The Time-Varying Dependency Patterns of NetFlow Statistics

Alexander J. Gibberd*†, Marina Evangelou‡ and James D. B. Nelson*
*Department of Statistical Science, †Department of Security and Crime Science
University College London, Gower Street, London WC1E 6BT
Email: alexander.gibberd.12@ucl.ac.uk, j.nelson@ucl.ac.uk
‡Department of Mathematics, Imperial College London, South Kensington Campus, London SW7 2AZ
Email: m.evangelou@imperial.ac.uk

***Abstract*—We investigate where and how key dependency structure between measures of network activity change throughout the course of daily activity. Our approach to data-mining is probabilistic in nature, we formulate the identification of dependency patterns as a regularised statistical estimation problem. The resulting model can be interpreted as a set of time-varying graphs and provides a useful visual interpretation of network activity. We believe this is the first application of dynamic graphical modelling to network traffic of this kind. Investigations are performed on 9 days of real-world network traffic across a subset of IP's. We demonstrate that dependency between features may change across time and discuss how these change at an intra and inter-day level. Such variation in feature dependency may have important consequences for the design and implementation of probabilistic intrusion detection systems.**

I. Introduction

Networked systems are increasingly being targeted by sophisticated cyber-criminals and other parties. Rather than cause immediate damage and expose themselves to defenders, attackers are increasingly choosing to infiltrate and remain active within a network for extended periods of time. These *Advanced Persistent Threats (APT)* are hard to detect due to the massive complexity and volume of activity within networks which can mask the subtle movements of an attacker [3]. A common strategy for infiltration is to initially target a single, or small subset of devices in a network. Once this objective is secured, an attack can be escalated, the intruder compromises further machines within the network, gaining further control over the network [11].

Traditional *Intrusion Detection Systems (IDS)* such as SNORT operate on a rule-based approach [12], they can react very quickly to detect known threats which correspond to previously modelled and coded patterns. Given the efficiency of such pattern matching, these methods can be applied at the packet level. Unfortunately, such hard-coded rules require frequent updating to keep up with current threats. Furthermore, given their high levels of specificity, rule-based methods are also increasingly bypassed using so-called polymorphic attacks. These attacks are capable of subtly transform their observed behaviour to deliver similarly malicious payloads or maliciously effect network/system operation. [2].

To counter over-fitting and rule specification issues, a popular research direction in network anomaly detection is to adopt so called machine-learning based approaches [12], [15], [7]. Generally speaking, these aim to model different classes of network activity (i.e. anomalous/normal) based on some algorithm which is trained on real network data. Due to the size and complexity of network datasets, finding patterns which can identify network state (either malicious or benign) constitutes a substantial data-mining problem.

Discriminative and Generative Models

Two main strands of machine-learning methodologies are employed in the literature:

- *Discriminative methods* - classify activity as normal/abnormal based on an explicit labeling of normality, i.e. one has access to some labeled data where the state of the network is known. Sometimes, this may be extended to consider specific types of anomaly, in a task which is known as anomaly identification [7].
- *Generative models* - aim to describe an underlying statistical distribution from which observed data may be generated from. Anomalous activity can then be defined with respect to the estimated distribution [12]. Many correlation based, for example Principle Component Analysis (PCA) methods may be seen in this light [13]. Previous works [9], [8], [16] have demonstrated that correlation, or second order statistics of network traffic can be used to identify denial of service attacks.

Let $\boldsymbol{x}^t$ be a set of p features derived from network traffic and y^t represent the state of the network at a time t. Roughly speaking, a discriminative model will try and predict y^t given data $\boldsymbol{x}^t$, using a model built (or trained) on data from the past $\boldsymbol{x}^{t'}, y^{t'}$ where $t' < t$. Statistically, such a procedure aims to model the conditional distribution $P(Y|X)$, i.e. once we observe the random variable X we should be able to say something about the network state Y. However, in order to train such a model, we need observations that relate to both the random variable X and the network state variable Y. In many situations, we simply don't know whether the network is in an anomalous state a-priori, i.e. we cannot measure Y.

Such a setting may relate to *zero-day* threats where knowledge of the attack vector is not available in advance of the attack.

In this case, a generative approach can be useful for defining anomalies. Rather than model the conditional distribution $P(Y|X)$, a generative model aims to describe the *joint distribution* of the network features $P(X_1, X_2, \ldots, X_p)$. In particular, the joint distribution is desirable, rather than a set of marginal models which describe $P(X_1), \ldots, P(X_p)$ and do not model inter-dependencies between features. Such joint modelling of features is the focus of this paper. However, defining and learning a joint distribution, as opposed to a conditional discriminative, or set of marginal models is difficult due to the inherent parametric complexity of such models.

Understanding Feature Dynamics

It is common practice to assess correlation patterns (which can relate to the specification of a joint distribution) across a whole data set, one does not typically take into account how these may change as a function of time. However, it is well known that network traffic can exhibit non-stationary behaviour [14], [12]; it is therefore possible that dependency patterns may change depending on which time/scale they are observed.

In this work, we aim to quantify whether any temporal variation in dependency exists on real network traffic. To achieve this, we utilise a novel dynamic graphical modelling framework. Representing dependencies between features as a set of time-varying graphs, we uncover important relations between features and quantify their variation over several days of real data. To the authors' knowledge, this is the first time such a methodology has been applied to NetFlow derived features. We limit our analysis here to the consideration of feature dynamics and do not build an explicit anomaly detection engine. However, it is our belief that understanding dependencies between features, and how these can vary over time is an important step to building fully generative anomaly detection systems.

In the next section we introduce our data-set and the subset of features used for our analysis. We also provide some details of the dynamic graph modelling tools used to extract and study dependency patterns. Section III introduces a set of experiments that investigate feature dynamics at both an intra and inter-day scale. We then conclude with a discussion of estimated dependency structures, what they mean in the context of network traffic, and how knowledge of the investigated patterns can be used in future work for anomaly detection.

II. Methodology and Data

A. The Dataset

We aim to assess the dependency structure within a set of features ($p = 14$) that relate to the number of connections, packets, and size of packets. These features are extracted from a subset of 10 IP addresses within the Imperial College network. The IP addresses (to be understood as devices) under study are kept constant throughout the study, where data was collected for 13 consecutive days (including 4 weekend days which we discount in our analysis).

The features we use are constructed from what is known as NetFlow data. This provides a popular measure of network activity and records some properties of sessions or connections between devices from inside and outside the network. Raw measurements include; the time and date that connections are established; the type of connection (or protocol, i.e. UDP, TCP); the Internet Protocol (IP) addresses of source and destination; volume; and number of packets transferred. These events occur in continuous time, however, to combine information from multiple NetFlow records we choose to bin the events into time intervals. We transform the raw NetFlow measurements into binned features as described in Table I.

Name	Description
No_Events	Number of events that start and end in bin
No_StartEvents	Number of events that start in bin but end outside
Bytes_Median	Median of packet size within bin
Bytes_MAD	Median absolute deviance (MAD) of packet size
Bytes_SUM	Total number of bytes in bin
Bytes_SD	Standard-deviation of bytes
Packets_Median	Median number of packets
Packets_MAD	MAD of packer distribution within bin
Packets_SUM	Number of packets within bin
Packets_SD	Standard-deviation of packet distribution in bin
BP_ratio_Median	Median of ratio between bytes and packet
BP_ratio_MAD	MAD of byte-packet ratio
BP_ratio_SUM	Sum of ratio within bin
BP_ratio_SD	Standard-deviation of byte-packet ratio

Table I: List of extracted NetFlow features used in this analysis. Features are split into four rough categories; those which count the volume of sessions (or individual NetFlow records), the size of the packets in each record, the number of packets in each record, and the ratio of bytes per packet. For further details on the construction of features see Evangelou *et al.* [1].

The act of binning the records simplifies analysis by reducing the raw volume of features to be processed. One advantage of this approach to data-mining is that it enables enhanced interpretation and robustness over correlating individual data-points. For example, the statistical distribution of binned features may provide a high-level characterisation of a network that can augment more specific (and sensitive) analysis at the individual packet/session level (for example via deterministic pattern matching [17]). Whilst we use $p = 14$ features in this work, there is plenty of scope to increase the number of features used. For example, a similar methodology may be employed from more fine-grained features derived via deep-packet inspection.

The distribution of the features will change dependening on the size of the bin used in analysis. For example, Fig. 1 demonstrates what one of these features (Number of events) looks like for different bin sizes, we plot this at 10 minutes and 5 minutes. Clearly, the distribution of features (within a bin) changes depending on the size of the bin. We note that many of the features we use are based on count data, as the bin-

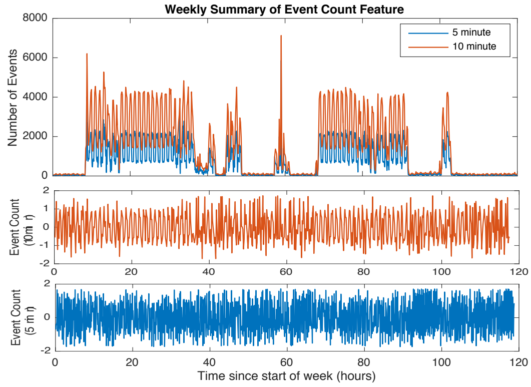

Figure 1: Top: Example traces of aggregated NetFlow packet counts (within 5 and 10 minute intervals) across 5 consecutive days (Mon-Fri). Bottom: The same traces as transformed by Eq. (1) with $h = 2$.

size tends to zero these feature distributions will demonstrate an excess of zero values [1], [14]. This is undesirable for our particular analysis as we wish to model the features as continuous random variables. To avoid such problems, we perform our analysis with a relatively large bin-size of 10 minutes, the limitations of this are discussed further in section IV.

Pre-processing

To enable stable estimation of dependency structure it is prudent to ensure that all features are measured on a similar *scale* level. To this end, we perform a localised *z-scoring* procedure which measures the empirical mean and standard-deviation of a feature flow within a window (of width $2h+1$). We perform a local de-trending and variance stabilising transform to each feature flow according to:

$$\tilde{X}_t = \frac{X_t - \hat{\mu}_t}{\hat{\sigma}_t} \;, \tag{1}$$

where $\hat{\mu}_t = (\sum_{i=t-h}^{t+h} X_i)/2h+1$ and $\hat{\sigma}_t = \big(\sum_{i=t-h}^{t+h}(X_i - \hat{\mu}_t)^2/2h\big)^{1/2}$. The result of such a transform can be seen in Fig. (1). The focus of this work is on understanding the contemporaneous relationship between features and how these change over time. The interest here is thus correlatory in nature, we desire to know how one feature i changes with (not necessarily in response) to another feature j. As we have rescaled the data, we can assume that the marginal variance of the data will be approximately 1 across time. Estimation of dependency structure should therefore no longer be sensitive to variable scaling (i.e. trends in the data).

B. Gaussian Graphical Models

An *undirected graphical model (UGM)* factorises a joint distribution $\mathcal{D}_p$ over a set of p variables by representing conditional dependencies as a graph structure $G(V,E)$. An example of a graphical model relating to network features can be seen in Fig. 2, where V indexes a set of nodes (relating to features) and E denotes edges associated with feature dependency. In our case, we assume our features are drawn from a *Gaussian graphical model (GGM)* whereby the distribution is Gaussian $\mathcal{D}_p = \mathcal{N}_p(\mathbf{0}, \mathbf{\Sigma}_0)$, with covariance matrix $\mathbf{\Sigma}_0$. An important property of GGM's, is that one may infer conditional independence between variables i,j by considering the relevant entry in the *precision matrix*, defined as $\Theta_{i,j} := (\Sigma)^{-1}_{i,j}$. In particular the following statements hold:

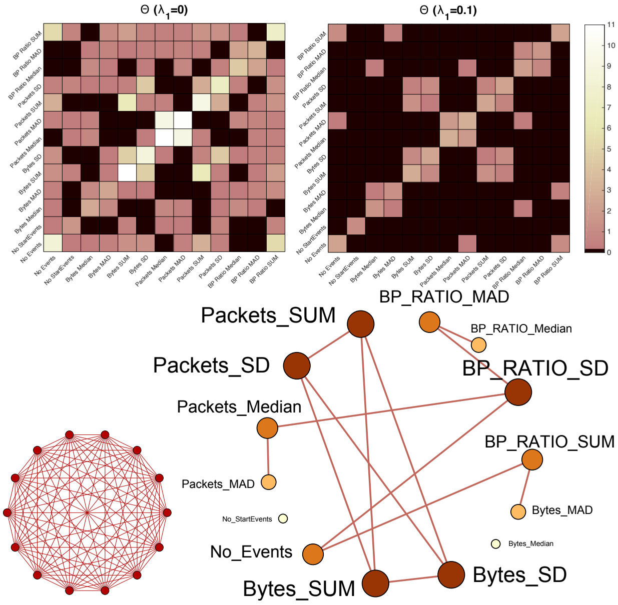

Figure 2: Top: Dense and sparse estimates of precision matrices ($\lambda_1 = 0$ and $\lambda_1 = 0.1$ respectively). Bottom: Corresponding graphical estimates for a static GGM. The size of the nodes in the estimated graph represent the relative degree (number of edges) associated with each node.

$$(j,k) \in E \iff X_j \perp X_k | X_{V\setminus\{j,k\}} \iff \Theta_{i,j} \neq 0 \;,$$

where $X_j \perp X_k | X_{V\setminus\{i,j\}}$ means the variable X_j is independent of X_k *conditioned* on the rest of the variables (see Lauritzen *et al.* [10] for more details). These are indexed by the complete set of vertices excluding the j,kth elements, i.e $V\setminus\{j,k\} := \{i = 1,\ldots,p \mid \forall i \neq j,k\}$. Estimating the precision matrix $\hat{\mathbf{\Theta}}$ is thus of great importance when inferring the edge set of a GGM. Naive estimation of the precision matrix, for example via Maximum-Likelihood will generally result in a completely dense precision matrix and graph $G(V,E)$, i.e. all possible edges will be contained within E (see left plot in Fig. 2). Such an estimate is undesirable for two reasons. Firstly, we do not gain any intuition about the dependencies between variables, and secondly, the estimator is likely to have high-variance due to the number of parameters $\mathcal{O}(p^2)$ required to be estimated. A popular solution to this problem is to enforce *sparsity* on the graph, whereby the number of edges estimated $s = |E|$ is small in comparison to the number of possible edges, i.e. $s < p^2/2 - p$. Such a model has less degrees of freedom than its dense counterpart and therefore provides a more robust estimate of the joint distribution. As seen in Fig. (2), a sparse graphical model also enables enhanced interpretation, in the GGM setting the estimated edges obtained from $\hat{\boldsymbol{\Theta}}$ can be used to identify conditional dependencies between variables. Such dependencies are important as they enable us to suggest where features may be related, either by construction, or as a result of different traffic processes operating on the network.

Sparsity can be enforced on a GGM in several ways, either by assuming an a-priori pattern of zeros (and correspondingly edges in the graph), or by placing a prior on the number of edges in the graph (or non-zeros in the precision matrix). A popular choice is to adopt a log-prior proportional to the size of the off-diagonal precision entries, namely $\log P(\boldsymbol{\Theta};\lambda_1) = -\lambda_1 \sum_{i\neq j} |\Theta_{i,j}|$, where in this case a larger value of λ_1 corresponds to assuming a sparser graph. Performing either Bayesian (we find the full distribution $P(\boldsymbol{\Theta}|\boldsymbol{X},\lambda)$) or MAP (we find the most likely precision matrix $\arg\max P(\boldsymbol{\Theta}|\boldsymbol{X},\lambda_1)$) inference we can then identify a sparse GGM [4].

C. Dynamic Graphical Models

Traditionally, GGM's are estimated in a static setting and the data is assumed to be drawn independently and identically. However, as discussed, we wish to examine the temporal variation in dependency patterns. In order to achieve this, we utilise what is known as a dynamic GGM [6]. Such an extension permits temporal variation in the covariance structure according to:

$$(X_1^t,\ldots,X_p^t)^\top \sim \mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_0^t)\ , \tag{2}$$

where $\boldsymbol{\Sigma}_0^t$ is a localised covariance matrix, for times $t = 1,\ldots,T$. As it stands the model in (2) is not identifiable from data as we are required to estimate $\mathcal{O}(Tp^2)$ parameters from only $\mathcal{O}(T)$ data points.

To aid in this identification and permit consistent estimation, we must assume that the model has certain smoothness properties, i.e. that the covariance can not vary too much in adjacent time-intervals. There are variety of options over what kind of smoothness we may wish to impose on such models [6]. However, for the purposes of this work we will assume that temporal variation will be restricted by a total-variation type constraint, namely $\sum_{t=2}^{T}|\boldsymbol{\Theta}_0^t - \boldsymbol{\Theta}_0^{t-1}| \leq \gamma_2$. This results in a piecewise constant precision matrix, where one can expect a given entry in the precision matrix $\Theta_{i,j}^t$ to change only at a few (sparse) points in time, i.e. for many time points $t, t+1 \in [1,T]$ we expect $\Theta_{i,j}^t = \Theta_{i,j}^{t+1}$. Such a smoothness prior is particularly useful for modelling bursty and discontinuous network traffic as it can permit sudden jumps in structure. In order to estimate such structure we work in the MAP estimation paradigm and minimise the negative penalised log-likelihood function: $\mathcal{L}(\{\boldsymbol{\Theta}^t\},\boldsymbol{Y}) :=$

$$\sum_{t=1}^{T}\big(-\log\det(\boldsymbol{\Theta}^t)+\mathrm{trace}((\boldsymbol{y}^t)^\top\boldsymbol{y}\boldsymbol{\Theta}^t)\big)+\ldots \tag{3}$$

$$\ldots\ldots+\lambda_1\sum_{t=1}^{T}\sum_{i\neq j}|\Theta_{i,j}^t|+\lambda_2\sum_{t=2}^{T}\sum_{i,j=1}^{P}|\Theta_{i,j}^t-\Theta_{i,j}^{t-1}|\ ,$$

where λ_2 is a tuning parameter that effects the smoothness of the estimated graphs. This parameter can be understood in a similar way to λ_1 which enforces sparsity, a larger λ_2 results in smoother estimation of a dynamic graph. A set of dynamic graph estimates can now be obtained by minimising the objective in (3), we refer to this procedure as the *Fused Graphical Lasso (FGL)*. Since this function is convex, it can be minimised reliably utilising a variety of convex optimisation methods. In this work we utilise an *Alternating Directed Method of Multipliers (ADMM)* algorithm which has computational complexity of order $\mathcal{O}(p^3 T\log(T))$. In the interests of space we refer the reader to Gibberd *et al.* [5] for details of this implementation, code is availiable on request.

III. Experiments

In this section we detail the estimation of dynamic graphical models based on 9 days of NetFlow features. We aim to assess the within day and between-day variation in feature dependency and visualise this via the estimated graphical models.

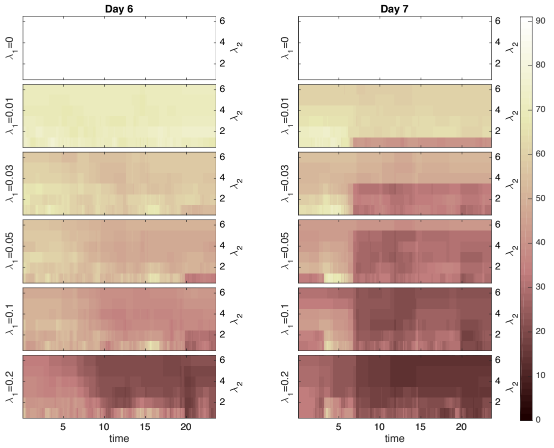

Figure 3: Solution paths depicting the number of edges estimated as a function of tuning parameters within two consecutive days.

A. Intra-day Dynamics

In this first experiment we aim to assess whether there is a common daily pattern to dependency dynamics. One might hypothesize such behaviour according to a day-night cycle, for example, activity during the day where users are working on a network may exhibit different patterns to those detected in the evening. To test such a hypothesis, we run FGL on each of the 9 working week-days within our dataset. For each day we find a *solution path* according to a grid of tuning parameters λ_1, λ_2. Each point in this path corresponds to a set of dynamic graphs with different sparsity and smoothness properties. Figure 3 gives a visualisation of such solution paths, comparing estimate edge structure across two days of data. As expected, one can see the clear sparsity inducing effect of λ_1, whereby large λ_1 have very few edges. Similarly, the smoothing effect of λ_2 can also be observed, larger values have visibly reduced variation in the number of edges estimated.

Cross-Validation

It is quite clear that the two days considered in Fig. (3) do not have very similar solution paths. However, we wish to know whether such difference between days is typical across the whole data-set, or just limited to the two days plotted. Hypothetically, if the data from across different days was generated according to the same process, then a model trained on one day should be able to describe some of the behaviour of another day.

We here formalise such descriptive ability using a measure of risk based on how well one *training* day can be explain the data corresponding to a held out *test* day. Exchanging the test data-set across the days in a leave-one-out cross-validation fashion, the risk can be used to describe how well the estimated models generalise between days. By minimising this risk surface one can estimate an optimal set of tuning parameters (λ_1, λ_2). To construct our risk function we adapt the idealised setting where we have knowledge of the ground-truth distribution. For a multivariate Gaussian distribution, the predictive risk for a pair of ground truth $\mathbf{\Sigma}_0$ and estimated $\hat{\boldsymbol{S}}$ covariance matrices is given as $R(\hat{\boldsymbol{S}}) = \mathrm{tr}((\hat{\boldsymbol{S}})^{-1}\mathbf{\Sigma}_0) + \log\det(\hat{\boldsymbol{S}})$. Zhou *et al.* [18] note that up to a constant $R(\hat{\boldsymbol{S}}) = -2E_0[\log(f_{\hat{\boldsymbol{S}}}(Z))]$, where $f_{\boldsymbol{S}}$ is the density for $\mathcal{N}(\mathbf{0}, \hat{\boldsymbol{S}})$ and the data is drawn under the ground truth structure $Z \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_0)$. The likelihood and the risk are thus related via the density function. In our case, we extend this measure of risk to cover the whole time-series. We define the leave one out cross-validation risk as: $R_{\mathrm{loo}}(\{\hat{\boldsymbol{S}}_t\}_{t=1}^{T}) :=$

$$\sum_{t=1}^{T}\left(\frac{1}{N}\sum_{i_{\mathrm{test}}=1}^{N}\sum_{i\neq \mathrm{i}_{\mathrm{test}}}\left[\mathrm{tr}(\hat{\mathbf{\Theta}}_t^{i}\boldsymbol{S}_t^{i_{\mathrm{test}}}) + \log\det\left((\hat{\mathbf{\Theta}}_t^{i})^{-1}\right)\right]\right), \tag{4}$$

where $\boldsymbol{S}_t^{i_{\mathrm{test}}} = \boldsymbol{y}_t\boldsymbol{y}_t^{\top}$ is an ill-conditioned estimate of the local empirical covariance. In effect, by averaging over the N different days of data, we can see how different (λ_1, λ_2) perform in terms of describing the data on other days.

The risk surface, as plotted in Fig. (4) tells us a lot about how the estimated dependency graphs generalise across days. In particular we note:

- The shrinkage inducing λ_1 appears to have a minima at around $\lambda_1 = 0.1$.
- The risk surface suggests that λ_2 should be set very large, there is no discernible minima with respect to λ_2.

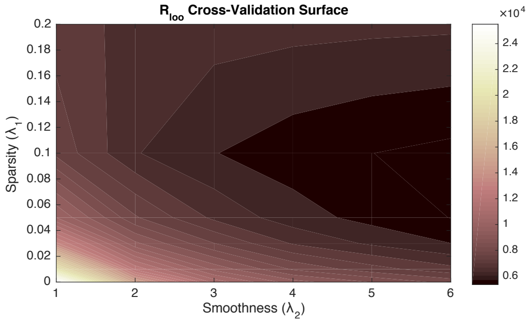

Figure 4: Leave-one-out risk surface, as averaged over $N = 9$ days of data.

The fact that the risk surface prefers a very large λ_2 suggest that almost constant precision matrices should be preferred, i.e. there will be no dynamic structure estimated. This should be interpreted as evidence against the hypothesis that there is a regular daily cycle of dependency patterns, i.e. there appears to be no consistent pattern to the dynamics across different days (at least in this data-set). A perhaps more interesting observation is the fact that *there is* a minima with respect to λ_1, whilst there are no regular temporal patterns, this does suggest that there is an optimal level of sparsity which generalises across different days.

B. Inter-day Dynamics

Whilst in the previous experiment, we didn't see any significant temporal patterns in the daily analysis, we might still be able to find some longer time-scale changes. In this experiment we concatenate 5 days worth of data together (Monday-Friday) in an effort to assess inter-day dynamics, we then run FGL on the whole dataset for a wide range of tuning parameters (λ_1, λ_2). For each setting of these parameters, we obtain a set of precision matrix estimates $\{\hat{\mathbf{\Theta}}^{t}(\lambda_1, \lambda_2)\}_{t=1}^{T}$ which encode a graphical model through the pattern of non-zero entries. In this mode of operation FGL is operating in a purely exploratory data-analysis role, we want to see what solutions for different λ_1 and λ_2 look like.

Figure 5 presents the output of this analysis, where we plot the number of edges estimated in the graph as a function of tuning parameters. There is clearly some temporal patterns contained within the solution path. In particular we note that there seems to be a slightly more dense region within the periods $t \in [10, 50]$ and $t \in [70, 100]$. It is interesting to note that these appear to coincide with periods of increased activity as measured by overall event count (see Fig. 1). Rather than a daily cycle, this pattern suggests that meaningful dynamics might be detected over a period of several days. Whilst this periodicity is very clear in the raw data, it is not necessarily obvious that this would propagate through to changes in feature dependency structure.

To investigate this structure further, we can harness the cross-validation analysis performed in the intra-day experiment to select an appropriate level of sparsity and set $\lambda_1 = 0.1$.

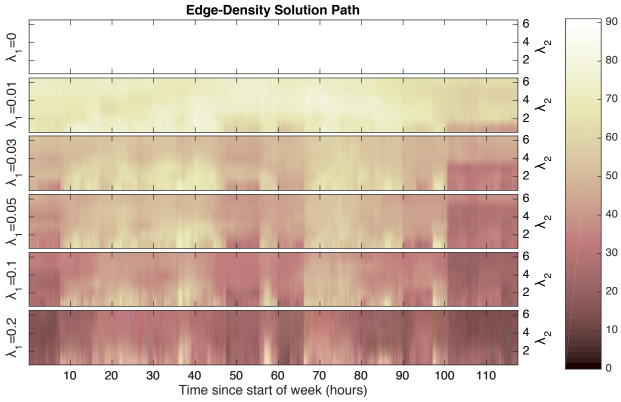

Figure 5: Edge density solution path for 5 consecutive days of data.

In Fig. 6 we plot the number of edges as a function of time at two specific smoothness parameters, $\lambda_2 = 1$ and $\lambda_2 = 6$. In addition to this we can visualise the graph directly, plotting snapshots of estimated graphs (for $\lambda_1 = 0.1, \lambda_2 = 6$).

IV. Discussion

In this section we discuss the interpretation of some of the dependency patterns in our analysis and present some future directions for research.

- *Dependency between features can and does change over time* - The analysis depicted in Figs. 5,6 clearly shows that dependency structure (i.e. edges in the graph) change as a function of time. In this particular data-set there are no obvious daily trends in this graph dependency. Instead, we find two distinctly similar regimes of activity (over a period of several days) where the graph estimates are denser than in the surrounding time-periods. We suggest that these may be used to describe different processes which are running across the network at these times.
- *Some dependency structures are persistent across time* - Whilst some dependencies change over time, our analysis suggests that a sub-set of these are persistent (or re-occurring) over time. In particular, the subset of features (Packets_SD, Packets_SUM, Bytes_SUM and Bytes_SD) appear to be highly inter-dependent. This is particularly obvious during periods of high-activity (see graphs in Fig. 6 at $t = 20, 40, 75, 90$). Alternatively, we can also detect variables which are relatively independent of the other variables, for example, the number of start events (No_StartEvents) appears to be relatively disconnected throughout the whole analysis window.

The characterisation of dependency patterns may enable future probabilistic IDS systems to focus their modelling power on only known dependencies. In theory this should lead to improved robustness (and hence reduced false positives) as the model complexity can be reduced without affecting the ability of the generative distribution to describe normal patterns of activity.

Limitations and Future Directions

In this work we assume feature flows are distributed as a continuous Gaussian random variable, however, this assumption is unlikely to be met in practice. The requirement of Gaussianity limits the time resolution of our analysis, specifically it requires us to have a large bin size (10 minutes) so that features can be treated as continuous random variables. In future work, one may consider using a different likelihood function, or non-parametric models which allow more flexibility with respect to feature distribution. This will potentially lead to a characterisation of network traffic at shorter time-intervals.

Larger studies which include more IP addresses or devices are planned and will investigate the reproducibility of the results obtained here. These will verify the persistence of extracted dependency structure between variables, and assess how this depends on the aggregation level (i.e. bin interval). To this extent, understanding dependency between features at different aggregation levels may be important for characterising network activity, such an analysis might be considered a multivariate extension of the work by Scherrer *et al.* [14].

Finally, it may be interesting to consider whether dependency arises between features due to construction (i.e. how we define the features), or as a product of actual varying network activity. Such a study would need access to information about the processes running on each device and how these produce network activity.

Conclusion

The primary contribution of this work is to demonstrate that dynamic graphical models can be applied to the modelling of network NetFlow data. In applying such methods we can gain some insight on feature dependency within network traffic. It is important to place these results in the wider cyber-security context. We emphasise here that we do note validate using such models to detect actual attacks, such validation is left as future work.

Our results demonstrate that network activity can be described by a sparse graphical model. This has potential implications for probabilistic IDS as it suggests that a statistical model can focus on a subset of dependencies between features. Such models should be more robust and therefore have capability to reduce false alarm rates. A key point that we demonstrate is that whilst dependency structure is sparse, the structure of this model may change over time. Such changes are not necessarily due to malicious activity, but may in future developments provide a means to automatically detect changes in usage patterns. For example, if one detects dependency patterns not seen in a large volume of historical data this may indicate a new, unseen, and possibly malicious process operating on the network. It is envisaged that high-level statistical characterisations of network traffic can therefore augment traditional rule-based IDS systems and provide context to decision making and alert generation.

In summary, it is our firm belief that understanding and characterising the dependency of network activity features, across different time-periods, and scales, is crucial for building

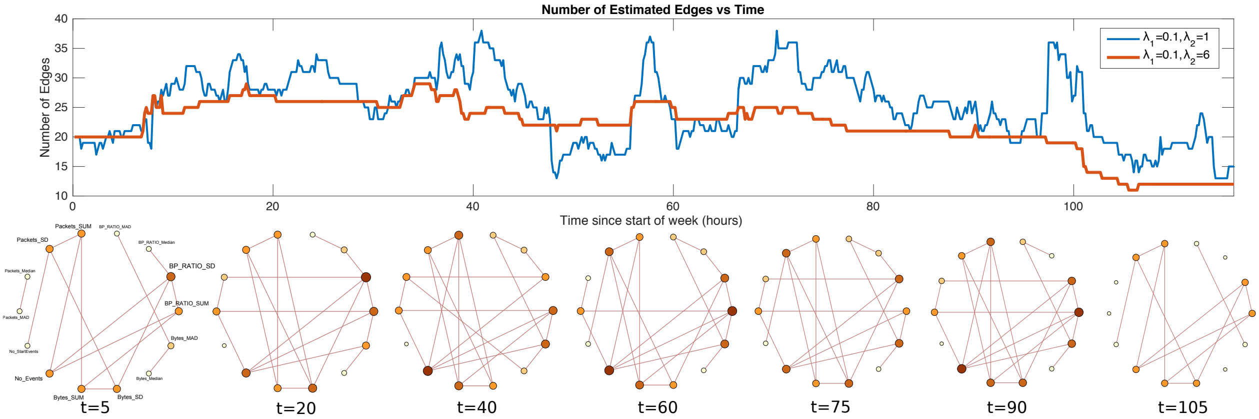

Figure 6: Top: Choosing $\lambda_1 = 0.1$ the number of edges are plotted as a function of time for two solutions, one with high smoothness $\lambda_2 = 6$ and one with low smoothness $\lambda_2 = 1$. Bottom: Some snapshots of graph structure are given at different points in the week (measured in hours). These graphs correspond to the solution with $\lambda_1 = 0.1, \lambda_2 = 6$.

effective generative anomaly detection systems. In this work we have shown on real data that such dependencies can and do vary with respect to time. As such, this serves to highlight the importance of considering dynamics in statistical models of network traffic.

Acknowledgements

We are grateful to Imperial College's ICT department and Andy Thomas for facilitating access to the NetFlow data used in this work. Alex Gibberd acknowledges funding from the Defence Science Technology Laboratory (Dstl) National PhD Scheme.

References

[1] M. Evangelou and N. M. Adams. Predictability of NetFlow data. *IEEE International Conference on Intelligence and Security Informatics*, 2016.

[2] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee. Polymorphic Blending Attacks. *15th USENIX Security Symposium*, pages 241–256, 2006.

[3] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler. Combating advanced persistent threats: From network event correlation to incident detection. *Computers and Security*, 48:35–57, 2015.

[4] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–41, 2008.

[5] A. J. Gibberd and J. D. B. Nelson. Regularized Estimation of Piecewise Constant Gaussian Graphical Models : The Group-Fused Graphical Lasso. *http://arxiv.org/abs/1512.06171*, pages 1–32, 2015.

[6] A. J. Gibberd and J. D. B. Nelson. Estimating Dynamic Graphical Models from Multivariate Time-series data : Recent Methods and Results. *Lecture Notes in Artificial Intelligence (in press)*, 2016.

[7] F. Iglesias and T. Zseby. Analysis of network traffic features for anomaly detection. *Machine Learning*, 101(1-3):59–84, 2014.

[8] S. Jin and D. S Yeung. A Covariance Analysis Model for DDoS Attack Detection. *IEEE International Communication Conference (ICC04)*, 00(c), 2004.

[9] S. Jin, D. S. Yeung, and X. Wang. Network intrusion detection in covariance feature space. *Pattern Recognition*, 40(8):2185–2197, 2007.

[10] S. L. Lauritzen. *Graphical Models*. Oxford, 1996.

[11] J. Neil, C. Hash, A. Brugh, M. Fisk, and C. B. Storlie. Scan Statistics for the Online Detection of Locally Anomalous Subgraphs. *Technometrics*, 55(4):403–414, 2013.

[12] A. Patcha and J. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, Aug. 2007.

[13] H. Ringberg, A. Soule, J. Rexford, and C. Diot. Sensitivity of PCA for traffic anomaly detection. *Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems - SIGMETRICS '07*, page 109, 2007.

[14] A. Scherrer and N. Larrieu. Non-gaussian and long memory statistical characterizations for internet traffic with anomalies. *IEEE Transactions on Dependable and Secure Computing*, 4(1):56–70, 2007.

[15] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu. A System for Denial-of-Service Attack Detection Based on Multivariate Correlation Analysis. *IEEE Transactions on Parallel and Distributed Systems*, 25(2):447–456, Feb. 2014.

[16] M. Thottan. Anomaly detection in IP networks. *IEEE Transactions on Signal Processing*, 51(8):2191–2204, 2003.

[17] G. C. Tjhai and M. Papadaki. The problem of false alarms: Evaluation with snort and DARPA 1999 dataset. *Trust, Privacy and Security in Digital Business, Lecture Notes in Comuputer Science*, pages 139–150, 2008.

[18] S. Zhou, J. Lafferty, and L. Wasserman. Time varying undirected graphs. *Machine Learning*, 80(2-3):295–319, Apr. 2010.