RESEARCH Open Access

Cloud repository as a malicious service: challenge, identification and implication

Xiaojing Liao[1,2*], Sumayah Alrwais[1], Kan Yuan[1], Luyi Xing[1], XiaoFeng Wang[1], Shuang Hao[1] and Raheem Beyah[1]

Abstract

The popularity of cloud hosting services also brings in new security chal- lenges: it has been reported that these services are increasingly utilized by miscreants for their malicious online activities. Mitigating this emerging threat, posed by such "bad repositories" (simply *Bar*), is challenging due to the different hosting strategy to traditional hosting service, the lack of direct observations of the repositories by those outside the cloud, the reluctance of the cloud provider to scan its customers' repositories without their consent, and the unique evasion strategies employed by the adversary. In this paper, we took the first step toward understanding and detecting this emerging threat. Using a small set of "seeds" (i.e., confirmed Bars), we identified a set of collective features from the websites they serve (e.g., attempts to hide Bars), which uniquely characterize the Bars. These features were utilized to build a scanner that detected over 600 Bars on leading cloud platforms like Amazon, Google, and 150 K sites, including popular ones like `groupon.com`, using them. Highlights of our study include the pivotal roles played by these repositories on malicious infrastructures and other important discoveries include how the adversary exploited legitimate cloud repositories and why the adversary uses Bars in the first place that has never been reported. These findings bring such malicious services to the spotlight and contribute to a better understanding and ultimately eliminating this new threat.

Keywords: Cloud, Cybercrime, Malicious hosting service

Introduction

Cloud hosting service today is serving over a billion users world-wide, providing them stable, low-cost, reliable, high-speed and globally available resource access. For ex- ample, Amazon Simple Storage Service (S3) is reported to store over 2 trillion objects for web and image hosting, system backup, etc. In addition to storing data, these ser- vices are moving toward a more active role in supporting their customers' computing missions, through sharing the repositories (a.k.a. *bucket* for Google Cloud (Buckets n.d.)) hosting various dynamic content and programming tools. A prominent example is Google's Hosted Libraries (Google. Google hosted libraries 2015), a content distribution network (CDN) for disseminating the most popular, open-source JavaScript resources, which web developers can easily incorporate into their websites through a simple code snippet. In addition to benign users, the popularity of these services has also attracted cybercriminals. Compared with dedicated underground hosting services, repositories on legitimate commercial clouds are more reliable and harder to blacklist. They are also much cheaper: for example, it is reported that 15 GB on the dark net is sold at $15 per month (Servnet n.d.), which is actually offered for free by Google to every Google Driver user. Indeed, it has been reported (solutionary 2015) that malware distributors are increasingly using the commercial clouds to process and deploy malicious content.

Understanding bad cloud repositories: challenges

Although there have been indications of cloud hosting misuse, understanding how such services are abused is challenging. For the service providers, who are bound by their privacy commitments and ethical concerns, they tend to avoid inspecting the content of their customers' repositories in the absence of proper consent. Even when the providers are willing to do so, determining whether a repository involves malicious content is by no means

* Correspondence: xliao@indiana.edu
[1]Indiana University, King Saud University, University of Texas at Dallas, Georgia Institute of Technology, Atlanta, USA
[2]Department of Computer Science, Indiana University Bloomington, Bloomington, USA

trivial: nuts and bolts for malicious activities could appear perfectly innocent before they are assembled into an attack machine; examples include image files for Spam and Phishing as shown in Fig. 1. Actually, even for the repository confirmed to serve malicious content like malware, today's cloud providers tend to only remove that specific content, instead of terminating the whole account, to avoid collateral damage (e.g., compromised legitimate repositories). Exploring the issue becomes even more difficult for the third party, who does not have the ability to directly observe the repositories and can only access them through the websites or sources that utilize the storage services. Further adding to the complexity of finding such a repository is the diverse roles it may play in attack infrastructures (e.g., serving malware for one attack and serving Phishing content for another), due to the mixed content a single repository may host: e.g., malware together with Phishing images. As a result, existing techniques (e.g., those for detecting dedicated malicious services (Li et al. 2014; Nelms et al. 2015)) cannot be directly applied to capture the repository, simply because their original targets often contain more homogeneous content (e.g., just malware) and contribute to different campaigns in the same way. So far, little has been done to understand the scope and magnitude of malicious or compromised repositories on legitimate clouds (called *Bad Repository* or simply *Bar* in our research) and the technical details about their services to the adversary, not to mention any effort to mitigate the threat they pose.

Finding "bars" online

In this paper, we present the first systematic study on the abuses of cloud repositories on the legitimate cloud platforms as a malicious service, which was found to be highly pervasive, acting as a backbone for large-scale malicious web campaigns (Section "Measurement and Discoveries"). Our study was bootstrapped by a set of "seeds": 100 confirmed malicious or compromised buckets (Buckets n.d.), each of which is a cloud resource repository with stored objects (often of different types) organized under a unique identification key. These buckets were collected from Spam messages or the malicious URLs cached by a popular malware scanner. Comparing them with those known to be legitimate, we found that despite various roles each bucket plays in different types of attacks (due to the diversity in the content it serves), still the websites connecting to those buckets exhibit prominent common features (see Section "Features of Bad Repositories"), particularly, the presence of "gatekeeper" sites that cover the Bars (a valuable asset for the adversary) and remarkably homogeneous redirection behavior (i.e., fetching repository resources indirectly through other sites' references) and sometimes similar content organizations, due to the same attack payload the compromised sites upload from their back-end (i.e., the Bars), or the templates the bucket provides to the adversary for quick deployment of her attack sites. By comparison, a legitimate bucket (e.g., reputable jQuery repository) tends to be *directly* accessed by the websites with highly diverse content.

Based on this observation, we developed *BarFinder*, a scanner that automatically detects Bars through inspecting the topological relations between websites and the cloud bucket they use, in an attempt to capture Bars based on the external features of the websites they serve. More specifically, for all the sites connecting to a repository, our approach correlates the domains and URLs (particular those related to cloud repositories) across their redirection chains and content features across their DOM structures to identify the presence of gatekeepers and evading behavior, and also measure the diversity of their content organization. A set of new *collective features* generated in this way, including *bucket usage similarity*, *connection ratio*, *landing similarity* and others (Section "Features of Bad Repositories"), are further utilized by a classifier to find out suspicious buckets. Running the scanner over all the data collected by the Common Crawl (Crawl 2015), which indexed five billion web pages, for those associated with all major cloud storage providers (including Amazon S3, Cloudfront, Google Drive, etc.), we found around 1 million sites utilizing 6885 repositories hosted on these clouds. Among them, BarFinder identified 694 malicious or compromised repositories, involving millions of files, with a precision of 95% and a coverage of 90% against our ground-truth set.

Our discoveries

Looking into the Bars identified by our scanner, we are surprised by the scope and the magnitude of the threat. These buckets are hosted by the most reputable cloud service providers. For example, 13.7% of Amazon S3 repositories and 5.5% of Google repositories that we

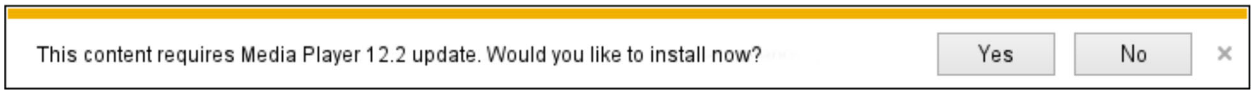

This content requires Media Player 12.2 update. Would you like to install now?

Fig. 1 Example of deceptive images in Amazon S3 bucket *cicloudf ront* used for malvertising. The image was shown at the bottom of a webpage as an update notification to lure visitors to download malware

inspected turned out to be either compromised or completely malicious.[1] Among those compromised are popular cloud repositories such as Groupon's official bucket. Altogether, 472 such legitimate repositories were considered to be contaminated, due to a misconfiguration flaw never reported before, which allows arbitrary content to be uploaded and existing data to be modified without proper authorization. The impact of these Bars is significant, infecting 1306 legitimate websites, including Alexa top 300 sites like groupon.com, Alexa top 5000 sites like space.com, etc. We reported our findings to Amazon and leading organizations affected by the infections. Groupon has already confirmed the compromise we discovered and awarded us for our help.

When it comes to malicious buckets, our study brings to light new insights into this new wave of repository based cyber-attacks, including the importance of Bars to malicious web activities and the challenges in defending against this new threat. More specifically, we found that on average, one Bar serves 152 malicious or com- promised sites. In one of the large campaigns discovered in our research, the Bar cloudfront_file.enjin.com hosts a malicious script that was injected into at least 1020 websites (Section "Prevalence and sharing"). These Bars sit right at the center of the attack infrastructure, supporting and coordinating other malicious actors' operations at dif- ferent stages of a campaign. Interestingly, we found that they could be strategically placed on different cloud platforms, making them hard to block (due to the popularity of their hosting clouds like Google) and detect (scattered across different providers), and easy to share across multiple campaigns. As an example, the Potentially Unwanted Programs (PUP) campaign we found first loads a redirection script from a Bar on Akamaihd (the world's largest CDN platform) to lead the victim to the attack web- site, then fetches Phishing pictures from an Amazon S3 Bar, and finally delivers the malware stored on Cloudfront to the target systems (Section "Case Studies"). In the presence of such meticulously planned attacks, the cloud service providers apparently are inade- quately prepared, possibly due to the privacy constraints in touching their customers' repositories. We found that many *Bars* survive a much longer lifetime than that of the malicious content hosted on websites. Further complicating the mission of Bar identification are other evasion techniques the adversary employs, including code obfuscation and use of a redirection chain and cloaking techniques to avoid exposing malicious payloads to a malware scanner.

Contributions

The contributions of the paper are as follows:

New understanding. We performed the *first* systematic study on cloud repositories as a malicious service, an emerging security threat. For the first time, our study reveals the scope and magnitude of the threat and its significant impact, particularly on the infrastructures of illicit web activities. These findings bring to the spotlight this important yet understudied problem and lead to a better understanding of the techniques the adversary employs and their weaknesses. This will contribute to better defense against and ultimate elimination of the threat.

New technique. Based on our understanding of bad cloud repositories, we take a first step toward automatically detecting them. The technique we developed relies on the topological relationship between a cloud repository and the websites it serves, which are difficult to change and effective at capturing malicious or compromised buckets. Our evaluation over a large number of popular websites demonstrates the potential of the technique, which could be utilized by both cloud providers and third parties to identify the threats posed by *Bars*.

Background

Cloud hosting

Cloud hosting is a type of infrastructure as a service (IaaS), which is rented by the cloud user to host her web assets (e.g., HTML, JavaScript, CSS, and image files). These web assets are organized into cloud repositories referred to as *buckets* which are identified by unique,[2] user-assigned keys, that are mapped as sub-domains. For example, the subdomain aws-publicdatasets.s3.amazonaws.com identifies Amazon S3 as the cloud platform and aws-publicdatasets as the user's cloud bucket and repository. Such name assignment is labeled as `s3.amazonaws.com_ aws-publicdatasets` throughout this paper. Also, each bucket is protected by an access control list configured by the user to authorize requests for her resources.

In recent years, we have seen an increase in popularity of these services. A key feature of cloud hosting is built-in site publishing (Google. Publish website content 2015), where the web assets in the bucket can be served directly to users via file names in a relative path in the bucket (i.e., *cloud URL*). For instance, JavaScript files hosted in the cloud bucket can be directly run in the browser. Also, the pay-as-you-go hosting is well received as an economic and flexible computing solution. As an example, Google Drive today offers a free web hosting service with 15GB of storage, and an additional 100GB for $1.99/month, and GoDaddy's web hosting starts at $1/month for 100GB.

Besides such front-end websites, mainstream cloud providers today (Amazon S3, Microsoft Azure, Google Drive, etc.) all allow their customers to store different kinds of web content and other resources in their cloud buckets, serving as back-end repositories that can be easily accessed by front-end applications (like the website) and shared across different parties. Figure 2 illustrates an example, in which the resource owner creates a bucket on the cloud hosting platform and uploads a script there (CD); this resource (i.e., the script) is made public through a cloud URL, which can be embedded into any website (@); whenever the site is visited (°), requests will be generated for fetching the script (©) and delivering it to the visitor's browser (°). The bucket in the example is typical of a service repository, whose resources can be fetched and updated through a cloud URL: for example, the visitor statistics of a website can be collected through a link (`s3.amazonaws.com/trk.cetrk.com/t.js`), which downloads a tracking script from `s3.amazonaws.com_trk.cetrk.com`, a bucket owned by the tracking website Crazy Egg. This is different from a "self- serving" bucket, whose resources can only be accessed by the bucket owner's sites. Note that our study focuses on abuses of this type of cloud repositories, regardless of the additional functionalities they may have (e.g., CDNs, DDoS protection, etc.), since these functionalities do not affect the way the repositories are used by either legitimate or malicious parties.

Adversary model

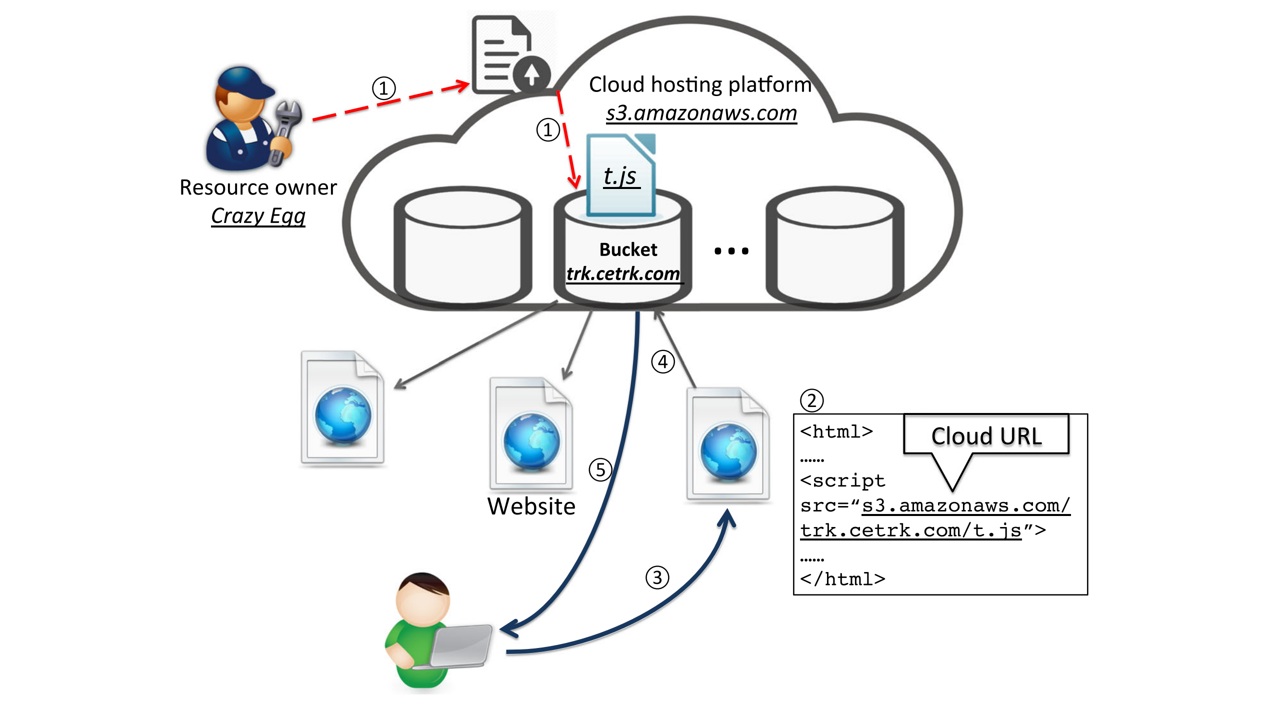
In our research, we consider the adversary who tries to use cloud buckets on legitimate cloud platforms as service repositories for illicit activities. For this purpose, the attacker could build her own malicious bucket or compromise legiti- mate ones, and store various attack vectors there, including Spam, Phishing, malware, click-hijacking and others. These buckets are connected to front-end websites, which could be malicious, compromised or legitimate ones contaminated only by the Bar.

Finding bars online

In this section, we elaborate on our analysis of a set of known Bars (the seed set) and the features identified for differentiating benign repositories and Bars. These features are utilized in our research to build a simple web scanner, *BarFinder*, for detecting other malicious or compromised high-profile, previously-unknown repositories and the malicious campaigns in which they serve.

Features of bad repositories

Our study is based on a small set of confirmed good and bad repositories and their related domains, which we analyzed to find out how Bars (bad repositories) differ from legitimate repositories. In the absence of direct access to these buckets, good or bad, all we can do is to *infer* their legitimacy from who use them and how they are used (by different domains), that is, the features of the domains and their interactivities on the redirection paths leading to the cloud repository. Of particular interest here are a set of *collective* properties identified from the resource fetching chains (a.k.a., redirection chains) for serving the content of Bars, which is hard to change by the adversary, compared with the content features of



```
<html>
……
<script
src="s3.amazonaws.com/
trk.cetrk.com/t.js">
……
</html>
```

Fig. 2 Overview of the cloud hosting process

individual Bars. Below, we elaborate on the way such data was collected and the salient features discovered in our research, which describe how the adversary attempts to hide Bars or use them to cover other attack assets, a redirection pattern never observed on legitimate repositories.

Data collection

To build the seed set, we collected a set of confirmed malicious or compromised buckets (called *Badset*) and legitimate buckets (called *Goodset*) as well as their related domains, as illustrated in Table 1.

Badset We utilized two feeds as the ground truth for gathering bad cloud buck- ets: the *Spamtrap* feed and the *CleanMX* feed (Clean-MX 2015). The former comes from a Spam honeypot we constructed (A. authors n.d.) that receives around 10 K Spam emails per day, from which cloud URLs promoted by the emails were extracted which may include spam resources such as HTML, images, and scripts. The latter includes the historical data of CleanMX, a popular domain scanning engine, from which cloud-related URLs were collected. For both feeds, we further validate them by VirusTotal (VirusTotal 2015) and manual inspections (e.g., looking for Phishing content) to ensure that they were indeed bad (to avoid contaminating the dataset with legitimate buckets used in malicious activities). Using the collected set of malicious cloud URLs from both feeds, we extracted their repositories, which led to 100 confirmed Bars.

Goodset The good buckets were gathered from the Alexa top 3 K websites, which are considered to be mostly clean. To this end, we visited each website using a crawler (as a Firefox add-on) to record the HTTP traffic triggered by the visit, including network requests, responses, browser events, etc. From the collected traffic, we extracted the HTTP cloud request URLs corresponding to 300 cloud buckets hosted on 20 leading cloud hosting services like Amazon S3, Google Drive, etc. (see Table 6 in Appendix for the complete list). Note that even though some of them provide CDN service or DDOS protection, they are all provided hosting service to act as cloud repository.

Bucket-served sites and their HTTP traffic We collected HTTP traffic using the crawler mentioned above to visit a list of websites using buckets for feature extraction. Rather than blindly crawling the web to find those sites, we adopted a more targeted strategy by crawling the sites found to contain links to the cloud in the past. We built the site list with the help of *Common Crawl* (Crawl 2015), a public big data project that crawls about 5 billion webpages each month through a large-scale Hadoop-based crawler and maintains lists of the crawled websites and their embedded links. Searching the *Common Crawl* (Crawl 2015) dataset, collected in February 2015, for the websites loading content from the 400 clean and malicious buckets identified above, we found 141,149 websites, were used by our crawler.

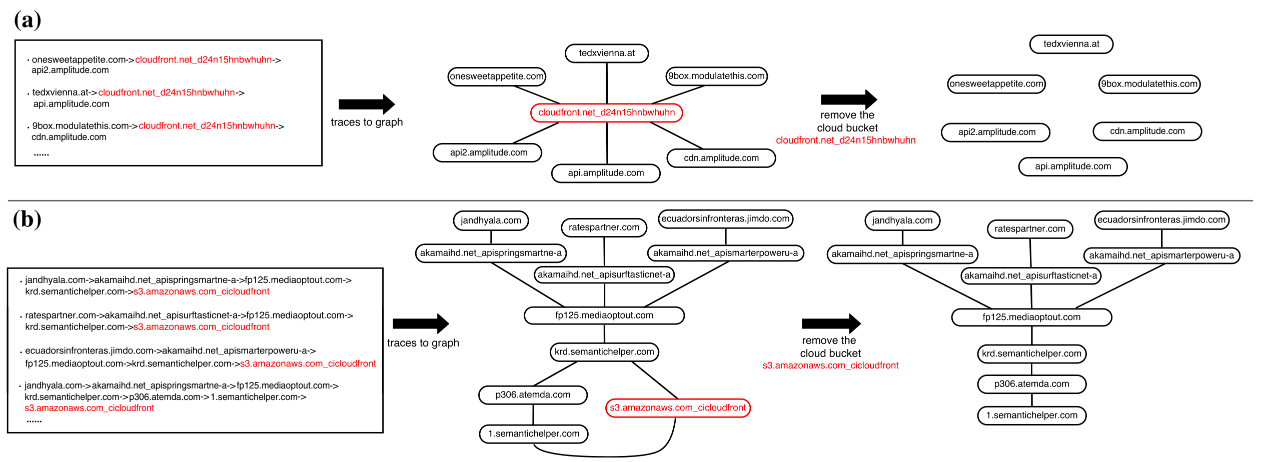
Topological features

We first inspected the *topology* of the redirection infrastructure associated with a specific bucket. Such an infrastructure is a collection of redirection paths, with each node being a *Fully Qualified Domain Name* (FQDN). On each path, the bucket is either a node when it directly participates in a redirection (e.g., its cloud URL delivers a redirection script to the visitor's browser) or simply a passive repository providing resources like pictures to other domains. Figure 3 illustrates examples of redirection paths leading to two real-world repositories, one for a legitimate bucket `cloudfront.-net_d24n15hnbwhuhn` and the other for a Bar `s3.amazonaws. com_cicloudfront`.

A key observation from our study is that *the redirection infrastructure leading to a Bar tends to include the features for protecting the Bar from being detected by web scanners*, presumably due to the fact that the repository is often considered to be a valuable asset for the adversary. Specifically, we found that typically, there are a few *gatekeeper* nodes sitting in front of a Bar, serving as an intermediary to proxy the attempts to get resources from the Bar. Examples of the gatekeepers include `fp125. mediaoptout.com` and its downstream nodes in Fig. 3b. On the topology of such an infrastructure, these gatekeepers are the hubs receiving a lot of resource- access connections from entry sites (the first node on a redirection path, see Fig. 3). Also interestingly, our research shows that some gatekeepers can access the Bar through multiple paths. For example, in Fig. 3b, `krd.semantichelper.com` can either go straight to `s3.amazonaws.com_cicloudfront` or take a detour through `p306.atemada.com`. This structure could be caused by the cloaking of the gatekeeper for hiding the Bar, or constructed to maintain access to the repository

Table 1 Summary results of the seed dataset

	# of buckets	# of linked websites	# of average linked website	# of redirection paths
Badset	100	12,468	133	468,480
Goodset	300	128,681	864	2,659,304



```
(a)
· onesweetappetite.com->cloudfront.net_d24n15hnbwhuhn->
  api2.amplitude.com
· tedxvienna.at->cloudfront.net_d24n15hnbwhuhn->
  api.amplitude.com
· 9box.modulatethis.com->cloudfront.net_d24n15hnbwhuhn->
  cdn.amplitude.com
  ......

(b)
· jandhyala.com->akamaihd.net_apispringsmartne-a->fp125.mediaoptout.com->
  krd.semantichelper.com->s3.amazonaws.com_cicloudfront
· ratespartner.com->akamaihd.net_apisurftasticnet-a->fp125.mediaoptout.com->
  krd.semantichelper.com->s3.amazonaws.com_cicloudfront
· ecuadorsinfronteras.jimdo.com->akamaihd.net_apismarterpoweru-a->
  fp125.mediaoptout.com->krd.semantichelper.com->s3.amazonaws.com_cicloudfront
· jandhyala.com->akamaihd.net_apispringsmartne-a->fp125.mediaoptout.com->
  krd.semantichelper.com->p306.atemda.com->1.semantichelper.com->
  s3.amazonaws.com_cicloudfront
  ......
```

(a)
traces to graph
tedxvienna.at
onesweetappetite.com
9box.modulatethis.com
cloudfront.net_d24n15hnbwhuhn
api2.amplitude.com
api.amplitude.com
cdn.amplitude.com
remove the cloud bucket cloudfront.net_d24n15hnbwhuhn
(b)
traces to graph
jandhyala.com
ratespartner.com
ecuadorsinfronteras.jimdo.com
akamaihd.net_apispringsmartne-a
akamaihd.net_apisurftasticnet-a
akamaihd.net_apismarterpoweru-a
fp125.mediaoptout.com
krd.semantichelper.com
p306.atemda.com
1.semantichelper.com
s3.amazonaws.com_cicloudfront
remove the cloud bucket s3.amazonaws.com_cicloudfront

Fig. 3 Example of the redirection infrastructure leading to the le- gitimate bucket *cloudf ront.net d24n15hnbwhuhn* (**a**) and the Bar *s3.amazonaws.com cicloudf ront* (**b**), which are in RED color

even when nodes (like `1.semantichelper.com`) are down (detected, cleaned, etc.). Note that such a protection structure does not exist on the paths to a benign repository (Fig. 3a): normally, the resources hosted in a repository (e.g., jQuery) is directly fetched by the website using it, without going through any redirection; even in the presence of redirections, there will not be any gatekeeper, not to mention attempts to cloak or build a backup path.

To identify this unique "protection" structure, we utilize two *collective features*: *bucket usage similarity* (BUS) that captures the topology involving hubs (gatekeepers) and *connection ratio* (CR) that measures the interactivities across different redirection paths (which point to the existence of cloaking behavior or the attempts to maintain back-up paths to the Bar). Specifically, consider a redirection graph $G = (V, E)$ (as illustrated in Fig. 3), where V is the set of nodes (the FQDNs involved in a redirection) and E is a set of edges from one node to the next one on individual paths: $E = \{e_{i,j} | node\ i\ precedes\ node\ j\ on\ a\ path\}$. The BUS is measured by

1 ^{i}, where i is the number of immediate predecessor nodes to a repository (the domains connecting to the repository) and s is the total number of entries of the repository's redirection graph. To find out the CR, we first remove the bucket b and all the edges to which it is attached (if they exist) to get another graph $G^t = G\ G_b$, where $G_b = (b, E_b)$ and $E_b = e_{b,j.}$ Note that each graph G^t is associated with one bucket. Then, from G^t, we find out the number of connected components n and calculate $CR = 1\ ^{n}$ (see Fig. 3 for an example).

Both collective features were found to be discriminative in our research. Figure 4a and b compare the cumulative distributions (CDF) of the ratios between Bad and Good sets. As we can see from the figures, Bars tend to have higher ratios than benign ones: the average BUS is 0.87 for the Bars and 0.79 for the legitimate repositories and the CR is 0.85 for the bad repositories and 0.67 for the good one. As mentioned earlier, this is caused by the fact that a small set of gatekeepers nodes are often placed there for protecting the Bars while the redirection chains towards the good repositories are much more direct and independent: different organizations typically do not go through an intermediary to indirectly access the public repository like jQuery, and even within the same organization, use of such a resource is often direct. Although there can be exceptions, our measurement study shows that in general, the structural differences between malicious and legitimate repositories are stark.

Also, we found that occasionally, a Bar itself may serve as a gatekeeper, running scripts to hide more valuable attack assets, such as the attack server or other malicious landing sites. When this happens, almost always the Bar leads to a small set of successors on redirection paths (e.g., attack servers, land sites). This is very different from the redirection performed by the script from a benign repository, for example, `cloudfront.net_d24n15hnbwhuhn`. In such cases, the targets of redirections are often very diverse. Based on this observation, we further measure the *landing similarity*, $LS = 1\ ^{l}$, where l is the number of the unique last nodes on the redirection paths associated with a repository. Again, as illustrated in Fig. 4c, our study shows that redirection paths involving Bars share fewer end nodes than legitimate ones, and therefore, the related redirection graphs (for Bars) have a higher landing similarity (0.94 vs 0.88).

Content and network features

In addition to their distinctive topological features, we found that the nodes on the redirection paths attached to a Bar often exhibit remarkable homogeneity in their content and network properties. Particularly, for the websites directly connecting to the repository, we found that they typically use a small set of templates (like

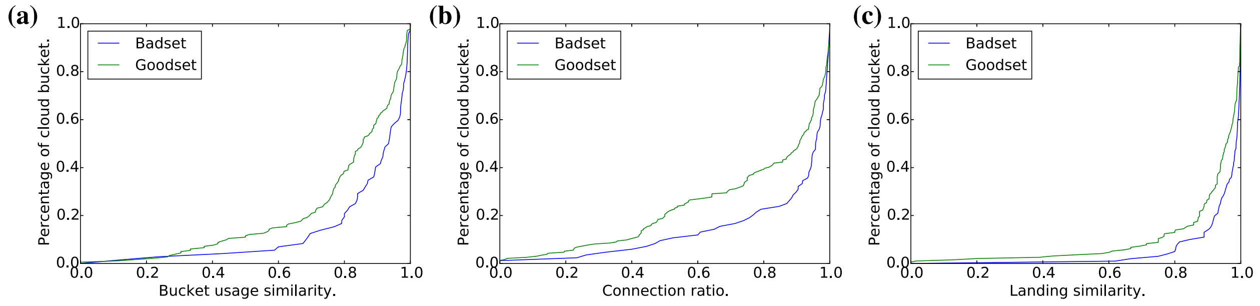


Fig. 4 Bars show smaller topological diversity. **a** Cumulative distribution of bucket usage similarity per of connected ratio per cloud landing similarity per cloud bucket. **b** Cumulative distribution bucket usage similarity per of connected ratio per cloud landing similarity per cloud bucket. **c** Cumulative distribution of bucket usage similarity per of connected ratio per cloud landing similarity per cloud bucket

WordPress) to build up their web pages, include similar DOM positions for script injection, carrying similar IP addresses or even having the same content management system (CMS) vulnerabilities, etc. These properties turn out to be very diverse among those utilizing a legitimate cloud repository. For example, all websites linking to a Google Drive Bar have their malicious cloud URL (for injecting a script) placed at the bottom of the DOM of each website. In another example, we found that the front-end sites using a Cloudfront Bar actually all include a vulnerable JCE Joomla extension.

To better understand the diversity of such websites, we try to compare them according to a set of content and network properties. In our research, we utilized the properties extracted by *WhatWeb* (WhatWeb 2015), a popular webpage scanner. WhatWeb is designed to identify the web technologies deployed, including those related to web content and communication: e.g., CMS, blogging platforms, statistic/analytics packages, JavaScript libraries, social media plugins, etc. For example, from the content

```
<link rel="search" type="application/
opensearchdescription+xml"
href="https://wordpress.com/open-
search.xml" title="WordPress.com" />
```

we obtain the property p as a key-value pair $p = (k, v) = (wordpress, opensearch)$, which indicates the website using wordpress plugin opensearch.

From our seed dataset, the scanner automatically extracted 372 keys of 1,596,379 properties, and then we clustered the keys into 15 classes such as Analytics and tracking, CMS and plugin, Meta-data information, etc., following the categories used by *BuiltWith*, a web technology search engine (BuiltWith 2015). Some examples of these properties are presented in Table 2. In addition to these properties extracted by *WhatWeb*, we added the following properties to characterize cloud URLs, including the position of the URL, the order in which different buckets appear in the web content and the number of cloud platforms used in a page.

Based on these properties, again we utilized a topological metric to measure the overall similarity across sites. Specifically, the relations among all the sites (connecting to the same bucket) in the same category (Analytics and tracking, CMS and plugin, etc.) are modeled as a graph $G^t = (V^t, E^t, P)$, where V^t is the set of the websites, which are characterized by a collection of properties P, and E^t is the set of edges:

$E^t = \{e_{i,j} | website\ i\ and\ j\ share\ p \in P\}$, that is, both sites having a common property. Over this graph, the *site similarity* is calculated as $SiS = 1 - \frac{n}{|V^1|}$. Here n is the number of connected components in the graph.

In our research, we computed SiS across all the categories summarized from the seed dataset, and compared those with Bars against those with the legitimate buckets. Again, the sites using Bars are found to share many more properties and therefore achieve a much higher similarity value than those linking to a good bucket. This is likely caused by mass-production of malicious sites using the same resources (templates, pictures, etc.) provided by a Bar or utilization of the same exploit tool stored in a Bar for compromising the sites with the same vulnerabilities. Therefore, such similarity is inherent to the attack strategies and can be hard to change.

BarFinder

Design

The design of BarFinder includes a web crawler, a feature analyzer, and a detector. The crawler automatically scans the web for cloud buckets (embedded in web content) and then clusters websites according to the buckets they use. From each cluster, the analyzer constructs a redirection graph and a content graph as described earlier (Section "Features of Bad Repositories"), on which it further calculates the values for a set of collective features including disconnection ratio (D), bucket usage similarity (B), landing similarity (L) and a series of content property/network property similarities ($S1$ S_n) for n web-technology categories (e.g., analytics and tracking,

Table 2 Examples of content and network features

Category	Feature	Example
Content	CMS platform information and their plugin	(wordpress, all in one SEO pack)
	Meta-data information	(metagenerator, drupal7)
	CloudURL information	(position, bottom)
	Advertising	(adsense, asynchronous)
	Javascript library	(JQuery, 1.9.1)
	Analytics and tracking	(Google-Analytics, UA-2410076-31)
	Widget	(addthis, welcome bar)
	DocInfo technologies	(open graph protocol, null)
Network	Identity	(IP, 216.58.216.78)
	Cookie	(Cookie, harbor.session)
	Server framework version	(Apache, 2.4.12)
	Custom HTTP header	(X-hacker, If youre..)

CMS and plugin, meta-data information, etc.). The output of this feature analysis is then passed to the detector, which maintains a model (trained on the seed dataset) to determine whether a bucket is malicious, based on its collective features.

Specifically, the crawler visits each website, inspecting its content, triggering events, recording the redirection paths it observes and parsing URLs encountered using the patterns of known cloud platforms to recognize cloud buckets. For ex- ample, the repository on Amazon S3 is accessed through the URL formatted as w + *.s3* w + [?].*amazonaws.com*, and Amazon CloudFront produces resource URLs in the form of w + *.cloudf ront.net*. In our research, 20 cloud platforms were examined to identify the buckets they host. At the feature-analysis stage, for each bucket, BarFinder inspects all its redirection paths, converts every node into an FQDN to compute their topological features, and then connects different nodes according to their content and network properties to find out their site similarities, as described in Section "Features of Bad Repositories".

Next, each cloud bucket i is uniquely characterized by a vector: D_i, B_i, L_i, $S_{i,}1$ $S_{i,n}$, with each element a collective feature. Individual features have different power in differentiating good and bad buckets, which we measured using the F-Score (Bishop 2006) (see Table 3). Note that the feature with a large score can better classify these vectors than the one with a small value. Therefore, a binary classifier with a model for weighing the features and other parameters can be used to classify the vector set and determine whether individual buckets are legitimate or not. Such a model is learned from the seed dataset. In our research, we utilized a Support Vector Machine (SVM) as the classifier, which showed the best performance among other classification algorithms (see Table 4). Its classification model is built upon the F-Scores for the collective features (D, B, etc.) and a threshold set according to the false positive and negative discovery expected to achieve. For each bucket classified, the SVM can also report the confidence of the classification.

Implementation

This simple design was implemented in our study into a prototype system. The web crawler was built as a Firefox add-on. In total, 20 such crawlers were deployed. We further developed a tool in Python to recover cloud URLs from the web content gathered by Common Crawl. The feature analyzer includes around 500 lines of Python code for processing the data collected by the crawler and computing the collective features (Section "Features of Bad Repositories"). Each feature in the vector is normalized using the L1 norm before passed to the SVM classifier. In our system, we incorporated the SVM provided by the scikit-learn open-source machine learning library (Sklearn 2015).

Evaluation

Here we report our evaluation of BarFinder on both the ground truth and the Unknown sets. All the experiments were conducted within an Amazon EC2 C4.8xlarge

Table 3 F-score of features

Feature	Label	Metric	F-score
Connection ratio	D	$1 - \frac{n}{\lvert V \rvert}$	0.084
Bucket usage similarity	B	$1 - \frac{i}{s}$	0.076
Landing similarity	L	$1 - \frac{l}{s}$	0.072
CMS information	S_1	$1 - \frac{n}{\lvert V' \rvert}$	0.037
Meta-data information	S_2	$1 - \frac{n}{\lvert V' \rvert}$	0.033
Analytics and tracking	S_3	$1 - \frac{n}{\lvert V' \rvert}$	0.032
Widget	S_4	$1 - \frac{n}{\lvert V' \rvert}$	0.031
CloudURL information	S_5	$1 - \frac{n}{\lvert V' \rvert}$	0.024

Table 4 Performance comparison under five-fold across validation

Classifier	Precision	Recall
SVM	0.94	0.89
Decision Tree	0.9	0.83
Logistic Regression	0.91	0.87
Naive Bayes	0.9	0.79
Random Forest	0.85	0.82

instance equipped with Intel Xeon E5–2666 36 vCPU and 60GiB of memory.

Evaluation on the seed set

We tested the effectiveness of BarFinder over our ground-truth dataset (i.e., the seed set) through the standard five-fold cross validation: that is, 4/5 of the data was used for training the SVM and the remaining 1/5 for evaluating the accuracy of Bar detection. Specifically, we randomly chose 80 Bars (out of 100) from the Badset and 240 (out of 300) legitimate buckets from the Goodset, together with the related websites (out of 141,149). These data were first processed by our prototype to adjust the weights and other parameters for its model. Then we tested the model on the remaining dataset (20 Bars, 60 legitimate buckets). The process is then repeated 5 times. BarFinder achieved both a low false discovery rate (FDR: 1- precision) and a high recall in detection: only 5.6% of reported Bars turned out to be legitimate (i.e., 1.6% of false positive rate), and over 89.3% of the Bars were detected. We further show the Area Under Curve (AUC) of the Receiver Operating Characteristics (ROC) graph, which comes very close to 1 (0.96), demonstrating the good balance we strike between the FD rate and the coverage. This preliminary analysis shows that the collective features of the sites connecting to cloud repositories are promising in detecting Bars.

Evaluation on the unknown set

We now use BarFinder to scan an unknown set. This unknown set contains HTTP traffic collected using a crawler as described in Section "Features of Bad Repositories" to visit a list of websites. This list of websites is also extracted from common crawl (Crawl 2015) by searching for websites that have loaded some content in the past from the cloud platforms listed in Table 6 in Appendix. As a result, the unknown data set contained HTTP traffic generated from dynamically visiting 1 M websites loading content from 20 cloud platforms and 6885 cloud buckets.

To validate our evaluation results, we employ a methodology that combines anti- virus (AV) scanning, blacklist checking, and manual analysis. Specifically, for the Bars flagged by our system, we first scan their cloud URLs with VirusTotal for malware and check them against the list of suspicious cloud URLs collected from our Spamtrap honeypot for Spam, Phishing, blackhat Search Engine Optimization (SEO), etc. In the case of VirusTotal, a URL is considered to be suspicious if at least two scanners raise the alarm. All such suspicious URLs (from either VirusTotal or the Spamtrap list) are cross-checked against the blacklist of CleanMX. Only those also found there are reported to be a true positive. Once a URL is confirmed malicious, its corresponding bucket is labeled as bad. Those unlabeled but flagged (by BarFinder) buckets are further validated manually.

In the experiment, BarFinder reported a total of 730 Bars, about 10.6% of the 6885 buckets. Among them, the AV scanning and blacklist verification confirmed that 502 buckets were indeed bad. The remaining 228 were manually analyzed through, e.g., inspecting the resources in the buckets for phishing or scam content, running scripts in the VM to capture binary code download. This validation further confirmed 192 Bars. The FDR was found to be at most 5% (assuming those not confirmed to be legitimate), in line with the finding from the seed set.

Measurement and discoveries

Based on the discoveries made by BarFinder, we further conducted a measurement study to better understand the fundamental issues about Bar-based malicious services, particularly *how the cloud repositories help facilitate malicious activities*, *how the adversary exploited legitimate cloud buckets* and *why the adversary uses Bars in the first place*. Our research shows that on the infrastructure, Bars play a pivotal role, compared with the content kept on other malicious or compromised sites, possibly because they are hosted on popular cloud services, and therefore hard to blacklist and also easy to share across different campaigns. Also, in a malicious campaign, the adversary may take advantage of multiple Bars, at different attack stages, to construct a complicated infrastructure that supports her mission (Section "Prevalence and sharing"). More importantly, we discovered that the adversary effectively exploited misconfigured legitimate buckets to infect a large number of their front-end web services (Section "Bucket Pollution"). Such observations, together with the challenge in blocking Bars, offer insights into the motivation for moving toward this new trend of repository-based attacks.

Prevalence and sharing

Landscape

As mentioned earlier, BarFinder reported 730 suspicious repositories from 6885 cloud buckets over 20 cloud platforms. Among them, we utilized 694 confirmed Bars (through AV/blacklist scanning or manual validation, see Section "BarFinder") for the measurement study. These Bars were found to directly serve 156,608 domains (i.e., front-end websites), through which they are further attached to 6,513,519 redirection paths involving 166,772 domains. Figure 5 illustrates the number of Bars we found on different cloud platforms. Among them, Amazon S3 is the most popular one in our dataset, hosting the most Bars (45%), which is followed by CloudFront (Amazon's CDN) 25.1% and Akamaihd 9.3%. Note that of these 20 clouds, seven of them provide free storage services (e.g., 15GB free space on Google Drive, 5GB for Amazon S3), and therefore easily become the ideal platforms for low-budget miscreants to distribute their illicit content. Also, eleven of them support HTTPS, on which malicious activities are difficult to catch by existing signature-based intrusion detection systems like snort and Shadow (Snort 2015; Symantec 2015). Interestingly, on some of the most prominent platforms, the miscreants are found to take advantage of the cloud providers' reputations to make their Phishing campaigns look more credible: for example, we found that the adversary continuously spoofed Gmail's login page on Google Drive, and the software download page for Amazon FireTV in an Amazon S3 bucket.

Figure 6 shows the distribution of Bars' frontend websites across 81 countries, as determined by the geolocations of the sites. The number of Bars' frontend sites in each country is ranked and described with different levels of darkness in the figure. We observe that most of these frontends stay in United States (14%), followed by Germany (7%) and United Kingdom (5%).

Content sharing

Our research reveals that Bars have been extensively shared among malicious or compromised websites, also across different positions on malicious redirection chains. Figure 7c illustrates the cumulative distribution of Bars' in- degrees in their individual redirection graphs: that is, the number of the sites utiliz- ing these Bars. On average, each Bar shows up on 252 sites and 12% of them are used by more than 200 websites. Table 5 lists the 10 most popular Bars we found. Among them, eight, including `s3.amazonaws.com_content.sitezoogle. Com`, `s3.amazonaws.com_publisher_configurations.shareaholic`, etc., host services for website generation, blackhat SEO or Spam. Particularly, `akamaihd. net_cdncache3-a` turns out to be a distributor of Adware, whose scripts are loaded into the victim's browser to redirect it to other sites for downloading different Adware. Also, we found that another Bar `s3.amazonaws.com_files.enjin. Com` hosts exploits utilized by 1020 bad sites. Finding Bars can help to effectively detect more sites with malicious contents.

Another interesting observation is that malicious content is also extensively shared across different Bars. To understand such content reuse, we grouped the malicious programs retrieved from different Bars based on the similarity of their code in terms of edit distance. Specifically, we removed the spaces within the pro- grams and ran the Python library *scipy.cluster.hierarchy.linkage* (Scipy 2015) to hierarchically cluster them (now in the form of strings) according to their Jaro scores (Cohen et al. 2003). In this way, we were able to discover three types of content sharing: intra-bucket sharing, cross-bucket sharing, and cross-platform sharing. Specifically, within the Amazon bucket `akamaihd.net_asrv-a`, we found that many of its cloud URLs are in the form of `http://asrv-a.akamaihd.net/sd/[num]/[num].js`. The JavaScript code turns out to be all identical, except that each script redirects the visitor to a different website. The similar code also appears in another Amazon bucket `akamaihd.net_cdncache-a`. As another example, we discovered the same ma- licious JavaScript (`JS.ExploitBlacole.zm`) from the Bars on CloudFront and Qiniudn respectively, even under the same path (i.e., `media/system/js/modal. js`). Moreover, we found that attackers used sub-domain generation algorithm to automatically generate sub-domain for Bars, then further reused the same malicious contents for these Bars. Specifically, we found that 28 content sharing Bars on Akamaihd have the same format in their names. Attackers

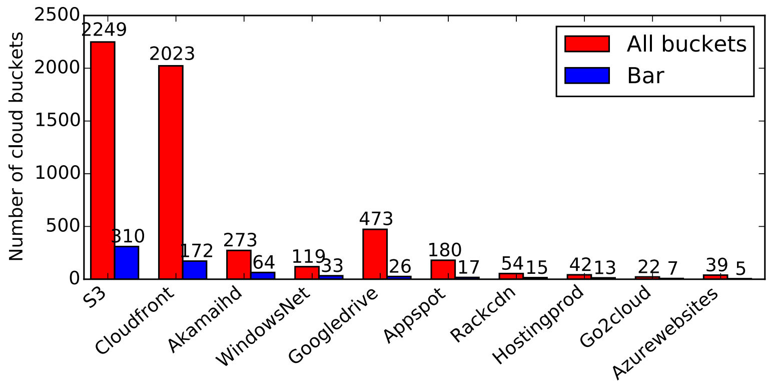


Fig. 5 Top 10 cloud platforms with most Bars, compared with their total number of cloud buckets in our dataset

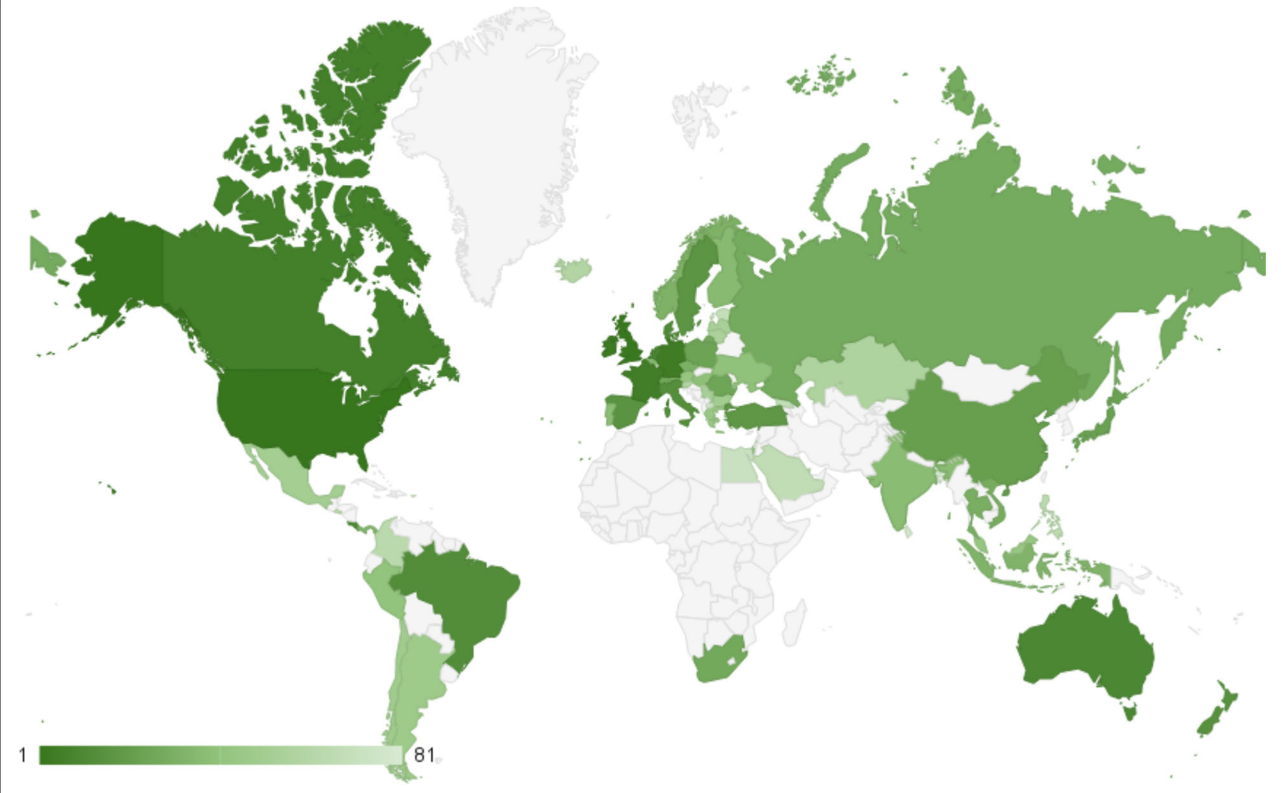


Fig. 6 Impact of Bars' front-end websites around the globe

utilized a word bank based sub-domain generation algorithms (damballa 2015), which concatenates fixed terms and a se- ries of domain names (remove dot), then truncates the string if its length is over 13, e.g., `apismarterpoweru-a` (truncated from `smarterpowerunite.com`). The common patterns of Bars indicate the potential of developing an accurate detection procedure.

Correlation

We further studied the relationships between different Bars, fetched by the same websites. From our dataset, 11,442 (3.5%) websites are found to access at least two Bars. Among them, 8283 were served as front-end websites, and 3159 other sites on redirection chains. Also, 60.9% of these sites link to the repositories on the same cloud platforms and 39.1% use those on different platforms. In some cases, two buckets are often used together. For example, we found that a click-hijacking program was separated into the code part and the configuration part: the former is kept on CloudFront while the latter is on Akamaihd; the two buckets always show up together on redirection chains. Such a separation seems to be done deliberately, in an attempt to evade detection. Also we saw that Bars carrying the same

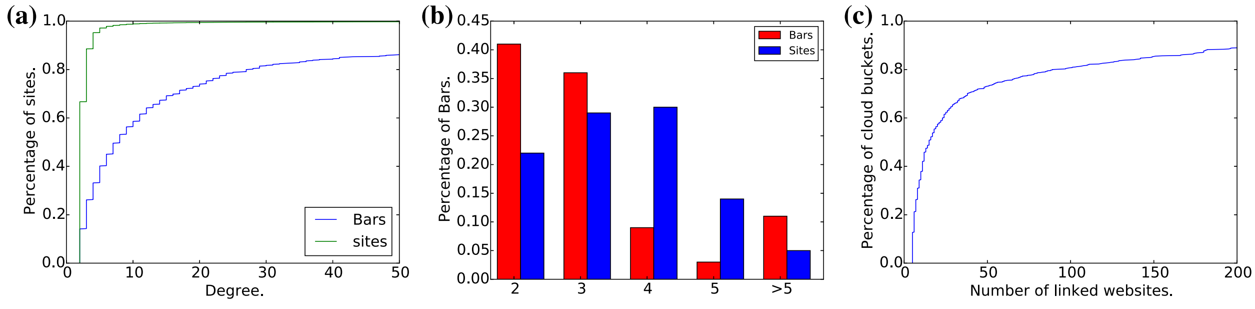


Fig. 7 Bars play critical roles in attack infrastructures. **a** Cumulative distribution of degrees per sites. **b** Percentage of Bars in each position of redirection path number of in-degrees per Bar. (Ignoring those traces with length of 2). **c** Cumulative distribution of position of redirection path number of in-degrees per Bar. (Ignoring those traces with length of 2)

Table 5 Top 10 most popular Bars

Rank	Cloud bucket	# of front-end sites	Avg path len	Popularity
1	s3.amazonaws.com content.sitezoogle.com	4429	2.9	2.8%
2	cloudfront.net d3n8a8pro7vhmx	1829	3.3	1.4%
3	s3.amazonaws.com assets.ngin.com	1643	3.2	1.2%
4	s3.amazonaws.com publisher configurations.shareaholic	1434	2.7	0.9%
5	cloudfront.net d2e48ltfsb5exy	1340	4.0	0.9%
6	cloudfront.net d1t3gia0in9tdj	1297	3.2	0.9%
7	cloudfront.net d2i2wahzwrm1n5	1249	2.5	0.8%
8	cloudfront.net d202m5krfqbpi5	1062	2.8	0.8%
9	s3.amazonaws.com files.enjin.com	1020	7.1	0.7%
10	akamaihd.net cdncache3-a	976	6.4	0.6%

attack vectors are often used together, which are apparently deliberately put there to serve parties of the same interests: as another example, a compromised website was observed to access four different Bars on different cloud platforms, redirecting its visitors to different places for downloading Adware to the visitor's system. Our findings show that Bars are widely deployed in attacks and serve in a complex infrastructure.

Bar-based malicious web infrastructure

Role in attack infrastructures

Actually, most nodes on a malicious infrastructure are the malicious websites with newly registered domains and those that are compromised. To better understand the critical roles of Bars, we compared those nodes with the bad cloud buckets. Specifically, we first identified both types of nodes from the redirection paths and then analyzed the number of unique paths each member in either category is associated with and the position of the member on the path. Figure 7a presents the cumulative distribution of the paths going through a Bar and that of a compromised or malicious site. As seen in the figure, compared with other nodes on the infrastructure, Bars clearly sit on much more paths (47.4 on average vs. 8.6), indicating their importance.

Further, Fig. 7b shows the histogram of position distributions (again, Bars vs. bad sites). The observation is that more Bars (41%, 11%) show up at the beginnings and the ends of the paths than bad websites (22%, 5%), which demonstrates that they often act as first-hop redirectors or attack-payload repositories. For example, in our three-month-long monitoring of the campaign based on the Spyware distribution Bar `akamaihd.net_rvar-a`, we found that besides the Bar, 320 newly-registered websites participated in the attack; here the Bar acted very much like a dispatcher: providing JavaScript that identified the victim's geolocation and then using an iframe to redirect her to a selected bad site.

Attack vectors and payloads

In a malicious infrastructure, the attackers run an attack vector to compromise the victim's systems (browser, client, server, etc.), and then deliver an attack payload (e.g., malware). We found that Bars often serve the infrastructure as repositories for such vectors and payloads. In our study, we ran AV/blacklist scanners and also checked fingerprints collected through manual analysis on the websites in our dataset (Section "BarFinder"). We identify attack vectors using the labels generated by the AV/blacklist scanners whenever available. Figure 8a and b illustrate our findings, in which Spam, Phishing, Fake AV scams and vulnerability exploits are considered to be the attack vectors, and virus, Spyware, Trojan, malicious scripts and other malware-related content are treated as payloads. As we can see from the figure, both types of malicious content are extensively hosted by Bars. On the other hand, malicious or compromised websites typically only serve attack vectors (and retrieve the payload from dedicated servers or the cloud repositories) (Li et al. 2014; Li et al. 2013). Interestingly, we found a malicious payload (CVE-2015-0029) on S3 Bar, which was uploaded in 2013 while the vulnerability is released in February, 2015. The malicious payload can stay active for a relatively long time in Bars.

Revenue estimate

We also investigated the revenues that could be received by the adversary leveraging malicious infrastructure. To this end, we utilized a model as proposed in the prior research (Alrwais et al. 2014; Moore et al. 2011): $R(t) = N_v(t)\ P_a\ R_a$ where the total revenue $R(t)$ during the time period t is calculated from the total number of actions taken (i.e., click-through number, number of visitors $N_v(t)$ weighed by the probability that the visitors take action P_a) and the average revenue per action R_a. Here, we assume the price model as pay per action: that is, the adversary gets paid only when a specified action

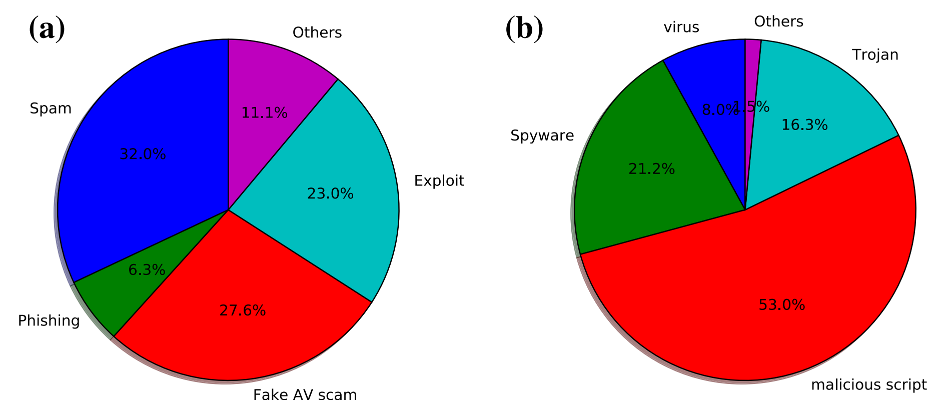


Fig. 8 Distribution of attack vectors and payloads of Bars. **a** Distribution of attack vectors of Bars. **b** Distribution of paypoads of Bars

(e.g., software installation) is taken by the victims. Using a PassiveDNS dataset (DNSDB 2015), which contains DNS lookups recorded by the Security Information Exchange, we estimated the daily number of visitors in February 2015 (data crawling time, i.e., attack time) $N_v = 32$ M. Note that the probability of a visitor taking action P_a is difficult to estimate due to the various malicious pages on different sites. According to the prior works (Alrwais et al. 2014), we set $P_a = 0.02$ and $R_a = \$1.25$. These parameters yield a daily revenue for attackers utilizing Bars of 32 M 0.02 \$1.25 = 0.8 million US dollars per day, which corresponds to a huge amount of illicit profit.

Bucket pollution

Polluted repositories

To find polluted buckets, we searched the Alexa top 20 K websites for the Bars in our dataset and 276 Bars were found. When a legitimate site links to a Bar, the reason might be either the website or the repository is hacked. Differentiating these two situations with certainty is hard, and in some cases, it may not be possible. All we could do is to get an idea about the prevalence of such bucket pollution, based on the intuition that if a website is less vulnerable, then it is less likely to be compromised. To this end, we ran WhatWeb, a powerful web vulnerability scanner, on these sites and found 134 Bar's front-end websites contain various flaws, such as using CMS in vulnerable version (e.g. wordpress 3.9), vulnerable plugins (e.g., JCE Extension 2.0.10) and vulnerable software (e.g., Apache 2.2). The remaining 142 Bar's front-end websites look pretty solid in web protection and therefore it is likely that the Bars they include were polluted. This set of potentially compromised buckets takes 19% of all the Bars flagged by BarFinder. These buckets, together with the additional 30 randomly sampled from the set, went through a manual analysis, which shows that indeed they were legitimate buckets contaminated with malicious content.

Misconfiguration and impact

It is even more challenging to determine how these buckets were compromised, which could be caused by exploiting either the cloud platform vulnerabilities or the bucket misconfigurations. Without an extensive test on the cloud platform and the repositories, which requires at least direct access to them, a comprehensive study on the issue is impossible. Nevertheless, we were able to identify a misconfiguration problem widely existing in popular buckets. *This flaw has never been reported before but was likely known to the underground community and has already been utilized to exploit these repositories. We reported the flaws to the vendors and they confirmed our finding.*

Specifically, on Amazon S3, one can configure the access policies for her bucket to defines which AWS accounts or groups are granted access and the type of access (i.e., list, upload/modify, delete and download): this can be done through specifying access control list on the AWS Management Console. Once this happens, the cloud verifies the content of the `authorization` field within the client's HTTP request header before the requested access is allowed to go through. However, we found that by default, the policy is not in place, and in this case, the cloud only checks *whether the authorization key (*i.e., *access key and secret key) belongs to an S3 user, not the authorized party for this specific bucket*: in other words, anyone, as long as she is a legitimate user of the S3, has the right to upload/modify, delete and list the resources in the bucket and download the content. Note that this does not mean that the bucket can be directly touched through the browser, since it does not put anything into the authorization field. However, the adversary can easily build his own HTTP header, filling in his own S3 key, as illustrated in Fig. 10, to gain access to the misconfigured repository. In our research, we verified that all such operations can be performed on any repositories with the configuration flaw, which suggests that site operators need to take more caution when setting the configuration rules.

To understand the impact of this problem, we developed a simple web testing tool, which checked a bucket's configuration using our own S3 key. By scanning all 6885 repositories (including both Bars and legitimate buckets), we discovered that 472 are vulnerable, which were associated with 1306 front-end websites. The Alexa global ranks and the bounce rates of their front-end websites are illustrated in Fig. 9a and b. Sixty-three percent of them have bounce rates from 20% to 60%; 9 sites are ranked within Alexa top 5000 (e.g., `groupon.com`, `space.com`).

Focusing on the 104 bad buckets with the flaws, we further manually sampled 50 and confirmed that these buckets were indeed legitimate, including high-profile ones like `s3.amazonaws.com_groupon`. Further, looking into the these buckets' file uploading time (retrieved from the buckets through the flaw), we found that in some cases, the attack has been there for six years. Particularly the Amazon bucket `s3.amazonaws.com_groupon`, Groupon's official bucket, was apparently com- promised five times between 2012 and 2015 (see Section "Case Studies" for details), according to the changes to the bucket we observed from the bucket historical dataset we collected from `archive.org`. We also estimated the volume of traffic to those Bar-related sites using a PassiveDNS dataset (DNSDB 2015), which contains DNS lookups recorded by the Security Information Exchange. Figure 9c illustrates the traffic of the websites during the time period when their buckets were compromised, which was increased significantly compared with what those sites received before their compromise, indi- cating that they likely received a lot of visits. This provides evidence that the impact of such compromised buckets is indeed significant.

Case studies

In this section, we discuss two prominent examples.

PUP campaign

Our study reveals a malicious web campaign dubbed *Potentially Unwanted Programs* (PUP) distribution: the attack redirects the victim to an attack page, which shows her fake system diagnosis results or patch requirements through the images fetched from a Bar, in an attempt to cheat the victim into downloading "unwanted programs" such as Spyware, Adware or a virus. This campaign was first discovered in our dataset. Altogether, at least 11 Bars from 3 different cloud platforms and 772 websites (not hosted on the cloud) were involved in.

Through analyzing the redirection traces of the campaign, we found that two Aka- mai Bars, `akamaid.net_cdncache3-a` and `akamaihd_asrv-a`, frequently inject scripts into compromised websites, which serve as first-hop redirectors to move a visitor down the redirection chain before hitting malicious landing pages (that serve malicious content). Interestingly, all the follow-up redirectors are compromised or malicious websites that are not hosted on the cloud. The scripts in the Bars were found to change over time, redirecting the visitor to different next-hop sites (also redirectors). On average, the life span of such sites is only 120 h, but the Bar was still alive when we submitted this paper. Such redirections end at at least 216 mali- cious landing sites, which all retrieve deceptive images from an Amazon S3 bucket `s3.amazonaws.com_cicloudfront` (a Bar never reported before and is still alive). An example is a system update warning, as shown in Fig. 1. From the reposi- tory, we collected 134 images, including those for free software installation, updates on all mainstream OSes, browsers and some popular applications. If she clicks and downloads the program promoted on the site, the code will be fetched from multiple Bars, such as `s3.amazonaws.com_wbt_media` where the PUP puts a Bitcoin miner on the victim's system, and `cloudfront.net_d12mrm7igk59vq`, whose program modifies Chrome's security setting.

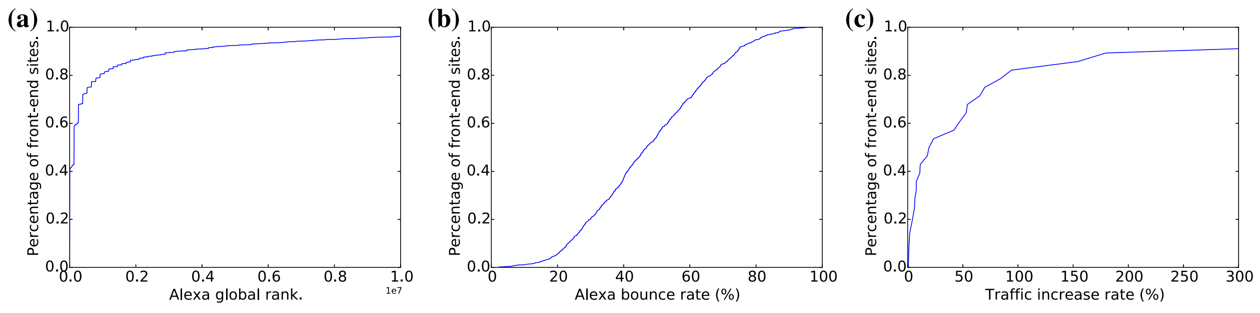


Fig. 9 Alexa global rank, bounce rate and traffic increase rate of Bar's front-end websites. **a** Cumulative distribution of Alexa global ranks per Bar's front-end sites. **b** Cumulative distribution of Alexa global ranks per Bars'Alexa bounce rate. **c** Cumulative distribution of Bar's traffic increase rate per Bar's front-end sites

```
GET /?delimiter=/ HTTP/1.1
Host: (bucket-name).s3.amazonaws.com
Accept-Encoding: identity
content-length: 0
Authorization: AWS (access key):(secret key)
```

Fig. 10 Constructed request header

Groupon Bar

We discovered that a misconfigured Amazon S3 bucket `s3.amazonaws. com_groupon` belongs to Groupon (Alexa global rank 265) (Fig. 10), a global e-commerce marketplace serving 48.1 million customers worldwide. The bucket was used as the resource repository for Groupon's official website (i.e., `groupon.com`) as well as its marketing sites (12 websites observed in our dataset). When tracking its historical content from `archive.org`, we were surprised to see that the Groupon S3 bucket has been compromised at least eight times in the past five years (e.g., 2015/08/06, 2014/12/18, 2014/06/25, 2014/01/27, 2014/02/26, 2013/06/23, 2011/11/08, 2010/09/28). These attacks caused different types of malicious payloads to be uploaded to their repository, including Adware, Trojan, virus and others. Even though the bucket owner changed the access control policy in 2012 to prevent the unauthorized party from directly listing the bucket content through browser, it remained accessible by our tool mentioned in Section "Bucket Pollution", which constructs an *Authorization* field in HTTP header, and unauthorized listing, upload and even modification can still occur.

Discussion

Limitations of BarFinder

As mentioned earlier, Bar detection is hard, since cloud repositories cannot be directly accessed by the parties outside the cloud. Therefore, the goal of BarFinder is to leverage the sites served by Bars to find suspicious repositories. For this purpose, we chose to utilize the *collective* features of these sites, such as their topological relations, content shared across sites, etc. This strategy could make the approach more robust, as the collective features are more difficult to evade compared with those from individual sites. On the other hand, it requires that the party running the system first makes efforts to gather the sites using cloud buckets, the more the better. Further, there are repositories that only serve a small set of front-end sites: e.g., we found that among the Alexa top 3 K sites, 67 sites are connecting to the cloud buckets only used by themselves. Those "self-serving" buckets are rather popular in reputable websites such as `appspot.-com_android-site` only used by `android.com`, `s3.amazonaws.com_ttv-backgroundart` only used by `twitch.tv`, etc. This fact makes the bad apples among them hard to catch by BarFinder simply because not enough sites using them are out there to allow us to differentiate these two types of repositories. Detection techniques covering this type of Bars need to be developed in the follow-up research.

Other defenses against bars

Besides the detection effort made by the third party, as BarFinder does, more can be done to mitigate the threats posed by Bars, from the ends of the website owner, the bucket owner and the service provider. The website owner could perform integrity checks on the resources her website retrieves from the bucket, making sure that it is not compromised. The cloud bucket owner should carefully configure her cloud bucket to avoid the issue we found and other misconfiguration flaws. In this case, an automatic configuration checker could be helpful. Most im- portantly, the cloud provider does have the responsibility to move more aggressively on detecting and removal of Bars from their systems. This, however, is non-trivial, given the privacy concern and the fact that some Bars can only be considered to be malicious by looking at the malicious activities they are involved in, such as those hosting Phishing pictures. Further research is needed to better understand what the provider can do to address the issue.

Ethical issues

Most findings of the paper were made through analyzing the data crawled from the public domain. Regarding the study on the misconfiguration problem we found, our scanner was designed to minimize the privacy impacts on vulnerable repositories: specifically, it only tried out the functionality like file listing, uploading and downloading. The impact of such operations are very much in line with those of running online web testing tools (e.g., Sucuri (Sucuri 2015)) on others' websites. Most importantly, we did this with the full intention to protect such repositories from future exploits, and also carefully avoided changing any existing content there and deleted from our system all the files downloaded. Further, we have already contacted the major vendors such as Groupon and the cloud providers like Amazon about those security breaches, and will continue to notify others and help them fix the configuration problem. So far, Groupon has acknowledged the importance of our findings and expressed gratitude for our help.

Related work

Cybercrime hosting service

The cybercrime hosting infrastructure is a basic building block of the cyber-crime ecosystem. It provides servers and networking infrastructure for the cyber criminals, and it also persist in the face of takedown attempts and complaints of illicit activities. Alrwais et al. (Alrwais et al.

2014) investigated a trending and stealthy cybercrime hosting service, that malicious activities were hosted on the sub-allocations of legitimate service provider Networks. Li et al. (Li et al. 2013) also found that Internet Service Provider (ISP) were used for abusive hosting. Compared with those prior studies, we shed light on the emerging and evasive cybercrime hosting platform: cloud service, which has never been studied before.

Bad site detection

Malicious web activities have been extensively studied (Invernizzi et al. 2012; Invernizzi et al. 2014; Moore et al. 2011; Nelms et al. 2015). Most related to our work here is the use of HTML content and redirection paths to detect malicious or compromised websites. Examples for the content-based detection include a DOM-based clustering systems for monitoring Scam websites (Der et al. 2014), clas- sification of websites for predicting whether some of them will turn bad based on the features extracted from HTML sources, and a monitoring mechanism (Borgolte et al. 2013) (called Delta) to keep track of the changes made to the content of a website for detecting script-injection campaigns. For those using malicious redirection paths, prominent prior approaches use short redirection sequences to capture bad sites (Li et al. 2014), unique properties of malicious infrastructure (its density) for detecting drive-by downloads (Invernizzi et al. 2014) or malware distribution (Stringhini et al. 2013) and a trace-back mechanism that goes through the redi- rection paths (Nelms et al. 2015) for labeling malware download sites. Compared with those prior studies, which all rely on the properties of the targets they try to capture, BarFinder utilizes the features found from the front-end websites using cloud buckets, as those repositories may not be directly accessible. Also, our approach leverages a set of unique collective features, based on the connected components of a graph, which, to our knowledge, has never been used before.

Cloud security

Previous studies on security and privacy issues in cloud storage primarily focus on the confidentiality of the data stored in the cloud or the attacks targeting the cloud computing infrastructure. Examples include the study on co- locating attack virtual machines (VM) with the target one on Amazon EC2 (Ristenpart et al. 2009), which enables a cache-based side-channel attack to infer sensitive user information from the target (Zhang et al. 2014), and the work on controlled-channel attacks on multi-user cloud hosting environment (Xu et al. 2015), which allows an untrusted VM to extract sensitive information from protected applications. More recently, attention has moved to abuse of cloud- based services for fraudulent activities. For example, prior research (Mulazzani et al. n.d.) analyzed Dropbox client software and discovered that it can be exploited to hide files with unlimited storage capacity. Additionally, (Han et al. 2015) studied the use of the Amazon EC2 to host malicious domains acting as command and control centers, exploit servers by downloading malware samples and executing them in sandbox environments to analyze their interactions with the cloud. (Liao et al. 2016) studied the long-tail SEO spam on cloud platforms and measured its effectiveness. Our study differs from these works by proposing BarFinder to identify malicious cloud repositories and provide an in-depth analysis of the use of cloud repositories in malicious campaigns and how they correlate with the websites they serve. In another study, researchers (Idziorek et al. 2011) inspected the fraudulent traffic to cloud-hosted pages for the purpose of squandering the user's resources and raising her cloud-usage cost. They also proposed detection methods based on the consistency of the requests. Unlike these works, our research investigated the abuse of cloud bucket as a malicious service, an emerging new cloud-based security threat that has never been studied before.

Conclusion

The emergence of using cloud repositories as a malicious service presents a new challenge to web security. This new threat, however, has not been extensively studied and little is known about its scope and magnitude and the techniques the adversary employs. In this paper, we report the first systematic study on malicious and com- promised cloud repositories and the illicit online activities built around them. We collected a small set of seeding Bars and identified a set of collective features from the websites connecting to them. These features describe the effort made by the adversary to protect Bars and utilize them to quickly build up a large campaign. Using these features, we developed a new scanner that detected over 600 Bars on top-of-the-line cloud platforms, including Google, Amazon, and others. Over these Bars, we per- formed a large-scale measurement study that led to surprising findings of such attacks. Examples include the central roles those buckets play at each stage of a web attack (redirection, displaying Phishing content, exploits, attack payload delivery, etc.), the strategy to separate malware code and configuration files to avoid detection, and a configuration flaw never reported before that was likely exploited to compromise many cloud buckets. Our findings made an important step toward better understanding and effective mitigating of this new security threat.

Endnotes

[1]We have manually examined and confirmed all those instances.

[2]The terms repositories and buckets are used interchangeably throughout this paper.

Appendix

Table 6 A List of cloud hosting platforms

Cloud Platform	Domain
heroku	herokuapp.com
amazon S3	s3.amazonaws.com
cloudfront	cloudfront.net
windowsnet	windows.net
azure	azurewebsites.net
google	googledrive.com
appspot	appspot.com
msecdn	msecdn.net
bitbucket	bitbucket.org
github	github.io
sina	sinaapp.com
olympe	olympe.in
rackcdn	rackcdn.com
baiduyun	duapp.com
qiniu	qiniucdn.com
akamaihd	akamaihd.net
yahoo	hostingprod.com
sogo	sogoucdn.com
go2cloud	go2cloud.org
aliyun	aliyuncs.com

Acknowledgements
We thank our anonymous reviewers for their useful comments.

Funding
This work was supported in part by the National Science Foundation (grants CNS-1223477, 1223495, 1527141 and 1618493);

Availability of data and materials
All public dataset sources are as described in the paper.

Authors' contributions
XL: design and experiment; KY, LX: experiment; SA, XW, SH, RB: design. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Received: 24 May 2018 Accepted: 10 September 2018
Published online: 11 October 2018

References

Hao, Shuang, et al. (2013) Understanding the domain registration behavior of spammers. Proceedings of the 2013 conference on Internet measurement conference. ACM

Alrwais S, Yuan K, Alowaisheq E, Li Z, Wang X (2014) Understanding the dark side of domain parking. In: Proceedings of the 23rd USENIX security symposium

Bishop CM (2006) Pattern recognition and machine learning. springer

Borgolte K, Kruegel C, Vigna G (2013) Delta: automatic identification of unknown web-based infection campaigns. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, ACM, pp 109–120

Buckets n.d.. https://cloud.google.com/storage/docs/json_api/v1/buckets. Accessed Aug 2018. [On- line]

BuiltWith. Builtwith. http://builtwith.com/, 2015. Accessed Aug 2018. [Online]

Clean-MX. Clean mx realtime database. http://support.clean-mx.de/clean-mx/viruses.php, 2015. Accessed Aug 2018. [Online]

Cohen W, Ravikumar P, Fienberg S (2003) A comparison of string metrics for matching names and records. In: Proceedings of Kdd workshop on data cleaning and object consolidation

C. Crawl. Common crawl. https://commoncrawl.org/, 2015. Accessed Aug 2018. [Online]

damballa. Dgas in the hands of cyber-criminals:examining the state of the art in malware evasion techniques. https://www.damballa.com/downloads/r_pubs/WP_DGAs-in-the-Hands-of-Cyber-Criminals.pdf; 2015. Accessed Aug 2018. [Online]

Der MF, Saul LK, Savage S, Voelker GM (2014) Knock it off: profiling the online storefronts of counterfeit merchandise. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 1759–1768

DNSDB. Passivedns. https://www.dnsdb.info/, 2015. Accessed Aug 2018. [Online]

Google. Google hosted libraries. https://developers.google.com/speed/libraries/?csw=1, 2015. Accessed Aug 2018. [Online]

Google. Publish website content. https://developers.google.com/drive/web/publish-site, 2015. Accessed Aug 2018. [Online]

Han X, Kheir N, Balzarotti D (2015) The role of cloud services in malicious software: trends and insights. In: DIMVA 2015, 12th Conference on Detection of Intrusions and Malware & Vulnerability Assessment, July 9–10, 2015, Milan, Italy, Milan

Idziorek J, Tannian M, Jacobson D (2011) Detecting fraudulent use of cloud resources. In: Proc. 3rd ACM workshop on cloud computing security workshop, Chicago

Invernizzi L, Comparetti PM, Benvenuti S, Kruegel C, Cova M, Vigna G (2012) Evilseed: A guided approach to finding malicious web pages. In: Security and Privacy (SP), 2012 IEEE Symposium on. IEEE, pp 428–442

Invernizzi L, Miskovic S, Torres R, Saha S, Lee S, Mellia M, Kruegel C, Vigna G (2014) Nazca: detecting malware distribution in large-scale networks. In: Proceedings of the Network and Distributed System Security Symposium (NDSS)

Li Z, Alrwais S, Wang X, Alowaisheq E (2014) Hunting the red fox online: Understanding and detection of mass redirect-script injections. In: Security and Privacy (SP), 2014 IEEE Symposium on. IEEE, pp 3–18

Li Z, Alrwais S, Xie Y, Yu F, Wang X (2013) Finding the linchpins of the dark web: a study on topologically dedicated hosts on malicious web infrastructures. In: Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, pp 112–126

Liao X, Liu C, Mccoy D, Shi E, Beyah R (2016) Characterizing long-tail seo spam on cloud web hosting services. In: Proceedings of the International World Wide Web Conference

Moore T, Leontiadis N, Christin N (2011) Fashion crimes: trending-term exploitation on the web. In: Proceedings of the 18th ACM conference on Computer and communications security. ACM, pp 455–466

Mulazzani M, Schrittwieser S, Leithner M, Huber M Dark Clouds on the Horizon: Using cloud storage as attack vector and online slack space. In: Proc. 20th USENIX security symposium, San Francisco, p 2011

Nelms T, Perdisci R, Antonakakis M, Ahamad M (2015) Webwitness: Investigating, categorizing, and mitigating malware download paths. In: 24th USENIX Security Symposium (USENIX Security 15). USENIX Association, Washington, D.C., pp 1025–1040

Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM conference on Computer and communications security. ACM, pp 199–212

Scipy. scipy.cluster.hierarchy.linkage. http://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html, 2015. Accessed Aug 2018. [Online]

Servnet n.d.. https://servnetshsztndci.onion. Accessed Aug 2018. [Online]

Sklearn. sklearn.svm.svc. http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, 2015. Accessed Aug 2018. [Online]

Snort. Snort ssl and tls. http://manual.snort.org/node147.html, 2015. Accessed Aug 2018. [Online]

solutionary. Threat-intelligence. https://www.solutionary.com/_assets/pdf/research/sert-q4-2013-threat-intelligence.pdf, 2015. Accessed Aug 2018. [Online]

Stringhini G, Kruegel C, Vigna G (2013) Shady paths: Leveraging surfing crowds to detect malicious web pages. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, pp 133–144

Sucuri. Sucuri. https://sucuri.net/, 2015. Accessed Aug 2018. [Online]

Symantec. The future of ids. http://www.symantec.com/connect/articles/future-ids, 2015. Accessed Aug 2018. [Online]

VirusTotal. Virustotal. https://www.virustotal.com/, 2015. Accessed Aug 2018. [Online]

WhatWeb. Whatweb. http://www.morningstarsecurity.com/research/whatweb, 2015. Accessed Aug 2018. [Online]

Xu Y, Cui W, Peinado M (2015) Controlled-channel attacks: deterministic side channels for untrusted operating systems. In: Proceedings of the 36th IEEE Symposium on Security and Privacy (Oakland). IEEE Institute of Electrical and Electronics Engineers

Y. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. Cross-tenant Side-Channel attacks in PaaS clouds. In Proc. 21st Conference on Computer and Communications Security (CCS), Scottsdale, 2014