Generation of a new IDS Test Dataset: Time to Retire the KDD Collection

Gideon Creech, *Student Member, IEEE*, and Jiankun Hu†, *Member, IEEE*

***Abstract*—Intrusion detection systems are generally tested using datasets compiled at the end of last century, justified by the need for publicly available test data and the lack of any other alternative datasets. Prominent amongst this legacy group is the KDD project. Whilst a seminal contribution at the time of compilation, these datasets no longer represent relevant architecture or contemporary attack protocols, and are beset by data corruptions and inconsistencies. Hence, testing of new IDS approaches against these datasets does not provide an effective performance metric, and contributes to erroneous efficacy claims. This paper introduces a new publicly available dataset which is representative of modern attack structure and methodology. The new dataset is contrasted with the legacy datasets, and the performance difference of commonly used intrusion detection algorithms is highlighted.**

I. Introduction

Computer security is a fundamental part of modern life. Regardless of whether a computer is a stand-alone system operated by a private user or a single node in a large corporate network, the security principles remain the same. With the prevalence of wireless systems in modern society and the increasing research focus on these critical information conduits, the imperative of ensuring effective security for these networks cannot be overstated.

Wireless networks present a unique security challenge. Unlike hardwired networks, where physical access to the network infrastructure is required to access data, wireless networks fundamentally expose all data by virtue of RF propagation. Whilst modern cryptographic algorithms are used to provide a high level of data security, they are less effective at protecting against packet injection and other remote attacks [1], [2], [3]. Indeed, wireless networks are particularly vulnerable to attackers as their very nature requires communication with unknown devices to be allowed; if this function was disabled, no new hosts could join the network, completely negating the flexibility provided by this communication medium.

It is computationally expensive for a hacker to penetrate a modern wireless network, but the task is by no means impossible. Indeed, many open source tools such as the Metasploit Framework (MSF) [4] exist which can automate much of this task, and it is widely acknowledged that the penetration of wireless networks is a persistent threat in the modern cyber environment. Like other types of network, if the outer perimeter is bypassed then only internal security measures stand between an attacker and critical data.

Host-based Intrusion Detection Systems (HIDS) represent one of the most effective means of defeating an attacker who has bypassed the outer perimeter of the network [5]. HIDS are a subset of Intrusion Detection Systems (IDS), and monitor the activity on a single host. By contrast, Network IDS monitor communications between hosts, and attempt to detect malicious activity present in that network traffic. Significant research effort has been expended over the past decades in improving HIDS performance [6]. This research has centred on two main methodologies.

The first methodology is signature-based detection. Using this approach, templates of known attack patterns are distributed by a central authority, and the IDS then matches observed behaviour against these templates. This approach is similar to antivirus methods, and provides a reasonable level of accuracy against known attacks. The critical weakness of signature-based systems is that they are unable to recognise new zero day, previously unseen attacks.

The second method commonly used by IDS is anomaly detection. This method constructs an accurate baseline of the system using known normal behaviour and then compares contemporary system behaviour against this baseline. If a large divergence is observed, the actions are classified as suspect and a detection is recorded. Anomaly-based systems have a good inherent capability to detect zero day attacks, but suffer from potentially high False Alarm Rates (FAR). This high FAR arises from the difficulty of creating accurate baselines in a highly dynamic modern operating system environment [7].

Testing of intrusion detection systems is a difficult task. Comparison between different approaches requires the use of a standard dataset. Historically, most IDS research had used the KDD datasets, which were compiled in 1998 and 1999 [8], [9]. These datasets represented the first systematic approach to IDS data generation, and were an incredibly valuable and innovative resource at the time of release. Over time, these datasets have lost most of their relevance through a natural aging process, but are still used as the basis for IDS validation by most researchers, nonetheless. Their continued use is largely because of the absence of any viable alternative.

This paper seeks to address the issues discussed above by introducing a new, contemporary IDS dataset representing a modern Linux operating system, Ubuntu 11.04, and cutting edge attack methodology [4], [10]. The remainder of this paper is organised as follows: Section II outlines the major existing criticisms of the KDD data. Section III discusses the generation of the new ADFA Linux Dataset (ADFA-

Gideon Creech is a PhD Student at the University of New South Wales.

Jiankun Hu is the Professor of Cyber Security at the Canberra Campus of the University of New South Wales.
†Correspondent author

Manuscript submitted on January 16, 2013.

978-1-4673-5939-9/13/$31.00 ©2013 IEEE

LD12), and presents the design decisions used in selecting its structure. Section IV presents the performance of several common decision engines (DE) used in IDS applications on the new dataset for comparison with historical performance against the KDD datasets. A discussion of these results is conducted in Section V, with concluding remarks in Section VI.

II. Existing Criticism of the KDD Datasets

Criticism of the KDD datasets [8], [9] has grown over the last ten years, and centres on the dated nature of the underlying experiment. These datasets represented the first real attempt to provide a benchmark for intrusion detection research and undeniably had a seminal impact in the field. At the time, there was no standardised way of evaluating IDS performance and DARPA attempted to remedy this through collaboration with various academic institutions.

The KDD experiments used a Solaris-based system to collect a wide range of data. System calls were available by processing the BSM audit data to extract the sequences. The work of Forrest et al [11] introduced and then further developed [12], [13], [14], [15] the concept of using system calls as the basis for anomaly centric classifications. As such, the KDD data represented the first significant collection of this important new information source [8].

Given the dynamism of computer technology, however, the relevance of the KDD data rapidly diminished in practical terms [16], [17], [18]. The Solaris operating system was selected largely because of its ease of integration and monitoring; whilst a perfectly functional OS, Solaris did not and does not have a large market share of the computer industry. Logically, any results produced on this operating system hence have limited generalisability to the cyber community as a whole [19]. Indeed, accessing system calls in modern computers can often be extremely challenging. Whilst the BSM functionality of Solaris is attractive in this regard, it is not representative of the utility of this approach in real-world applications.

Furthermore, the technology of the late 1990s has very little resemblance to the technology of the current era [16], [19], [20]. Operating systems have undergone many fundamental revisions in that time, and innovative development has drastically changed the basic nature of computer systems. Even something as simple as the conversion from 32 bit architecture to 64-bit systems has a profound effect on the exploitability of contemporary systems.

Additional concerns relate to the methodology employed when collecting the KDD data, with various researchers [16], [19], [20], [21] commenting adversely on the numerous data artefacts which reside throughout the KDD collection. This is a critical and very significant point. Machine learning is extremely sensitive, and artefacts can have a pronounced effect on classifications. Selection of model parameters itself may be influenced by data impurity, with consequently erroneous decisions made and faulty conclusions drawn.

Some limited attempts have been made to provide a more contemporary dataset, such as the UNM offering of 2004. Whilst undeniably more current than the KDD datasets, the UNM dataset [22] is extremely limited in scope and does not represent a full sampling of the target operating system, focusing instead on single processes. As can be seen from Table I, performance against the KDD dataset is saturated. It is unlikely that any of these IDS algorithms could hope to perform at these levels in a real-world deployment; unfortunately, the limitations of the KDD datasets makes further expansion of the knowledge horizon all but impossible.

Experiments continue to be conducted on the KDD data largely because of expedience [16], [18], [19], [20], [21]. It is acknowledged that this dataset lacks relevance, yet the absence of an alternative drives its continued employment. This paper introduces a new dataset representing modern attack technology, in the hope that a new and more relevant baseline can be established for future generations of IDS research.

TABLE I: Results from Other Algorithms

Algorithm	**Detection Rate** %	**False Alarm Rate** %
Data mining of audit files [23]	80.2	Not cited
Multivariate statistical analysis of audit data [24][1]	90	40
HMM and entropy analysis of system calls [25]	91.7	10.0
System call n-gram sliding window (assorted decision engines) [14][2]	$95.3 < DR < 96.9$	~ 6.0
RBF ANN analysing system calls [26]	96 mean	5.4 mean
MLP on subset of KDD98 [27][3]	99.2	4.94
SVM on subset of KDD98 [27][3]	99.6	4.17
kNN with Smooth Binary Weighted RBF [28]	96.3	6.2
Rough Set Clustering [29]	95.9	7.2

III. Dataset Generation

Ubuntu Linux version 11.04 was selected as the host OS for the ADFA-LD12. This operating system is a modern Linux distribution, currently installed by many users world-wide [30]. To allow for a web based attack vector, Apache Version 2.2.17 [31] running PHP Version 5.3.5 [32] was installed and enabled. The operating system was fully patched, and FTP, SSH and MySQL Version 14.14 [33] were started as services with their default ports. TikiWiki Version 8.1 [34] was installed as a web-based collaborative tool. This version was selected as it has a known vulnerability, detailed in [35], which allows for exploitation by web based attacks.

This configuration represents a reasonable generalisation of a modern Linux local server, offering file sharing, database services, remote access and web server functionality, with some small residual vulnerabilities.

[1]Extracted from ROC curves with DR set at 90%.

[2]Whilst good, these results were obtained using a small dataset so decision engine performance may not generalise well.

[3]Results potentially lack generalisability.

A. Attack selection and rationale

Table II shows the breakdown of payloads and vectors used to attack the Ubuntu OS. In selecting these attacks, careful consideration was given to the methodologies adopted by contemporary penetration testers and hackers, with a delicate trade-off between vulnerability of the target system and realism required. Whilst remote exploits are extremely powerful and potentially catastrophic for the recipient, it is rare to find them in modern patched systems. As such, the target server for the ADFA-LD12 was fully patched in order to provide a realistic target. A small vulnerability was engineered in, however, namely the TikiWiki remote code execution vulnerability [35]. This represents a realistic and small flaw in an otherwise well configured server, which is a good simulation of the real world. Overly porous and trivially exploitable servers are rare, and hence their use in this environment would be utterly artificial.

TABLE II: Attack Structure

Payload/Effect	Vector
Password bruteforce	FTP by Hydra [36]
Password bruteforce	SSH by Hydra [36]
Add new superuser	Client side poisoned executable
Java Based Meterpreter	Tiki Wiki vulnerability [35] exploit
Linux Meterpreter Payload	Client side poisoned executable
C100 Webshell	PHP Remote File Inclusion vulnerability

The first two attack classes represent brute force password guessing attempts on the open services. Often the last resort of attackers due to their large footprint, this vector is nonetheless a valid tool in the hacker's arsenal. FTP and SSH are commonly available services, and are regularly attacked if exposed to external sources. Hence, their inclusion represents a realistic threat, albeit one that probably originates from a less skilled attacker or one who has exhausted all other options.

Privilege escalation in a Linux environment can be extremely difficult. One way around this is to create a new user with superuser privileges. This can be done in many different ways, but for this dataset a client side attack was used. A payload encoded into a Linux executable using Metasploit [4] was uploaded to the server in a simulation of a social engineering attack. This payload was then triggered both remotely and locally using a collection of techniques such as local file inclusion and simulated social engineering.

Meterpreter is a custom payload unique to the freely available Metasploit framework [4]. Metasploit is arguably the most popular open source hacking toolkit, and is widely used by penetration testers, security professionals and hackers. Meterpreter is an enhanced functionality command shell, which greatly facilitates the remote compromise of a targeted system. The TikiWiki vulnerability was used to upload a copy of the Java Meterpreter payload, which initiated a reverse TCP connection to the attacking computer when executed. Once the shell was established, various actions were taken on the host system, including local privilege escalation, attempts to access the shadow password file, and the installation of persistent backdoor tools.

Similarly, a Linux executable version of the Meterpreter payload was uploaded using social engineering and similar manipulations performed. This represents a distinct attack from the Java version, as both Meterpreter implementations use different methods to achieve the same command shell functionality. Whilst the hacker sees the same interface, the mechanics are vastly different and hence the system call pattern for each version will differ significantly. Finally, a PHP-based remote file inclusion vulnerability was used to upload the C100 webshell which was then used to further compromise the host system and escalate privileges. The C100 shell is a sophisticated piece of PHP code which provides an illicit GUI interface to the attacker through a web browser, hence allowing manipulation of the underlying operating system.

This collection of attacks represents the assault methodology which might be employed by a hacker of intermediate skill level. It exploits all available avenues of attack, ranging from low level password guessing through social engineering and web-based attacks to remote exploits. It is representative of current practices, and provides a modern attack set for IDS evaluation.

B. Dataset structure

The ADFA-LD12 is designed for anomaly based systems, not signature recognition IDS. It consists of three different data groups, with each group containing raw system call traces. Each training or validation data trace was collected during normal operation of the host, with activities ranging from web browsing to LaTeX document preparation. Traces were generated using the *auditd* Unix program and then filtered by size. For training data, traces outside the size range 300 Bytes to 6 kB were rejected, and for validation data, traces outside the range 300 Bytes to 10 kB. This represents an effective trade-off between data fidelity and unnecessary processing burden.

Table III shows the number of traces contained in each class. The current structure provides a ratio of approximately $1:5$ between training data and validation data, with each attack set consisting of all traces meeting the filtering criteria for the period of the different attacks.

TABLE III: Dataset Structure

Data Type	Trace Count
Normal Training Data	833 Traces
Normal Validation Data	4373 Traces
Attack Data	10 Attacks per Vector (See Table II)

System call traces are widely regarded as the most accurate way of detecting intrusions for anomaly based systems, and were first used by Forrest et al [11], [12], [13], [14], [15]. They are extensively used in the IDS research community, and were selected as the metric for collection by this dataset as a result.

IV. Results and Validation of Dataset

After generation of the dataset as discussed in Section III, the validity of this new test data was examined by evaluating the performance of several well-known IDS DE algorithms. The four algorithms used were hidden Markov models [37],

[38], the STIDE approach [11], K-Means clustering, and the K-Nearest Neighbour algorithm.

As can be seen in Figure 1, performance was significantly worse than when evaluated against the KDD98 and 99 datasets [8], [9], [39]. Contrasting the seemingly mediocre performance demonstrated in Figure 1 against that shown in Table I, it is clear that the ADFA-LD12 is much more challenging that the KDD family.

A. Analysis of the Complexity Level of the ADFA-LD12

The new ADFA dataset has a much larger degree of similarity between attack data and normal data than the KDD collections. Consequently, the new dataset is much more representative of current cyber attacks, and forms a realistic and relevant metric for IDS performance evaluation.

To demonstrate the complexity of the ADFA-LD12 data, the clustering analysis discussed in [40] was implemented, and the feature charts for KDD98 and ADFA-LD12 compared. [40] uses a 'bag of system calls' methodology, similar to the 'bag of words' approach to text mining. The resultant clustering feature produced is $n-$dimensional, where n represents the number of system calls present in the traces. [40] claims 100% detection rate for 1.29% false alarm rate against the UNM live LPR dataset using this feature, indicating strong performance against legacy IDS data. The UNM dataset is of comparable vintage to the KDD data, and focuses on single processes rather than a whole of system evaluation.

As can be seen from Figure 2, the complexity of the new ADFA dataset is significantly greater than that of KDD98 when viewed in terms of the feature proposed by [40]. In the KDD plot, the difference between the red attack data and blue normal data is so profound that it can be easily observed by the naked eye; there are clearly different maxima and minima, and significantly more pronounced peaks at several key system calls. The ADFA datset, however, is much more homogeneous, with only a few discernible differences.

This demonstrated complexity explains the discontinuous performance of clustering algorithms and the poor overall performance of existing sequential pattern algorithms presented in Figure 1. It is also a compelling argument for the adoption of a new benchmark dataset, as modern cyber activity is clearly and demonstrably fundamentally different from legacy data.

V. Discussion

When compiling the data set, an unexpected system call distribution pattern was observed. It was forecast that the compromised system process hosting the attacking shell would demonstrate a spike in the frequency of system calls, providing an indication of suspicious behaviour which would then be detected by the evaluating DE. This prediction was proved wrong by observation. As the data collection progressed, it became clearly apparent that beyond an initial burst of system calls, most of the functionality of the advanced payloads was provided by processes other than the initially compromised program. This is particularly significant because it represents a fundamental change in attack methodology.

In KDD-era attacks, only a single process was usually affected. This in turn meant that this process would have a high degree of evidence present in its system call trace. Existing IDS algorithms optimised for these datasets leveraged this fact to produce strong classification performances. The difficulty of the ADFA-LD12 arises from the fact that modern attacks no longer follow this pattern, with evidence of their activities spread between numerous system call traces. As such, either a new data rich feature must be used to provide the input to existing algorithms, or decision fusion avenues must be aggressively pursued to combine the information present in these disparate traces.

Of particular note is the performance of the two clustering approaches. Both algorithms produced a discontinuous ROC curve, jumping from performance no greater than 3% DR for 3% FAR to 98% DR for 98% FAR. This is represented in Figure 1 by the two large black clusters in the bottom left and upper right of the plot. These two algorithms were implemented using publicly available Python [41], [42] and Matlab toolboxes [43], as well as purpose written code produced by the authors.

All implementations delivered the same range of results, confirming the initially surprising discontinuity. The authors hypothesize that the clustering algorithms perform so poorly against this dataset because of the similarity of the attack data to the normal data. This is a feature of modern hacking attacks, which present a distributed approach to system compromise, spreading their activity through multiple processes. By contrast, older techniques and vulnerabilities such as exploited by the KDD experimenters were limited to a single process, thus creating a larger and more detectable system call footprint.

The 'bag of system calls' approach used in [40] and other clustering algorithms inherently loses all positional data of the system call traces, relying instead on a frequency analysis to provide a discrimination feature. Whilst this approach is able to classify legacy datasets with reasonable accuracy, it fails against the more complex ADFA-LD12. This strongly indicates that the reduced entropy feature suggested by [40] is not sufficient for use in the modern environment. Indeed, the mediocre performance of algorithms which do not dispense with positional information, shown in Figure 1, suggests that any increase of IDS performance will be contingent on the development of either a new feature or a new decision engine.

These results clearly demonstrate that the methods that have been perfected on older data are no longer relevant as they stand. The traditional features which have been selected for anomaly-based intrusion detection do not adequately capture the nature of attacks which are launched from a modern framework against contemporary operating systems. This is a natural evolution of hacking methodology, and is most likely tied to the evolving nature of software design rather than conscious attempts by the hacker collective to avoid intrusion detection systems. Regardless, it is clear that the modern environment is much more challenging than the existing public datasets represent.

Intrusion detection researchers must acknowledge this fact, and actively search for a richer data feature upon which to base the next generation of IDS. The poor performance of clustering

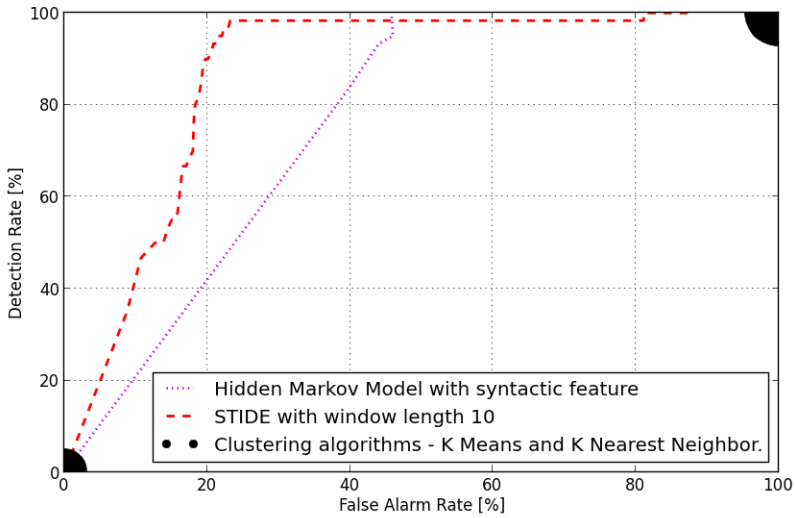

Fig. 1: Performance of Common IDS DE Algorithms against ADFA-LD12

algorithms against this dataset makes it clear that current measures of operating system behaviour are not of a high enough dimension to truly capture the subtlety of anomalous behaviour. Whilst the algorithmic methods themselves remain valid, the inputs are clearly not providing the required level of information and hence the decision engines are unable to reproduce the high fidelity of classification which they can achieve on older datasets.

VI. Conclusion

The ADFA-LD12 represents a worthy successor to the KDD collection. This new dataset uses a modern Linux operating system and the most recent publicly available exploits and methods. Two key discoveries were made in the course of generating this dataset: first, performance on the KDD datasets does not represent true performance against contemporary attacks, and second, current features used for decision engine classification are not sufficiently data rich to produce good results.

Future research will seek to address these problems by the development of a new feature, suitable for high quality classification. Researchers wishing to obtain copies of the ADFA-LD12 should email j.hu@adfa.edu.au or visit the UNSW@ADFA SEIT webpage.

VII. Acknowledgement

The authors would like to acknowledge the support from ARC (Australian Research Council) Projects LP110100602, LP100100404 & LP100200538, LP120100595.

References

[1] H. Zhao, J. Qin, and J. Hu, "Energy Efficient Key Management Scheme for Body Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. PrePrints, p. 1, 2012.

[2] B. Tian, S. Han, J. Hu, and T. Dillon, "A mutual-healing key distribution scheme in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 80 – 88, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804510001645

[3] B. Tian, S. Han, S. Parvin, J. Hu, and S. Das, "Self-Healing Key Distribution Schemes for Wireless Networks: A Survey," *The Computer Journal*, vol. 54, no. 4, pp. 549–569, 2011. [Online]. Available: http://comjnl.oxfordjournals.org/content/54/4/549.abstract

[4] "Metasploit Penetration Testing Software," http://www.metasploit.com, Accessed 24 Apr. 2012.

[5] Defence Signals Directorate - Cyber Security Operations Centre, "Top 35 Mitigation Strategies for Targeted Cyber Intrusions," http://www.dsd.gov.au/publications/Top_35_Mitigations.pdf, Accessed 13 Mar. 2012.

[6] Y. Xie, S. Tang, Y. Xiang, and J. Hu, "Resisting Web Proxy-based HTTP Attacks by Temporal and Spatial Locality Behavior," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. PrePrints, p. 1, 2012.

[7] J. Hu, "Host-based anomaly intrusion detection," in *Handbook of Information and Communication Security*, P. Stavroulakis and M. Stamp, Eds. Springer Berlin Heidelberg, 2010, pp. 235–255. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04117-4_13

[8] International Conference on Knowledge Discovery and Data Mining-98, "KDD98 Intrusion Detection Dataset," 1998. [Online]. Available: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1998data.html

[9] International Conference on Knowledge Discovery and Data Mining-99, "KDD99 Intrusion Detection Dataset," 1999. [Online]. Available: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html

[10] "Offensive Security Training and Services," http://www.offensive-security.com, Accessed 22 May 2012.

[11] S. Forrest, S. Hofmeyr, A. SoMayaji, and T. Longstaff, "A sense of self for Unix processes," in *Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on*, May. 1996, pp. 120–128.

[12] S. Forrest, S. A. Hofmeyr, and A. SoMayaji, "Computer immunology," *Commun. ACM*, vol. 40, no. 10, pp. 88–96, Oct. 1997. [Online]. Available: http://doi.acm.org/10.1145/262793.262811

[13] S. A. Hofmeyr, S. Forrest, and A. SoMayaji, "Intrusion detection using sequences of system calls." *Journal of Computer Security*, vol. 6, no. 3, p. 151, 1998. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&db=tsh&AN=1531432&site=ehost-live

[14] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: alternative data models," in *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*, 1999, pp. 133–145.

[15] S. A. Hofmeyr and S. Forrest, "Architecture for an Artificial Immune System," *Evolutionary Computation*, vol. 8, no. 4, pp. 443–473, 2000.

[16] C. Brown, A. Cowperthwaite, A. Hijazi, and A. SoMayaji, "Analysis of the 1999 DARPA/Lincoln Laboratory IDS evaluation data with NetADHICT," in *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*, Jul. 2009, pp. 1–7.

[17] P. Owezarski, "A Database of Anomalous Traffic for Assessing Profile Based IDS," in *Traffic Monitoring and Analysis*, ser. Lecture Notes in

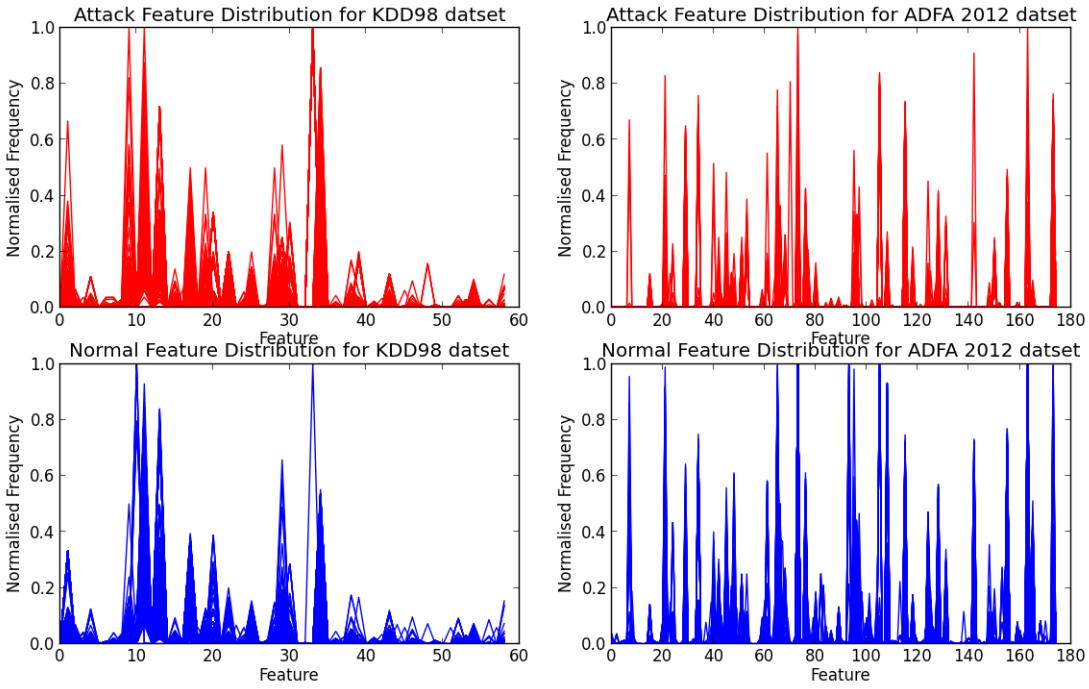

Fig. 2: Complexity of ADFA-LD12 and KDD98 Dataset

Computer Science. Springer Berlin / Heidelberg, 2010, vol. 6003, pp. 59–72.

[18] S. Petrovic, G. Alvarez, A. Orfila, and J. Carbo, "Labelling Clusters in an Intrusion Detection System Using a Combination of Clustering Evaluation Techniques," in *System Sciences, 2006. HICSS '06. Proceedings of the 39th Annual Hawaii International Conference on*, vol. 6, Jan. 2006, p. 129b.

[19] J. McHugh, "Testing Intrusion Detection Systems: a critique of the 1998 and 1999 DARPA Intrusion Detection System evaluations as performed by Lincoln Laboratory," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 4, pp. 262–294, Nov. 2000. [Online]. Available: http://doi.acm.org/10.1145/382912.382923

[20] V. Engen, J. Vincent, and K. Phalp, "Exploring discrepancies in findings obtained with the KDD Cup '99 data set," *Intelligent Data Analysis*, vol. 15, no. 2, pp. 251–276, 2011.

[21] M. Mahoney and P. Chan, "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection," in *Recent Advances in Intrusion Detection*, ser. Lecture Notes in Computer Science, vol. 2820. Springer Berlin / Heidelberg, 2003, pp. 220–237.

[22] Computer Science Department, "University of New Mexico Intrusion Detection Dataset," 2012. [Online]. Available: http://www.cs.unm.edu/~immsec/systemcalls.htm

[23] W. Lee, S. Stolfo, and K. Mok, "A data mining framework for building intrusion detection models," in *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*, 1999, pp. 120–132.

[24] N. Ye, S. Emran, Q. Chen, and S. Vilbert, "Multivariate statistical analysis of audit trails for host-based intrusion detection," *Computers, IEEE Transactions on*, vol. 51, no. 7, pp. 810–820, Jul. 2002.

[25] D. Yeung and Y. Ding, "Host-based intrusion detection using dynamic and static behavioural models," *Pattern Recognition*, vol. 36, no. 1, pp. 229–243, 2003.

[26] U. Ahmed and A. Masood, "Host based intrusion detection using RBF neural networks," in *Emerging Technologies, 2009. ICET 2009. International Conference on*, Oct. 2009, pp. 48–51.

[27] W.-H. Chen, S.-H. Hsu, and H.-P. Shen, "Application of SVM and ANN for intrusion detection," *Computers and Operations Researc*, vol. 32, no. 10, pp. 2617–2634, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054804000711

[28] A. Sharma, A. K. Pujari, and K. K. Paliwal, "Intrusion detection using text processing techniques with a kernel based similarity measure," *Computers and Security*, vol. 26, no. 78, pp. 488–495, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404807001113

[29] S. Rawat, V. Gulati, and A. Pujari, "A Fast Host-Based Intrusion Detection System Using Rough Set Theory," in *Transactions on Rough Sets IV*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3700, pp. 144–161.

[30] "Ubuntu Linux," http://www.ubuntu.com, Accessed 19 May 2012.

[31] "The Apache Software Foundation," http://apache.org, Accessed 19 May 2012.

[32] "PHP: Hypertext Processor," http://www.php.net, Accessed 19 May 2012.

[33] "MySQL," http://www.mysql.com, Accessed 19 May 2012.

[34] "TikiWiki: CMS groupware," http://info.tiki.org/Tiki+Wiki+CMS+Groupware, Accessed 19 May 2012.

[35] "Tiki Wiki CMS Groupware Remote PHP Code Injection."

[36] "THC-Hydra," http://www.thc.org/thc-hydra/, Accessed 19 May 2012.

[37] X. Hoang and J. Hu, "An efficient hidden Markov model training scheme for anomaly intrusion detection of server applications based on system calls," *IEEE International Conference on Networks (ICON 2004), Singapore*, vol. 2, pp. 470–474, Nov. 2004.

[38] J. Hu, X. Yu, D. Qiu, and H.-H. Chen, "A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection," *Network, IEEE*, vol. 23, no. 1, pp. 42–47, Jan.-Feb. 2009.

[39] M. Xie, J. Hu, S. Han, and H.-H. Chen, "Scalable hyper-grid k-nn-based online anomaly detection in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. PrePrints, p. 1, 2012.

[40] D.-K. Kang, D. Fuller, and V. Honavar, "Learning classifiers for misuse and anomaly detection using a bag of system calls representation," in *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, Jun 2005, pp. 118 – 125.

[41] "Python Programming Language Official Website," http://www.python.org, Accessed 12 Feb. 2012.

[42] "Numerical Python," http://numpy.scipy.org, Accessed 12 Feb. 2012.

[43] "Matlab Programming Language," http://www.mathworks.com.au/index.html, Accessed 12 Feb. 2012.