Investigating Web Defacement Campaigns at Large

Federico Maggi, Marco Balduzzi, Ryan Flores, Lion Gu, Vincenzo Ciancaglini
Forward-Looking Threat Research Team - Trend Micro, Inc.

ABSTRACT

Website defacement is the practice of altering the web pages of a website after its compromise. The altered pages, called *deface pages*, can negatively affect the reputation and business of the victim site. Previous research has focused primarily on detection, rather than exploring the defacement phenomenon in depth. While investigating several defacements, we observed that the artifacts left by the defacers allow an expert analyst to investigate the actors' modus operandi and social structure, and expand from the single deface page to a group of related defacements (i.e., a *campaign*). However, manually performing such analysis on millions of incidents is tedious, and poses scalability challenges. From these observations, we propose an automated approach that efficiently builds intelligence information out of raw deface pages. Our approach streamlines the analysts job by automatically recognizing defacement campaigns, and assigning meaningful textual labels to them. Applied to a comprehensive dataset of 13 million defacement records, from Jan. 1998 to Sep. 2016, our approach allowed us to conduct the first large-scale measurement on web defacement campaigns. In addition, our approach is meant to be adopted operationally by analysts to identify live campaigns in the real world.

We go beyond confirming anecdotal evidence. We analyze the social structure of modern defacers, which includes lone individuals as well as actors that cooperate with each others, or with teams, which evolve over time and dominate the scene. We conclude by drawing a parallel between the time line of World-shaping events and defacement campaigns, representing the evolution of the interests and orientation of modern defacers.

ACM Reference Format:
Federico Maggi, Marco Balduzzi, Ryan Flores, Lion Gu, Vincenzo Ciancaglini. 2018. Investigating Web Defacement Campaigns at Large. In *ASIA CCS '18: 2018 ACM Asia Conference on Computer and Communications Security, June 4–8, 2018, Incheon, Republic of Korea.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3196494.3196542

1 INTRODUCTION

Website defacement, or simply *defacement*, is the practice of visibly altering one or more web pages of a website upon compromising it. The intent of the actor, so-called *defacer* in this context, is to arbitrarily change or replace the original content of victim website to advertise the success of her compromise. The resulting page, called *deface page*, may contain information on the motive behind the attack, team affiliation of the defacer(s), or nicknames of the supporting actors. Over the years, defacers have abandoned their interested in defacing for the mere purpose of advertising the compromise, pursuing defacement more as a mean to broadcast strong messages "to the World"—by compromising popular websites.

Despite several actors are still driven by the desire of promoting their own reputation, an increasing number of defacers strive instead to promote their ideologies, religious orientation, political views, or other forms of activism, often closely following real-world events (e.g., war, elections, crisis, terrorist attacks). We refer to this phenomenon as *dark* propaganda, to highlight that legitimate resources are abused for pushing the actors' viewpoints. For example, in 2013–2014 "Team System Dz" defaced over 2,800 websites, and planted pro-ISIL/ISIS[1] content to protest against the US military action in the Syrian civil war[2]. In one of these incidents, the actors replaced the homepage of the British Rugby League club team (Keighley Cougars) with disturbing, war-related pictures, along with the message "I love you ISIS". In Jan 2014, when the NSA was found exploiting Angry Bird for mass-surveillance purposes[3], a group of defacers modified the homepage of Rovio's Angry Birds, and placed the NSA logo modified with the text "Spying Birds".

The inappropriate or offensive content placed by the defacers affect the reputation and continuity of the legitimate businesses behind the targeted websites, making these campaigns not as innocuous as they appear. In response to this, past research has proposed monitoring solutions to *detect* whenever defacement content is planted on a website [3, 5, 4, 2]. Although these systems are operationally useful, they do not provide the analysts with in-depth knowledge to help them *understand* and follow the web-defacement phenomenon from an investigation standpoint. For example, the detection method [2] uses the most accurate yet very *opaque* machine-learning technique—convolutional neural networks trained via deep learning. Thus, despite yielding excellent detection rates, the outcome is not very helpful to the analyst who needs to *track* the actors behind these attacks, or reconstruct a campaign of defacements.

We observe a lack of methodologies to analyze, understand, and track web defacements at large. Previous work along this research line relied on metadata alone (e.g., reason for the attack, vulnerability used, nickname of the attacker). Such metadata is spontaneously provided by the attacker and thus should *not* be considered trustworthy. The only researches that have inspected the *content* of the planted page are either outdated ([10] is from 2004, and the defacement phenomenon have evolved since then) or limited to manual inspection of a handful of pages [8]. We conclude that there is a need for a comprehensive and large-scale analysis approach, especially given the availability of datasets spanning over 19 years, which so far have been used only for detection purposes.

ASIA CCS '18, June 4–8, 2018, Incheon, Republic of Korea

ACM ISBN 978-1-4503-5576-6/18/06…$15.00
https://doi.org/10.1145/3196494.3196542

[1] Islamic State of Iraq and the Levant (ISIL) / and Syria (ISIS)
[2] https://kevin.borgolte.me/notes/team-system-dz-isis-isil-defacement-campaign/
[3] http://www.bbc.com/news/technology-25949341

To fill this gap, we take a data-driven approach that supports the analyst at exploring and visualizing the data, eliciting the relevant web-defacement campaigns. To this end, we identify a set of core characteristics that "translate" the unstructured content planted by the defacers (i.e. web page and linked resources) into useful features. Using such features to represent a (large) dataset, we feed a data-analysis and machine-learning pipeline that leverages scalable clustering techniques to automatically group related defacement records into campaigns. We then label these campaigns in a convenient and human-friendly representation that the analyst can easily use for classic data-exploration tasks (e.g., graphing, plotting, drilling, pivoting).

Not only the outcome of our approach leads to the detection of relevant campaigns, but sheds light on the actors' modus operandi, social structure and organization, and allows flexible tracking and investigation of defacements. Operationally, analysts can use the results to provide early warning (e.g., sites that are more prone to be defaced), understand how this happens (given the knowledge of upcoming geo-political events such as elections), and ex-post investigation.

Overall, the contributions of our work are:

- We conduct a measurement on a comprehensive dataset of 13 million defacement records (Jan 1998 to Sep 2016);
- We introduce a novel approach that builds intelligence information to identify defacement campaigns, out of raw deface pages;
- We show how our approach empowers the analyst in understanding modern defacements, including the social structure of the actors, their modus operandi and motive;
- Through real-world cases, we show how defacement is leveraged for dark propaganda purposes, for example in support of specific religious or political ideologies.

2 DEFACEMENT DATASET

This paper is based on a dataset comprising a unique collection of defacement records from 5 major reporting sites, as summarized in Table 1. These reporting sites provide feeds of defacement records, aggregated from various sources such as sharing initiatives, CERTs, or victims. Often, the defacers themselves voluntarily submit their defacements, to advertise and "show off" their mischief. The defacers are generally interested in submitting correct data, to the extent that certain web-exploitation kits include automatic submission routines that notify the reporting sites automatically upon successful execution of the payload[4]. Despite the maintainers of popular sites such as Zone-H strive to manually validate each submitted defacement, no strong assurance that the data is free from (deliberate or incidental) errors.

As Table 2 shows, each defacement record consists of metadata (e.g., timestamp of the deface event, target URL) and raw content (e.g., the planted HTML or multi-media content). Moreover, to enable our analysis, we derive additional attributes such as the category of the defaced site (e.g., News, Media), the TLD, and so on.

Zone-H is the largest and richest archive, because of its popularity and because it receives cross-submitted contributions. Therefore, we decided to purchase a copy of the dataset as of Sep 2016, along with a full snapshot of the deface pages captured at the time the defacement was reported[5]. We complement this dataset by crawling the other reporting sites listed in Table 1.

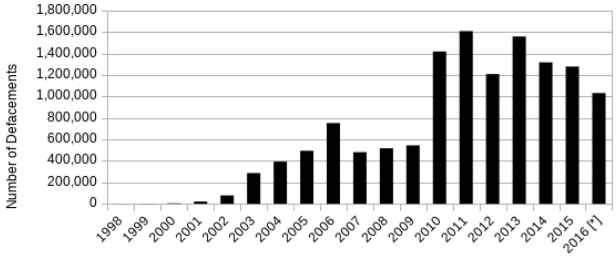


Figure 1: Records per year from Jan 1998 to Sep 2016.

As shown in Figure 1, our collection spans over almost 19 years, from Jan 1998 to Sep 2016. During this time frame, the number of reported incidents per year grew from few thousands to more than one million, showing the increasing importance of the phenomenon explored in this paper.

2.1 Metadata vs. Content

Given the heterogeneous and possibly unknown origins of this data, these feeds are to be used cautiously, especially in operational environments, because the risk of false or misleading information is quite high. Indeed, as highlighted in Table 2, there are various degrees of trustworthiness for each data attribute. Although throughout this paper we may use metadata attributes to draw statistics or trends, the core of our analysis is based exclusively on the actual content planted by the attacker on the deface pages (e.g., HTML text, images, URLs, other linked resources, both internal and external), which is the most reliable data. We refer to a set of deface pages setup by a group of actors—with a given goal in mind—to as *campaign*. In the next section we will present the concepts needed to better define a campaign. Except for the timestamp, which the attacker "cannot" forge[6], we ignore all low-trustworthiness metadata, like the defacer's declared nickname, target webserver, exploited vulnerability, reason for hacking, and so on. This is one of the core differences between our approach and previous work [8].

Source	Site URL	#Records
Zone-H	www.zone-h.org	12,303,240
Hack-CN	www.hack-cn.com	386,705
Mirror Zone	www.mirror.zone.org	195,398
Hack Mirror	www.hack-mirror.com	68,980
MyDeface	www.mydeface.com	37,843
Total		12,992,166

Table 1: # of Records per Reporting Site

[4]https://gist.github.com/dreadpiratesr/798b21f2aa88bc651803

[5]The entire dataset takes about 1GB for metadata (in CSV) and slightly less than 1TB for the deface pages.

[6]Note that the attacker might still wait (a long time) before submitting a defacement incident, but this would be against her interest. Moreover, we are not really trusting the accuracy of the timestamp, nor we are using it as a feature.

Type	Attribute	Example	Description	Trustworthiness	Explanation
Metadata (~1GB)	Timestamp	1998-01-02T15:14:12+00:00	Time of reported defacement incident	Medium-high	The attacker will get poor visibility by forging this datum
	Nickname	Team CodeZero	Pseudonym of the attacker or submitter	Low-medium	The attacker has no interest in but can forge this datum
	URL	http://janet-jackson.com/	URL of the planted deface page	High	The attacker has no interest in forging this datum and can be verified by the submission site
	Webserver	Nginx	Name and brand of the webserver running at the time of defacement	Low	The attacker or submitter can forge it at no cost
	Reason	Political reasons	Motive of the defacement	Low	The attacker or submitter can forge it at no cost
	Hack mode	SQL-Injection	Vulnerability used to enable the upload of the defacement content	Low	The attacker or submitter can forge it at no cost
Raw content (~1TB)	Main page	HTML or TXT file	File storing the source code of the main defaced content (at the given URL)	High	The planted content is the subject of the defacement
	Embedded resources	Various formats	Images or other web resources linked by the defaced page and hosted within the same compromised web server	High	Collected as part of the main page by the submission site
	External resources	Various formats	External resources used in the main page	Medium-high	Can change over time or become unavailable

Table 2: Metadata and raw content available in our dataset, along with a description of the trustworthiness of each attribute.

2.2 Overall Statistics and Trends

We hereby provide an overview of the key statistics and trends that we observed in our dataset, before running our automated analysis.

2.2.1 Topics Over the Years. To observe the evolution of the messages left by the defacers, we use an off-the-shelf machine-learning technique called *topic modeling*, which is widely used in news classification to determine the subject of a written story (e.g., politics, technology, finance).

For the scope of this section, a topic-modeling algorithm can "fit" a large corpus of documents (i.e., our deface pages) to an arbitrarily small set of high-level concepts. We use the latent-semantic analysis technique [1], which assumes that words that are close in meaning will also occur in similar documents.

Year	Most relevant topics
1998	question, student, **security**, number, place
1999	cowboy, *team*, **security**, think
2000	baby, tabloid, people, provided
2001	lord, prime, provided, saved, better
2002	worry, sind, **lame**, care, **encryption**
2003	**backup**, gift, *team*, came, take
2004	best, *group*, micro, look, total
2005	normal, **pope**, time, familia, contact
2006	**terror**, saved, intruder, energy, user
2007	badger, since, high, **turk**, **turkey**
2008	*crew*, speech, warning, saved, *team*
2009	knowledge, acker, *team*, album, **country**
2010	posted, member, protocol, kernel, security
2011	contact, security, village, holding, highlander
2012	saved, contact, *team*, underground
2013	*team*, forgive, security
2014	eagle, *crew*, electronic
2015	clash, king, **terrorism**, visit, alligator
2016	**marocain**, **turk**, steel, anonymous, *team*

Table 3: Most relevant topics each year.

In practice, after extracting the text from each of the planted web pages, we remove stop words[7], words containing non-ASCII letters, and any occurrence of team or defacer nicknames (obtained from the metadata), because they do not contribute to defining the overall message. We then feed the topic-modeling algorithm with the entire collection of pages.

Focusing for simplicity on the English language, we notice from Table 3 that early defacements (e.g., 1998–2004) are primarily focused on exposing the scarce security of the target web site (e.g., 'security', 'backup', 'lame'), which made the defacement possible. From 2005 on we notice an interesting shift, with terms such as 'pope', 'terror', 'country', 'marocain', 'turk'. Although we cannot draw any strong conclusion yet, we understand that modern defacers are all but disconnected from real-world events, and seem to use defacement to express their thoughts. The most striking examples are the papal conclave of 2005, the Turkish general election in 2007, the Moroccan general election in 2016, and the coup d'état attempted in Turkey in 2016. In Section 5.3 we show how our analysis approach allows to easily draw these parallels.

In addition to highlighting the scarce level of security, another common theme throughout the years is the sense of "belonging" of the defacers to a team, as expressed by words like 'team', 'crew' or 'group'. In Section 6.1 we show how our analysis approach allows to easily explore the social structure of the defacers, which resembles that of gangs, characterized by strong, longstanding relationships.

2.2.2 Targets Over the Years. Excluding generic top-level domains (gTLDs)[8], the top TLDs are .com.br (4%), .de (3.5%), .co.uk (3.2%), .nl (2.5%), .it (2.3%), and .ru (2.2%). Interestingly, looking at the yearly breakdown (Table 14 in the Appendices), in 2002–2009 German websites (.de) were the main target, while recently (2010–2016) defacers seem to find most targets in Brazilian (.com.br) and Russian (.ru) web sites. Note that, the choice of the target could be guided by the type of message (e.g., attacking a national or .gov website to protest against the government), or simply by chance (e.g., small websites are more vulnerable). Indeed, most of the targets are popular web applications (WordPress, Joomla, and Drupal),

[7] From the comprehensive list at https://github.com/igorbrigadir/stopwords
[8] .com, .org, .net, .edu, .gov, .mil

frequently targeted by automated exploitation scripts whenever a new vulnerability is found. Unsurprisingly, the most targeted platforms are Linux (72%), with Windows 2003 (12.3%) and 2000 (3%) being the second and third most popular OSs. Despite this breakdown is derived from a low-trustworthiness attribute, these values are aligned with server-OS usage statistics [9].

2.2.3 Malicious Content. As Figure 2 shows, there is an increasing trend in the adoption of malicious client-side content (e.g., JavaScript) in deface pages. To draw this trend, we processed our dataset with Trend Micro's Site Safety Center[9], which can detect a wide variety of web threats, including obfuscated and complex payloads.

Given the historical nature of our dataset, and that deface pages are ephemeral, we can only speculate on the reasons behind this trend. The attackers may have used an already-malicious page, or might have been targeted with such malicious payload, or maybe they wanted to monetize their deface page by selling so-called "installation services" to third parties. Investigating the reasons behind this practice is beyond the scope of this paper.

2.3 Key Observations

We conclude the section with two key observations that we use as foundations for building our system and measurements.

2.3.1 Teams and Campaigns. According to the psychological analysis conducted in 2001 by Woo et al. [10], web defacers cooperate in teams. Especially if driven by strong ideologies, defacers are not lone wolves, but their modus operandi resemble that of well-organized cyber gangs acting in a coordinated fashion.

After manually inspecting several thousands of deface pages, we confirm that modern defacers tend to be affiliated in teams and by no means act as "script kiddies". In addition, we found out that they organize their defacements into *campaigns*, which involve multiple target sites and can be repeated over time. Each group may conduct multiple campaigns, and one campaign may be supported by multiple (partnered) groups.

Practically, we found out that the names of the teams as well as their members appear in the content of deface pages. We take these observations into account, such that our system can automatically map the relationships between campaigns, teams, and actors.

2.3.2 Deface Campaign Templates. Because of how these teams operate, the resulting deface pages often present notable characteristics. In practice, when a team prepares and runs a campaign, it

[9]https://global.sitesafety.trendmicro.com/

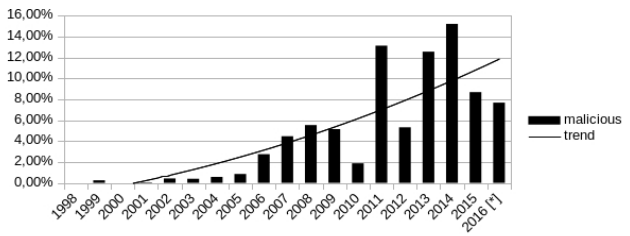


Figure 2: Adoption of malicious content in deface pages.

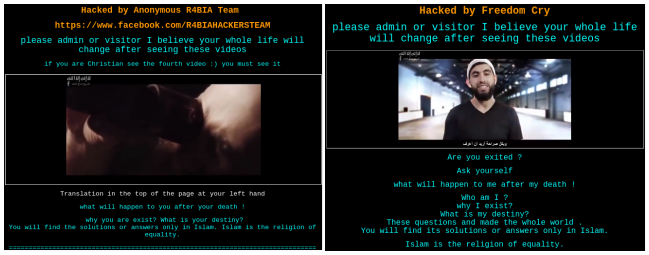


Figure 3: Two deface pages based a common template.

tends to re-use a common template that each member can personalize based on the target or other factors. For example, the two pages shown in Figure 3 look similar to the human eye, because they both have a black background, turquoise text, orange header, and 4 embedded videos[10]. Moreover, although not (always) visible to the human eye, these two pages use the same character encoding (western). Upon manual inspection, we conclude that the actor `Freedom Cry` (on the right-hand side) adopted a template, supposedly created by the `Anonymous R4BIA TEAM` group (on the left-hand side), and applied slight personalization.

Although the 'how' and 'why' are out of the scope of this work, we cannot exclude that novice actors spontaneously copy and reuse a deface pages taken from existing defacements, maybe to show their will to be part of, or to glorify, a team.

Regardless of why, how, and to what extent a page is personalized, our key observation is that deface pages within the same campaign are very similar to each other, if not identical. This is a strong attribution indicator, which allows the analyst to group them together and understand the relationships between teams and actors. This indicator is the foundation of our approach for automated campaign detection and tracking.

From hereinafter, we use the term *campaign template* (or, simply, template) to indicate the content (e.g., bits of text, color scheme, language, character encoding) that is common to most of the pages within a campaign, which in turn can be leveraged to recognize and identify each campaign.

However, performing this attribution manually is tedious and extremely time consuming.

3 AUTOMATED ANALYSIS APPROACH

The previous section suggested that defacers are organized and act in a way that leaves visible traces of their modus operandi in their defacements. In this section we describe how we leverage such traces in order to analyze millions of deface pages automatically to find groups of similar ones, where the term "similar" indicates that they belong to the same campaign.

Despite finding commonalities between web pages is a generic and well-researched problem, deface pages from the same campaign may still differ substantially. Therefore, the concept of "page similarity" is more brittle than in classic information-retrieval settings.

Overview. Our approach is summarized in Figure 4.

The first phase, called *Deface Page Analysis*, extracts the raw content from the deface pages, both dynamically (in a browser) and statically (from the files on disk). From this content (e.g., rendered

[10]One is shown, 3 are at the bottom of the page

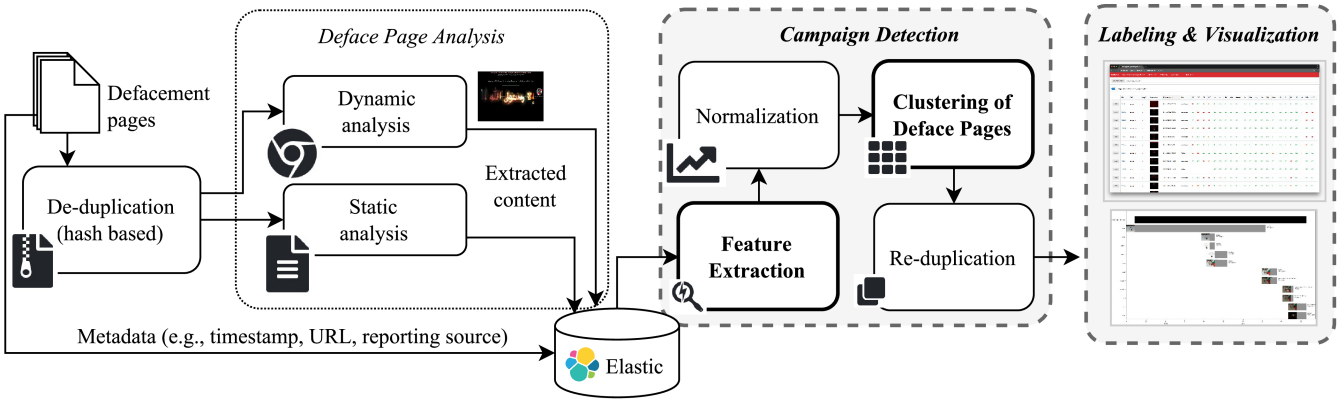


Figure 4: Our approach first extracts the raw content from the deface pages, both dynamically (in a browser) and statically. Then, we use clustering to detect campaigns as groups of "similar" pages. Last, we label each cluster and visualize the campaigns across several dimensions. The core parts highlighted in bold are described in this section, while the details are in Section 4.

HTML, images, and other media files), we extract a set of features—the foundation for the remainder of our analysis.

In the *Campaign Detection* phase, we detect campaigns as groups of "similar" deface pages, using an unsupervised machine-learning pipeline. More specifically, we use data clustering as the core of our analysis system, where each deface page is an object represented as a tuple of numerical and categorical features. Similar pages will have similar features, and thus will end up being clustered together. Although one might be tempted to rely on a supervised machine-learning approach (i.e., classification), the lack of ground truth is a show stopper. As a matter of fact, if any reliable ground truth existed, our work would not be needed in the first place.

Last, in the *Labeling & Visualization* phase, we label each cluster based on its content, and visualize the detected campaigns across several dimensions (e.g., time, actors, targets).

In the remainder of this section we describe the core parts of our approach, deferring the implementation details to Section 4

3.1 Feature Extraction

Engineering the set of features is central to any clustering problem.

Based on our experience from manually analyzing thousands of deface pages, we design the features listed in Table 4 in order to capture the following aspects: visual (e.g., color, images, video, audio), structural (e.g., HTML tags), lexicon (e.g., distribution of character classes), social (e.g., use of social network handlers), and so on.

We extract these features both statically (with pattern-matching on the source files of the deface pages) and dynamically (from the DOM obtained through a headless browser).

Although these features are, technically, extracted on attacker-supplied data (i.e., the deface page itself), the fact that we extract several aspects and the fact that we are not trusting the metadata, makes our approach more resilient to feature evasion than the existing approaches purely based on metadata.

3.1.1 Visual Features. As shown in Figure 3, the appearance of a page immediately characterizes a campaign. This has been confirmed by previous work: [2] demonstrates that a deep-learning algorithm finds that small portions of the screenshot are strong features that can automatically tell defaced vs. clean pages apart.

We extend this concept and include the perceptual hash of the page screenshot, the 5 most common web-safe colors, and the number of images (i.e., `<img>` tags) found in the page. Indeed, as Figure 5 shows, web defacers have always been using images.

Moreover, Figure 5 suggests that modern defacers include embedded audio files that play a song whenever the deface page is rendered. Songs are typically related to strong symbols or ideologies, which the defacers want to promote. These audio files are included via external URLs (e.g., pointing to YouTube, SoundCloud, or other streaming services) using JavaScript or the `<embed>` tag.

We use two features: the first is numerical and counts the occurrences of "sound URLs"; the second is categorical and captures the type of "sound URLs" – i.e. `<service>_srv` (`service` is YouTube, SoundCloud, and so on) and `<ext>_file` (`ext` is MP3, M4A, WAV, and so on).

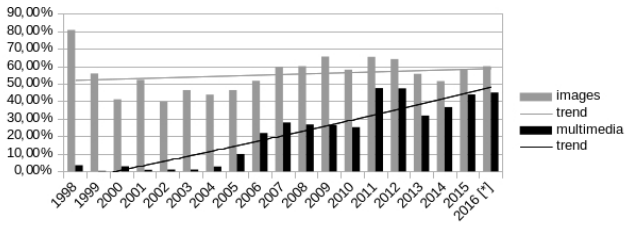


Figure 5: Use of multi-media and image files in deface pages.

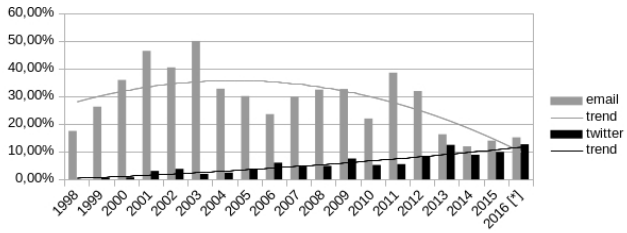


Figure 6: Use of email and Twitter handlers in deface pages.

3.1.2 Structural Features. The same visual aspect can be obtained through several different combinations of HTML tags. Therefore, we use a set of features that count the occurrences of each tag, focusing on the most relevant tags found in deface pages: `style`, `meta`, `embed`, and `object`,`script`, `iframe`, and `a`.

3.1.3 Geographical Features. This group of features captures the ethnicity of the page author. Character encoding is a good indicator of the region (real or mimicked) where the author creates the deface content. For example, ISO-8859-1 is popular among computer users in Western Europe, ISO-8859-6 in Arabic-speaking regions, and so on. We do not trust the encoding declared with the `<meta>` tag, if any, because some defacers deliberately use an encoding that differs from the declared one, probably inma an attempt to confuse automatic analyzers. Instead, we use the popular `chardet` library, the same used by modern web browsers such as Mozilla.

Moreover, we detect the language the text is written in. Although language detection has become a common task and many reliable tools exist, the co-presence of multiple languages in the same deface page can confuse such tools. Moreover, the (typically long) list of nicknames creates long sentences of existing and non-existing words in mixed languages, making language detection prone to errors. For this reason, although we have a language-guessing step in our pipeline, we use this feature for descriptive purposes, not for clustering nor to take any decision.

3.1.4 Domains Features. The inclusion of external resources characterizes how a page is built (e.g., using libraries or pointing to external URLs vs. self-contained page with embedded resources). We capture this aspect with two features: the ratio of external domains is the fraction of URLs included in the page that point to domains that are different from the defaced one; the second feature describes the "syntax" of such external domains i.e. the fraction of ASCII letters in each domain name. We keep the average of this latter value for each page.

3.1.5 Social Features. Defacers tend to follow the evolution of Internet technologies and adopt mainstream communication and social networking tools (including Internet Relay Chats in the past). As Figure 6 shows, the number of Twitter @handlers and email addresses over time present two opposing trends, with email slowly leaving room to Twitter.

3.1.6 Page Title Features. The title is key element of a web page, and so is for deface pages: The actors use a title that delivers the core of their message, such that to ensure that search engines and automated scrapers capture it, ensuring good visibility. We indeed noticed—by manually analyzing thousands of deface pages—that defacers seem to put a reasonable amount of effort to fit their high-level message in the title, and sometimes the team member names.

This group of features captures an approximated representation of the lexical aspects of the title, encoded as the ratio of each main character family (ASCII letters, punctuation, white spaces and digits), normalized to the title length.

3.2 Clustering

Since our dataset contains millions of records, each represented by tens of features, the choice of the clustering algorithm is constrained by the available memory and time. Therefore, any clustering algorithm that needs to materialize the entire distance matrix—or that needs to perform pairwise comparisons across all the combinations of elements—is not a suitable choice. The state of the art for memory efficient, scalable clustering is BIRCH (balanced iterative reducing and clustering using hierarchies) [11]. Without going into the details, instead of calculating and storing the distance between points according to the *entire* feature space, BIRCH keeps 3 statistics for each cluster (the number of elements in the cluster, the sum and the square sum). These values are efficient to compute, and are sufficient to calculate the distance between two clusters, their centroids, and diameters. Moreover, BIRCH maintains a B+-tree-like structure, which allows to quickly find the closest cluster for each new data point (i.e., deface page feature vector), and updates the tree as new points come in. BIRCH requires one main parameter, the threshold, which is used to decide whether a new sample should be merged into an existing cluster, or it should start a new one. The branching factor of the tree, which can be tuned, only influences the speed and memory requirements, without drastically affecting the final result.

Optionally, BIRCH clusters can be further post-processed with any other clustering method. However, we manually validated the results with BIRCH alone, and the clusters produced were correct, containing tightly clustered data points, and overall matching the output of manually clustered data.

Despite libraries and computation services help streamlining machine-learning tasks nowadays, we still have to deal with the peculiarities of the application domain, which are described in the reminder of this section.

3.2.1 Categorical Features. Since BIRCH requires all features to be real valued (or, at least, numerical), we reduce the number of categorical features to the bare minimum, with each categorical feature taking a small number of category values. This allows to map each categorical type onto M-sized binary features (valued zero or one, accordingly), where M is the cardinality of the categorical features. This procedure, known as one-hot encoding, has the downside of increasing the number of features by a factor M, for each categorical feature.

According to Table 4, we one-hot encoded the type of first sound URL (21 categories), and character encoding. We do not use the language for clustering.

To keep the problem feasible, we reduced the cardinality of the encoding feature to 10 macro-categories. In fact, there exist 40 character encodings, which would mean 40 additional binary features. To this end, we grouped the character encodings by region of use, obtaining 10 values for this feature[11] in no particular order.

3.2.2 Feature Selection and Weighting. Table 4 shows a selection of the features that we originally designed. Indeed, we eliminated features with near-to-zero variance, because they would have no discriminant power in the clustering process. In our case, we eliminated the counts of the `<resource>` and `<link>` HTML tags.

During our validation (described in Section 6), we noticed that the perceptual hash was too discriminant, causing clusters to break

[11] European, Cyrillic, Greek, Turkish, Hebrew, Arabic, Chinese, Thai, Korean, and Japanese

Group	Feature Name	Type and range	Description
Visual	No. of Images	integer $[0, \infty]$	Number of <img> tags
	Perceptual Hash	binary (64 bits)	Calculated on the north-centered 1600x900 screenshot crop
	Average Color	3 floats (RGB)	Average of the 5 most common colors in the screenshot
	No. of Sound URLs	integer $[0, \infty]$	Number of URLs pointing to sound-hosting services or files
	Type of first Sound URL	categorical	File and service type of the first (usually the only) sound URL
Structural	No. of each <tag>	7 integers $[0, \infty]$	Number of `style`, `embed`, `script`, `meta`, `object`, `iframe`, and `a` tags
Geographical	Encoding	categorical	Detected text encoding
	Language	categorical	Detected language (for labeling only)
Domains	External domains	real $[0, \infty]$	Ratio of links pointing to cross-origin domains
	Letters in external domains	real $[0, \infty]$	Avg. ASCII letters in the external domains string
Social	No. of online handlers	int $[0, \infty]$	Twitter @handlers, #hashtags, e-mail addresses
Title	Letters, digits, punctuation, white-spaces in title	4 real $[0, \infty]$	Ratio of the listed character classes in the page title

Table 4: Clustering features that we extract from each deface page.

too often for minimal visual variations. Indeed, the perceptual hash can be very sensitive to object replacement (e.g., the defacer changes the background image, while the campaign is the same). For this reason, we applied feature weighting, assigning 30% weight to the perceptual hash feature, and 70% to the remaining features. We set this weighting empirically, as described in Section 5.1, starting from 50–50% and gradually shifting the weight towards the other features.

3.2.3 Distance Metric. At this point, we have obtained a feature vector with 6 high-level features (visual, structural, geographical, domains, social, title), comprising 22 real-valued features and 95 binary-valued features, that is, 64 (perceptual hash) + 10 (encoding) + 21 (type of first sound URL). For real-valued features, the Euclidean distance (L2-norm) is the natural choice, whereas binary-valued features are typically compared by means of the Hamming distance, which is very time efficient.[12]

3.3 Labeling & Visualization

To provide the analyst with an explainable and humanly-readable view of the clustered deface pages, we represent each cluster as a succinct report that includes the time span (oldest and newest deface page), thumbnail of the screenshots grouped by perceptual hash, and, most importantly, a list of patterns that create a meaningful label of that cluster.

To this end, we rely on a set of regular expressions that we built semi-automatically by inspecting thousands of deface pages and clusters. Through these regular expressions, we can reliably extract the name of the deface actor, team, and the set of terms used by the attackers to name their campaign. For example, we built patterns that capture all the variations of sentences such as "`hacked by TEAM / ACTOR NAME`" or "`#CAMPAIGN NAME`" or "`ACTOR NAME defacer`" (including l33t speech normalization, removal of unnecessary punctuation, and so on). Although far from perfection, having these strings extracted semi-automatically from each cluster allows to assign a meaningful name to an *otherwise unidentified* set of similar pages, which speeds up considerably the manual validation process.

Last, we annotate each cluster with the list of categorized targets (e.g., news site, government site), which we obtain from the aforementioned Trend Micro's Site Safety Center.

Armed with these additional attributes, the analyst can easily explore, drill-in, and pivot the data, as showcased in Section 6. For example, we can flexibly group the clusters by labels, period, target (TLD, category), and so on.

Note (on Campaigns). We acknowledge that the concept of campaign is fuzzy, and comes with some limitations that the reader must be aware of. Clusters are meant to find very similar—with respect to the defined features—yet not identical, deface pages. Campaigns are meant as a higher-level grouping of clusters, based on patterns provided by the analyst. In our experience, grouping solely on such patterns does not eliminate the burden of inspecting the large number of similar-yet-no-identical pages. Similarly, clustering alone does not provide semantic labels for the analyst to understand, at a glance, the content of a cluster without inspecting it. For this reasons, the definition of campaign encompasses both aspects.

4 IMPLEMENTATION DETAILS

In this section we describe the essential implementation details that are needed to reproduce our data-processing pipeline.

4.1 Static & Dynamic Page Analysis

We extract the features both statically (i.e., from the page's raw HTML and the page's source file), and dynamically from the DOM that we obtain by rendering the page in a full-fledged headless browser. Technically, the recent Chrome's "headless browser mode" resulted in the fastest and most robust choice.

For a given feature, when these two static and dynamic values are different, we keep the greatest of the two; we assume that a greater value unveils more content (e.g., obfuscated content is revealed during rendering and in the dynamic analysis only).

[12] The pairwise Hamming distance between perceptual hashes is a real value in [0, 1], where 0 indicates that two images are essentially identical and 1 indicates that two images are far from being visually similar.

Despite the defacers are interested in creating functional pages that can be rendered by a browser, the main challenge is that external resources could become unavailable at the time of analysis (e.g., 404). To overcome this limitation, we set a rendering timeout of 10 seconds before storing a (partial, but valuable) screenshot and DOM representation.

4.2 De-duplication and Re-duplication

Often, defacers resort to re-using the very same page, resulting to near if not exact duplicate records. We take advantage of this fact to increase the throughput the analysis system is capable of. Indeed, from a clustering viewpoint, two identical pages are treated as one, single page. Therefore, before any expensive computation—from Feature Extraction onward—we calculate a hash of the main file of the deface page (e.g., "`index.html`") and use it as a de-duplication key. After clustering on the de-duplicated data, we re-duplicate the data in each cluster using the same key, thus obtaining the "expanded" clusters with the full set of original records. We opted for the conservative and most precise choice of SHA over SSDEEP.

4.3 Normalization

We experimented with a time-dependent data scaling—using yearly min-max bounds—rescaling each feature on a per-year basis, but we obtained very poor results. To understand this outcome, let's consider the self-explanatory example of Twitter. Before 2006, when Twitter was founded, the value of this feature was essentially zero—apart for defacers using the "`@name`" syntax to highlight keywords. Had we used a time-dependent scaling, a value of "1 Twitter handler" in 2006 would certainly be the highest value found around that time, whereas the same value in 2016 would be abysmal. As a result, low values would be penalized in recent times, and inflated in the old times, resulting in an overall flattened feature space. In terms of clustering, looking at this example feature only, a page from 2006 having only 1 Twitter handler would be clustered together with a page from 2016 having a dozen Twitter handlers, because the two values (1 and 12) would have a similar normalized (low) value. After this analysis, we decided not to perform time-dependent scaling, and simply resort to dataset-wide normalization, using an L2-norm function, which is a widely accepted practice in the machine-learning community.

We do not scale binary-valued features, because the Hamming distance do not require any scaling by design.

The BIRCH clustering algorithm is implemented in Scikit-learn [7], the de-facto choice for Python-based machine learning, which is well integrated with Pandas and NumPy [6] for data manipulation.

5 VALIDATION

Validating large-scale clustering results is hard. Without a ground truth, the only option would be to use so-called *internal* validation metrics, which are computed over the feature space, and characterize, for example, how "close" the elements within a clusters are to each other vs. how far the elements of two separate clusters are. These metrics do not predicate on the actual quality of a clustering, but simply give an indication on the features' discriminant power. For instance, a perfectly good clustering in practice might have decent internal metrics, and vice versa. Moreover, these metrics do not scale, because they are computed over the pairwise distance.

To overcome these obstacles, we adopt a threefold approach.

First, we manually create a small ground truth, comprising 10 deface campaigns, which we expect our system to cluster accordingly. This allows us (1) to calculate so-called *external* metrics, which indicate how much a clustering "agrees" with the ground truth, and (2) to find the best clustering parameters. Secondly, we run our system on a large portion (1%) of our dataset, and manually inspect the obtained clusters without any pre-built ground truth. Last, we run our system on the entire dataset, isolate the largest campaigns that are discovered, project them onto a time line, and search for confirmatory evidence (e.g., online stories) to corroborate such findings.

5.1 Initial Validation and Tuning

During our manual analysis of thousands of deface pages, we learned about 10 well-known campaigns[13], comprising of 4,827 pages overall.

On this dataset, we run our system on all the combinations of feature groups (from Table 4), with a 0.1 to 0.9 BIRCH threshold. For each iteration, we calculate both internal and external metrics. We use the silhouette score (internal), which is close to 1.0 when clusters are compact and sharply separated clusters. Having a ground truth, we calculated the V-measure, which is the harmonic mean between homogeneity and completeness. The homogeneity is 1.0 if all the clusters contain only deface pages that are members of a single true campaign, whereas the completeness is 1.0 if all the deface pages of a given true campaign are member of the same cluster. Therefore, a V-measure score equal to 1.0 means perfect match between ground truth and obtained clusters.

We obtain the best results with a BIRCH threshold equal to 0.5, resulting in a silhouette score equal to 0.822 and V-measure equal to 0.856 (0.907 homogeneity and 0.810 completeness). Therefore, we set this threshold for the remaining experiments.

We examined the cause of few mis-clustered pages, and we learned that such pages did not have a full screenshot available (which affect the visual features), because some of the external resources were not available anymore. After eliminating such pages from the ground truth and input dataset, we obtained a clustering that matched the ground truth perfectly.

As a final validation, we ran DBSCAN on the centroids of the 10 clusters obtained by BIRCH, which were confirmed. Given time and memory efficiency of BIRCH, we decided to use BIRCH alone.

5.2 Large-Scale Validation

We validated our system on a larger—but still manually explorable—dataset (1 month worth of records). We chose a time range (Jan 2015) where we knew from external sources that there were active campaigns, without actually knowing where, in our dataset, those campaigns were located.

Our system produced 2,722 clusters, of which we inspected 10% (totaling 1,702 deface pages). For this, we followed a semi-automated approach with the aid of the *Labeling & Visualization* phase in

[13] We named them: Syria.1, Syria.2, Bader Operation, Muslim Liberation Army, End the Occupation, Operation France, Charlie.1 and Charlie.2

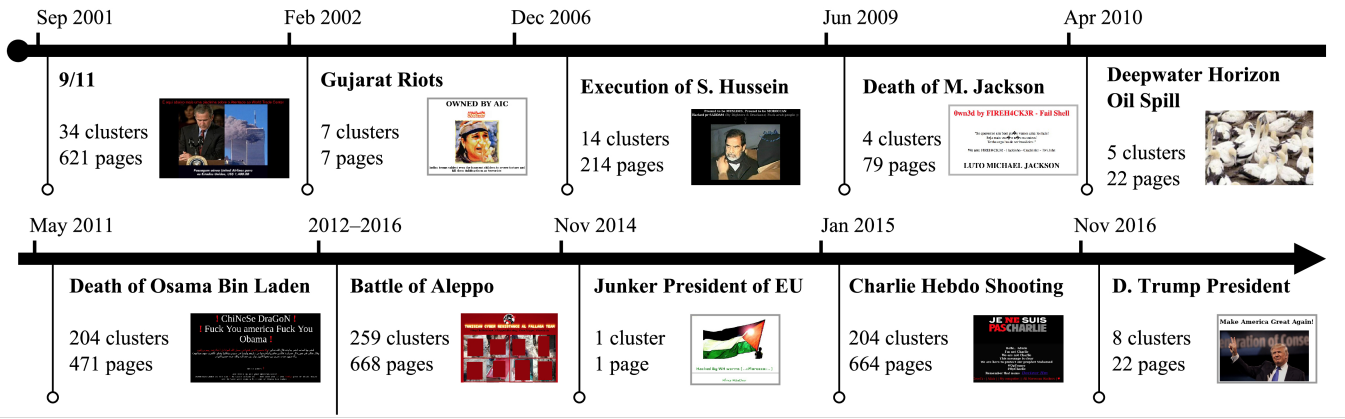


Figure 7: Timeline of major real-world events reflected in the cyber space as defacement attacks.

Figure 4: The screenshots of the web pages, keywords, and the fuzzy hash (SSDEEP) of the text of the web pages, helped speeding up the validation process—although of course not a substitute for a complete ground truth.

Through this process, we confirmed the identified campaigns, which included: `opdesaparecidos` (Jan 10–25) with targets in Mexico and Argentina, `opthailand` (Jan 05), and `nasaanang` (Jan 29–31) with targets in the Philippines, including government-related sites.

We did not encounter any spurious cluster (i.e., with unrelated defacements). However, we found 27 "split" clusters, despite their content looked very similar. This expected result is due to the conservative approach that favors fine-grained, yet very compact clusters as opposed to spurious ones. Recall that, such clusters could be automatically merged during the *Labeling & Visualization* phase.

5.3 Real-World Validation

We performed clustering on our whole dataset. Then, starting from 20 major events occurred in the World, we searched for such evidence in our results. As Figure 7 shows, we found out key matches that confirm the belief that actors use the cyber space as a parallel "territory" for their propaganda.

Right after the 9/11, pro-Al-Qaeda cyber actors planted supporting web pages celebrating the outcome of the terrorist attacks, whereas other cyber actors were against the event. We observe a similar pro/against situation after the Charlie Hebdo shooting in Jan 2015.

In Feb 2002, the riots in the Indian state of Gujarat were closely followed by cyber actors, who protested against the violence by planting deface pages in compromised Indian sites, asking to stop such riots. We observe a similar reaction during the long and devastating battle of Aleppo, for which we had to redact part of the screenshot in Figure 7 because of the disturbing images.

In the case of the execution of Saddam Hussein (Dec 2006) and death of Osama Bin Laden (May 2011), we observe several deface campaigns condemning the executors.

Even election events are followed by the cyber actors. Interestingly, unidentified actors have compromised and defaced the website of the European People's Party to protest when its leader Jean-Claude Junker was elected president of the European Commission in Nov 2014. Contrarily, D. Trump supporters planted defacement campaigns to celebrate after the Nov 2016 elections.

Year	Actors	Teams	Clusters	Year	Actors	Teams	Clusters
1998	50	30	31	2008	12,169	6,936	85,085
1999	410	248	826	2009	13,779	8,762	76,567
2000	820	492	2,385	2010	16,762	8,156	96,599
2001	2,283	1,167	11,726	2011	19,203	8,959	117,396
2002	2,122	1,244	14,684	2012	21,640	10,051	121,243
2003	2,778	2,948	23,183	2013	21,366	10,032	125,195
2004	4,041	4,459	29,722	2014	19,318	8,811	112,760
2005	6,729	5,789	48,043	2015	20,659	12,164	167,031
2006	14,335	9,504	90,632	2016*	16,317	10,521	113,085
2007	12,941	7,323	76,000				

Table 5: Yearly distribution of actors, teams and clusters as reported by our clustering system from Jan 1998 to Sep 2016*

5.4 Limitations

During our validation, on top of successful cases, we also found clusters with unrelated pages. After manual inspection, we identified the two main corner cases that challenged our approach.

First, defacers using almost empty pages, with a short sentence as proof of defacement (e.g.,"Admin, you been HACKED! By Attacker"), yielding feature values with poor discriminant power. Fortunately, such pages seldom form proper campaigns.

Secondly, the defacement leaves the original, functional web application almost unaltered (e.g., planting traffic-redirection or drive-by code). Strictly speaking, such cases do not qualify as defacements, and our system groups the "deface" pages based on the features of the original page, forming clusters containing classes of web applications (e.g., WordPress, Drupal).

6 CLUSTERING RESULTS

In this section we present a general overview of the results of our clustering on the entire dataset.

Overall, our clustering took 35 hours of processing time, including 2 hours for Labeling and Visualization. In comparison, the fastest alternative for large-scale clustering (DBSCAN) would

have required prohibitive (exponentially growing) computational resources, as shown in Figure 8. Attempts to cluster 128,000 deface pages with DBSCAN crashed for main memory exhaustion on machine with 256GB of RAM. This further confirms our choice of BIRCH.

Table 5 reports the number of clusters detected by the system over the year, together with the number of teams and attackers. On average, the clusters that we obtained include 8–9 records (with a 136–137 standard deviation), and span across 9.23 days (with a 77.26 standard deviation).

From a practical point of view, each cluster found by our system represents the joint work of multiple actors, working for the same attack, using visually similar deface pages, and, most importantly, believing in the same ideologies. This represents a unique analysis pivot, towards better understanding on how attackers work together—beyond anecdotal evidence.

In this section we provide example of the measurements that an analyst can obtain with our system.

6.1 How Attackers are Organized

First, our system produces insights on how attackers are organized in groups when conducting defacement campaigns. A good summary is given in Figure 9.

Overall, when looking at the general picture, about half of the attackers (53%) are lone wolves, as they do *not* identify themselves using a team name. The remaining attackers belong to one or more groups. More in detail, our data suggests that most of the defacers (80%) are devoted to the same affiliation(s) throughout their "career," while only 20% of them migrate from one group to another.

The same Figure 9 also describes well the concept of *joint campaigns*. This phenomenon is quite common and only 30% of the campaigns do not cooperate. On the other side, campaigns sharing common motives, objectives or targets, are often driven by similar geopolitical or religious ideologies and therefore interested in targeting similar ethnic groups or races. For example, the following campaigns—operated by hacking groups of different countries like Spain and Italy—all advocate for the stop of the well-known Syrian war in Aleppo: `alepo_se_pierden`, `savesyria`, `save_halab`, `stoptheholocaust`, `aleppo_is_burning` and `aleppo_é_in_fiamme`. In another example, actors from anarchist groups operate joint campaigns to benefit from larger outreach: `freedom`, `opanarchy` and `delirium`.

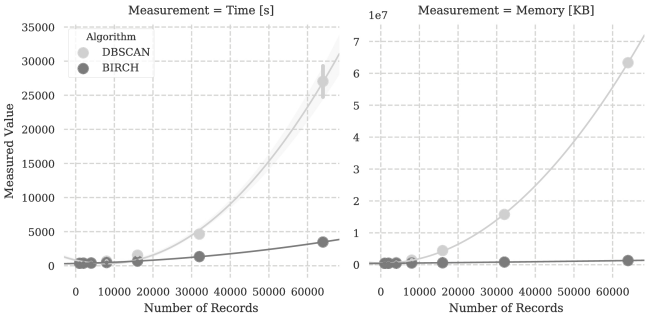


Figure 8: Scalability of BIRCH vs. DBSCAN (10 runs).

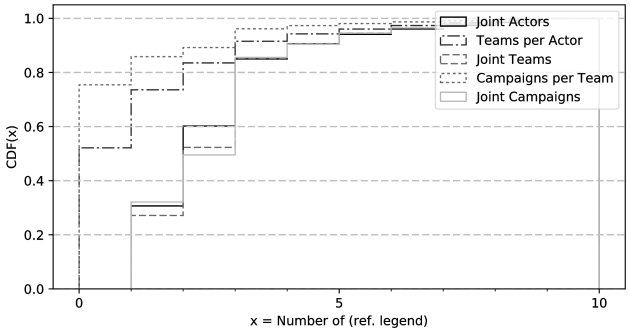


Figure 9: Cumulative distribution of the number of: joint actors, affiliations (teams) per actor, joint teams, campaigns per team, and joint campaigns. The term "joint" means "appearing together in the same cluster of deface pages."

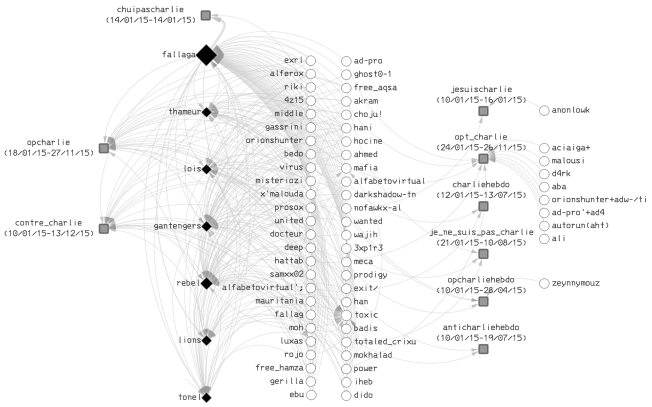


Figure 10: Teams (diamond nodes) and actors (circle nodes) cooperate (arrows) in defacement campaigns (square nodes). The "fallaga" team drives the largest of these campaigns, while smaller teams (second column of nodes) refer to "fallaga" in their defacements. A minority of actors (nodes on the right) conduct campaigns almost independently.

Digging deeper with an example, Figure 10 provides a view of various campaigns pro and against the "Charlie Hebdo" attacks. The "fallaga" team[14] participate to the largest of these campaigns ("chuipascharlie" slang for "I am not Charlie"), whereas smaller teams are still referring to "fallaga" in their deface pages. Also, we notice a vast majority of actors affiliated with at least one team, and a minority of actors working almost independently.

6.2 Long Term vs. Aggressive Campaigns

Campaigns can differ in terms of durability and intensity. With the following experiment, we capture such differences and visualize them through the heatmaps in Figure 11.

Figure 11 *left* shows the longest lasting campaigns over our entire dataset. For example, campaigns `samarindahack` and `syshack`

[14] "6 Members Of Fallaga Team Hacker Group Arrested" http://cjlab.memri.org/lab-projects/monitoring-jihadi-and-hacktivist-activity/6-members-of-fallaga-team-hacker-group-arrested-by-tunisian-authorities-over-opfrance/

score 10+ years of presence in the underground with hundreds of attacks distributed over the entire life of the campaign.

In contrast, the campaigns on the *right* last shorter and are more aggressive with respect to the number of attacks they conduct. All of them, including major geopolitical campaigns like `bangladeshi` and `savegaza` reacting to war events in India and Palestine, prefer visibility over stealth. This aspect is also highlighted from the category of websites been targeted, with major news and media portals been defaced for additional visibility.

6.3 Case Studies

Throughout this section we provide examples on how our system can be adopted by an analyst to conduct real-world investigations on historical and ongoing campaigns.

6.3.1 Timeline Analysis. Figure 12 shows a long-running campaign named `h4ck3rsbr`. The 60 clusters are represented as horizontal bars, annotated with the most targeted TLDs (on the vertical axis) and the number of defacements in each cluster (on the magnified detail on the right-hand side).

Although the entire campaign spans over 4 years, each cluster (i.e., horizontal bar) does *not* go over 4 months. However, thanks to these clusters sharing a common label like the name of the campaign they affiliate to (i.e., `h4ck3rsbr`), we can automatically draw a timeline.

As opposed to targeted campaings, `h4ck3rsbr` is generic, as it targets websites hosted under various TLDs. Some of the groups that contributed to the campaign are `c0d3rz` with a cluster of 24 defacements, `Trustix` (38 defacements), *Fatal Error* (6 defacements), and `Infektion Group` (17 defacements). These clusters contain templates with (pages having) either white or black background, dating back to Oct 2006.

6.3.2 Targeted Campaigns (Victims' TLDs and Categories). As we previously discussed in Section 6.1, single campaigns often cooperate for the benefit of the entire community. Figure 9 shows that 70% of the campaigns cooperate and about half of the joint campaigns are larger than 3. As said, this is often the case with campaigns sharing common motives and objectives, e.g. in support of certain ideologies like religion or politics.

In the following we present two examples of large-scale joint campaigns that we observed in our dataset. Figure 13 visualizes the Israeli-Palestinian conflict (left) and Anonymous operations (right). In each of these graphs, one node is proportionally as big as the number of connecting arcs (i.e. defacements). Similarly one arc is as thick as the number of defacements.

For example, while the entire Israeli-Palestine conflict involves 12 campaigns, `opisreal` and `opsavealaqsa`[15] represent the most aggressive and active ones. With respect to Anonymous operations (on the right), these campaigns mainly focus on governmental websites, as expected.

These are only some of the analyses that our system can provide via automated correlation of the cluster labels.

[15]The Al-Aqsa Mosque in Jerusalem

7 RELATED WORK

Despite website defacement being a longstanding issue, only limited peer-reviewed prior work exists. We divide them roughly between measurements and detection approaches.

Measurements on Web Defacement. Woo at al. [10], from the psychology research community, have analyzed the content of 462 defaced web pages collected between January and April 2001. They found out that only about 30% of the defacements had a political motive. Although drawing statistics about the motive of defacements is not among the goals of our work, we have briefly highlighted in Section 2 that we have noticed an increased presence of keywords that suggest that the defacers are more and more driven by real-world conflicts and ideologies. Interestingly, the same authors in [10] also notice that web defacers are not isolated individuals, but are part of active and extensive social networks. Unfortunately, 462 hand-picked web deface pages are just a drop in the ocean if compared to the data available as of 2009: Indeed, Zone-H has been active since 1998.

The most recent work is [8], which is very aligned with our research goal. However, authors in [8] are more interested in understanding whether and to what extent there is a link between web defacement and hacktivism. To answer such question, the authors focused on deface records reported to Zone-H throughout 2016. Interestingly, they also find an increase in the use of defacement for political and patriotism reasons, especially in the past 3 years. However, the authors relied on the metadata (see Table 2) rather than on the actual content of the deface page, which gives only part of the picture, and is very limited in number.

Detection of Web Defacement. In 2007–2008, Davanzo et al. [3] compared the effectiveness of 7 anomaly detection techniques—versus domain knowledge—at detecting web defacement, on a set of 480,000 deface pages obtained from Zone-H. Interestingly, all the 7 automatic techniques required a feature-space reduction to a dozen features in order to deliver acceptable results. Instead, the use of expert knowledge delivered good results, sometimes better than the automatic techniques, even with the full array of 1,466 features that the authors have selected. This result, extended in [4], supports our decision of proposing a decision-support system, as opposed to creating a pure detection system.

Overall, the problem that approaches such as [3, 5] tackled is fundamentally different: The authors were looking for features that recognize when a *monitored* benign page is altered by a defacement—a binary decision, whereas we are looking for features that tell the various defacement campaigns apart—a multi-outcome and much more complex decision.

In this direction, Borgolte et al. [2] is the most advanced technique presented so far. Instead of doing feature engineering and selection, the authors let a deep-learning pipeline figure out the best features to recognize deface pages. The input domain to the learning algorithm, which uses convolutional neural networks, is a screenshot of the page, obtained with a headless browser. As a result, the neural networks (automatically) give importance to the regions of the page that contain the logos of the hacking groups, or the special "symbols" displayed using non-ordinary fonts. Indeed, such visual peculiarities immediately catch the attention of the

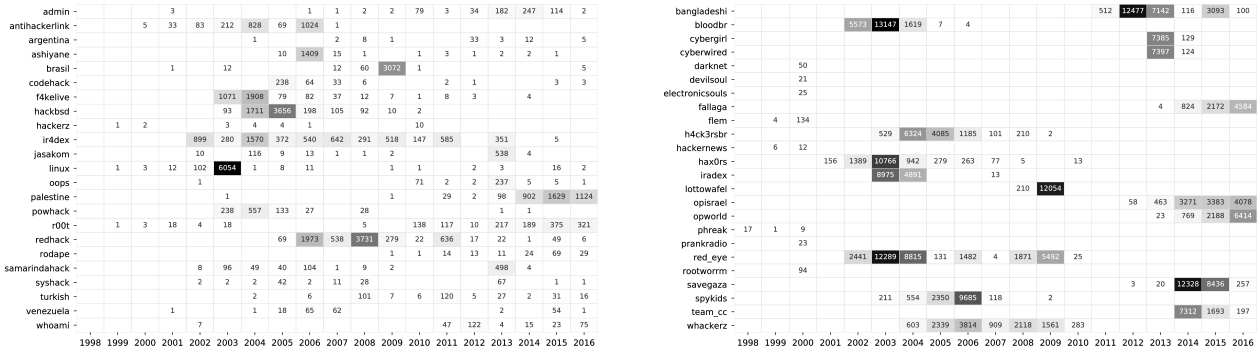


Figure 11: Longest lasting vs. most aggressive campaigns: the heatmap highlights their opposite nature. The cell represents the number of attacks conducted by the campaign in the year. Longest-lasting campaigns (left) conduct slower and longer attacks, while aggressive campaigns (right) react to geographical events (like terrorist attacks) and prefer massive attacks (conducted few days after the event, for example).

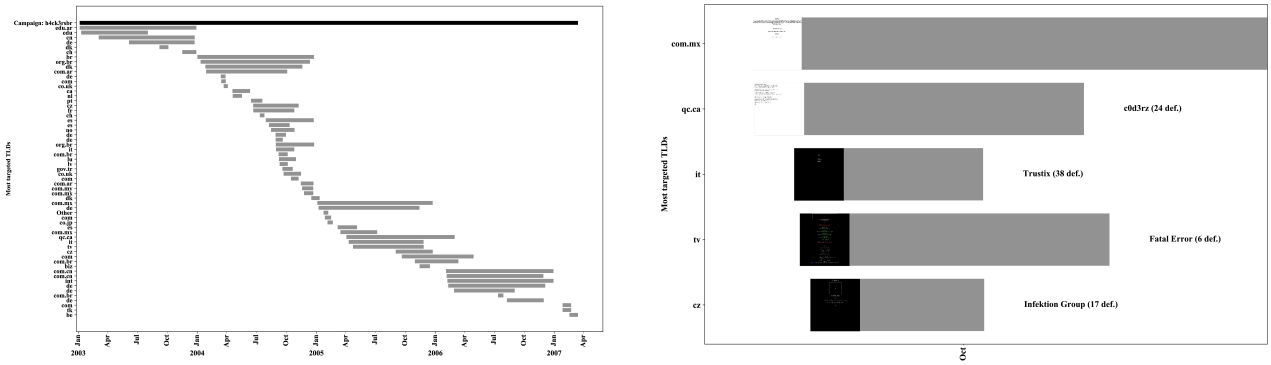


Figure 12: The long-running `h4ck3rsbr` campaign, with 60 clusters overall, each represented with an horizontal bar, with top-targeted TLDs on the horizontal axis, and a magnified view on the right-hand side, showing two of the templates (white and black) used by the actors.

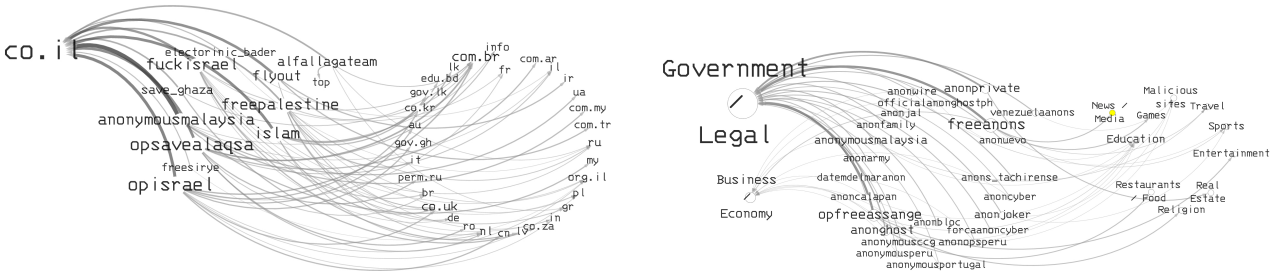


Figure 13: Israeli-Palestinian conflict (left) and Anonymous operations (right). Two groups of *joint campaigns* targeting Israeli websites (left) and Governmental websites (right). The campaigns cooperating in each group share common motives and objectives.

domain expert, who finds them unusual in a regular benign page. Despite our work goes in a different direction, we were inspired by how the authors leveraged the visual appearance of the page, and we designed some of our features to capture this aspect.

8 CONCLUSIONS

Attackers compromise and deface websites for various reasons, from supporting their personal reputation to, more interestingly, promoting a certain ideology, or religious or political orientation. Using an automated approach, we conducted a large-scale measurement on 13 million records of defacement incidents spanning over a period of almost 19 years. We shed light on how attackers collaborates, how are organized into teams to run long-lasting and impactful campaigns. In particular, we explored the dark propaganda phenomenon, that is the abuse of websites to deliberately place content that may affect the public opinion.

REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. of machine Learning research*, 3(Jan):993–1022, 2003.

[2] K. Borgolte, C. Kruegel, and G. Vigna. Meerkat: Detecting website defacements through image-based object recognition. In *USENIX Security Symp.*, pages 595–610, 2015.

[3] G. Davanzo, E. Medvet, and A. Bartoli. A comparative study of anomaly detection techniques in web site defacement detection. In *IFIP ISC*, pages 711–716. Springer, 2008.

[4] G. Davanzo, E. Medvet, and A. Bartoli. Anomaly detection techniques for a web defacement monitoring service. *Expert Systems with Applications*, 38(10):12521–12530, 2011.

[5] T. Kanti, V. Richariya, and V. Richariya. Implementing a web browser with web defacement detection techniques. *WCSIT*, 1(7):307–310, 2011.

[6] W. McKinney. Data structures for statistical computing in python. In S. van der Walt and J. Millman, editors, *Proc. of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, Nov. 2011.

[8] M. Romagna and N. Jan van den Hout. Hacktivism and website defacement: motivations, capabilities and potential threats, 2017. VB Conference, Madrid.

[9] W3Techs. Usage of operating systems for websites, 2017.

[10] H.-j. Woo, Y. Kim, and J. Dominick. Hackers: Militants or merry pranksters? a content analysis of defaced web pages. *Media Psychology*, 6(1):63–82, 2004.

[11] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. In *In Proc. of the ACM SIGMOD Intl. Conference on Management of Data (SIGMOD)*, pages 103–114, 1996.

Appendices

Year	Top 5 eTLDs	Excluding gTLDs
1998	com co.uk net edu mil	co.uk edu mil it ac.cr
1999	com org net gov se	gov se de edu gov.br
2000	com com.br org net edu	com.br edu gov.br co.uk gov
2001	com com.br net org edu	com.br edu com.tw com.cn de
2002	com net com.br de	com.br de it co.uk
2003	com de net com.br org	de com.br co.uk it
2004	com de net org com.br	de com.br it co.uk nl
2005	com net org de it	de it com.br co.uk ro
2006	com net org de co.uk	de co.uk com.br info nl
2007	com net org de com.br	de com.br info nl co.uk
2008	com net org de com.br	de com.br co.uk nl info
2009	com net org com.br de	com.br de nl co.uk dk
2010	com net org de co.uk	de co.uk nl com.br info
2011	com net org com.br info	com.br info co.uk ru nl
2012	com net com.br org co.uk	com.br co.uk info nl ru
2013	com net org com.br de	com.br de co.uk it ru
2014	com org net ru com.br	ru com.br de co.uk it
2015	com org net com.br co.uk	com.br co.uk ru in it
2016	com net org ru com.br	ru com.br in pl it

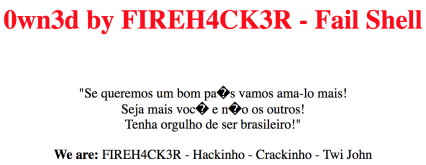
Table 6: Top-level domains targeted each year, according to metadata. The second column is obtained by excluding the main generic TLDs (com, org, net). Ref. Section 2.2.2.

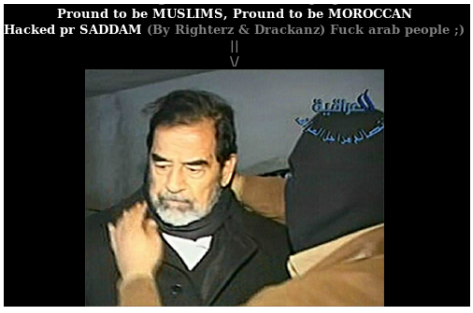
Team	Size	# Defacements	Nationality
Mafia Hacking Team	47	714,863	Mashhad, Iran
Infektion Group	23	606,309	Brazil
h4x0rteam	31	604,597	Unknown
Hmei7	12	591,388	Indonesia
1923turk	2	547,208	Unknown
red devils crew	15	507,886	Saudi Arabia, China
team falcons hackers	41	491,361	Morocco
nopo team	12	474,379	Iran
ksg-crew	14	406,618	Unknown
anonscorpattackteam	12	356,248	Unknown

Table 7: Top 10 teams overall, according to metadata.

Year	Top 5 Actors
1998	milw0rm, Team CodeZero, Zyklon, Giftgas, Magica de Bin
1999	AntiChrist, Fuby, PHC, Fl3m, ytcracker
2000	GForce, Prime Suspectz, Hackweiser, pimpshiz, WFD
2001	Silver Lords, BHS, PoizonB0x, Hi-Tech Hate, WoH
2002	Red Eye, Fatal Error, hax0rs lab, ISOTK, BYS
2003	TechTeam, PsychoPhobia, Red Eye, BloodBR, SHADOW BOYS
2004	iskorpitx, Ir4dex, r00t_System, Infektion Group, int3rc3pt0r
2005	iskorpitx, Infektion Group, ArCaX-ATH, Simiens, SPYKIDS
2006	iskorpitx, Thehacker, crackers_child, CyberLord, SPYKIDS
2007	iskorpitx, 1923Turk, crackers_child, GHoST61, Mafia Hacking Team
2008	iskorpitx, r00t-x, Crackers_Child, GHoST61, Dark_Mare
2009	iskorpitx, NobodyCoder, M0µ34d, 1923Turk, Fatal Error
2010	GHoST61, iskorpitx, TheWayEnd, 1923Turk, ByLenis
2011	TiGER-M@TE, 1923Turk, iskorpitx, KriptekS, GHoST61
2012	Hmei7, kinG oF coNTroL, TiGER-M@TE, 1923Turk, T0r3x
2013	Sejeal, Hmei7, misafir, BD GREY HAT HACKERS, SA3D HaCk3D
2014	d3b X, Hmei7, Index Php, Th3Sehzade, 1923Turk
2015	Index Php, w4l3XzY3, Kuroi'SH, d3b X, KingSam
2016	chinafans, GeNErAL, ifactoryx, Freedom Cry, 4Ri3 60ndr0n9

Table 8: Top 5 actors per year, according to metadata.





14

Figure 14: Each screenshot is taken from one of the cluster that we identified in our dataset in support of the 20 major World events used for the real-world validation of the clustering (ref. Section 5.3).