Performance Evaluation of Classification and Feature Selection Algorithms for NetFlow-based Protocol Recognition

Sebastian Abt, Sascha Wener, and Harald Baier
da/sec – Biometrics and Internet Security Research Group,
Hochschule Darmstadt, Darmstadt, Germany
{sebastian.abt,harald.baier}@h-da.de, sascha.wener@stud.h-da.de

Abstract: Protocol recognition is a commonly required technique to deploy service-dependent billing schemes and to secure computer networks, e.g., to reliably determine the protocol used for a botnet command and control (C& C) channel. In the past, different deep packet inspection based approaches to protocol recognition have been proposed. However, such approaches suffer from two drawbacks: first, they fail when data streams are encrypted, and second, they do not scale at high traffic rates. To overcome these limitations, in this paper we evaluate the performance in terms of precision and recall (i.e., accuracy) of different feature selection and classification algorithms with regard to NetFlow-based protocol recognition. As NetFlow does not rely on payload information and gives a highly aggregated view on network communication, it serves as a natural data source in ISP networks. Our evaluation shows that NetFlow based protocol detection achieves high precision and recall rates of more than 92% for widespread protocols used for C&C communication (e.g., HTTP, DNS).

1 Introduction

One fundamental aspect of Internet communication is standardization of port numbers for well-known applications. As of today, all port numbers in the range 0-1024 have been assigned to specific applications by the Internet Assigned Numbers Authority (IANA). Thus, it is well-known that, for instance, HTTP servers utilize TCP port 80 to listen for incoming requests or that SSH servers accept incoming login attempts on TCP port 22. This standardization allows users to consume specific services without knowing and specifying the port number component of a remote socket. However, port numbers are not always used in accordance to this standardization. A deviation from standardized port numbers can have various legitimate reasons (e.g. business policy reasons, IP address conservation reasons, security reasons, etc.) as well as illegitimate reasons (e.g. covert channel communication, bypassing firewall rules, payment fraud, etc.). An important application of protocol recognition is the reliable identification of the actual protocol used for a network connection, e.g., to detect botnet command and control (C& C) communication.

To tackle illegitimate deviation of port number standardization, protocol recognition approaches based on raw packets, i.e. Deep Packet Inspection (DPI), are usually applied [BTS06, FMMR10, HSSW05, IKF+09, KPF05, KcF+08, MP05, VRM+09, KBB+04].

Such DPI-based approaches have three strong disadvantages: First, the use of DPI potentially conflicts with data privacy and protection laws or regulatory requirements. This especially counts for its application in ISP networks. Second, the amount of data to process is very high in today's networks. Even with an access bandwidth of 50 Mbit/s, which can typically be found in residential access networks (e.g. DSL networks), up to 3.75 GiB worth of traffic have to be processed within a 10 minutes time frame per customer. Third, DPI-based approaches utilizing payload information usually fail when communication is encrypted.

To overcome the aforementioned limitations of DPI-based approaches, in this paper we systematically evaluate the utility of NetFlow data for protocol recognition. As NetFlow provides a highly aggregated description of network traffic, it effectively performs a reduction of the data to process. Furthermore, as NetFlow data records do not contain any payload information, utilization of NetFlow is more privacy-preserving and immune to encryption. More specifically, in this paper we assess the information provided within NetFlow records with regard to its capability of solving a multi-class classification problem, i.e. that of reliably recognizing HTTP, SMTP and DNS traffic without requiring any port number information.

To achieve our goal, we assess classification results of two different classifiers, i.e. C4.5 decision tree algorithm and SVM, using data sets derived by applying three different feature selection approaches, i.e. correlation-based feature selection, consistency-based feature selection and principal component analysis, to attributes directly exported in NetFlow version 5 records. Our evaluation is performed on NetFlow data exported from a German Internet service provider (ISP). It shows that NetFlow based protocol detection achieves high precision and recall rates of more than 92% for widespread protocols used for C&C communication (e.g., HTTP, DNS).

The remainder of this paper is structured as follows: Section 2 describes related work in the area of protocol recognition. Section 3 briefly describes the necessary foundations. Section 4 details our proceeding and section 5 discusses our results. Finally, section 6 proposes future work and concludes.

2 Related Work

As of today, different approaches to protocol recognition using raw packet samples [BTS06, FMMR10, HSSW05, IKF+09, KPF05, KcF+08, MP05, VRM+09, KBB+04] and NetFlow traces [RS10, CEBRSP09, JMG+07] have been published. A detailed survey of IP traffic classification techniques before the year 2008 was conducted by Nguyen and Armitage in [NA08]. A performance comparison of ML algorithms in context of IP traffic classification has been published in [WZA06].

Generally, existing approaches can be classified as follows:

- Approaches looking for known signatures in packet payload (e.g. [MP05, KBB+04, SSW04]).

- Approaches deriving statistical descriptors of packet payload (e.g. [FMMR10] and [HSSW05]).
- Approaches that rely on statistical properties of network flows created by applications (e.g. [BTS06, CEBRSP09, JMG+07, KcF+08])
- Approaches characterizing the behavior of applications by recognizing the traffic pattern generated [IKF+09, KPF05, VRM+09]

In [RS10], Rossi et al. describe an approach that makes use of histograms created for received bytes and packets and hereby forms signatures which characterize different applications. It has been discovered that most application protocols show distinct frequency distributions of packets and bytes transmitted. A Support Vector Machine (SVM) classifier is then trained on the created signatures to perform classification. NetFlow records required for the analysis are derived from raw packet traces.

[CEBRSP09] investigates the impact of sampled NetFlow as input for classification. The authors note to rely only on features of NetFlow Version 5 in order to conduct classification by using a C4.5 decision tree. The required NetFlow data set has been extracted from raw packet traces. Labeling was performed using deep packet inspection. As a result, the authors observed a significant drop of classification accuracy with extensive sampling being applied. Similar to that, [JMG+07] evaluates impact of sampling to NetFlow-based protocol detection using naïve Bayes kernel estimation. In contrast to the observations of [CEBRSP09], the authors observed no significant accuracy drop when using naïve Bayes kernel estimation. In fact, the authors report a drop of accuracy of only 8% when the training data set size is reduced from 100% to 0.1%.

3 Foundations and Methodology

This section describes foundations and methodology of our approach. More specifically, section 3.1 introduces NetFlow, section 3.2 describes the feature selection approaches we use throughout this paper, and section 3.3 describes the machine learning approaches we use. Finally, section 3.4 describes the metrics we use to measure our results.

3.1 Network Flows

In a nutshell, a flow, in terms of a computer network, is a distinct, unidirectional stream of packets directed from one endpoint to another within a certain period of time. A proper definition for IPv4 networks can be given by the 5-tuple: source and destination IP addresses, source and destination ports as well as the Layer 4 protocol number. Within a certain period of time, a flow is uniquely defined by this 5-tuple, also called flow keys [CBL+08].

NetFlow is a proprietary format for flow information export by Cisco Systems and was

originally designed by Cisco as a cache to improve IP flow lookups in network routers. In recent years, NetFlow has established itself as the dominant solution for flow-level measurements. Thus, it is widely implemented and can be used easily for many purposes. As of today, several versions of NetFlow exist. While Version 5 is the most common one, Version 9, is the newest and offers more flexibility by enabling the sender to choose the composition of the exported information. Version 9 as an advancement called Internet Protocol Flow Information Export (IPFIX), has recently been standardized in an Internet Engineering Task Force (IETF) draft [CBL+08]. For the experiments performed throughout this paper, NetFlow version 5 has been used.

Besides the already mentioned flow keys, NetFlow format contains information about duration, ToS flags, TCP flags, Autonomous System (AS) numbers as well as byte and packet count per flow [Cis07].

3.2 Feature Selection

When utilizing ML for IP traffic classification, one assumes that there is a certain relation between features of each instance and classes to be identified. Ideally, the ML algorithm will then be capable of separating instances into different groups or classes by analyzing the supplied feature vector. As the storage and computing cost rises with the dimensionality of the feature vector used to describe each instance, it is important to choose the minimum amount of features which lead to maximum differentiation. Feature selection only selects the most important features and hence reduces computational complexity [NA08]. In addition, irrelevant or redundant features have a negative impact on most ML techniques [DLM00, NA08, WF05].

For the remainder of this work, we compare three different popular feature selection methods which are correlation-based feature selection (CFS), consistency-based feature selection (CON), and principal component analysis (PCA), which are briefly described next.

3.2.1 Correlation-based Feature Selection

In correlation-based feature selection (CFS), features which are highly correlated or predictive of a class are expected to be relevant for classification, otherwise they are not. As already mentioned, features are redundant if they are closely correlated and contain similar information. According to [Hal99], CFS is based on the hypothesis that a good feature subset is one that contains features highly correlated with the class, yet uncorrelated with each other. CFS therefore calculates the degree of correlation between the features and simultaneously evaluates their predictive ability. Based on this simple principle, best performing features are then chosen for the recommended feature set.

3.2.2 Consistency-based Feature Selection

Full consistency means that there is zero inconsistency. The measure for consistency-based feature selection is an inconsistency rate over the data set for a given set of features. Two sets of values are considered inconsistent if they match with all attributes but their class labels. For instance, inconsistency is caused by two instances $(0, 1, ClassLabelA)$ and $(0, 1, ClassLabelB)$. The inconsistency count for a set of values is calculated by the number of times it appears in the data set minus the largest number of this set among other classes. The inconsistency rate is then computed as the sum of all inconsistency counts of value sets for the regarded feature subset divided by the total number of different value sets in the data set. In other words, the inconsistency rate describes how inconsistent a certain feature subset is [DLM00]. From these metrics one can derive which features are the most consistent ones and, consequently, present a selection of features which most sufficiently fulfill this requirement.

3.2.3 Principal Component Analysis

The Principal Component Analysis (PCA) treats instances of a specific data set as vectors of a P-dimensional space, with P denoting the number of attributes per instance. The basic idea behind PCA is to transform the given data set into a Q-dimensional space, with $Q < P$. I.e., into a set of linearly uncorrelated variables, named principal components, maintaining roughly the same information as in original space. Principal components are computed iteratively as the vectors that approximate the given data set best, i.e. having minimum distance to all data points, with the requirement that each new principal component is orthogonal to previously identified principal components. As a result, each principal component accounts for as much of the variability in the data as possible. The eigenvalues in PCA reflect how much of the overall variance of all variables is covered by a single principal component. Subsequently, one can infer how the new principal components are made up of the original attributes. By ranking these results, one can then conduct a feature selection that contains the most important information.

3.3 Machine Learning Algorithms

Machine Learning (ML) is a branch of artificial intelligence and describes a computer based process of finding and describing structural patterns in a given data set. Supervised learning is a specific ML technique which requires a set of a-priori labeled instances from which a model is built. The phase during which a model is built based on labeled data is called training. Afterwards, the model can be used to classify unknown data, i.e. to assign a label to a specific unknown instance.

Throughout this paper, we use two different ML approaches which are commonly used in, e.g., IP traffic classification and anomaly detection: the C4.5 decision tree algorithm and Support Vector Machine (SVM) algorithm. Both algorithms are briefly described next.

3.3.1 C4.5 Decision Tree Algorithm

C4.5 is a decision tree algorithm which is the predecessor of ID3 and its commercial successor C5.0 [WF05, Hal99]. Decision trees are commonly used in practice due to their robustness, model build time and classification speed. A decision tree is a tree structure that consists of decision nodes and leaves. Decision nodes consist of tests on features and leaves represent classes. C4.5 tries to fit a model to a given classification problem by recursively generating such trees. For each decision node, C4.5 computes normalized information gain of the remaining features. A feature is chosen as a decision criterion, i.e. as a new child node, if normalized information gain of a test computed on that feature is the maximum. This process is repeated for each decision node.

3.3.2 Support Vector Machine

Support vector machines are a mixture of linear modeling and instance based learning, which use linear models to implement nonlinear boundaries. Generally, SVM is supposed to offer good generalization capabilities and little overfitting, but on the other hand shows a lack of transparency concerning the results. Moreover, SVM is very expensive in terms of computational complexity, for both model building and classification [WF05]. Unlike other classification algorithms, SVMs tend to use all the available features by combining them in a linear way. In general, an SVM tries to fit a model to a given classification problem by computing a set of maximum-margin hyperplanes separating the classes of interest. If the classes of a specific classification problem cannot be separated linearly using the feature vectors at hand, SVMs transform the input data, i.e. feature vectors, to higher-dimensional feature vectors by applying a kernel function $K(x, y)$ to it. This process is referred to as the kernel trick.

3.4 Classification Metrics

A common method to characterize performance of a classifier is to use metrics known as False Positives (FP), False Negatives (FN), True Positives (TP) and True Negatives (TN) [NA08]. Derived from that, further commonly used metrics are $Recall$, $Precision$ and $F - Measure$.

$Recall$ is defined as $Recall = \frac{TP}{TP+FN}$. $Recall$ serves as a measure of the number of correctly assigned positive labels. Thus, $Recall$ quantifies the sensitivity, i.e. the class coverage, of a classifier.

In contrast, $Precision = \frac{TP}{TP+FP}$ on the other hand measures the number of correctly assigned labels with respect to the overall number of positives. Hence, $Precision$ quantifies the noise or false alarms introduced by a specific classifier.

As weighted measure taking $Precision$ and $Recall$ into account, $F - Measure$ is commonly used to evaluate different algorithms. $F - Measure$ combines $Precision$ and $Recall$ by computing the harmonic mean, i.e. $F - Measure = 2\frac{Precision \cdot Recall}{Precision+Recall}$.

Throughout the remainder of this paper we primarily use these measures to compare the results we obtain using different feature selection and machine learning approaches.

4 Evaluation of Protocol Recognition Performance

This section describes our methodology in order to analyze performance of classification and feature selection algorithms with regard to protocol recognition. As mentioned earlier, for this analysis we focus on recognizing HTTP, SMTP and DNS traffic in NetFlow records. We deliberately chose to limit the number of protocols to those three as we assume them to be the currently most dominantly used protocols in the Internet. In the following sections, we describe our methodology (section 4.1) as well as the data sets (section 4.2) and features (section 4.3) we use.

4.1 Methodology

The general procedure we adhere to is to first derive two ground-truth data sets from a corpus provided to us by a German ISP as described in section 4.2. One data sets contains labels of HTTP, DNS and SMTP traffic, as these are the protocols we try to recognize. The second data set additionally introduces a class label "other" which is used to aggregate other NetFlow records which have been filtered out of the more basic data set. The rationale behind this second more advanced data set is to assess the recognition performance in a more realistic setup, as in real-world we will typically not face a closed and controlled environment.

After data sets have been compiled, feature selection is conducted for every resulting data set. In total, we create three feature sets using each of the aforementioned algorithms CFS, CON and PCA. For each feature set we train both classification algorithms, i.e. C4.5 and SVM. This gives us the possibility to observe the features selected by different feature selection approaches and to assess classification performance of different classification algorithms using different feature sets. In total, we perform 12 runs on two different data sets with three different feature selection algorithms and two distinct classifiers.

We apply C4.5 using stratified ten-fold cross-validation. All SVM runs are conducted using a stratified holdout with 66% split for the training data. Furthermore, all settings for the classification algorithms have been set to the default of the WEKA machine learning framework for the time being. All classification metrics presented in the following sections are measured in terms of number of NetFlows records.

4.2 Dataset

For our analyses, we obtained a NetFlow data set from a German ISP. The data set was collected in the ISP's data center behind a firewall. Thus, the NetFlow traces were collected in a server environment. All NetFlow records were recorded on July 30, 2012 between 11:47 am to 4:36 pm, containing approximately 5 hours or 1,763,547 NetFlow records. Based on these NetFlow records, a ground truth data set has been derived by applying ISP expert knowledge to the data. As all traces were collected in a controlled environment, we were able to assure that communication is performed on standardized ports. Hence, we assign labels based on the port number attribute reported in NetFlow data.

From this set of NetFlow traces, we generate two different corpora that we use for training and evaluation. The basic data set contains only the three aforementioned protocols, namely HTTP, DNS and SMTP. Our advanced data set additionally aggregates all other traces into a class "other". This "other" class consists of the remaining top 10 protocols we found in the sample set provided by the ISP: Hypertext Transfer Protocol Secure (HTTPS), Secure Shell (SSH), Internet Message Access Protocol (IMAP), IMAP Secure Sockets Layer (SSL), Post Office Protocol (POP3) SSL, Simple Network Management Protocol (SNMP), NTP. For each data set, we derive per-NetFlow record features as described next.

4.3 NetFlow Features

From the data set as described above, we derive the following features that we use per NetFlow record:

- td: transmission duration - Some protocols have a typical magnitude of duration, e.g. DNS tends to have rather short queries, HTTP also performs longer transfers of data.
- ipkt/ibyt: input packets/bytes - The number of packets/bytes transmitted is a valuable protocol characteristic, some protocols tend to send more packets than others.
- flg: TCP flags - These flags contain information about the status of the TCP connection.
- stos: source ToS - The Type of Service value is set according to the containing content and priority of the transmission, hence one can conclude which protocol was used to transfer the content .
- bps: bits per second - Bits per second reflects the bandwidth used by the regarded flow.
- pps: packets per second - PPS is assumed to be a differentiating metric to distinguish between protocols as different protocols tend to communicate in different intensity..
- bpp: bytes per package - We assume that depending on the underlying protocol we may notice a different average packet size.

Apparently, we deliberately chose to not use source and destination IP addresses and source and destination port numbers as features, whereas they are present in NetFlow records. The reason for this is that we do not want our algorithms to learn behavior of specific hosts within the network, but recognize characteristics of protocols. Thus, we believe that neglecting this feature will make our models more adaptable to different networks. Additionally, we do not use protocol numbers as this would lead to a logic conflict: as our ground-truth labeling is based on protocol numbers, including protocol numbers as features would lead to falsified results due to oversimplification of the classification task at hand.

Subsequently, these features are fed to the feature selection methods CFS, CON and PCA, as described above in order to derive the most relevant features. This process is conducted individually for each of our two data sets.

5 Results and Discussion

The following sections present and discuss some results we achieve after applying our methodology as described above.

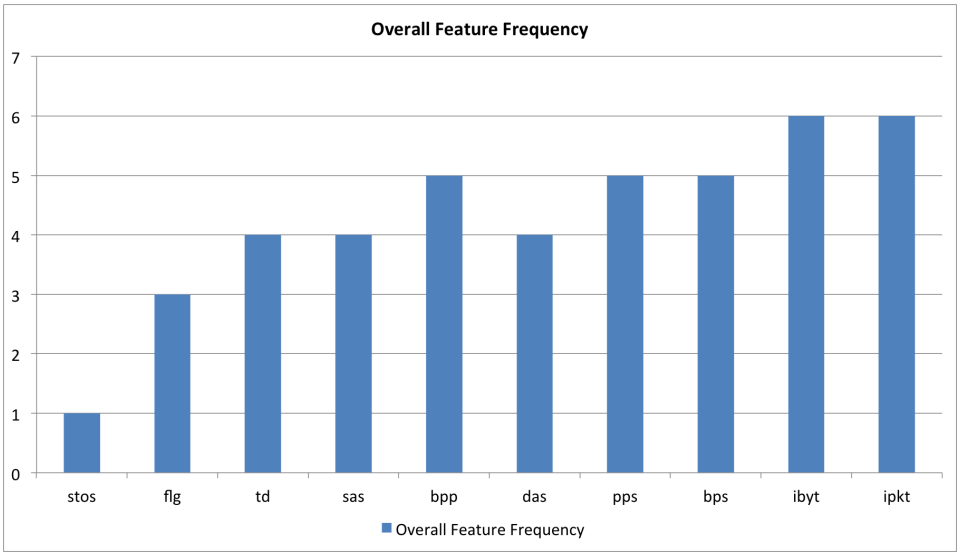
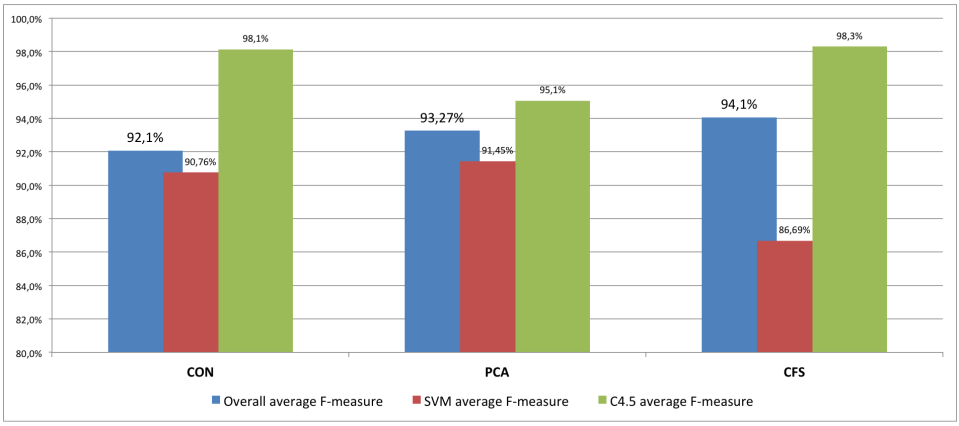


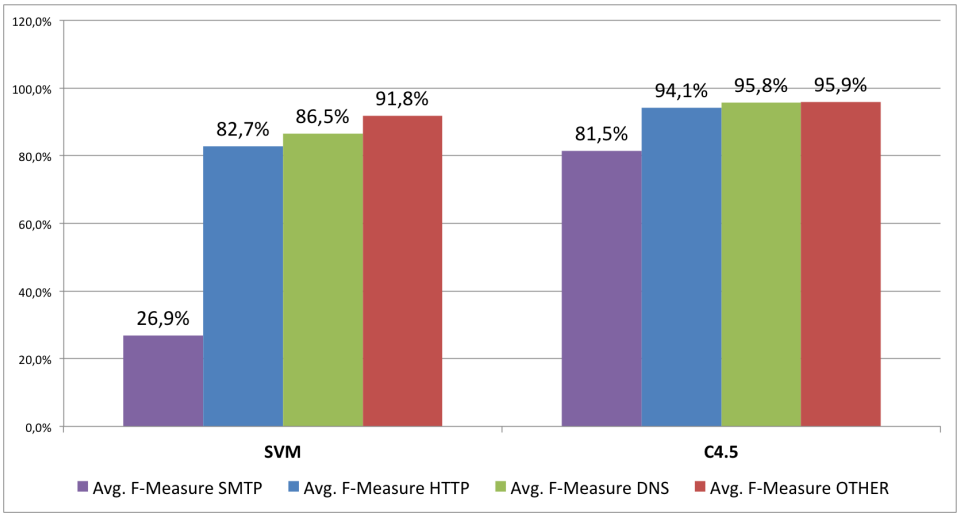
Figure 1: Frequency of Selected Features over all feature selection runs.

5.1 Feature Selection Methods

Figure 1 lists the frequency of each feature over all feature selection runs. When regarding the frequency distribution of the feature selection results, we notice that ipkt, ibyt and bps, pps and bpp were selected as the most significant features over all feature selection approaches. From this we learn that features related to communication intensity are best used to perform protocol recognition.



(a) $F - Measure$ per classification and feature selection algorithm.



(b) Average $F - Measure$ of SVM and C4.5 over all feature selections split according to class label.

Figure 2: Comparison of SVM and C4.5 performance.

5.2 Comparison between C4.5 and SVM

Figure 2a denotes the average $F - Measure$ we achieve for SVM and C4.5 using our two data sets and different feature selection algorithms. While all results are quite high, with the lowest $F - Measure$ score being 86.99% for SVM and CFS, the best results are generally achieved using C4.5 and the consistency-based feature selection algorithm.

Subsidiary, Figure 2b shows the average $F - Measure$ scores over all feature selection algorithms for the classifiers under investigation. The results are split into the different classes that have to be recognized. Again, it can clearly be seen that C4.5 shows slightly better results than SVM. Interestingly, the order in protocol recognition performance is is the same for both classifiers. However, the $F - Measure$ performance on SMTP by SVM is pretty poor compared to C4.5.

5.3 Classification Results

As one can see in figure 3, SMTP shows the poorest classification performance in terms of $F - Measure$, $Precision$ and $Recall$ of all three classes. Overall, most of the bad

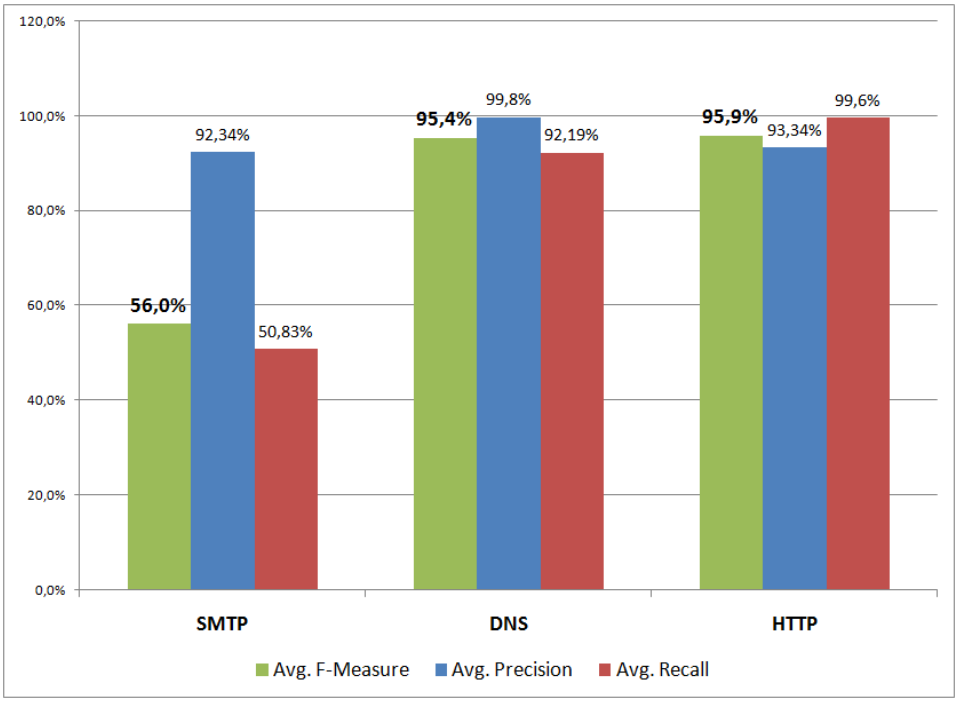


Figure 3: Basic data set $F - Measure$, $Precision$ and $Recall$

$Recall$ and $F - Measure$ values for SMTP are contributed by runs conducted through SVM, C4.5 performs significantly better. DNS results show a higher $Precision$ but less $Recall$ compared to HTTP. Nevertheless, in terms of $F - Measure$ HTTP is ranked first with a one percent gap. In general, however, the results achieved are quite high and indicate good protocol recognition performance using NetFlow. For HTTP and DNS, we receive an average $F - Measure$ over all feature selection approaches and classifiers of 95.9% and 95.4%, respectively.

For the advanced classification (cf. figure 4) the "other" class is ranked first in terms of $Precision$ and $F - Measure$ but only second in terms of $Recall$. In this matter HTTP is ranked first. In context of $Precision$ HTTP and SMTP hold the last place. SMTP holds the last place in terms of $F - Measure$. Compared to basic classification SMTP still is ranked in the last place but HTTP and DNS switched places, presumably due to the bad $Precision$ performance of HTTP. Also, the recognition performance generally drops in contrast to our basic data set. Nevertheless, for DNS and HTTP we sill achieve scores in the range of 90%.

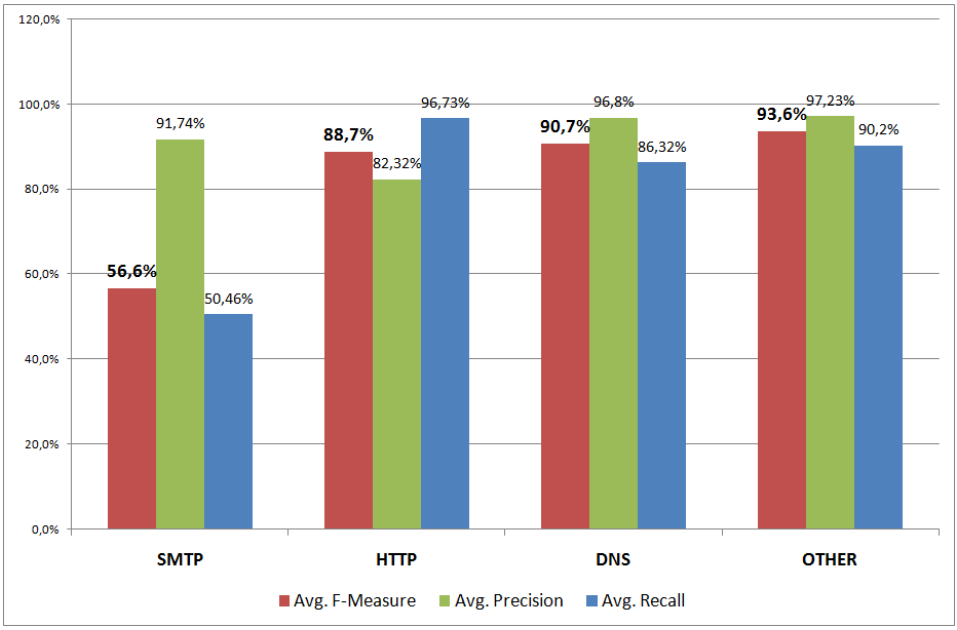


Figure 4: Advanced data set $F - Measure$, $Precision$ and $Recall$

6 Conclusion

In this paper we propose and evaluate different feature selection and classification algorithms with regard to NetFlow-based protocol recognition. Our evaluation on NetFlow

data from a German ISP shows that NetFlow based protocol detection is reliable and achieves high precision and recall rates of more than 92% for widespread protocols used for C&C communication (e.g., HTTP, DNS). In addition our work reveals a number of interesting relationships between NetFlow features as well as ML and feature selection algorithms.

In general, one can say that the order of classes to be detected best in descending order is DNS, HTTP and SMTP for both ML techniques. SVM performs very bad for the detection of SMTP in terms of $Recall$ and $F - Measure$ values. In average, C4.5 performs better than SVM. Surprisingly, classification performance for HTTP was found to be independent of any traffic direction. In terms of features, intensity-based feature are most significant. Concerning feature selection methods, C4.5 performs best using consistency-based and correlation-based feature selection, whereas SVM is ranked best using consistency-based feature selection and principle component analysis.

The next step is to integrate the NetFlow based protocol detection into a general NetFlow based botnet C&C detection framework to increase its precision and recall, respectively.

References

[BTS06] Laurent Bernaille, Renata Teixeira, and Kavé Salamatian. Early Application Identification. *ACM CoNEXT'06*, pages 1–12, 2006.

[CBL+08] Benoit Claise, Stewart Bryant, Simon Leinen, Thomas Dietz, and Brian Brian Trammell. RFC 5101 - Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information, 2008.

[CEBRSP09] Valentın Carela-Español, Pere Barlet-Ros, and Josep Solé-Pareta. Traffic Classification with Sampled NetFlow, 2009.

[Cis07] Cisco Systems. Catalyst Switched Port Analyzer (SPAN) Configuration Example - Cisco Systems, 2007.

[DLM00] Manoranjan Dash, Huan Liu, and Hiroshi Motoda. Consistency Based Feature Selection. *PADKK '00 Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, pages 98–109, 2000.

[FMMR10] Alessandro Finamore, M. Mellia, Michela Meo, and Dario Rossi. KISS: Stochastic Packet Inspection. *IEEE Transactions on Networking*, 18(5):1505–1515, 2010.

[Hal99] Mark A. Hall. Correlation-based Feature Selection for Machine Learning, 1999.

[HSSW05] Patrick Haffner, Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang. ACAS: Automated Construction of Application Signatures. *In SIGCOMM'05 MineNet Workshop*, 2005.

[IKF+09] Marios Iliofotou, Hyun-chul Kim, Michalis Faloutsos, Michael Mitzenmacher, Prashanth Pappu, and George Varghese. Graph-Based P2P Traffic Classification at the Internet Backbone. *12th IEEE Global Internet Symposium (GI2009)*, 2009.

[JMG+07] Hongbo Jiang, Andrew W. Moore, Zihui Ge, Shudong Jin, and Jia Wang. Lightweight application classification for network management. *ACM SIGCOMM Internet network management (INM '07)*, pages 299–304, 2007.

[KBB+04] Thomas Karagiannis, Andre Broido, Nevil Brownlee, kc claffy, and Michalis Faloutsos. Is P2P dying or just hiding? *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, 3:1532–1538, 2004.

[KcF+08] Hyunchul Kim, kc claffy, Marina Fomenkov, Dhiman Barman, Michalis Faloutsos, and KiYoung Lee. Internet traffic classification demystified myths, caveats, and the best practices. *ACM CoNEXT'08*, 2008.

[KPF05] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. BLINC: multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, pages 229–240, 2005.

[MP05] Andrew W. Moore and Konstantina Papagiannaki. Toward the Accurate Identification of Network Applications. *Passive and Active Measurement (PAM'05)*, 2005.

[NA08] Thuy T.T Nguyen and Grenville Armitage. A Survey of Techniques for Internet Traffic Classification using Machine Learning. *Communications Surveys & Tutorials, IEEE vol. 10*, pages 56–76, 2008.

[RS10] Dario Rossi and Valenti Silvio. Fine-grained traffic classification with Netflow data, 2010.

[SSW04] Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang. Accurate, scalable in-network identification of p2p traffic using application signatures. *Proceedings of the 13th international conference on World Wide Web, WWW'04*, pages 512–521, 2004.

[VRM+09] Silvio Valenti, Dario Rossi, Michela Meo, Marco Mellia, and Paola Bermolen. Accurate, fine-grained classification of P2P-TV applications by simply counting packets. *Traffic Measurement and Analysis (TMA)*, 2009.

[WF05] I. H. Witten and Eibe Frank. *Data mining: Practical machine learning tools and techniques*. Morgan Kaufman, Amsterdam and and Boston and MA, 2 edition, 2005.

[WZA06] Nigel Williams, Sebastian Zander, and Grenville Armitage. A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification, 2006.