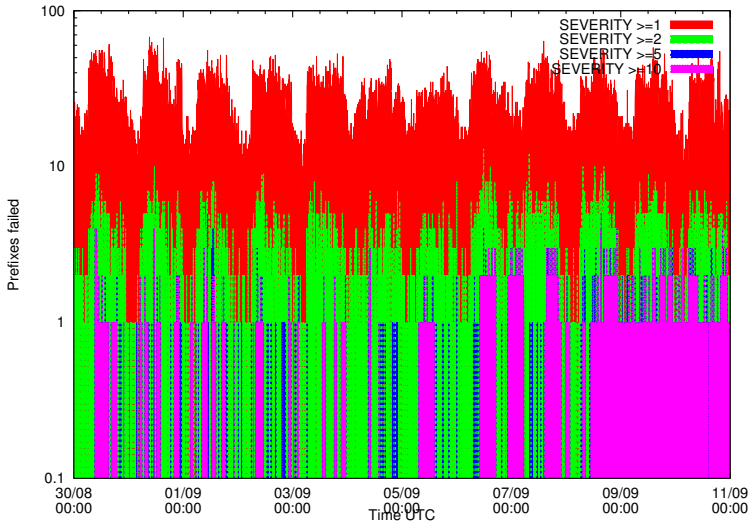
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Daniel Aschwanden
asdaniel@ee.ethz.ch
07-907-769

Identification of Connectivity Issues in Large Networks using Data Plane Information



Semester Thesis SA-2010-21
September 2010 through February 2011

Advisors:
Dominik Schatzmann
Dr. Wolfgang Mühlbauer

Supervisor:
Prof. Dr. Bernhard Plattner

Communication Systems Group – CSG
Computer Engineering and Networks Laboratory – TIK
Department of Information Technology and Electrical Engineering – ITET
Swiss Federal Institute of Technology – ETH

Abstract

Monitoring remote connectivity issues is really important since network complexity continuously grows. FACT - a Flow-based Connectivity Tracking System - was proposed a year ago and is efficient in alerting network operators about connectivity problems which really affects his network users. However, FACT has some shortcomings like no IPv6 support, no standardized data input or the lack of a smart reporting engine. These three shortcomings are addressed in this thesis and implemented. For demonstrating how effective and easy connectivity issues may be detected with FACT a two week long traffic trace of the SWITCH network is analyzed and some extracted connectivity problems are presented.

Keywords

monitoring, connectivity problems, flow-based

Acknowledgments

During this semester thesis several people and organizations supported me. At this place, I would like to thank them.

At first, I am deeply grateful to thank Dominik Schatzmann and Dr. Wolfgang Mühlbauer for their support and their patience. During the entire semester they provided good remarks, excellent advisory not only in technical details and they have always pointed me the right way. In particular, I would like to thank Dominik Schatzmann for his excellent development of the FACT framework upon my own work is based and for showing me the ”ruby way of doing things”.

Furthermore, I am appreciative to Prof. Dr. Bernhard Plattner for supervising this thesis and asking good questions and giving some remarks in occasional meetings. In addition, I enjoyed the support of the entire Communication System Group (CSG) of Prof. Dr. Bernhard Plattner in various aspects.

Finally, I would like to thank SWITCH for providing the data used in this study as well as the nfdump team of SWITCH, particularly Peter Haag, for his competent and uncomplicated guidance in nfdump related questions.

Daniel Aschwanden

Contents

CHAPTER 1

Introduction

1.1. Motivation

The complexity of the Internet is still growing in a rapid pace. Hence, some tools to easily monitor and resolve connectivity issues are required by network operators and Internet service providers. Predominantly, the root cause of a connectivity problem lies not within the control scope of a network administrator, e.g. a disrupted peering or congested paths of an upstream provider. However, if such problems are known, customers and providers may be informed and the problem may be resolved within convenient time.

Several researches ([Zhang et al., 2008], [Katz-Bassett et al., 2008], [Madhyastha et al., 2006]) and commercial vendors ([CA technologies, 2011], [Arbor Network, 2011]) have proposed systems to detect and troubleshoot such events [Schatzmann et al., 2011]. Nevertheless, the success of these proposed systems is questionable since most of them ”rely on active measurements using ping, traceroute, etc”[Schatzmann et al., 2011]. Bush et al.(2009) illustrated the ”limitations and biases that arise when trying to assess data-plane reachability from control-plane observations”[Bush et al., 2009]. They stated that it is dangerous to rely on control-plane information. Therefore, Schatzmann et al.(2011) proposed ”FACT, a system that implements a **F**low-based **A**pproach for **C**onnectivity **T**racking”. The big advantage is that FACT fully relies on passive measurements with data-plane information.

FACT tries to passively detect remote connectivity problems through matching outgoing and incoming connections of internal network users. Connections of unresponsive external hosts are leached. If no other internal host succeeded to reach this host, it is classified as unreachable. Afterwards, FACT tries to aggregate these unresponsive external hosts up to network or AS level. ”This requires a careful data processing to correctly handle

asymmetric routing and to eliminate the impact of noise due to scanning, broken server, late TCP resets, etc."[Schatzmann et al., 2011]. Consequently, FACT is able to dependably detect remote connectivity problems which really affect internal network users. Due to that, FACT is an interesting tool for small- to medium-sized Internet service providers and network operators which allows to detect connectivity issues and react on time.

Nevertheless, FACT has some shortcomings like no IPv6 support, no standardized data input or the lack of a smart reporting engine. These three shortcomings are addressed in this thesis and implemented. Since FACT intends to be a forward-looking tool for network operators, it have to support IPv6, because of the need of deploying IPv6 in the near future. Moreover, to guarantee the interoperability of FACT with existing netflow traces of network operators FACT has to support a standardized netflow data format as input.

Kleefass et al.(2009) developed the idea of flow-matching based detection of connectivity issues. So this thesis is based on their work and the work of Schatzmann et al.(2011). The rest of thesis is structured as follows: Chapter 2 explains the basic concept of FACT and outlines the problems of FACT which will be addressed in this thesis. Afterwards, chapter 3 presents how these problems are solved and implemented. The results from a brief analysis of a two week traffic trace are presented in chapter 4 and evaluated. Chapter 5 concludes this thesis. Due to privacy concerns, all IPv4 addresses are 16-bit truncated and IPv6 addresses are 96-bit truncated.

CHAPTER 2

Background

2.1. Basic Concept of FACT

The main purpose of FACT is to detect remote connectivity issues in the Internet through matching the outgoing and incoming connections of all internal network users. This is done with flow accounting.

2.1.1. Flow Accounting

All border router of the network are exporting their flow data to a central instance. Since in a large network the incoming and the outgoing connections must not necessarily pass the same border router, it is important to account the flow data of all border routers. Then, the central instance is processing the data for FACT as follows: The flow records are filter so that only the interesting traffic remains. All network internal traffic is never recorded, since these flows are not passing border routers. In addition, transit traffic is dropped, because these connections are not helpful to detect remote connectivity issues which have an impact on internal users. Furthermore, only incoming traffic as blocked scans are not considered as well. Consequently, the remaining traffic is outgoing with either a corresponding incoming connection or not. For tracking connectivity issues in the Internet, the goal is to extract the connections of the latter case and pinpoint their external hosts. Hence, the main computation of FACT is to extract so called *unbalanced flows*, which means to figure out which flows are only outgoing and have no corresponding incoming flow. Afterwards, FACT tries to aggregate these unresponsive hosts up to an unresponsive network or even to an unresponsive autonomous system. For this step does FACT group all hosts which are sharing an common prefix. If a group contains only unresponsive hosts, this group or more precisely their prefix are classified as unresponsive. Therefore, a connectivity issues

may be viewed as an outage of an entire network and not only as an outage of several external hosts. This helps to identify the root cause of the problem.

2.2. Shortcomings

In its first version, FACT had only a basic functionality which was enough for analyzing connectivity issues on a research level. However, for deploying FACT at network operators various aspects have to be improved and replenished. This thesis focusses on three problems: IPv6 support, no standardized data input and the lack of a smart reporting engine.

2.2.1. IPv6 Support

Up to now, FACT is only processing IPv4 flows and is dropping all IPv6 flows. Since FACT should be a forward-looking tool for global network operators it has to support IPv6 flows, because the emergence of IPv6 in a dominant disposition will be unquestionable - sooner or later. Therefore, FACT must be able to identify connectivity issues of IPv6 networks. On a scientific point of view it is non-advisable to neglect the entire IPv6 traffic. Moreover, there are suspenseful questions to be answered, e.g. it would be interesting to correlate IPv4 and IPv6 network outages. On that account, this thesis purposes the enabling of IPv6 in FACT.

2.2.2. Standardized Data Input

Due to the fact that FACT is requiring an own proprietary data input format so far, it is not portable to a network operator yet. Hence, FACT has to be able to read a standardized data input so that a network operator may reuse already existing traces or easily collecting new traces with a wide-spread netflow tool. Therefore, a second important task of this thesis is to include the ability of reading a standardized data format in FACT.

2.2.3. Smart Reporting Engine

The output of the analyser of FACT is in the majority of cases pure csv formatted text, spread over a vast amount of directories. Manually analyzing these files is very time-consuming and the visualization of connectivity issues is done through a post-processing script. It is obvious that this behavior of FACT is neither suitable for network operators nor user-friendly for researchers. For convenience reasons these output files must be automatically parsed, summarized and the likelihood of a connectivity issue must be classified. So a smart reporting engine should be developed which parses the output of the analyser, determines the top connectivity issues for a given period and classifies the likelihood of an issue.

CHAPTER 3

Implementation

3.1. IPv6 Support

3.1.1. IPv6 and the SWITCH Network

Since the end of January 2011, there are no more Internet Protocol version 4 (IPv4) address blocks available at IANA[1]. Hence, only the region Internet registries (RIR) are able to allocate few IPv4 address blocks. Estimations [Huston, 2011] predicts the first RIR IPv4 address pool exhaustion in mid 2011 (APNIC[2]). This means that there are no longer IPv4 address blocks available for the affected region. Hence, this may influence the growth of the Internet as a whole. The fact of IPv4 address depletion is known since decades and a working solution is provided by version 6 of the Internet Protocol (IPv6). However, the deployment of IPv6 is running very slowly, although, IPv6 was officially announced as successor of IPv4 in December 1998 by the RFC2460.

The Swiss national research and education network SWITCH[3] is providing by June 2004 a fully enabled dual-stack backbone network for the Swiss universities [SWITCH, 2011a]. Seven years later, none of the Swiss universities have fully enabled IPv6 on their own networks. SWITCH states that "there are technologies that are exceedingly important but slow to get off the ground. IPv6, the new version of the Internet protocol, is one of these."[SWITCH, 2011b]. Therefore, SWITCH have recently launched an incentive program to motivate the universities to enable IPv6 on their network [SWITCH, 2011a]. Hopefully, Swiss universities will deploy IPv6 on their networks in the near future to be ready for the future of the Internet.

[1] Internet Assigned Numbers Authority `http://www.iana.org`
[2] Asia and Pacific Network Information Center `http://www.apnic.net`
[3] SWITCH: `http://www.switch.ch`

Since FACT mainly intents to analyze the traffic from the SWITCH network and other network operators, the FACT code framework has to be able to process IPv6 flow-level traces. This adaption to fully support IPv6 is done as an essential part of this semester thesis. In section 4.1.1, some results about the traffic volumes within the SWITCH network are presented.

3.1.2. IPv6 and FACT

Since FACT processed so far only IPv4 flows - all IPv6 flows were filtered and dropped - there are several adjustments to be made in the framework of FACT. One aim of this semester thesis is to enable IPv6 in FACT such that FACT is able to track connectivity issues for IPv4 and IPv6 networks.

The work for enabling IPv6 in FACT contained the following parts:

- Creation of a filter for all IPv4 connections (FilterIPv4), so that only IPv6 connections can be processed, e.g. for IPv6 only networks, analogous to FilterIPv6.
- Adjustment of the filter for incoming/outgoing connections(FilterInOut), in particular the configuration file for the FilterINOUT. This configuration file contains all prefixes of the home network - in this case all prefixes of the SWITCH network. For creating this file automatically, a little perl script (switchextract.pl) is included in the tool folder of the FACT code. This script needs the output of bgpdump for the desired timeframe. Moreover, this script is generating a file prefixes.txt, which is needed by the analyser of FACT.
- Adjustment of all prefix routines for aggregating and matching IPv6. Since the analyser of FACT aggregates IPv4 hosts up to a /24 network, IPv6 hosts are aggregated up to a /48 network by the analyser.
- Separation of IPv4 and IPv6 flows into own connection matrices - one for matching IPv4 flows and the other for IPv6 flows.
- Adoption of the analyser to analyze each connection matrix and to output the analyser results into an own IPv4 and IPv6 folder.

Problems

While enabling IPv6 in FACT, several problems appeared. Since this task was the first to implement, a deep contemplation and comprehension of the existing FACT code framework was inevitable, which was very time consuming and required several further assistance by the developer of FACT. Afterwards, the essential parts of enabling IPv6 had to be identified. The key troubles have been appeared while implementing the separation of flows into two separate connection matrices. The working solution is to initialize for each type of flow - IPv4 and IPv6 - an own connection matrix hash table (Connection_Matrix_HT). After solving this issue, the residual tasks of enabling IPv6 in FACT were quite straightforward.

3.2. NfDump

3.2.1. NfDump in a nutshell

NfDump is a toolset similar to tcpdump, but for collecting and processing flow level traces instead of packet level traces. It is developed by SWITCH and widely used by network operators over the world. NfDump supports netflow versions 5, 7 and 9 [Haag, 2011].

The NfDump toolbox mainly consists of the following parts:

nfcapd is the *netflow capture deamon*. nfcapd is able to collect netflow data from several exporting routers and saves the collected data.

nfdump is the main processing tool. It allows the user for example to sort the data and to create some top-N-statistics.

nfprofile is the filter instance of nfdump. Nfprofile is able to filter the stored data according to a predefined set of filter rules and saves the filtered data.

nfreplay is intended to replay the saved flows. This means that nfreplay is able to send the stored data to another host in the network.

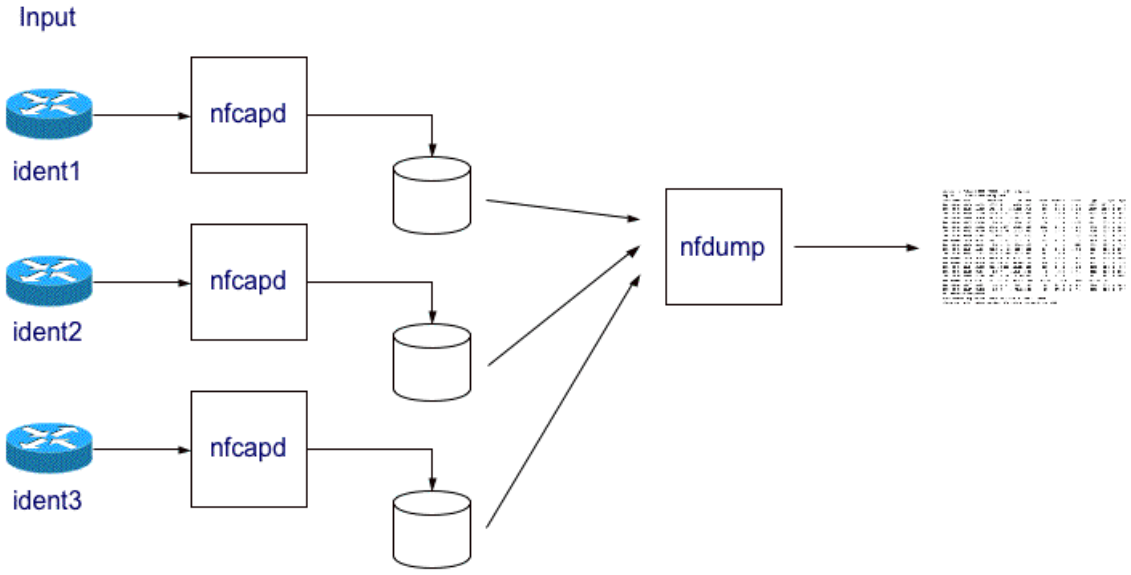


Figure 3.1.: functional overview of NfDump [Haag, 2011]

The entire toolbox is called NfDump, while the main processing tool is called nfdump. Figure 3.1 shows systematically how nfcapd and nfdump are interacting. Nfcapd is collecting netflow exports from routers and saving them into predefined directories. Then nfdump is able to get these saved traces and is processing these traces into some statistics.

3.2.2. Integration with NfDump

The main motivation of integrate NfDump data is that FACT has to be able to read some standardized data format. Up to now, FACT is only processing a proprietary netflow format of the Communication Systems Group (CSG). A big goal of FACT is that it will be deployed at SWITCH or other network operators in the near future. Hence, the data input format must be somehow standardized to guarantee the interoperability with the flow-level traces of the network operator which are often already captured with nfcapd. For example SWITCH is capturing netflow traces with nfcapd. Therefore, this task was about to integrate the ability of reading nfdump data as input data in FACT.

The jobs related to the integration with NfDump were the following:

- Adjustment of the main routine connectivity_extract, so that it accepts a nfdump input data folder as argument.
- Adoption of the data parser which calls nfdump to extract the needed information out of the data files and correctly initializes the connection entries.

Because of the limited amount of time, only an input of comma separated value (csv) output of nfdump is implemented so far. An direct import of the binary nfdump data format is planned and will be finished soon.

Problems

Because NfDump is a open-source software, the source code is publicly available and thus an insight into this code yields the functionality of NfDump including its data format. There is also an example for a nfdump data reader included in the sources. Although, due to the lack of time, only a csv data input was implemented. This is done through calling the nfdump function directly from the ruby C++ extension of the data parser and then parsing the piped output. The function call includes a special format and is specified as the following:

```
1 nfdump -m -o "fmt:%ts,%te,%sa,%sp,%da,%dp,%pr,%ra,%nh,%in,%out,%pkt,%byt" -r
      nfcapd_dir
```

Listing 3.1: nfdump function call with special format specification for FACT csv input

After exporting some local traces with fprobe[4] and capturing these traces with nfcapd, some tests have been made. During these tests, a weird bug in the nfdump code[5] have been discovered. This bug disposed nfdump to setting some fields in a random manner, e.g. the router address (%ra). After confirming this bug on several systems, Peter Haag from SWITCH was contacted and notified about this bug. The root cause of this bug was a falsely initialized master record of an flow within the nfdump data. For fixing this bug a patch was provided and since February 2011 the new version 1.6.3 of nfdump is available with included patch.

[4] fProbe `http://fprobe.sourceforge.net`
[5] NfDump version 1.6.2

3.3. Smart Reporting Engine

3.3.1. Approach

The main goal of this smart reporting engine is to provide an easily understandable overview of the connectivity situation for a given period. Since the analyser is creating information in several distributed files, the reporter must collect, combine and process this information. One important part of the reporting engine is the classification of the connectivity problem. An important role in classifying the connectivity problems is the *severity*, which means how many internal hosts are affected by a problem. A problem is more severe the more internal hosts are affected. For classifying connectivity issues thresholds are playing an important role as well. The border between a classification of a non-severe problem and a severe problem is called threshold. This threshold must be set cleverly in order to achieve a good classification. The intention of the smart reporting engine was to keep the model as simple as possible, regardless to achieve an usable classification with a low false positive rate. This is done with the so called *8to8 model*.

8to8 Model

The basic idea of this 8to8 model is that if there are a large amount of network users during the day (from 8am till 8pm), a lot of minor connectivity problems may be detected. Therefore, to classify only the really important connectivity issues a higher threshold value must be set than the threshold of the night. So according to the time of the report, a threshold variable is set to threshold_day or threshold_night.

To classify a connectivity issue several variables are required and defined as follows:

`score` is a count variable. Upon this variable the decision is made.

`top_number` is the number of prefixes in the top five problem causing prefixes and can be an element of $\{0,1,2,3,4,5\}$

`top_sum` is the sum of the severity from each prefix of top five problem causing prefixes.

`problem_sum` is the overall sum of the severity from each problem causing prefix, i.e. not only from the top five prefixes.

In order to capture several events this count variable is set by the following three events:

1. `top_sum` exceeds the threshold which means that there are enough internal hosts affected to form a severe connectivity problem.
2. `problem_sum` exceeds the `top_sum` by factor 2: This means that there are a lot of minor problems so that more than twice as many users are affected by these minor issues than by the five major issues.
3. `top_number`: The number of prefixes in the top five - usually less than 5 - is intended to capture severe routing failures.

Classification	Explanation
unlikely	`top_sum` does not exceed the threshold and the overall `problem_sum` is not twice the `top_sum`. This means that there are not many connectivity issues and therefore, probability of a severe connectivity issue is unlikely.
very likely	`top_sum` exceeds the threshold and either there are five top five failed prefixes or there are a lot of minor problems, i.e. `problem_sum` exceeds twice the `top_sum`.
likely	everything else, individual resolution and classification is required.

Table 3.1.: Classifications

3.3.2. Report

The implemented reporting engine is creating three sections in a report file:

- `Connectivity Problem`: Classification of the likelihood of a connectivity issue during the reporting period. Table 3.1 shows the different classifications and their explanations.
- `Top Problems`: sorted delineation of the five most severe prefix outages.
- `Problem Structure`: well-arranged overview of each prefix outage in a tree display. The top five problem causing prefixes are partitioned into the top five problem causing /24 network of this prefix and these are again partitioned into the top five problem causing hosts in this network.

Problems

Due to the intention to keep the model as simple as possible, the implementation of the reporter was quite straight-forward. The drawback of this reporter is that there is a need for parameter tuning so that it will fit the needs of a particular network. Especially the threshold and/or the individual scores for each of the three events must be tuned. Therefore, events must be identified manually and then the variables must be set to an appropriate value. This is done only in a very limited extend. Nevertheless, the reporter fulfills the requirements quite well.

```
1  FACT REPORT
2  ———————————
3
4  EPOCH: 1283159700
5  DATE: 2010-08-30 11:15:00 +0200
6  ::::::::::::::::::::::::::::::::::::::
7  CONNECTIVITY PROBLEM: VERY LIKELY
8  ::::::::::::::::::::::::::::::::::::::
9
10 TOP PROBLEMS:
11 1: PREFIX: 88.221.XXX.0/21, SEVERITY: 615
12 2: PREFIX: 92.123.XXX.0/22, SEVERITY: 547
13 3: PREFIX: 92.122.XXX.0/22, SEVERITY: 204
14 4: PREFIX: 95.101.XXX.0/22, SEVERITY: 200
15 5: PREFIX: 207.107.XXX.0/16, SEVERITY: 2
16
17
18 PROBLEM STRUCTURE:
19 1: PREFIX: 88.221.XXX.0/21, SEVERITY: 615
20     1.1 NET: 88.221.XXX.0/24, SEVERITY: 154
21         |— HOST: 88.221.XXX.XXX/32, SEVERITY: 1
22         |— HOST: 88.221.XXX.XXX/32, SEVERITY: 1
23         |— HOST: 88.221.XXX.XXX/32, SEVERITY: 1
24         |— HOST: 88.221.XXX.XXX/32, SEVERITY: 1
25         |— HOST: 88.221.XXX.XXX/32, SEVERITY: 1
26     1.2 NET: 88.221.XXY.0/24, SEVERITY: 142
27         |— HOST: 88.221.XXY.XXX/32, SEVERITY: 1
28         |— HOST: 88.221.XXY.XXX/32, SEVERITY: 1
29         |— HOST: 88.221.XXY.XXX/32, SEVERITY: 1
30         |— HOST: 88.221.XXY.XXX/32, SEVERITY: 1
31         |— HOST: 88.221.XXY.XXX/32, SEVERITY: 1
32     1.3 NET: 88.221.XXZ.0/24, SEVERITY: 134
33         |— HOST: 88.221.XXZ.XXX/32, SEVERITY: 1
34         |— HOST: 88.221.XXZ.XXX/32, SEVERITY: 1
35         |— HOST: 88.221.XXZ.XXX/32, SEVERITY: 1
36         |— HOST: 88.221.XXZ.XXX/32, SEVERITY: 1
37         |— HOST: 88.221.XXZ.XXX/32, SEVERITY: 1
```

Listing 3.2: an excerpt of a report showing the presentation of the problem in tree structure

CHAPTER 4

Results and Evaluation

4.1. Results

4.1.1. IPv6 and the SWITCH network

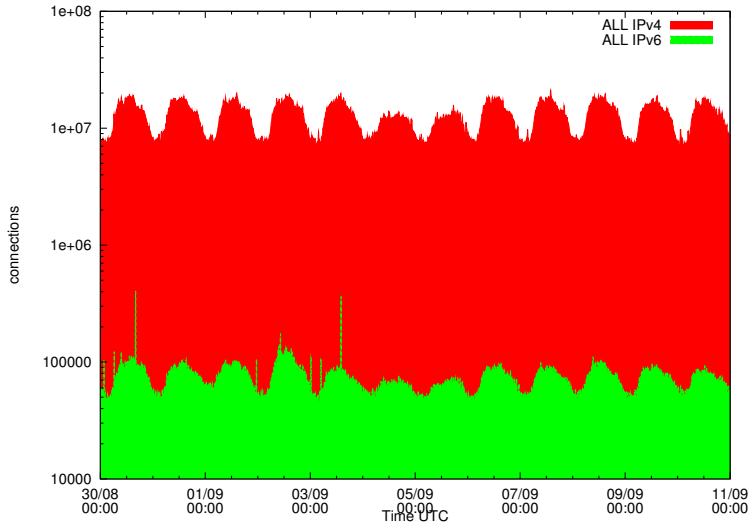


Figure 4.1.: Comparison of the traffic levels of IPv4 and IPv6 in the SWITCH network in terms of connections per 5 minutes

This section is to get an idea of the deployment of IPv6 in the SWITCH network at the end of August 2010. As figure 4.1 shows that there are around 8M-20M IPv4 connections per 5 min time slot. On the other hand, there are only 50k-100k IPv6 connections per 5 min time slot. So the overall traffic level of IPv4 is a factor 160 - 200 higher than the traffic

level of IPv6. This means that there is at least 160 times more IPv4 traffic than IPv6 traffic. Consequently, FACT has some trouble in detecting connectivity issues in IPv6 networks with the same reliability as in IPv4 networks because of this significant lower traffic volume, i.e. most likely due to a significant lower number of IPv6 user.

4.1.2. Analysis of a two week traffic trace

In order to demonstrate how efficient and comfortable FACT works in practice, a few details of a two week traffic trace from the entire SWITCH network are elaborated. This traffic trace has started on 30/08/2010 at midnight and ended around 12 days later on 11/09/2010. Since FACT is analyzing the data split up in 5 minute time slots, there are 3492 reporting files generated.

Data

SWITCH is collecting unsampled traffic traces at their boarder router which exports netflow traces to a central instance within the SWITCH network. They are saved so that these traces are available for several years back. The CSG has an exclusive access to this traffic traces in order to a research contract with SWITCH.

IPv4

A brief statistic of this two week IPv4 traffic trace:

- 2170 time slots are classified as `UNLIKELY`, this yields that there are no problems reported in 62% of the time (180 hours within 291 hours).
- 912 time slots are classified as `LIKELY`. In 26% of the cases there may be a connectivity issue. However, some further investigation is required to definitely denote a connectivity problem within these time slots.
- 410 time slots are classified as `VERY LIKELY`. Hence, in 34 hours (11%) of the trace a connectivity issue is reported.

Figure 4.2 is presenting the number of failed prefixes over time. This semi-log plot shows how many external prefixes are classified as unreachable over time. The severity is indicated by colors, red stands for $severity \geq 1$ which means that at least one host is affected by the given number of failed prefixes. According to that, green stands for $severity \geq 2$, blue for $severity \geq 5$ and purple for $severity \geq 10$. It is visible that the red curve is heavily fluctuating, because there is a high impact of some noise like port scans, DDoS backscatter, etc. Consequently, the green curve is showing the number of failed external prefix whose outage affects two or more internal hosts. This is to be considered as more robust to noise. Therefore, if at least 10 internal hosts are affected one is attempted to state that this is certainly a real connectivity problem. Hence, for each purple spot there may be a connectivity issue on a very high probability.

IPv6

Figure 4.3 is showing again how many external prefixes are classified as unreachable, but for IPv6 instead of IPv4 as in figure 4.2. Obviously, there are hardly any IPv6 connectivity issues detected in contrast to the IPv4 plot. This may be a result of the low traffic level of IPv6 in the SWITCH network or just because there are no severe connectivity issues within the observed timeframe. The red spots which can be seen on figure 4.3 are prefix failures which only affect one internal host and this is - as stated above - not very reliable. Moreover, there are five green spots which indicate a prefix failure which affects 2 or more internal hosts. Further investigation though yields that these spots are caused by the same two hosts within a single /48 network. This may be a real connectivity issue. However, only two internal hosts are affected by this problem which is of course still not as reliable as severity 10.

4.2. Evaluation

As the two plots above shows, prefix failures may be extracted quite well with FACT. So further investigation of the problematic time instances is needed. This may be done through the consultation of the report file of these time slots. At 30/08/2010 11.15 CEST for example there are 4 IPv4 prefixes reported as failed with $severity \geq 10$. The investigation yields that some prefixes of a content distribution network provider have not been reachable during 2 hours and thus, caused that some content of this provider and their customers were partially unavailable. The larger purple blocks around the 08/09/2010 till the 11/09/2010 exhibits again one of these prefixes as broken. This shows that it was not possible to completely resolve that problem within at least 12 days. If FACT were deployed at that moment, this prefix failure, which may be a result of a broken peering or another routing failure, would have been resolved faster.

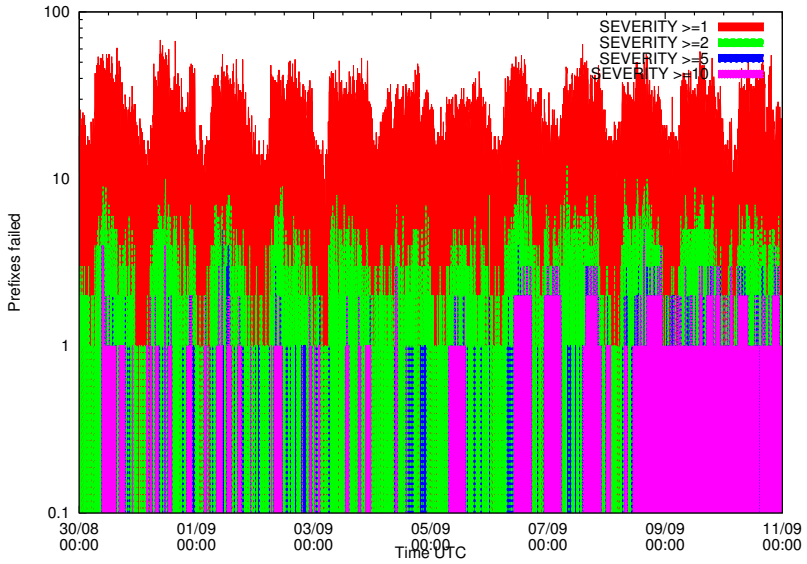


Figure 4.2.: number of failed IPv4 prefixes within the two week traffic trace

This analysis of the two week trace shows how easy it is to classify and identify connectivity issues with FACT. However, FACT requires a certain level of traffic to reliably classify connectivity issues. Moreover, there are some methodical details to evaluate:

- How precise is the classification of connectivity issues?
- How complete is the classification of connectivity issues?
- How low is the false positive rate?

4.2.1. Precision

The precision is defined as the fraction of the true positives and the sum of the true positives and the false positives. Further investigation of the report files and of the above plot yields that there are hardly any False Positives in this data set which leads to a very low false positive. Therefore, the Precision should be quite high, at least if there is enough traffic to get a high severity.

4.2.2. Recall

The recall is an indicator for the completeness of the connectivity issue tracking. Firstly, the complete set of connectivity issues has to be defined. Either all connectivity issues of the entire Internet are defined as set of connectivity issues or all connectivity issues which concern the network users. In the first case the recall would be very low, because it is very unlikely that the connections of all internal users will track all connectivity issues in the entire Internet. Consequently, there are a lot of false negatives and therefore, the recall is very low. The second case is harder to examine and needs a very detailed examination which will not fit the extent of this thesis. This is due to the uncertainty of how robust and

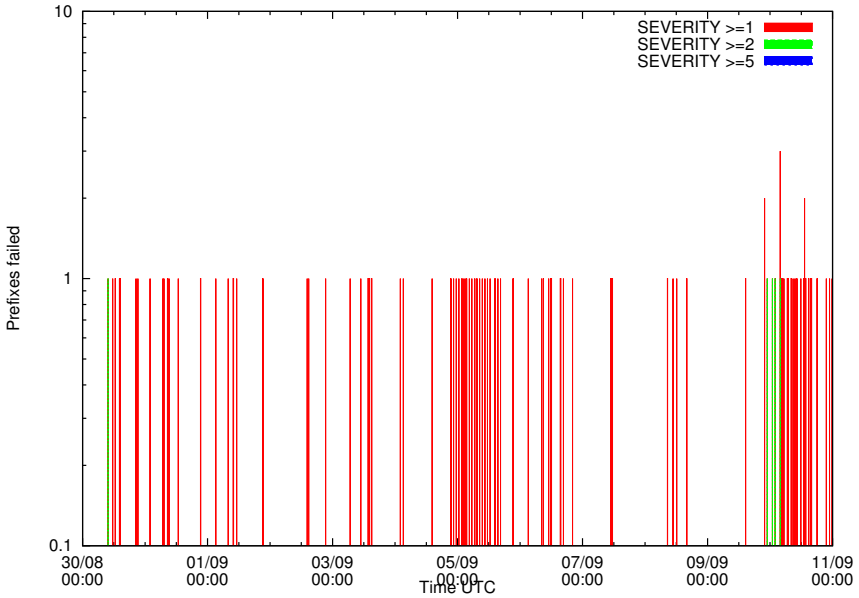


Figure 4.3.: number of failed IPv6 prefixes within the two week traffic trace

reliable the severity one is. If all connectivity issues of severity one are considered, the false negatives will increase by this amount what decreases the recall in turn.

4.2.3. False Positive Rate

To determine the false positive rate the robustness and significance of the severity has to be estimated again. Since the consideration of severity one will lead to a high number of false positives due to port scans or other blocked traffic, the false positive rate will be higher than in the cases of severity bigger than one. This assumes that the likelihood of independently creating scanning traffic or some blocked traffic towards the same external network by two or more internal hosts is very low. Nevertheless, relying on a severity bigger than one should lead to a quite low false positive rate. However, there is a conflict between the false positive rate and the recall. The overall traffic level impacts again the ability to aggregate prefix failures by internal hosts. The higher the traffic level is, the higher the probability to aggregate a prefix failure to more than one host and the more reliable a classification gets.

CHAPTER 5

Conclusion

5.1. Conclusion

This thesis intended to demonstrate how efficient and comfortable FACT is for tracking connectivity issues. FACT has been made even more powerful by adding some new futures like IPv6 support, standardized data input and a smart reporting engine. The big goal of pushing FACT towards a state in which it could be used is still not reached and some further work is required. The entire framework should be deamonized so that FACT could run in the background, process traffic traces online and alert - when required - the network operator about new connectivity issues and require further investigation. Moreover, the false positive rate, recall and precision have to be determined which includes a detailed analysis of the severity and its properties. The future work should also include an advice for choosing a reliable level of the severity constrained by the size of the network.

APPENDIX A

Appendix

A.1. Prefix Files for FACT

Switchextract is a neat little perl script for generating the required prefix files for the FilterInOut and the Analyser of FACT. It may be found in the tool folder of the FACT sourcecode. For correctly using these perl script, the following preliminary work have to be done:

1. Download and install bgpdump from the RIPE RIS project.
2. Download the bgpdump file of the desired date from a suitable route collector - the best from the default free rrc00.ripe.net.
3. Adjust your own AS number within the perl script switchextract.pl, i.e. replace 559 at line 24 with your own AS number.

Then the following steps must be executed:

1. Create a file bgpdump.txt with bgpdump:
   `bgpdump bview.XXXXXX.XXXX.gz -m > bgpdump.txt`
2. Call the perl script switchextract.pl:
   `perl switchextract.pl bgpdump.txt prefixes.txt`
3. prefixes.txt has to be moved or linked to the analyser configuration folder of FACT
4. switch_prefixes.txt is needed by the FilterInOut and has to be moved or linked to the configuration folder of FilterInOut.

A.2. Orginalproblem

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

Semester Thesis
for
Daniel Aschwanden

Tutors: Dominik Schatzmann, Wolfgang Mühlbauer

Issue Date: 20.09.2010
Submission Date: 31.12.2010

Identification of Connectivity Issues in Large Networks using Data Plane Information

1 Introduction

More than 20 years after the launch of the public Internet, web forums are still full of reports about temporary unreachability of complete networks. We propose a system that helps network operators to detect *any* type of reachability problem with other autonomous systems and networks. In contrast to existing solutions, our approach is both *passive*, based on *data-plane* information, *faster than real time* in terms of data processing, and *highly efficient* in alerting about critical events.

2 Tasks

The task of this thesis is to improve the existing analysis framework.

This work should cover at least the following aspects : (i) include IPv6 support, (ii) include NfSen/NF-Dump support, (iii) improve the existing detector, and (iv) evaluate the detector on real-world events

2.1 IPv6 support

The framework does not yet provide IPv6 support. This should be fixed.

2.2 NfSen/NfDump

The data interface should be improved to be able to work with the well known flow collector system NfSen/NfDump.

2.3 Detector

The current version of the framework does not yet provide a smart detector. The student should design and implement a detector with a low false positive rate. This could possibly be achieved by considering the history of the traffic or active measurements.

2.4 Evaluation

The improved framework should be evaluated on real world data.

3 Deliverables

The following results are expected:
– Implementation of an improved framework
– Evaluation of added features based on real world data
– A final report, i.e. a concise description of the work conducted in this project (motivation, related work, own approach, implementation, results and outlook). The abstract of the documentation has to be written in both English and German. The original task description is to be put in the appendix of the documentation. The documentation needs to be submitted electronically. The whole documentation, as well as the source code, slides of the talk etc., needs to be archived in a printable, respectively executable version on a CDROM.

4 Assessment Criteria

The work will be assessed along the following lines:

1. Knowledge and skills
2. Methodology and approach
3. Dedication
4. Quality of results
5. Presentations
6. Report

5 Organisational Aspects

5.1 Documentation and presentation

A documentation that states the steps conducted, lessons learned, major results and an outlook on future work and unsolved problems has to be written. The code should be documented well enough such that it can be extended by another developer within reasonable time. At the end of the project, a presentation will have to be given at TIK that states the core tasks and results of this project. If important new research results are found, the results may be published in a research paper.

5.2 Dates

This project starts on September 20, 2010 and is finished on December 31, 2010. At the end of the second week the student has to provide a schedule for the thesis, that will be discussed with the supervisors.

One intermediate presentations for Prof. Plattner and all supervisors will be scheduled after one month.

A final presentation at TIK will be scheduled close to the completion date of the project. The presentation consists of a 20 minutes talk and reserves 5 minutes for questions. Informal meetings with the supervisors will be announced and organised on demand.

5.3 Supervisors

Dominik Schatzmann, schatzmann@tik.ee.ethz.ch, +41 44 632 54 47, ETZ G 95
Wolfgang Mühlbauer, muehlbauer@tik.ee.ethz.ch, +41 44 632 70 17, ETZ G 90

1st March 2011

Bibliography

[Arbor Network, 2011] Arbor Network (2011). Peakflow. http://www.arbornetworks.com.

[Bush et al., 2009] Bush, R., Maennel, O., Roughan, M., and Uhlig, S. (2009). Internet optometry: assessing the broken glasses in internet reachability. In *Internet Measurement Conference'09*, pages 242–253.

[CA technologies, 2011] CA technologies (2011). NetQoS. http://www.netperformance.com.

[Haag, 2011] Haag, P. (2011). NfDump. http://nfdump.sourceforge.net/.

[Huston, 2011] Huston, G. (2011). IPv4 Address Report. http://www.potaroo.net/tools/ipv4/index.html.

[Katz-Bassett et al., 2008] Katz-Bassett, E., Madhyastha, H. V., John, J. P., Krishnamurthy, A., Wetherall, D., and Anderson, T. E. (2008). Studying black holes in the internet with hubble. In *NSDI*, pages 247–262.

[Madhyastha et al., 2006] Madhyastha, H. V., Isdal, T., Piatek, M., Dixon, C., Anderson, T. E., Krishnamurthy, A., and Venkataramani, A. (2006). iplane: An information plane for distributed services. In *OSDI*, pages 367–380.

[Schatzmann et al., 2011] Schatzmann, D., Leinen, S., Kgel, J., and Mhlbauer, W. (2011). FACT: Flow-based Approach for Connectivity Tracking. In *Passive and Active Measurement conference (PAM 2011)*, Atlanta, Georgia, USA.

[SWITCH, 2011a] SWITCH (2011a). SWITCH IPv6 Service. http://www.switch.ch/network/services/ipv6/index.html.

[SWITCH, 2011b] SWITCH (2011b). Which Swiss university will be the first to implement IPv6? http://www.switch.ch/about/initiatives/community/contest/index.html.

[Zhang et al., 2008] Zhang, Y., Mao, Z. M., and Zhang, M. (2008). Effective diagnosis of routing disruptions from end systems. In *NSDI*, pages 219–232.