Spatio-temporal Mining of Software Adoption & Penetration

Evangelos E. Papalexakis[*], Tudor Dumitras[†], Duen Horng (Polo) Chau[‡], B. Aditya Prakash[§], Christos Faloutsos[*]
[*]Carnegie Mellon University, [†]Symantec Research Labs, [‡] Georgia Tech, [§] Virginia Tech

E-mail: epapalex@cs.cmu.edu, Tudor_Dumitras@symantec.com, polo@gatech.edu, badityap@cs.vt.edu, christos@cs.cmu.edu

***Abstract*—How does malware propagate? Does it form spikes over time? Does it resemble the propagation pattern of benign files, such as software patches? Does it spread uniformly over countries? How long does it take for a URL that distributes malware to be detected and shut down?**

In this work, we answer these questions by analyzing patterns from 22 million malicious (and benign) files, found on 1.6 million hosts worldwide during the month of June 2011. We conduct this study using the WINE database available at Symantec Research Labs. Additionally, we explore the research questions raised by sampling on such large databases of executables; the importance of studying the implications of sampling is twofold: First, sampling is a means of reducing the size of the database hence making it more accessible to researchers; second, because every such data collection can be perceived as a sample of the real world.

Finally, we discover the SHARKFIN temporal propagation pattern of executable files, the GEOSPLIT pattern in the geographical spread of machines that report executables to Symantec's servers, the Periodic Power Law (PPL) distribution of the life-time of URLs, and we show how to efficiently extrapolate crucial properties of the data from a small sample. To the best of our knowledge, our work represents the largest study of propagation patterns of executables.

***Index Terms*—Malware Propagation, Internet Security, Data Analysis**

I. INTRODUCTION

What are the main properties of malware propagation? How does it go about infecting new machines on the Internet? Does its temporal propagation pattern resemble that of *legitimate* files, such as software patches? How long does it take for a malicious URL that distributes malware to be spotted and shut down?

On a similar pace, for the hosts where we can collect telemetry on software adoption and propagation, how are they distributed in a global scale? Are they distributed uniformly across all countries or do they adhere to a different geographical spreading pattern?

To answer such questions, security researchers and analysts need comprehensive, field-gathered data that highlights the current trends in the cyber threat landscape. Understanding whether a data set used for research is representative of real-world problems is critical, because the security community is engaged in an arms race with the cyber criminals, who adapt quickly to the defenses introduced, creating increasingly specialized cyber attacks [5], [28]. For example, in 2011, security analysts have identified 403 million new variants of malware and 55,294 new malicious web domains [28].

One resource available to the research community for studying security problems at scale is the Worldwide Intelligence Network Environment (WINE), developed at Symantec Research Labs [23]. WINE includes field data collected by Symantec on millions of hosts worldwide, and it provides a platform for data intensive experiments in cyber security. The WINE data sets are updated continuously with data collected on real hosts that are targeted by cyber attacks, rather than honeypots or machines in artificial lab environments. For example, the *binary reputation* data set includes information on binary executables downloaded by users who opt in for Symantec's reputation-based security program (which assigns a reputation score to binaries that are not known to be either benign or malicious).

However, the researchers who use WINE must understand the properties of the data, to assess the selection bias for their experiment and to draw meaningful conclusions. For example, when analyzing the patterns of malware propagation around the world, researchers would want to know that the distribution of executable files over machines follows a power law (see Figure 1); many files are reported by few machines, and few files by many machines. Additionally, the WINE data covers a sampled subset of hosts running Symantec products; we must understand the effects that this sampling technique may have on the experimental results. This challenge is not limited to WINE: every corpus of field data is likely to cover only a subset of the hosts connected to the Internet, and we must understand how to extrapolate the results, given the characteristics of the data sets analyzed.

The first contribution of this paper is a list of 3 of the many questions that are of interest to security researchers:

- **Q1**: What is the temporal propagation pattern of executable files?
- **Q2:** Where are files downloaded on the Internet?
- **Q3**: What is the typical URL lifetime?

The remaining contributions form two thrusts: the first is modeling of the data, so that we can answer the above questions, and the second is how to extrapolate from samples (since, inevitably, nobody has the full picture - only a sample of activities).

- **Modeling:** We propose three new models, one for each of the motivating questions
  - SHARKFIN: It describes the temporal propagation pattern of high volume executables: exponential growth, followed by power-law tail, with periodicities.
  - GEOSPLIT: It captures the geographical (spatial) spread of machines that submit executables to the WINE database.
  - PPL: The distribution of the lifetime of software disseminating URLs, follows our "Periodic Power Law" (PPL), with slope -1.
- **Extrapolations:** Given a sample, we show how to exploit our above models, to guess measures of interest (like life-time, geographical footprint etc) of the full, unknown, dataset. Our specific contributions are:
  - Extrapolation of the propagation pattern of a file, given its sample.
  - Estimation of the footprint loss due to sampling, on the geographical distribution of machines that report executable files (malware and legitimate) to Symantec.

The rest of the paper is organized in the typical way: Description of the data, proposed models, extrapolations from a sample, discussion, related work, and conclusions.

II. DATA DESCRIPTION

We conduct our study using the Worldwide Intelligence Network Environment (WINE), a platform for data intensive experiments in cyber security [23]. WINE was developed at Symantec Research Labs for sharing comprehensive field data with the research community. WINE samples and aggregates multiple terabyte-size data sets, which Symantec uses in its day-to-day operations, with the aim of supporting open-ended experiments at scale.

Starting from the raw data available in WINE, we define a reference data set with the following pieces of information:

- File occurrence counts spanning a whole month (June 2011), both for legitimate files and malware. This piece of the dataset essentially consists of time series that capture the propagation patterns of both types of files. This dataset consists of the following attributes:
  `(File SHA2 ID, Occurrences, Timestamp)`
- Counts of personal computers where telemetry is collected, for each country, spanning June 2011. This piece of data is both in aggregate form and in a daily basis. The attributes of this dataset are:
  `(Country ID, count, Timestamp)`
- The lifetime of malicious URLs as crawled by humans using these personal computers during June 2011. This dataset consists of records of the form:
  `(URL, First-seen Timestamp, Last-seen Timestamp)`

For each one of the aforementioned datasets, we possess both *before* and *after* sampling versions. As noted before, however, even the *before* sampling parts of the dataset may be viewed as a sample of the real world, since the hosts that use Symantec software are a subset (or a sample) of all the machines that exist in the Internet.

Details on the WINE database and how sampling is done

The data included in WINE is collected on a representative subset of the hosts running Symantec products, such as the Norton Antivirus. These hosts do not represent honeypots or machines in an artificial lab environment; they are real computers, in active use around the world, that are targeted by cyber attacks. WINE also enables the reproduction of prior experimental results, by archiving the reference data sets that researchers use and by recording information on the data collection process and on the experimental procedures employed.

The WINE database is updated continuously with data feeds used in production by Symantec, and the data is sampled on-the-fly as the files are loaded on the database. Each record includes an anonymous identifier for the host where the data was collected. The WINE sampling scheme selects all the records that include a pre-determined sequence of bits at a pre-determined position in the host identifier, and discards all the other records. In consequence, WINE includes either all the events recorded on a host or no data from that host at all. Because the host identifier is computed using a cryptographic hash, the distribution of its bits is uniform, regardless of the distribution of the input data. This sampling strategy was chosen because it accommodates an intuitive interpretation of the sampled subset: the WINE data represents a slice of the Internet, just like the original data set is a (bigger) slice of the Internet.

In this paper, we focus on the *binary reputation* data set in WINE. This data set records all the binary executables—whether benign or malicious—that have been downloaded on end-hosts around the world. This information is submitted by the users who opt in for Symantec's reputation-based security program (which assigns a reputation score to binaries that are not known to be either benign or malicious). The binary reputation data has been collected since February 2008. In addition to the host identifier, each report includes geolocation information for the host, the download time, the hash (MD5 and SHA2) of the binary, and the URL from which it was downloaded. These files may include malicious binaries that were not detected at the time of their download because the threat was unknown. To study the effects of sampling, we compare the sampled data in WINE with the original data set for the month of June 2011.

III. Patterns, observations and analysis

In this section, we pose three different questions that are of particular interest to companies involved in Internet security, such as Symantec. The spirit of the questions posed is exploratory and mainly pertains to the spatio-temporal properties of legitimate and malicious pieces of software.

Even though we mentioned that due to the overwhelming volume of the original data, the goal of the WINE project is to provide researchers with a representative sample of the data, in this section, we do not delve deep into issues such as extrapolation from the sample. Instead, we follow a qualitative approach in order to describe these spatio-temporal attributes of such files, and in the process of accomplishing that, surprising patterns and observations present themselves, all of which are described in detail in the next few lines.

A note on notation

N is the number of machines that submit executables to Symantec. T is the number of time-ticks in file occurrence time-series. $X(n)$ is the file occurrence time-series ($n = 1 \cdots T$). $\Delta I(n)$ is the number of "Infected" hosts at time n. $U(n)$ is the number of Un-infected hosts at time n. $S(n)$ is the external shock/first appearance of an executable (see Appendix). SharkFin is the model that fits the temporal propagation of high volume executables (see Appendix). GeoSplit is the distribution that describes the geographic spread of hosts that submit executables. Ppl stands for Periodic Power Law distribution.

A. Q1: What is the temporal propagation pattern of executable files?

A worm that propagates through buffer-overflow exploits (e.g., the Blaster worm from 2003) will exhibit a propagation rate different from another malware that spread through drive-by-downloads. Additional patterns of the time series that describes the evolution of the number of infections provide further clues regarding the behavior of the malware; for example, a surge of infections hours after Microsoft's *Patch Tuesday*[1] may point to the use of automated techniques for reverse-engineering security patches into working exploits.

Our proposed analysis and modelling, with respect to the temporal propagation pattern, works for high volume files, i.e. files that have enough samples of occurrences such that any form of (meaningful) modelling is feasible. As "high volume" files we consider all files with more than 1000 occurrences in distinct machines. In Figure 1 we show that the file popularity (and hence its volume) follows a power law.

In Figure 2, we illustrate the propagation pattern of six high volume files coming from several, major software vendors. For instance, these files can be either patches of already existing software, or new software binaries; such files (e.g. security patches) tend to become highly popular very early in their lifetime. In fact, in Figure 2 we observe, for all those popular files, a steep *exponential rise* which follows shortly after they initially appear on the Internet.

This exponential rise is followed by, what appears to be a *power-law drop*. Intuitively, this observation makes sense: A few days after a new security patch by a major software vendor appears, nearly all

[1] Each month's second Tuesday, on which Microsoft releases security patches.

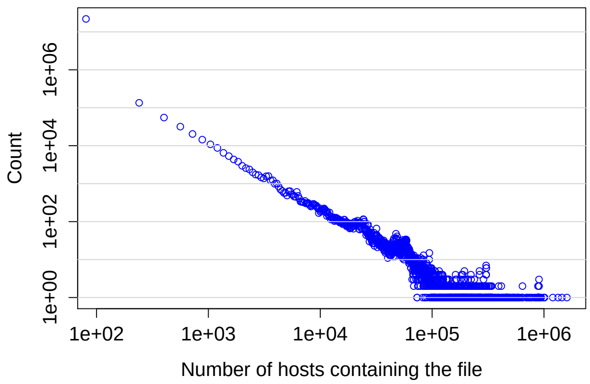


Fig. 1. Distribution of file popularity among machines. We observe that the popularity of a file, which also reflects its volume on the database, follows a power law.

users download it right away and only a few people tend to download it a couple of days after its release date; moreover, nearly nobody downloads the file one or two weeks after it has been released. We henceforth refer to this pattern as the SHARKFIN pattern, due to the resemblance of the spike to an actual shark fin.

Moreover, Figure 2 also captures a *daily periodicity* in the files' propagation pattern. An intuitive explanation for this periodic behaviour may be that a large number of these files are security patches, which are very often downloaded automatically; this would explain the relative increase of occurrences in a periodic manner, since the auto-update software usually runs the update at a standard time.

In order to model the propagation of high volume files, such as the ones shown in Figure 2, we take into account 1) the exponential rise and, 2) the power-law drop.

Recently, a model was proposed in [19] that is able to capture both the exponential rise and the power law drop, as well as periodicity in the data. This work was focused on meme propagation; however, it turns out that the SHARKFIN pattern bears a striking resemblance to the propagation pattern of memes that go viral on the Internet. Based on that observation, we leverage the work that focuses on meme propagation [19], and redirect its modelling power for the purposes of the task at hand.

A simplified version of our model is the following

$$\Delta I(n+1) = \left(U(n) \sum_{t=n_b}^{n} \left(\Delta I(t) + S(t) \right) f(n+1-t) \right)$$

where $\Delta I(n)$ is the file occurrences in time-tick n (i.e. the number of *I*nfected hosts), $U(n)$ is the number of machines that have not downloaded the file yet, at time-tick n, $S(t)$ is an "external shock" function that is zero for all n except for the first time that the file appears (which is denoted by n_b), and f emulates the power-law drop. If we denote as $X(n), n = 1 \cdots T$ the original data, then we essentially need to minimize: $\min_{\boldsymbol{\theta}} \sum_{n=1}^{T} \left(X(n) - \Delta I(n) \right)^2$ where $\boldsymbol{\theta}$ is the vector of the model's parameters. For a more detailed description of the SHARKFIN model, we refer the reader to the Appendix.

Figure 3 shows the result of our modelling; the proposed model almost perfectly captures both rise and fall patterns in the temporal

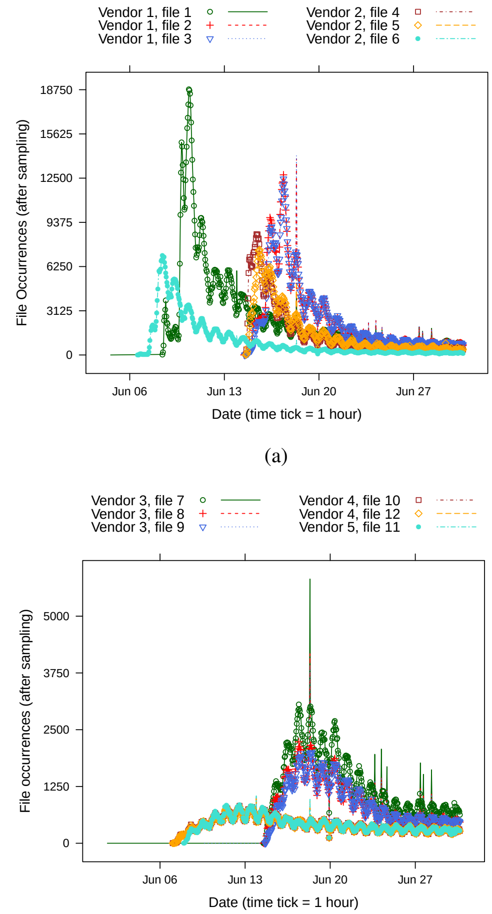


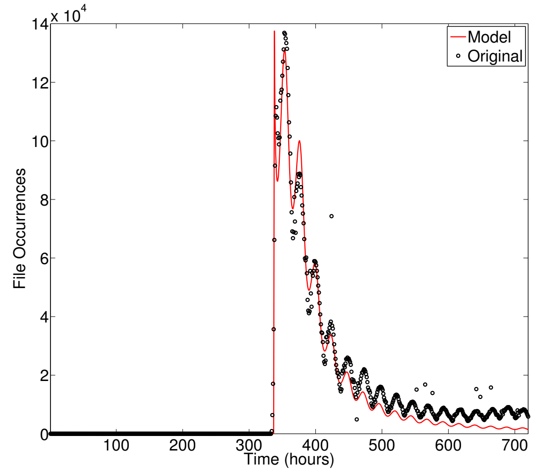
Fig. 2. Propagation of high volume files, before and after sampling (the symbol markers correspond to the sampled data, while the lines correspond to the original data scaled down by the sampling rate). These files all follow the SHARKFIN pattern that we describe on the main body of the text: A spike that grows exponentially and drops as a power law.

evolution of a high volume file's propagation. Both the exponential rise and the power law drop have been expressed through the SHARKFIN model, as well as the daily periodicity which we observed in the propagation pattern. There are a few outliers which do not follow the SHARKFIN spike, however, the vast majority of the file occurrences are aligned with the model.

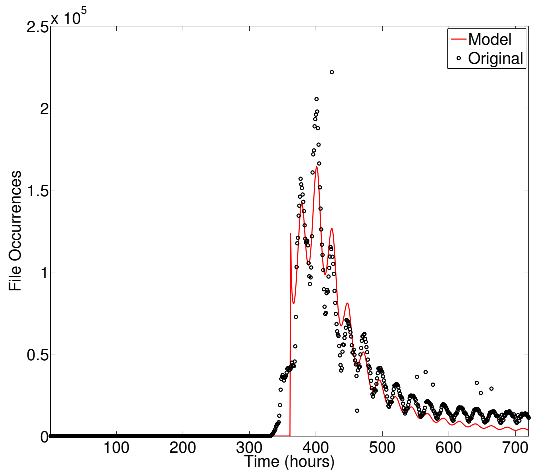
In addition to visual evaluation, we measured the relative squared error (RSE) between the original file time series X and the one produced by SHARKFIN, which we call $\hat{X}$; RSE is defined as $\frac{\|X-\hat{X}\|_2^2}{\|X\|_2^2}$. The median RSE for all the files that we tested was 0.071; the mean RSE was 0.244 ± 0.3617 and it is considerably higher due to a few files having very small number of occurrences, and thus being modelled poorly (which, in turn, causes the high deviation). However, for the majority of files, SHARKFIN performs very well, as it captures vital characteristics of the data.

B. *Q2: Where are files downloaded on the Internet?*

Understanding the geographical distribution of cyber attacks allows analysts to determine whether the malware tries to spread indiscriminately or it targets specific organizations. Similarly, understanding the geographical reach of update-dissemination infrastructures (e.g.,



(a)



(b)

Fig. 3. This Figure illustrates our modeling of the propagation pattern, compared to the actual file occurrence data, before sampling, for two high volume executables. Our SHARKFIN model seems to fit the data quite accurately. The median relative squared error (cf. Sec. III-A) for all the files that we tested was 0.071

Microsoft Update, Google Software Update) allows software vendors to optimize the delivery of critical security patches to their users. To answer both these questions using WINE, we must be able to reconstruct the histogram of host counts for different countries and ISPs from the sampled data.

We leverage data that record the number of hosts, covered in our WINE data set, where legitimate or malicious executables have been downloaded in June 2011, per country. Due to the sensitive nature of the data, we anonymize each country and we present only its id, which is merely determined by its ranking with respect to the host count. The total number of countries in the database is 229.

How are the WINE hosts distributed geographically? In Figure 4, we demonstrate the machine count per country as a function of a country's rank; we merely sort the counts in descending order and we assign a rank to each country according to that order. The figure shows the distribution both before and after sampling; there is an obvious displacement of the "sampled" line, which is to be expected.

In terms of the actual distribution that the hosts follow, we claim that the GEOSPLIT distribution fits the real data very well. In short, the GEOSPLIT distribution can be seen as a generalization of the 80-20 distribution, where additional freedom is given for the choice of the probabilities, i.e. p (which in the case of the 80-20 distribution is equal to 0.8) is now a parameter.

The model is closely related to the so-called "multifractals" [10]: Let N be the total count of machines that carry the executable, and assume that the count of countries is 2^k (using zero-padding, if necessary). Thus, we can do k levels of bisections of the set of countries; at each bisection, we give p fraction of the machines to the "rich" half, and the rest 1-p to the "poor" half. After k levels of bisections, we have 2^k pieces/countries; the "richest' has p^k fraction of the machines; the next k richest all have $p^{k-1}(1-p)$, and so on. Thus we construct a GEOSPLIT distribution, which fits very well the geographical spread of machines that submit files to the WINE database.

As we sample the dataset in a per machine basis, it is often the case that a few countries with very low volume will eventually disappear from the sample. In other words, if one is observing the sampled distribution, there is a part of the geographical footprint that is effectively lost. In the next section, we shall elaborate further on this footprint loss and will provide an efficient way to obtain an estimate of how many countries are ignored in the sample.

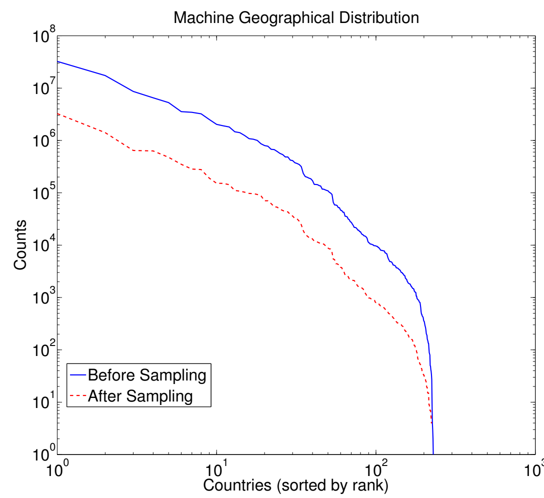


Fig. 4. Distribution of machines per country, in log-log scale. We have anonymized the countries, for privacy purposes, with the total number of countries in WINE being 229. The distribution follows the GEOSPLIT model, as we describe it in Section III of the text and the appendix. We observe that both the sampled and the original data follow the same distribution, with obvious displacements due to sampling.

C. *Q3: What is the typical URL lifetime?*

Malware-spreading sites often move, to conceal their presence, in ways that are not fully understood [6]. Therefore, estimating the time elapsed between the first and last file downloads from a malicious URL, as well as the next URL employed to spread the malware, allows analysts to characterize the attack. The WINE data set provides a large body of information on URL activity, collected using a distributed, but non-automated, crawler: the human users who download executable files around the world. However, the sampling strategy might distort these measurements by shortening the observed lifetimes and by omitting some URLs from the chain.

In Figure 5, we show the empirical distribution of the WINE URL lifetimes, as recorded for an entire month. We observe that sampling does not significantly alter the shape of the distribution; in fact, both before and after sampling distributions, for their most part seem to be aligned. The only part in which they deviate is the tail of the distribution, where the distribution obtained after sampling has generally fewer URLs for a given duration, which is not surprising, since sampling by default should cause this phenomenon.

However, both before and after sampling distributions (excluding outlying data points due to horizon effect, which is explained in the next few lines) *follow a power-law with slope -1*.

Additionally, we observe two rather interesting phenomena:

1) *Periodicity:* In both distributions, a periodic behavior is pronounced. This daily periodicity, however, is not an inherent property of the data, but rather a by-product of the data collection process. As we mentioned earlier, the URLs first and last appearances are crawled by human users, who manually download executable files around the world. Therefore, the periodic fluctuations of the URL lifetimes in Figure 5 are caused by the periodic pattern that the human crawlers operate on.
2) *Horizon Effect:* Since we are operating on a fixed time window of one month, it is very likely that we don't see the actual last appearance of some URLs which continue being on-line even after the end of the window we focus on. Because of that fact, the tail of both distributions (before and after sampling) contains many outliers; more specifically, it contains URLs that, in reality, have longer durations than the ones we observe. Furthermore, the horizon effect is even more pronounced on the distribution after sampling.

Putting everything together, we may characterize the distribution of the lifetime of URLs as a Periodic Power Law or PPL for short, with slope -1. It is important to note that both the periodicity and the slope is retained after sampling, excluding of course the horizon effect outliers.

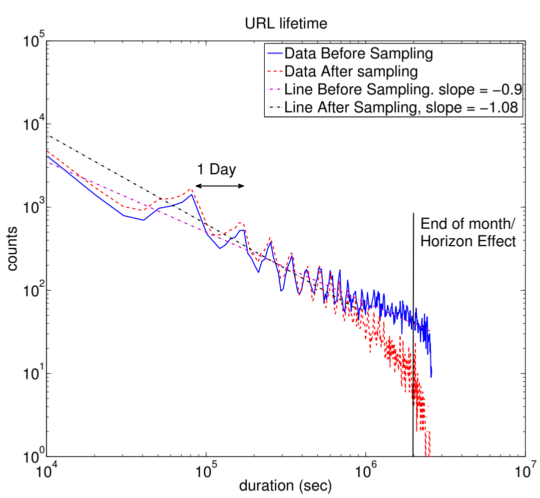


Fig. 5. Distribution of URL lifetimes, before and after sampling, in log-log scale. The lifetime of URLs follows the PPL model, which is a periodic power law distribution with slope -1 and daily periodicity. We also show the end of June 2011, which signifies the end of our measurement period and thus the start of the horizon effect.

IV. SEEING THROUGH THE SAMPLE

In the previous section we were concerned with both the original WINE database and a small sample thereof, but merely from an observatory perspective. In this section, however, we attempt to dive deeper into the implications of using a (representative) sample of the WINE database in lieu of the original, enormous dataset. In particular, we provide means to estimate/extrapolate crucial (from the security industry and research point of view) attributes of the data, based only on a sample. For instance, it is important for someone who works on a sample to be able to reconstruct the original propagation pattern of a file, given that sample. In the following lines, we pose such questions pertaining to the extrapolation from the sample and provide effective algorithms in order to "see through the sample".

A. *SQ1: Given a sample, can we extrapolate the propagation pattern of a file?*

Suppose we are given a sampled subset of the occurrences of a file, each accompanied with a time-stamp, as in **Q1** in the previous section. The sampling procedure is the same as before. How can we reconstruct the original, before sampling, propagation pattern of that particular file? Does the reconstruction resemble the original pattern? What are the inherent limitations imposed by sampling?

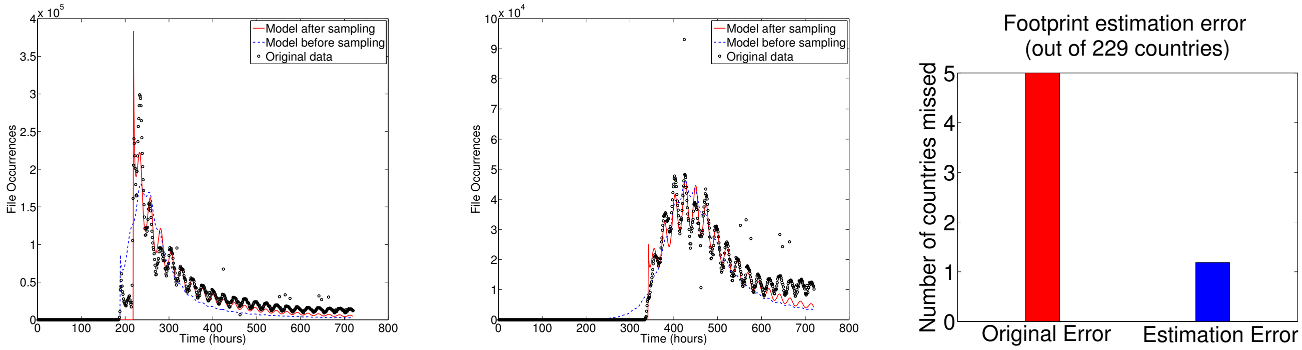
As we investigated in **Q1** of the previous section, we can successfully model the propagation pattern of legitimate files before sampling, as in Figure 3. In Figure 2 we observe that sampling does not severely alter the SHARKFIN shape of the time-series, at least for such popular, high volume files; the sampled time series seems to have consistently lower values than the before sampling ones, which is to be expected due to sampling (even though our sampling is per machine and not per file occurrence).

The main idea of our extrapolation technique lies exactly in the observation above. Since sampling has displaced the sampled time-series by a, roughly, constant amount, we follow these two simple steps:

1) Multiply every point of the sampled time series by the sampling factor, in order to displace it to, roughly, the same height as the original time-series.
2) Fit the model that we introduce in **Q1** on the multiplied time-series.

More formally, following the same notation as in **Q1**, and denoting the sampling rate by s, we need to minimize the following function:
$$\min_{\boldsymbol{\theta}} \sum_{n=1}^{T} \left(sX(n) - \Delta I(n)\right)^2$$
In Figures 6(b& c), we show the result of the proposed approach, for two popular, high volume files, by major software vendors. We can see that our extrapolation is perfectly aligned with the model of the data *before* sampling, which renders our simple scheme successful. On top of that, both models, as we also demonstrated on Figure 3 fit the original data very well.

As in modelling, here we employ RSE in order to further assess the quality of our extrapolation (by measuring the RSE between the original sampled vector of observations, and the extrapolated one). The median RSE was 0.0741; the mean RSE was 0.2377 ± 0.3648 (for the same reasons we mentioned in Sec. III). We see that for the majority of files, the extrapolation quality is very high, demonstrating that our extrapolation scheme, usingSHARKFIN, is successful for high density files.



(a) (b) (c)

Fig. 6. **(a) & (b)**: In these Figures we show i) our extrapolation after sampling, ii) our modelling before sampling, and iii) the original data before sampling, for two different, popular, legitimate files. We see that the extrapolation and the model before sampling are almost perfectly aligned, justifying our approach. Additionally, we see that they both fit very well the original data. The median RSE in this case was 0.0741. **(c)**: Estimation of lost footprint due to sampling. We recover the size of geographical footprint of machines that use Symantec's software, again starting from a ≈10% sample of the data: (blue: error of our estimation; red: actual error/footprint loss due to sampling

B. SQ2: Given a sample of the geographical distribution of the cyber attacks, can we estimate the footprint of the original distribution?

The empirical geographical distribution of machines around the world is shown in Figure 4. As we saw, before sampling, the footprint of the distribution spans 229 countries. Because of sampling, it is often the case that some countries in the tail of the distribution, that have low counts, will inevitable disappear from the sample. In this particular case, the countries that are left in the sample are 224. We refer to this problem as the footprint loss, due to sampling, and here we propose a way to accurately recover the number of countries that are indeed missing from the sample, i.e. the lost footprint.

Zero-th frequency moment of the GEOSPLIT distribution: In order to come up with a reliable means of estimating the footprint prior to sampling, we have to make a few assumptions with respect to the type of the distribution. As we showed earlier, in Figure 4, the geographical distribution of machines follows GEOSPLIT model. Under this assumption, we may leverage the work of [10] in order to perform our extrapolation.

More specifically, there is an immediate correspondence of the *zero-th frequency moment* of the GEOSPLIT distribution, to the number of countries that it spans. If we denote by m_i the count of machines for each country, then the q-th frequency moment is defined as $F_q = \sum_i {m_i}^q$. If $q = 0$, then F_0 is simply the number of countries in our distribution. Thus, if we are able to accurately estimate the F_0 given the sample, then we have an estimate of the lost footprint.

Given the distribution of machines across countries (C_1, ..., C_m, for m countries), we have that $N = C_1 + \ldots + C_m$, and we can estimate k and p: $k = \lceil \log_2(m) \rceil$, and $C_1 = Np^k$.

Then, we estimate F_0 (see [10] for more details). Specifically, if the j-th country has estimated count $\hat{C}_j < 1$, we round down to zero, and consider that country as having no machines that submit executables.

In Figure 6(c) we provide a comprehensive look at the performance of our approach. In particular (in red) we show the actual footprint loss, in other words the error incurred by sampling, which is 5 countries. With our proposed approach of estimating the zero-th frequency moment of the distribution, we are able to accurately argue that 228 countries originally exist in the footprint, resulting in an error of just 1 country.

V. DISCUSSION - SAMPLING IS UNAVOIDABLE

A. Lessons learned

Should we bother with sampling and extrapolations, given that major security companies, like Symantec, have huge amounts of data? The answer is "yes", for several reasons:

- *Nobody sees the full picture:* Since only a subset of all the machines in the Internet are using any security software at all (and are, thus, monitored for malware infections), and a subset of this subset uses software by a particular security software vendor, e.g. Symantec, the following natural issue arises: The data that each security vendor monitors is but a *sample* of the real world. Thus, if Symantec to estimate what is happening in the whole world, it still needs to use our models and the extrapolations formulas based on them.
- *Big Data is difficult to transfer and analyze:* When data sets reach petabyte scales and they are collected on hundreds of millions of hosts worldwide, keeping them up-to-date can require a prohibitive amount of storage and bandwidth. Moreover, analysis tasks can execute for several hours or even days on large data sets, which makes it difficult to experiment with new, unoptimized, data intensive techniques.
- *Security telemetry is collected at high rate:* In 2011, 403 million new malware variants were created (more than *1 million each day*) [28], and data sets that grow by several gigabytes per day are common in the security industry. This problem, also known as "data velocity," is present in other industries; for example, Akamai collects log data at a rate of 10 GB/s [17]. When faced with such high data collection rates, practitioners either apply aggressive compression techniques, which can render data processing difficult, or they store only part of the data set (i.e., a sample). Representative sampling techniques that can be applied on-the-fly, as the data is collected (as the sampling strategy adopted in WINE), can enable open-ended analyses and experiments on such data sets.
- *Restrictions in data access:* Because large data sets are often critical to a company's operations, the entire corpus of data collected by the company is kept confidential and the systems used to analyze the data in production are not opened up to research projects. Under these circumstances, prototypes and models are developed using sampled data, which further emphasizes the

need for model fitting and extrapolations.

In short, sampling and extrapolations are necessary, for everybody in this field, including the largest computer security companies.

B. Deployment & Impact

WINE is an operational system, used for experimenting with new Big Data ideas and for building data analysis prototypes. In 2012, several engineering teams within Symantec and five academic researchers have used WINE in their projects. The sampling techniques described and validated in this paper enable open-ended experiments at scale that often correlate several data sets, collected and sampled independently.

For example, WINE has provided unique insights into the prevalence and duration of zero-day attacks. A zero-day attack exploits one or more vulnerabilities that have not been disclosed publicly. Knowledge of such vulnerabilities gives cyber criminals a free pass to attack any target, from Fortune 500 companies to millions of consumer PCs around the world, while remaining undetected. WINE has enabled a systematic study of zero-day attacks that has shown, among other findings, that these attacks are more common than previously thought and that they go on undiscovered for 10 months on average [3]. Quantifying these properties had been a long-standing open question, because zero-day attacks are rare events that are unlikely to be observed in honeypots or in lab experiments; for instance, exploits for most of the zero-day vulnerabilities identified in the study were detected on fewer that 150 hosts out of the 11 million analyzed. This result was achieved by correlating the binary reputation data set, analyzed in this paper, with additional types of security telemetry (anti-virus detections, dynamic analysis traces, vulnerability databases) and was made possible by the fact that the WINE sampling algorithm makes consistent decisions: if the data collected in one data set about a certain host is included in one data set, then it will be included in all the other data sets as well.

In addition to the technical implications of these results, they also illustrate the opportunities for employing machine-learning techniques in cyber security (e.g., assessing the reputation of unknown binaries [7], which singles out rare events such as zero-day attacks). The SHARKFIN, GEOSPLIT and PPL models, introduced in this paper, represent another step in this direction. Understanding the basic properties of security telemetry opens up promising research avenues into preventing cyber attacks, by distinguishing malicious and benign software using their propagation patterns and by estimating the number of hosts and geographies reached by worms and by security updates.

VI. RELATED WORK

Propagation of Executables In July 2001, the Code Red worm infected 359,000 hosts on the Internet in less than 14 hours [21]. Code Red achieved this by probing random IP addresses (using different seeds for its pseudo-random number generator) and infecting all hosts vulnerable to an IIS exploit. This incident triggered a wave of research into the propagation of Internet worms. In 2002, Staniford et al. analyzed Code Red traces and proposed an analytical model for its propagation [27]. Based on this model, the researchers also suggested that a worm can infect one million vulnerable hosts on the Internet within 30 seconds by replacing random probing with a combination of hit-list scanning, permutation scanning, and use of Internet-sized hit-lists [27]. In follow-on work, they showed that additional optimizations may allow a worm to saturate 95% of one million vulnerable hosts on the Internet in less than 2 seconds [26]. Such techniques were subsequently employed by worms released in the wild, such as the the Slammer worm [20] (infected 90% of all vulnerable hosts within 10 minutes) and the Witty worm [31].

Gkantsidis et al. study the dissemination of software patches through the Windows Update service and find that approximately 80% of hosts request a patch within a day after it is released; the number of hosts drops by an order of magnitude during the second day, and is further reduced by factor of 2 in day three [12].

Influence Propagation Studies on virus and influence propagation are numerous, with popular books [1] and surveys [13], blog analysis [15], response times in linked-in invitations [14], spike analysis in youtube video [8] and the recent `SpikeM` model [19], which our SHARKFIN model generalizes. Recent research in malware detection [7] leverages propagation-based machine learning method (Belief Propagation) to infer files' reputations (e.g., malicious or benign).

Power Law Distributions Power laws appear in countless settings [32], [22], including network topology [11], web topology [2], [4] and are closely related to fractals and self-similarity (see [18] and [25] for long lists of settings with power-law distributions). Multifractals [25] and the multifractal wavelet model [24] are closely related to our GEOSPLIT model, and have been used to model local area network traffic, web traffic [9], disk accesses [30] [29].

VII. CONCLUSIONS

In this paper we analyzed one of the largest available security databases, comprised by both malware and benign executables. We provide intuitive insights on the data and we identify surprising patterns therein. Moreover we provide efficient techniques in order to extrapolate key attributes and properties of the full data, based on a small, uniform, random sample.

Our key contributions are:

- **Spatio temporal models** for malware/software propagation. Specifically:
  - *Spatial:* The GEOSPLIT model for the geographical spread of infected machines.
  - *Temporal:* The SHARKFIN model for the temporal evolution of executables.
  - *Lifetime:* The PPL (periodic power law), with slope -1, for the life-time of software-disseminating URLs.
- **Extrapolations from Sample**: Thanks to our spatio-temporal models above, we showed how to extrapolate the spatio-temporal properties, given a sample of malware propagation.

VIII. ACKNOWLEDGMENTS

We thank Vern Paxson and Marc Dacier, for their early feedback on the the design and effects of the WINE sampling strategy. The data analyzed in this paper is available for follow-on research as the reference data set `WINE-2012-006`. Research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053.

APPENDIX

The SHARKFIN model

The SHARKFIN model of executable propagation is a generalization of the `SpikeM` model [19] for the spreading of memes through blogs. We briefly describe the model in [19], adapting to the task at hand

The model assumes a total number of N machines that can be infected. Let $U(n)$ be the number of machines that are *not* infected at time n; $I(n)$ be the count of machines that got infected up to time $n-1$; and $\Delta I(n)$ be count of machines infected exactly at time n. Then $U(n+1) = U(n) - \Delta I(n+1)$ with initial conditions $\Delta I(0) = 0$ and $U(0) = N$.

Additionally, we let β as the strength of that executable file. We assume that the infectiveness of a file on a machine drops as a specific power law based on the elapsed time since the file infected *that machine* (say τ) i.e. $f(\tau) = \beta\tau^{-1.5}$. Finally, we also have to consider one more parameter for our model: the "external shock", or in other words, the first appearance of a file: let n_b the time that this initial burst appeared, and let $S(n_b)$ be the size of the shock (count of infected machines).

Finally, to account for periodicity, we define a periodic function $p(n)$ with three parameters: P_a, as the strength of the periodicity, P_p as the period and P_s as the phase shift.

Putting it all together, our SHARKFIN model is

$$\Delta I(n+1) = p(n+1)\left(U(n)\sum_{t=n_b}^{n}\left(\Delta I(t) + S(t)\right) f(n+1-t) + \epsilon\right)$$

where $p(n) = 1 - \frac{1}{2}P_a\left(\sin\left(\frac{2\pi}{P_p}\left(n + P_s\right)\right)\right)$, and ϵ models external noise.

No-sampling version: If $X(n), n = 1\cdots T$ is the sequence of file occurrences we want to model as a SHARKFIN spike, we want minimize the following:

$$\min_{\boldsymbol{\theta}} \sum_{n=1}^{T}\left(X(n) - \Delta I(n)\right)^2$$

where $\boldsymbol{\theta} = \begin{bmatrix} N & \beta & S_b & P_a & P_s \end{bmatrix}^T$ is the vector of model parameters.

With sampling: If we are dealing with a sample of file occurrences, with sampling rate s, then we solve the problem:

$$\min_{\boldsymbol{\theta}} \sum_{n=1}^{T}\left(sX(n) - \Delta I(n)\right)^2$$

In both cases, we use Levenberg-Marquardt [16] to solve for the parameters of our SHARKFIN model.

REFERENCES

[1] RM Anderson and RM May. Coevolution of hosts and parasites. *Parasitology*, 85(02):411–426, 1982.

[2] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.

[3] Leyla Bilge and Tudor Dumitraş. Before we knew it: An empirical study of zero-day attacks in the real world. In *ACM Conference on Computer and Communications Security*, Raleigh, NC, Oct 2012.

[4] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *WWW9 / Computer Networks*, 33(1–6):309–320, 2000.

[5] Juan Caballero, Chris Grier, Christian Kreibich, and Vern Paxson. Measuring pay-per-install: The commoditization of malware distribution. In *USENIX Security Symposium*. USENIX Association, 2011.

[6] Jean Camp, Lorrie Cranor, Nick Feamster, Joan Feigenbaum, Stephanie Forrest, Dave Kotz, Wenke Lee, Patrick Lincoln, Vern Paxson, Mike Reiter, Ron Rivest, William Sanders, Stefan Savage, Sean Smith, Eugene Spafford, and Sal Stolfo. Data for cybersecurity research: Process and "wish list". http://www.gtisc.gatech.edu/files_nsf10/data-wishlist.pdf, Jun 2009.

[7] D.H. Chau, C. Nachenberg, J. Wilhelm, A. Wright, and C. Faloutsos. Polonium: Tera-scale graph mining and inference for malware detection. *SIAM International Conference on Data Mining*, 2011.

[8] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences*, 105(41):15649–15653, 2008.

[9] M. Crovella and A. Bestavros. Self-similarity in world wide web traffic, evidence and possible causes. *Sigmetrics*, pages 160–169, 1996.

[10] C. Faloutsos, Y. Matias, and A. Silberschatz. Modeling skewed distribution using multifractals and the80-20'law. *Computer Science Department*, page 547, 1996.

[11] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *SIGCOMM*, pages 251–262, Aug-Sept. 1999.

[12] Christos Gkantsidis, Thomas Karagiannis, and Milan Vojnovic. Planet scale software updates. In Luigi Rizzo, Thomas E. Anderson, and Nick McKeown, editors, *SIGCOMM*, pages 423–434. ACM, 2006.

[13] H.W. Hethcote. The mathematics of infectious diseases. *SIAM review*, 42(4):599–653, 2000.

[14] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[15] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs. *arXiv preprint arXiv:0704.2803*, 2007.

[16] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[17] Bruce Maggs. Personal communication, 2012.

[18] Benoit Mandelbrot. *Fractals: Form, chance, and dimension*, volume 1. W. H. Freeman, 1977.

[19] Y. Matsubara, Y. Sakurai, B. Aditya Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: Model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 6–14. ACM, 2012.

[20] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the Slammer worm. *Security & Privacy, IEEE*, 1(4):33–39, 2003.

[21] David Moore, Colleen Shannon, and Kimberly C. Claffy. Code-red: a case study on the spread and victims of an internet worm. In *Internet Measurement Workshop*, pages 273–284. ACM, 2002.

[22] M. E. J. Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46, 2005.

[23] Tudor Dumitraş and Darren Shou. Toward a standard benchmark for computer security research: The Worldwide Intelligence Network Environment (WINE). In *EuroSys BADGERS Workshop*, Salzburg, Austria, Apr 2011.

[24] Ruldolf H. Riedi, Matthew S. Crouse, Vinay J. Ribeiro, and Richard G. Baraniuk. A multifractal wavelet model with application to network traffic. In *IEEE Transactions on Information Theory*, number 3, April 1999.

[25] Manfred Schroeder. *Fractals, Chaos, Power Laws*. W. H. Freeman, New York, 6 edition, 1991.

[26] Stuart Staniford, David Moore, Vern Paxson, and Nicholas Weaver. The top speed of flash worms. In Vern Paxson, editor, *WORM*, pages 33–42. ACM Press, 2004.

[27] Stuart Staniford, Vern Paxson, and Nicholas Weaver. How to 0wn the internet in your spare time. In *Proceedings of the 11th USENIX Security Symposium*, pages 149–167, Berkeley, CA, USA, 2002. USENIX Association.

[28] Symantec Corporation. Symantec Internet security threat report, volume 17. http://www.symantec.com/threatreport/, April 2012.

[29] Mengzhi Wang, Anastassia Ailamaki, and Christos Faloutsos. Capturing the spatio-temporal behavior of real traffic data. *Perform. Eval.*, 49(1/4):147–163, 2002.

[30] Mengzhi Wang, Tara Madhyastha, Ngai Hang Chang, Spiros Papadimitriou, and Christos Faloutsos. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic. *ICDE*, February 2002.

[31] Nicholas Weaver and Dan Ellis. Reflections on Witty: Analyzing the attacker. *;login: The USENIX Magazine*, 29(3):34–37, June 2004.

[32] G.K. Zipf. *Human Behavior and Principle of Least Effort: An Introduction to Human Ecology*. Addison Wesley, Cambridge, Massachusetts, 1949.