# Contextualizing and unlocking political web defacements for research

cc-by ©. Published by De Gruyter in cooperation with the University of Luxembourg Centre for Contemporary and Digital History. This is an Open Access article distributed under the terms of the Creative Commons Attribution License CC-BY

Defacement, Hacking, Archive, NLP, Topic Modeling

Despite their significance, political web defacements remain an under-utiled resource for research on both the history of activism on the web as well as subaltern political communication. Web defacements as a form of hacktivism are rarely archived and thus mostly lost for systematic study. When they find their way into web archives, it is often more as a by-product of a larger web archiving effort than as the result of a targeted campaign. Aside from large collections such as the Internet Archive, which might pick up a few hacked pages during a crawl, there also exists a small scene of community-maintained cybercrime archives that archive hacked websites, some of which are hacked in a hacktivist context and may involve web defacements. While community archives exist, no source is available to serve research needs. The data source for this section will be the publicly available attrition.org archive, combined with original pages retrieved from the Internet Archive. To unlock and make available political web defacements for research, their archives need to be stabilized and integrated in an environment that may be usable for academic research. Due to their disruptive nature, this integration effort needs to be complemented by methodological innovation. This article describes a workflow for processing, indexing and analyzing political web defacements.

## Introduction

Web defacements in the context of this article are alterations or replacements of websites by an unauthorized third party. This unauthorized access is usually achieved through compromising the target system (here very broadly named hacking). Of the thousands of websites defaced every day, some are defaced for the purpose of propagating political expression of whatever quality, while others are defaced as part of a campaign to disrupt the digital communication channels of a perceived enemy. Both of these types of defacements are broadly labeled political and are what this article focuses on. Specifically for the context of this article, engagement with public discourse, introduction of material into public discourse and online commemoration of events in defacements were in focus. These web defacements are part of the larger realm of hacktivist and web defacement practices. As defaced pages are generally short-lived, research cannot rely on larger, unspecified Internet archiving services to obtain copies, nor can it follow events in real time. Where web defacements are preserved, it is in community-maintained archives of varying scope and quality. One of these community archives was Attrition.org, which archived selected web defacements from 1995 until 2001. The research objective of this article is therefore to present a workflow for the stabilization of web defacements as an academic resource and provide methods for the investigation of large collections of web pages with a focus on the specificities of defacements.

Defacements are at a double disadvantage in regard to their preservation. On a technical level, defacements are more ephemeral than most other types of websites. Seen as unwanted and often illegal intrusions into computer systems, defaced pages tend to be restored or taken offline before large Internet archiving crawls can find them. Socially, defacements are often regarded as nuisances rather than artifacts worthy of cataloging and consideration. Consequently, and despite their significance, political web defacements remain an under-utilzed resource for research on both the history of activism on the web as well as subaltern political communication. According to Niels Brügger,

Web content may have been changed, moved, or deleted for various reasons, and sometimes in the hope of not being found out. ()

This is most certainly true for web defacements, which are usually deleted and the original page restored as quickly as possible. While understandable from the perspective of the admin or domain owner, these defacements nevertheless form a part of digital history.

Web defacements as a form of hacktivism are rarely archived and thus mostly lost for systematic study. When they find their way into web archives, it is often more as a by-product of a larger web archiving effort than as the result of a targeted campaign. Aside from large collections such as the Internet Archive, which might pick up a few hacked pages during a crawl, there also exists a small scene of community-maintained cybercrime archives that archive hacked websites, some of which are hacked in a hacktivist context and may involve web defacements. Web defacements may also be studies indirectly using press reports, or interview data. See, for example, Hopkins () and Samuel (). While community archives exist, no source is available to serve research needs. Especially in the context of creating a more complete historical record (), filling the described gaps in web archiving is important. Implied in this advocacy for unlocking web defacements is the delivery of a metadata system to adequately describe and thus make defacements usable for academic research. A recent study found that lack of metadata is one of the key barriers for scholars when engaging with archived web material. () As in many web archiving efforts, the problems arising from a large collection of unrelated, unstructured data are problems of identifying relevant material within the corpus and problems of making adequate and data-based statements about themes and sentiments within that corpus. () The methods and workflow presented in this article therefore present first steps towards making web defacements available as academic resources and enable larger-scale academic engagement with defaced web pages.

## Context and definitions

Political web defacements are situated within the larger context of hacktivism. Hacktivism as a political phenomenon has first been described in the context of the 1999 Kosovo War. () For the context of this article, Alexandra Samuels provides a comprehensive definition of the larger field of hacktivism and political web defacements:

[hacktivism is defined as] the nonviolent use of illegal or legally ambiguous digital tools in pursuit of political ends. These tools include web site defacements, redirects, denial-of-service attacks, information theft, web site parodies, virtual sit-ins, virtual sabotage, and software development. ()

This definition reflects on both the activism and hacking background of hacktivism. Hacktivism, and consequently web defacements, contains parts of political activism and digital infrastructure interference. This is accounted for throughout this article when defacements are described as communicating on both a technical and a contextual level. For a reflection of this dual nature of defacements, also see Vegh (). The defacements discussed in this article all date to a time period between 1996 and 2002, that is prior to the platformization of the web through social media companies. Research on hacktivism in the early 2010 years was largely shaped by Anonymous and the role of social media in the Arab spring uprisings. See Coleman (, ), Fuchs () and Salem (). Later discourses focussed on the role of “hacking” (the inflationary use of that term grew steadily throughout) in the context of the 2016 US presidential election. () The defacements discussed in this article offer a rare insight into the hacktivism and web defacement scene of the pre-social media era. It is at the same time a rare collection of pre-2000 websites, a time period rarely covered by web archiving initiatives. () The content of the Attrition archive allows one to see the origins and development of subversive political expression on the web.

### Data sources

Most data sources on defacements have already disappeared from the Web. Alexandra Samuel in her 2014 dissertation finds that

Thanks to the volume of defacements, the biggest mirrors (Attrition and alldas) have stopped archiving defacements. alldas has gone offline entirely; Attrition stopped maintaining its archive in April 2001 [...] but has preserved its records of defacements from 1995-2001. ()

Related work on political web defacements found that in 2020, 9 out of 12 surveyed defacements archives were offline, with only one still collecting new content (). The reason for this decline is unclear. A larger analysis of web defacements suggests web defacements have lost some of their appeal as a political tool (). While this is likely to be one factor, a change towards web content being distributed through centralized platforms rather than individual web pages is certainly another. The remaining websites are - very generally speaking - more resilient against intrusion than websites of the early 2000s.

One of the mirrors mentioned by Samuel - Attrition - will be the data source for this article. While Attrition stopped accepting defacements in 2001 (), they have preserved their archive and made it available to the public in 2021. This opening up of the archive (research access before that was granted on a case-by-case basis, but only to the front end) shows the awareness of the Attrition admins towards the value of the archived material. This is also expressed in the statement accompanying the re-opening of the archive:

Back in the day we had so many ideas for analyzing the mirror data. Extensive analysis of the HTML, language, heat maps, graphing the "greets" and "hates", and more. This data is extremely challenging to do any of that for a variety of reasons. Twenty years later, data processing tools and methodology is considerably more advanced though, so we hope someone will do just that. ()

But it is not only the relative ease of access that makes Attrition a valuable object of the study of web defacements. Attrition started collecting material in 1999, incorporating older material “copied from other defacement mirrors that ran before us” (), going back as far as 1996. At the time of closure in 2001, around 14.000 defaces websites were preserved in the archive. This offers exceptional access to very old web material and thus provides resources for research on early days web activism and political expression. Who the original audience was for Attrition.org is hard to exactly reconstruct. With some 20 years distance, this audience may broadly be defined as hackers, from [black hat](https://www.merriam-webster.com/dictionary/black%20hat) to [white hat](https://www.merriam-webster.com/dictionary/white%20hat) Also see the [original GitHub repository](https://github.com/attrition-org/web-hack-mirror).

Attrition stands out from the remaining defacement archives in that it explicitly states the use of its material as the basis for research. In contrast, Zone-H sells research access for a considerable price while other mirrors show no interest in research enquiries. While none of the mentioned archives provides insight into their collection strategy (or merely the rules for acceptance or rejection of submitted material), Attrition can be seen as leaning towards political expression in web defacements, while in no way downplaying the offensive and hateful content found in some of the archived material. The ethics of working with such content will be discussed in the following section.

### Ethical considerations

There are substantial ethical considerations when examining hacktivist materials. Research on and with hacktivist material is going to be research on the fringes of the Internet, as the content is only accidentally captured by means of web archiving, if not captured by dedicated archives. Thus, this study offers the opportunity to include little or unheard voices and perspectives into the historical record of the web. Ian Milligan describes the challenge and potential of using abandoned or ownerless web material:

I feel similarly uncomfortable with leaving the voices of everyday people completely outside the historical record when there is ample opportunity to include them. Moving to a full opt-in process [referring to Geocities accounts] would likely lead to the historical record being dominated by corporations, celebrities and other powerful people, tech males, and those wanted their public face and history to be seen a particular way. ()

Relating this to web defacements means to further expand the problem: hacktivist material is found within cybercrime archives, amongst a full range of gory images and defamatory content of all sorts. Not only can consent not be obtained, but the relevant material is hidden amongst plenty of other, largely unrelated websites. One explanation for this can be found returning to Alex Ball, who understands “Anonymity, combined with the ephemeral nature of posts [as a] competition for the limited resource of attention: a driver of excess and the extreme” ().

What is certain, however, is that reliance on one single archive, such as in the case of this study, carries the risk of reinforcing limitations and biases already present in the source material. In other words, we must not forget Attrition’s original function as a community site. This means that material found within the archive is likely created by individuals who felt drawn to this kind of material for a variety of reasons Further, whenever discussing the heterogeneity or diversity of defacements, it must be considered that a certain amount of privilege was necessary to be active on the Web of the early 2000s. This privilege may take the form of available time, equipment, Internet access and English language skills and is oriented along the general lines of privilege and opportunity within a society.

While the defacements used as source material for this article are freely available on GitHub (and before that were freely available on the Attrition.org website), research has to take appropriate precautions and, for example when selecting individual websites for public display, consider the ethical implications. A quantitative approach can mitigate problems with individual websites by focusing on the dataset' metadata.

When dealing with defacements on an individual level, we have to consider that there are at least three distinct layers involved. Each layer represents different types of authorship, different expectations of privacy and publicity and different levels of agency and vulnerability. What all three layers have in common, however, is that none left a signed form expressing their informed consent. These layers will be explored in more detail below and show how ethical considerations when dealing with defaced websites shape the research methodology.

The first layer is the original website, and with it, the original website owner. These people never consented to having their page hacked or their content defaced or altered in any way. For private websites, identifiable information regarding the original owner may be found in the URL or in the content. For example, a defacement may feature content that identifies the original owner of the website. To protect the privacy of these individuals, derived datasets from the Attrition archive must avoid private websites and focus on corporate and governmental website defacements. While there is no clear distinction between private and non-private websites applicable in all cases, this is a precaution that takes into account the increased attention a defacement may receive if it is prominently featured in a dataset.

The second layer is the attacker. While it is safe to assume that defacers seek publicity through their work, it is not possible to retroactively obtain consent. Even though the material in question has been available to the public since 2001, identification of individuals might pose a risk to them. To mitigate any risk in this area, only as much information as attackers are willing to provide about themselves through their work is used and no further steps are taken to identify individuals. The third and final layer is the content. While past research experience has shown that political defacements hardly make use of private information or original media content, it is still possible to come across identifiable information about third parties. Again, any derived dataset must take steps to remove this information for the risk of potentially exposing a third party’s information.

It is at this point necessary to explain important differences between using data obtained through hacking and data about hacking in my research. All content available on Attrition was a public website at one point. I relate this situation to the concept of “expected privacy” as defined by Milligan: “Did the author of the individual website they [researchers] are citing or using as evidence have a reasonable expectation of privacy?” (). Defacers hack websites with the intention of creating as much publicity as possible, even more so when they submit their work to be archived. While it is possible to come across identifiable information, this is not the focus of the study and this information will not become part of the dataset under a data minimization principle.

## Approach and Workflow

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 1: Distribution of content over time."  
 ]  
 }  
 }  
}  
display(Image("media/flowchart.drawio.png", width=1000), metadata=metadata)

### Introduction

The goal of this article is to contextualize and unlock political web defacements for research. To achieve this goal, three distinct steps are necessary:

* To clean and bring the downloaded data into a format usable for a web archive
* To index files and identify trends, themes and actors
* To create derived datasets based on identified content

This workflow is loosely based on the approach described by Emily Maemura who argues that “new frameworks are needed to inform descriptions of these collections [...], beyond characterising datasets, to also include the content of data curation systems, processes, procedures and organizational settings within which collecting takes place.” ()

In this workflow, recreating the entire Attrition archive through WARC files is only an intermediate step. As described by Ruest et al., derivative datasets together with a robust finding aid are of high value for web archive research (). Such derived datasets will be the final output of this described workflow. Step one includes the creation of rudimentary metadata, which is almost entirely absent in the Attrition download. The files provided via GitHub structure defacements as follows:

{Year}/{Month}/{Day}/{URL}/{individual files}

Because of the lack of metadata, defacement date and target URL must be extraced from the folder path through regular expressions.

Web defacements might spread more than political ideology. Either knowingly or unknowingly (by the defacers using infected systems themselves), defacements may be used to spread malware. () This aspect has been a concern in related research on web defacements (), and thus the mitigation of any risk is also relevant for this research. As part of the workflow, data was scanned using [ClamAV](https://www.clamav.net/), which produced a small number of detections. In one case, it seems that the malware found was used to compromise the target system ([CVE-2015-1761](https://cve.circl.lu/cve/CVE-2015-1761)). Other detections were less serious, such as the [Ramen Worm](https://www.f-secure.com/v-descs/ramen.shtml), and were probably unknowingly passed on. While it is none of the detected malware presented a serious threat to a modern system, malware was occasionally found in the downloaded Attrition.org data. Where possible, these files have either been cleaned or removed from the corpus.

### Reading all Files

Some pages in the Attrition archive are not a mirror of the defaced page, but a simple page offering “more info on this hack”, with a link to the actual defacement. Because this phrase would skew results, it is filtered out at a later stage. Where multiple pages were defaced using the same template (a tactic known as mass defacement), Attrition would sometimes provide a separate page with all defaced URLs. These pages also needed to be filtered out through stopwords.

Again reconstructing the metadata, the date for each defacement is derived from the folder structure using a regular expression similar to the solution for finding URLs. This search result is then converted to a datetime object and can be written to the dataframe.

Finally, each HTML document is written to the dataframe including

* The local path
* The extracted and converted date
* The extracted ULR
* The TLD
* The extracted and cleaned text
* All links found on the page

In a second step, all defacements are to be converted into WARC files with fixed datetime (so the actual date of the defacement is included) and fixed URLs (so the original URL becomes part of the WARC).

import glob  
import pandas as pd  
from IPython.display import display  
from bs4 import BeautifulSoup  
import re, datetime  
import os  
import matplotlib.pyplot as plt  
from tld import get\_tld  
import numpy as np

def cleanMe(i): #custom function to extract text from some HTML tags while removing others  
 # parse html content  
 with open(i, encoding='latin-1') as f:  
 content = f.read()  
 soup\_text = BeautifulSoup(content, "html.parser") #using bs4 to parse the HTML  
 for data in soup\_text(['style', 'script', 'code', 'a']): #select and remove these tags - they ususally contain text  
 # Remove tags  
 data.decompose()  
 # return data by retrieving the tag content  
 export = ' '.join(soup\_text.stripped\_strings) #exclude the default text on mirror.html  
 if export == "More info on this hack!":  
 export = "[None]" #use any placeholder value here and remove it with the stopwords. Links will still be extraced for the linkmap  
 return export

files = []  
clean = "None"  
progress = 0  
body\_text = []  
linklist = []  
date = 'NaN'  
url = 'NaN'  
tld = 'NaN'  
text = 'NaN'  
for i in glob.glob('/attrition\_analysis/\*\*\*\*/\*\*/\*\*/\*/\*.html', recursive = False):#set scope here #possible to filter by year, month or file name  
 if os.path.isfile(i) == True: #check if file or directory  
 with open(i, 'r', encoding='latin-1') as infile:  
 url = re.search("w{3}\.\S+\/", i) #use RE to search for url in path  
 if url is not None:  
 tld = get\_tld(url.group(), fail\_silently=True, fix\_protocol=True) #extracting tlds  
 url = url.group()  
 else:  
 tld = "No url extracted"  
 linklist = findURLs(infile) #extracting all links on documents  
 clean = cleanMe(i) #extracting clean text from html  
 match = re.search('\d{4}/\d{2}/\d{2}', i)   
 date = datetime.datetime.strptime(match.group(), '%Y/%m/%d').date() # get dates from folder structure IDEA: extract domain like that #done  
 files.append((i, date, url, tld, clean, linklist))  
 progress +=1  
 print('Processed '+ str(progress) + ' pages.')  
df = pd.DataFrame(files, columns= ['path', 'date', 'url', 'tld', 'text', 'linklist']) #turn everything into Dataframe  
df.to\_csv('file\_AA\_all.csv')

df=pd.read\_csv("https://raw.githubusercontent.com/mkrzmr/jdh/main/script/file\_AA\_sm.csv")  
df.drop('Unnamed: 0', axis=1, inplace=True)  
df.head() #using a smaller dataframe for the article. Full dataframe available on demand

id path date \  
0 0 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-12   
1 1 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-06   
2 2 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-25   
3 3 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-27   
4 4 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-20   
  
 url tld text \  
0 www.kriegsman.com/ com The \*Former\* Kriegsman Fur Page Whoa!! this si...   
1 www.tasr.sk/ sk WWW server Tlacovej Agentury SR bol napadnuty ...   
2 www.army.cz/ cz www.army.cz hacked WWW server Armady Ceskej Re...   
3 www.netlab.sk/ sk www.netlab.sk hacked WWW server Internet Provi...   
4 www.upjs.sk/ sk www.army.cz hacked WWW server Safarikovej Univ...   
  
 linklist \  
0 ['http://envirolink.org/arrs/faqtop.html', 'ht...   
1 ['#10', '#0', '#1', '#2', '#3', '#4', '#5', '#...   
2 ['povodny.html', 'modifikovany.html', 'clanok....   
3 ['povodny.html', 'hacked.html']   
4 ['http://www.upjs.sk', 'hack.html']   
  
 processed\_text   
0 former kriegsman fur page whoa site hit times ...   
1 server tlacovej agentury sr bol napadnuty tejt...   
2 wwwarmycz hacked server armady ceskej republik...   
3 wwwnetlabsk hacked server internet providera n...   
4 wwwarmycz hacked server safarikovej univerzity...

Also, data from the defaced pages is to be extracted and written to a dataframe for further analysis. In the first step, this data includes date, URL, Top-Level-Domain (TLD), all outgoing links and the full content of all HTML files. The general approach is quite simple: Using the YYYY/MM/DD/URL folder structure, the script recursively searches for HTML files using wildcards. For every HTML document found (say, 2000/01/01/www.example.com/index.html), a regular expression search extracts the URL (in this case, www.example.com) and the TLD (in this case, com). The original URL is not present in all cases, so the code needs to be able to account for that and replace a missing URL with "No url extracted".

To be able to analyze links between pages, all referenced URLs within the HTML doc are written to a list. At this step, no distinction is made between internal and external links. External links are identifiable through regular expression (http(s)...) and local links are identifiable through them being relative links (/page2.html). The list of outgoing links on pages can later be used to create a linkmap of the entire collection.

At this point in the processing, the HTML files are parsed as plain text files. This allows for easy search and retrieval of elements such as URLs. For the next step, the text and encoding need to be separated to extract the written text of the defacement. To achieve this, the file content is processed by BeautifulSoup as HTML content, and unwanted tags are removed. Because web defacers do not always use valid HTML, and because of the special situation of defacements, this is a fuzzy process which will not fit every single one of the ~15.000 defaced websites in the collection. Since the extracted text will be processed and filtered further later on, this stage only removes some elements.

### Creating WARC Files

So far, the analysis has been working with the downloaded files and folders. For replay in a web archiving solution, WARC files are necessary. This file format is not only included better suited for the preservation of pages, it also allows embedded metadata. For this step, it is necessary to go back to the file and folder structure one more time. The [warcit tool](https://github.com/webrecorder/warcit) allows us to easily convert local files and folders to WARC files. As the files are local and do not feature any metadata except what has been reconstructed through the previous section, it is necessary to change the information in the warcinfo section so that the WARC files show the correct harvest date and URL for later use:

WARC/1.0  
WARC-Date: 2011-02-01T00:00:00Z  
WARC-Creation-Date: 2017-12-05T18:30:58Z  
WARC-Source-URI: file://./path/to/somefile.html  
WARC-Type: resource  
WARC-Record-ID: ...  
WARC-Target-URI: http://www.example.com/to/somefile.html  
Content-Type: text/html  
Content-Length ...

To process all files, a separate script is necessary which calls the “warcit” tool command on all folders found within the YYYY/MM/DD folder structure. This is largely a repetition of the metadata extraction of the previous section, except the URL is defined as the section after the last slash, so for example in the string “1996/01/01/nasa.gov”, nasa.gov is identified as the URL. The reason for this unconventional approach (since the URLs were already extracted in the previous section and could just be read from the dataframe) is to compensate for the unstructured URL notation in the archive. Some URLs are given in full notation including the www prefix, while others only feature the domain name and TLD. Finally, the command

warcit -o --fixed-dt {date} {fixed URL} {local path})

Is passed on to the system and the output is recorded. This length process produces 14,133 warc.gz files and allows the analysis to leave the file and folder structure behind. Through this first step of the workflow, metadata has been reconstructed from the folder structure including date and target URL. This approach deliberately avoided metadata later added by attrition (such as motivation or single/mass defacement), but instead derived its descriptive metadata from the files themselves. The output of this first step so far is a collection of WARC files with embedded metadata in the warcinfo section, and a dataframe containing metadata and extracted HTML content.

for i in glob.glob('/attrition\_analysis/\*\*\*\*/\*\*/\*\*/\*', recursive = False): #again, scope can be set here  
 match = re.search('\d{4}/\d{2}/\d{2}', i)  
 date = datetime.datetime.strptime(match.group(), '%Y/%m/%d').date() # get dates from folder structure  
 url = i.split('/') #silly but works  
 url.reverse()  
 command = str('warcit -o --fixed-dt '+str(date)+' http://'+str(url[0])+'/'+' '+i)  
 print(command)   
 #res = os.system(command) #the command to convert local files into WARC files must run outside of Jupyter #uncomment and run locally  
 print("Returned Value: ", res) #fixed output file name

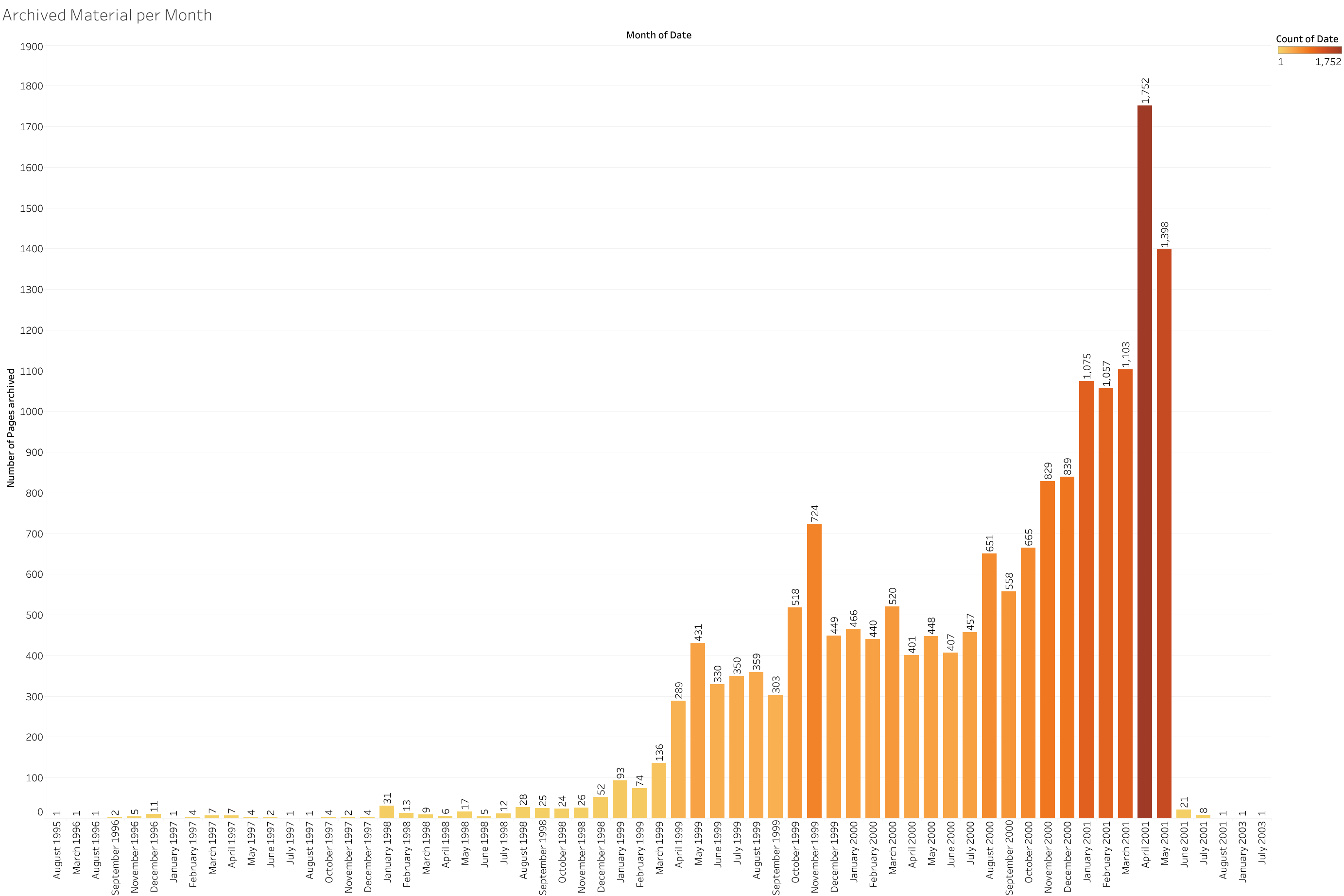
### Indexing and First Analysis

The output of this second step is a collection of usable WARC files to then replay in a Web archive solution ([SolrWayback](https://github.com/netarchivesuite/solrwayback)). Finally, all created WARC files can be indexed through a Solrwayback server. Solrwayback offers advanced search and analysis tools for web archives and also provides us with the ability to easily create derived datasets for further analysis. Indexing a larger number of WARC files through Solrwayback is a lengthy process depending on the hardware used, but once completed, the entire Attrition file collection has become an archive:

In terms of unlocking web defacements, this already is a significant contribution as it brings web defacements into a format usable for web archive research. However, political web defacements are a minority type of content in this large collection, accompanied by simple cyber vandalism, and other content unrelated to the research objectives. Furthermore, while the current state allows searching for domains and keywords, it remains somewhat difficult to judge the actual content of this defacement archive from the outside. Additionally, web defacements have unique features which are not attributed for by Solrwayback or any other web archive browsing application, such as attribution of attacks for defacer groups or individuals. To provide finding aids specific to web defacements, and to identify trends in content over time is the aim of the next step of the workflow.

Before delving into the specific analysis, it is important to remember just how much the content of this archive is skewed by the collection practices which produced it. To illustrate this, a visualization shows the distribution of material over time:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 1: Distribution of content over time."  
 ]  
 }  
 }  
}  
display(Image("media/Sheet 1.png", width=1000), metadata=metadata)



This graph shows the distribution of content over time. The oldest archived page dates back to August 1995, while the majority of material was collected in 2001. The graph also confirms Attrition’s narrative of being overwhelmed by submissions and therefore deciding to close the archive. While it is generally rare to find pre-2000 material in a web archive - and even more so in a community-maintained archive of defaced websites - the observed discrepancy in content distribution must be kept in mind when discussing any developments over time. Even when using relative measures such as frequency to describe how defacements change in reaction to geopolitical events, the very low number of early defacements and the sharp drop in post-2001 defacements must be considered.

Another useful way to visualize the collection is through a link network. This is possible since the first step of the analysis indexed all links to external sites, and email-addresses. The use of hyperlink network analysis for the analysis of large datasets has been described by Waldherr et. al (). Additionally, the application of hyperlink analysis for the attribution of political ideology has been described by de Bakker and Hellsten (). Finally, the use of link network graphs for the analysis of political content in archived web content has been described by Brügger (). The authors agree that hyperlink analysis is a useful tool for the analysis of web content.

The below graph shows a directed graph of all external link targets, with a minimum of 5 incoming links. The below graph shows the links throughout the entire dataset, irrespective of the date of the website. Clusters are highlighted by the color scheme. Using gephi and VOSviewer, the entire link network can be visualized as an [interactive page](https://app.vosviewer.com/?json=https://raw.githubusercontent.com/mkrzmr/jdh/main/media/network/VOSviewer-network.json&dark_ui=True):

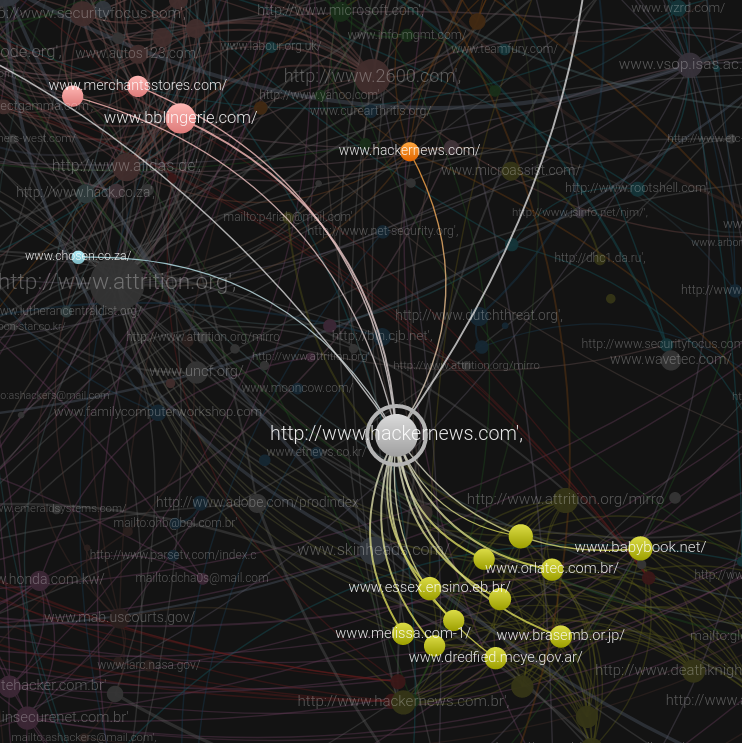
from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 2: Directed incoming link network of the archive content, size indicates number of incoming links, clustered, minimum connectivity = 5"  
 ]  
 }  
 }  
}  
display(Image("media/network/VOSviewer\_full.png", width=1000), metadata=metadata)



This approach is helpful in understanding the collection in multiple ways. First, it shows what external links are often referred to in defacements, thus indicating popular topics within defacements. Clusters showing in the above graph are, for example, relating to the Kashmir conflict (green, top right section) and Palestine (red, middle left section). Further, some nodes are not obviously political, but hacker-related news sites and e-mail addresses. Taking a closer look at these connections, two distinct types emerge:

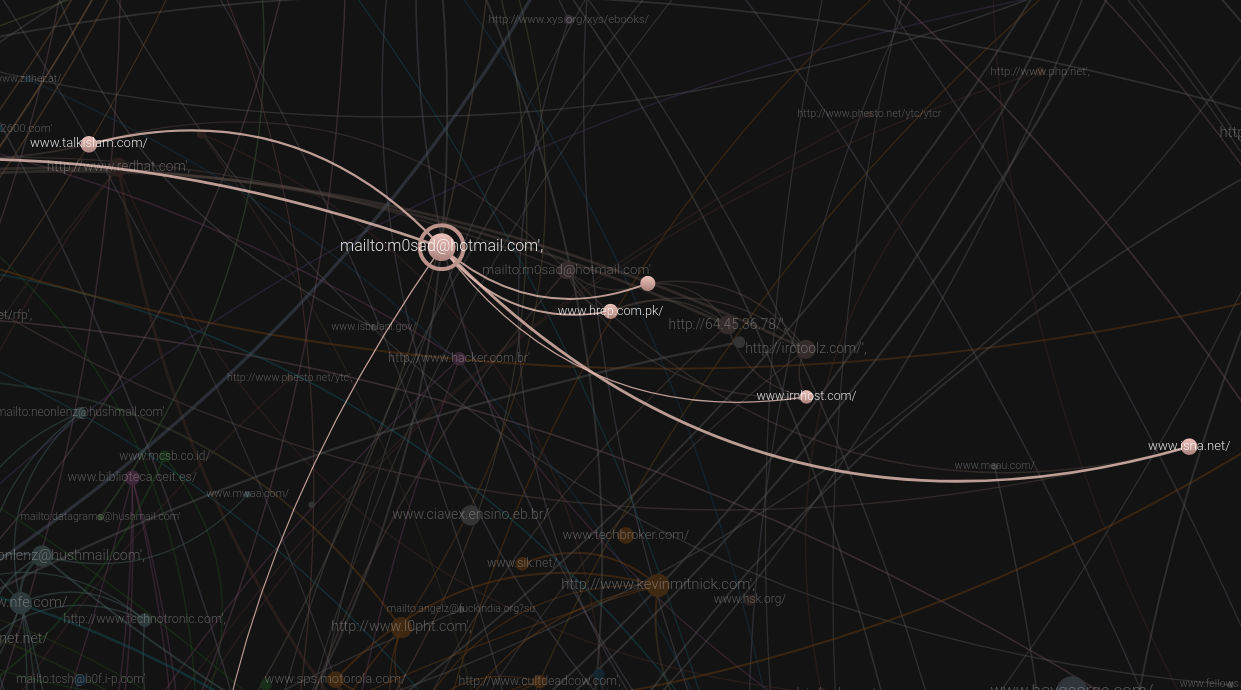
* URLs (such as hackernews.com, a hacker news website and magazine, but also web sites of defacers)

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 3: Link network detail for hackernews.com"  
 ]  
 }  
 }  
}  
display(Image("media/network/hackernews.png", width=1000), metadata=metadata)



E-Mail addresses, usually left by the attacker. This may seem surprising at first, but giving the administrator a way to learn and prevent further intrusions is a recurring element especially in political hacking. More information on this element can be found in the next section.

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 3: Link network detail for a defacer's email address"  
 ]  
 }  
 }  
}  
display(Image("media/network/email.png", width=1000), metadata=metadata)



These links allow for some attribution of attacks and community alignment of the attacker. Especially when referring to the same external links, be it websites belonging to political movements or scene-specific news outlets such as hackernews.com, it can be assumed that attackers are part of the same movement or scene. In some cases this alignment is quite clear, such as when pages link to anti-torture websites (stopthetorture.org), in some cases this assessment is more indirect as in the case of news websites whose political alignment needs to be understood.

There is a general distinction apparent in the visualization of the archive: While one type of defaces web sites has a surprisingly high level of incoming connections, the other appears to be largely isolated with no or only one or two incoming links. This second type forms clusters along the edge of the network, while the more connected type forms the center of the network.

This approach of creating a link network between the archived pages is useful for a quick visualization of the archive contents. It also makes use of the low level of interlinkage between defaced web pages and the fact that defacers generally avoid 3rd-party content. These conditions make the few existing link connections more valuable and, in many cases, indicate alignment either politically or socially. Where E-Mail addresses are given, these also help attribute the defacement. In lieu of descriptive metadata, this link network approach reads hypertext elements as indicators of relation between authors.

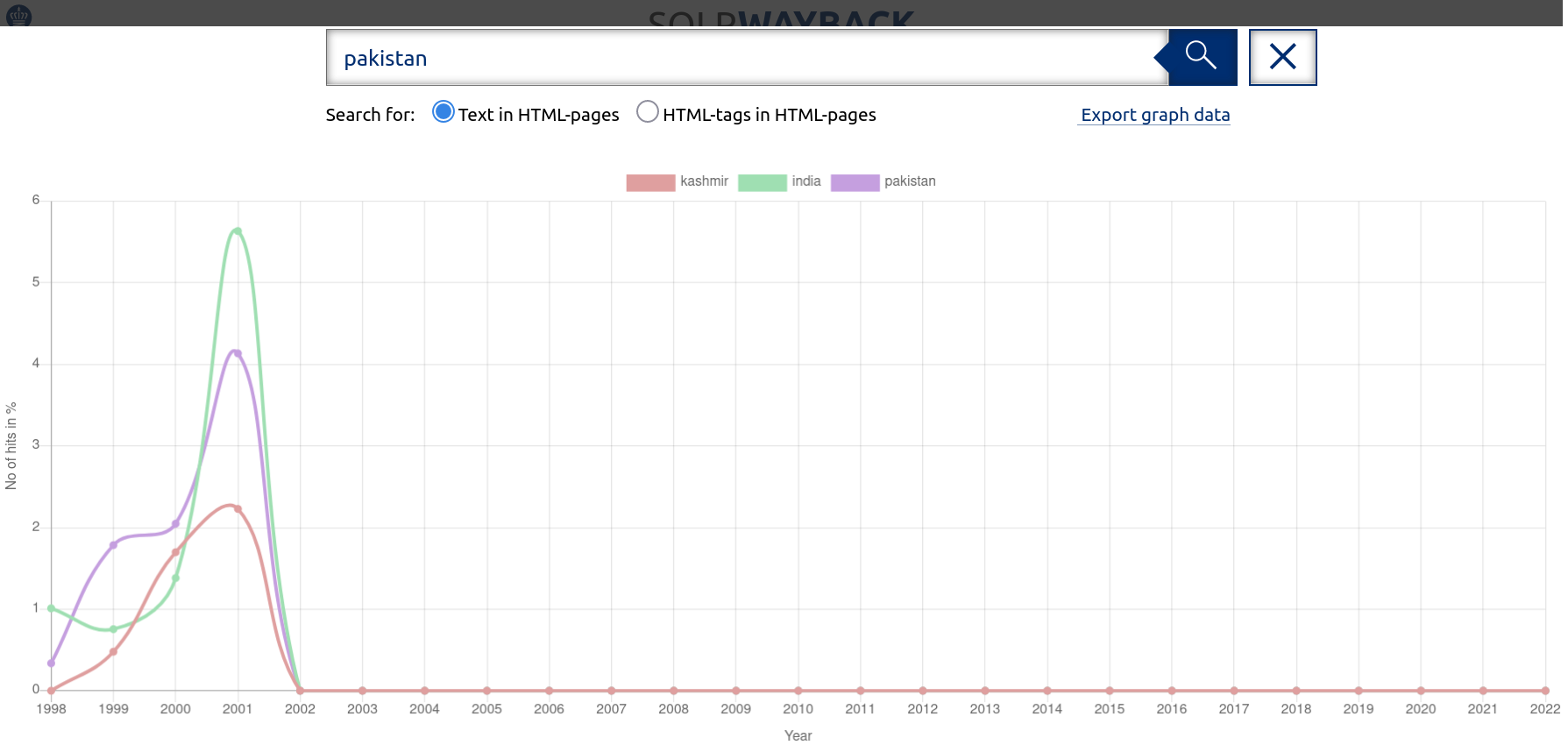
As mentioned, the data in the Attrition.org archive spans the time period between August 1995 and July 2003.

The final step of the workflow is the creation of derivative datasets for further analysis and more focussed research. These datasets are created within SolrWayback, using the supplied tools that come with the software package. While it would be possible to create derivative datasets manually, the objective of this step in the workflow is to provide researchers with easy access to a collection of WARC files appropriate to their research interests.

It has been confirmed in literature that web defacers often act in response to geopolitical events (, , ). Based on related research into hacktivism as a means of political expression (), the keywords “Kashmir”, “India” and “Pakistan '' are used in SolrWayback’s n-gram tool. These keywords relate to the Kashmir conflict. The Kashmir conflict is a territorial dispute between India, Pakistan and China over the Kashmir region. The conflict arose from the 1947 partition of India and has, with varying intensity, continued until today. This ongoing violent conflict has left its marks in the cultural history of Pakistan and India. It has also led to an ongoing cyber war between the two countries, a part of which features web defacements. The selected set of defacements consists of in-scope material relating to the Kashmir conflict. Reflecting on the value of event-based web archives () , the availability of material relating to the Kashmir conflict offers the opportunity to complement the historical record of the accompanying cyberwar. Most defacements are from 1999 to 2002 with some outliers. For a general history of the region and conflict, see (, , ). For the role of digital communication in the conflict, see (, ).

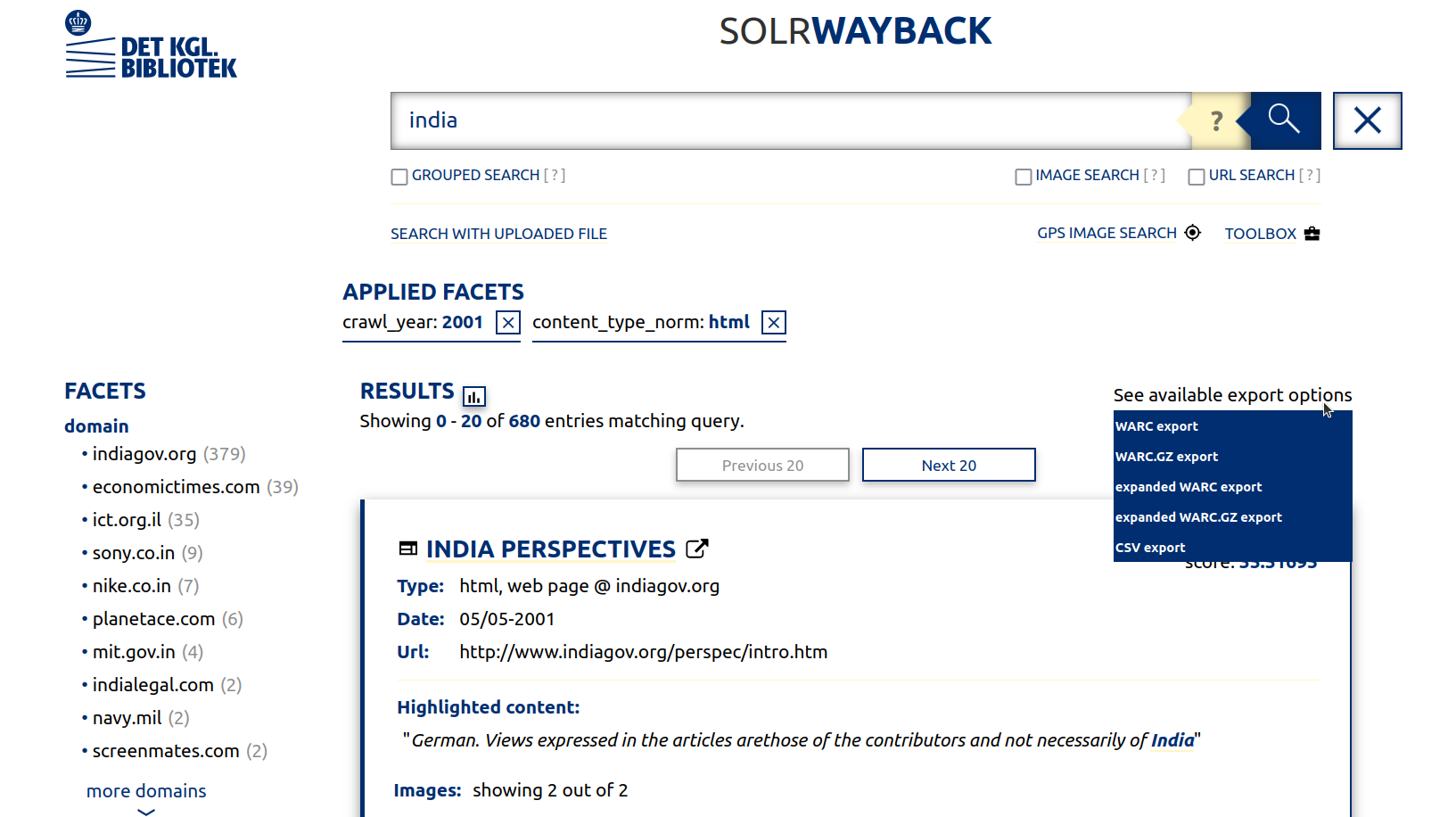
The result confirms the existence of these keywords in the Attrition archive:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 4: Frequencies for keywords relating to the Kashmir conflict over time"  
 ]  
 }  
 }  
}  
display(Image("media/distributionPakistanInadiKashmir.png", width=1000), metadata=metadata)



The results show that all three keywords are present from 1999 on (“India” and “Pakistan” also found in 1998 defacements), and that their usage increases over time to peak in 2001. The sudden drop in 2002 is again owed to the dataset structure - see the previous section on content distribution in the Attrition archive. The fact that all three pathways show similar trajectories indicates that these keywords relate to the Kashmir conflict rather than other geopolitical issues. While the presentation of results is far from ideal - SolrWayback continues the X-Axis into the present despite having no post-2002 material - the real value of this n-gram visualization lies in the connection between the graph and the archived material. Clicking on any data point opens a new window showing all material belonging to the data point. Moreso, the results can be exported to create a derived dataset for further research.

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 5: Exporting a derived dataset from SolrWayback"  
 ]  
 }  
 }  
}  
display(Image("media/SWBIndia.png", width=1000), metadata=metadata)



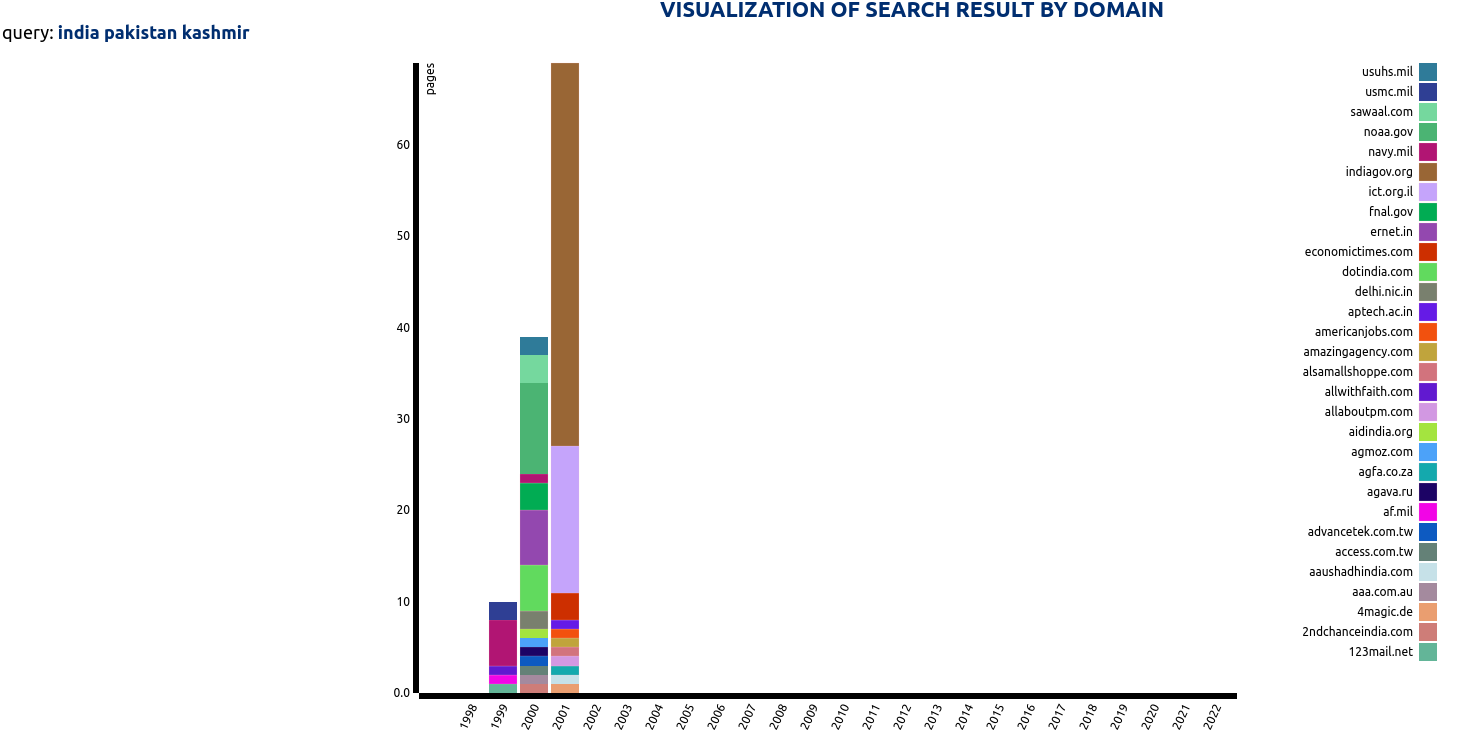
Extending the regular search function, the n-gram visualization allows to quickly scope the use of one or multiple expressions across the archive. Using the relative frequency rather than the mere count of the expression allows to mitigate the very uneven distribution of content over time in the archive. Through the export function, researchers can easily create any derived dataset for further analysis.

One of the characteristics of web defacements is their operation on different layers. While the content layer certainly is relevant, and largely responsible for the reception of the defacer’s message, the choice of target and with it the technical ability to carry out a defacement add another layer to web defacements. Broadly speaking, targets may be chosen based on the technical challenge presented (easy or hard to compromise the target system) and the target’s relationship to the message. When hackers choose targets based on the technical challenge, so-called mass defacements of outdated servers are often the result. See () for a breakdown of the potential targets presented by outdated Wordpress installations. On the other hand, websites with strong security setups might gain the attacker additional fame within the hacking community. Regarding the relationship between the target and the defacer, three general categories emerge:

* The target is entirely unrelated to the message and its selection was either the result of opportunistic behavior or the pursuit of a reputation as a skilled hacker.
* The target’s Top-Level-Domain (TLD) is related to the defacer’s message (such as the website of an Indian business in the context of the Kashmir conflict), while the content is not. These targets are often chosen because of their assumed allegiance with the perceived enemy.
* The target’s TLD and content is clearly related to the message, such as Indian or Pakistani government websites in the context of the Kashmir conflict. Here, the defacement serves a double purpose: It shows the vulnerability of perceived enemy infrastructure and propagates a message.

Because this distinction is an important part of understanding the different levels on which web defacements operate, the breakdown of target TLDs is important for analysis. Similar to the cluster search, this breakdown is part of SolrWayback’s tool kit:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 6: Visualization of search results by domain"  
 ]  
 }  
 }  
}  
display(Image("media/SWBTldDist.png", width=1000), metadata=metadata)



The results show a high level of focus on government domains such as indigov.org and ict.gov.il, yet also indicate a change of targets between 1999 and 2001. This is indicative of the mentioned level web defacements operate on. Looking at the domains, a few US domains were also defaced. This might be an indicator of who defacers target their work towards. Defacing US domains might be done in an attempt to rally Western support for their cause, while defacing Indian domains might be done in an attempt to attack a perceived enemy’s infrastructure.

This brief initial overview shows some of the content of the Attrition.org archive. It outlines a simple and easily reproducible way for researchers to either use the existing SolWayback installation to browse the archive, or to filter according to their research interests and export derived datasets. This makes the project the first academically usable web archive of defaced content.

It is, however, not always easy to contextualize defacements without further sources. Especially parodies are hard to detect when the original page is no longer available. As one of the secondary goals of this project is to foster integration of web defacements with established web archives, the following section will explorer the possibility of using automated requests to retrieve a copy of the original web site shortly before or after the defacement.

### Further contextualizing defacements through source combination

As the previous section has indicated, web defacements operate on different levels (for a basic classification, defacements can distinguish themselves through the technical expertise demonstrated and the content displayed). As much as SolrWayback comes with a range of tools to explore and visualize collections of WARC files, SolrWayback does not specialize on a specific type of archived web content. Since web defacements need additional contextual information to be used in research, and these information are neither accessible through Solrwayback’s interface nor are they part of the Attrition archive, the article at this point will demonstrate the combination of different web archives to contextualize defacements.

The underlying problem in further contextualizing defacements is that the original page is usually not preserved in specialized defacement archives and the defaced page usually is not preserved in other, public web archives. Especially defacements which interact with the original page through parody etc. are very hard to understand if the original page is not available. Using the example of the labour.org.uk website defacement, the use of APIs to combine sources and contextualize the defacement so that it can be used for research on the history of the Labour party website, hacktivism and the political communication through hacking.

To begin, this is the defaced website in playback through SolrWayback:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 7: The defaced Labour.org.uk website"  
 ]  
 }  
 }  
}  
display(Image("media/SWBLabourHacked.png", width=500), metadata=metadata)



What is remarkable about this defacement is its age (dated December 12th, 1996), and its unusual complexity. Each of the links seems to lead to different sub-pages addressing various issues the unknown author saw with the 1996 Labour-led government under Tony Blair. Of these links, only “The Road to nowhere” and “Policies - New Labour, New Britain” are available in the attrition archive. The defacement as a whole appears to be a parody of an original website, mimicking the layout and imagery used in an official party website. Without the original, however, this remains speculative.

SolrWayback provides an API to easily send queries to the server:

The API for linking to and browsing archived webpages is the same as for Internet Archive:

* Internet Archive: <https://web.archive.org/web/20080213093319/http://www.statsbiblioteket.dk/>
* SolrWayback: <http://server/solrwayback/services/web/20140515140841/http://statsbiblioteket.dk/>

Using this structure, it is possible to send a query to the Internet Archive to try and retrieve the nearest available snapshot. In this case, the snapshot is dated November 9th, 1996:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 8: The original website"  
 ]  
 }  
 }  
}  
display(Image("media/SWBLabourOrg.png", width=500), metadata=metadata)



Comparing the two versions, it is confirmed that the defacement is a parody of the original site, mimicking the structure and content of the original. How long this defaced version was online is hard to say, the next available snapshot in the Internet Archive dates to March 1997 and shows a blank page. The next confirmed change in content dates to April 1997, where it appears the site has been taken offline. This example further shows how defacement archives can enrich existing web archives and help fill in gaps in a website’s history. Both the defacement archives as well as the Internet Archive (or any web archive) are complementary sources in this example. This approach can be automated to try and retrieve the closest copy available for every defacement, however it is slow and only recommended in targeted defacements. Where derived datasets are used, and where the retrieval of the original sites is not constrained by bandwidth and access restrictions (such as in the case of a national web archive), this approach may well be automated.

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 9: Comparison of the two versions"  
 ]  
 }  
 }  
}  
display(Image("media/SWBLabourOrg.png", width=500), metadata=metadata)  
display(Image("media/SWBLabourHacked.png", width=500), metadata=metadata)

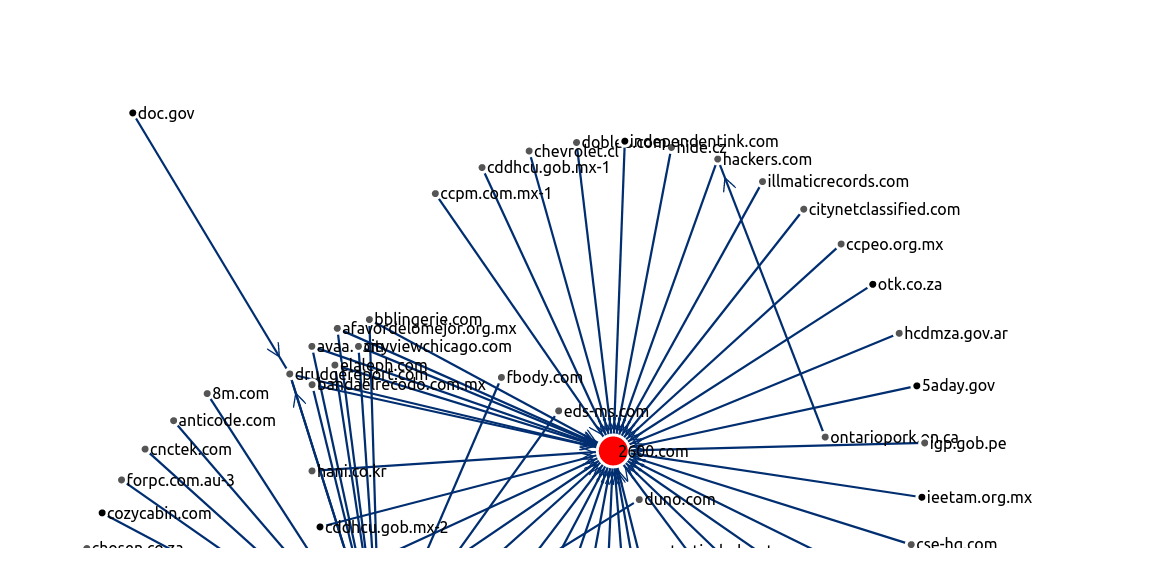




### Attributing Authorship and finding related material

In many cases, including the labour.org.uk defacement, it seems a logical next step to identify the author of the defacement and search the archive for related content. In relation to the ethics statement, this section will briefly investigate the attribution of authorship and the search for related material within the existing archive. In the case of the labour.org.uk defacement, the modified link “Hacking to be Heard” (Figure X, bottom right), seems to be a promising indicator towards the identity of the defacer. It is, however, just a link to 2600.com, a Hacking zine still in operation. Using SolrWayback’s link graph function, a network of all sites in the Attrition archive pointing links towards 2600.com can be created:

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 10: Link network relating to 2600.com"  
 ]  
 }  
 }  
}  
display(Image("media/SWB2600.png", width=1000), metadata=metadata)



While there is no certainty that the unknown author of the labour.org.uk defacement was in any way affiliated with 2600.com, this list of other websites also containing a link to 2600.com is a good starting point for the identification of related content. This problematic of attributing authorship or finding related content illustrates a general distinction between defacers: While some choose to remain anonymous like the author of the labour.org.uk defacement, others seek public attention with a curated public profile. One defacement reads

Nothing was deleted, Just renamed. Except the logs! Lol!!

This claim that no harm was done to the existing files is complemented by an invitation to contact the defacer in order to learn more about how the server was attacked:

Admin, solution for this site u find in [jshalom\_bhs@yahoo.com.br](mailto:jshalom_bhs@yahoo.com.br). Together with more contact information:

We're in BRASnet #BHS /server irc.unisys.com.br /join #BHS

This approach is quite common for defacers who choose to maintain a recognizable public profile. The claim to have done no harm to the target serves as a public expression of alignment with a community’s rules, while the extended contact information indicates a willingness to communicate beyond the mono-directional defacement. To take this public presence even further, some groups maintained websites or even gave interviews to various zines. () In these cases, attribution of ownership is easily possible by simply searching for the respective group name - BHS returns 149 defaced pages. Beyond that, related material may also be found by cross referencing known defacer groups (Gforce, sometimes called Gforce Pakistan is another high-profile defacement group with over 1200 results in the attrition archive). The limitations of this approach, be it through shared use of links or cross-referencing, are that it only returns results where defacers choose to be identifiable through their public profiles. The next section will therefore provide an outlook to further possible approaches.

### Natural Language Processing and Topic Modeling

So far, the analysis of the Attrition archive has been concerned with, firstly, creating the archive and transforming it into a usable format and secondly, the introduction of basic concepts to analyze web defacements as a form of political communication. In these analytical efforts, the approach has been to cross-reference existing material (such as geopolitical events, defacer pseudonyms or known domains). While this approach is an adequate first step in the engagement with defacements as a source of information, it is obviously prone to overlook other, unknown topics within the archive.

While SolrWayback comes with a range of tools available for the analysis of archived web material - some of which have been used throughout this article - it is not by any means a topic modeling software. This final section will therefore present first steps towards an automated analysis of the corpus, with the goal of identifying previously unknown topic clusters and defacers. For this part of the analysis, it is necessary to return to the data frame created earlier. The goal is to create a list of recognized named entities for each page, to create the foundation for further analysis and especially topic modeling approaches. The existing data frame for each HTML file contains: The text as extracted from the HTML The text in all lowercase The text in all lowercase and cleared of all punctuation

Using Spacy, two functions are created. The first outputs recognized entities and their labels - this is relevant for any custom named entity recognition (NER) projects. The second function outputs the entities, which is relevant for further analysis. For the scope of this article, the Spacy module “en\_core\_web\_lg” is used to output a list of entities based on the lowercase text. It is important to increase

nlp.max\_length = 3000000

To allow processing of the large corpus. This requires a system with 32GB ram to complete.

def ner(i):  
 entlist = []  
 doc = nlp(str(i)) #reduce scope for testing  
 for ent in doc.ents:  
 entlist.append((ent.text,ent.label\_))  
 return (entlist)  
  
def pure\_ent(i):  
 entlist = []  
 doc = nlp(str(i)) #reduce scope for testing  
 for ent in doc.ents:  
 entlist.append(ent.text)  
 return (entlist)

#!python3 -m spacy download en\_core\_web\_lg   
import spacy  
nlp=spacy.load('en\_core\_web\_lg')  
nlp.max\_length = 3000000 #increase max length for text, needs about 1 GB per 100.000 characters  
df['lowercase'] = df['text'].str.lower()  
df['ner\_ent'] = df['lowercase'].astype(str).apply(lambda x:pure\_ent(x)) #this only lists the entities, not the labels. Change depending on use case

#df.drop('Unnamed: 0', axis=1, inplace=True)  
df.head(100)

id path date \  
0 0 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-12   
1 1 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-06   
2 2 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-25   
3 3 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-27   
4 4 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-20   
.. .. ... ...   
95 95 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
96 96 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
97 97 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
98 98 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
99 99 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
  
 url tld \  
0 www.kriegsman.com/ com   
1 www.tasr.sk/ sk   
2 www.army.cz/ cz   
3 www.netlab.sk/ sk   
4 www.upjs.sk/ sk   
.. ... ...   
95 www.kalsel.go.id/ go.id   
96 www.tribuna.com.mx/ com.mx   
97 www.tribuna.com.mx/ com.mx   
98 www.i-mg.com/ com   
99 www.alaskayellowpages.com/ com   
  
 text \  
0 The \*Former\* Kriegsman Fur Page Whoa!! this si...   
1 WWW server Tlacovej Agentury SR bol napadnuty ...   
2 www.army.cz hacked WWW server Armady Ceskej Re...   
3 www.netlab.sk hacked WWW server Internet Provi...   
4 www.army.cz hacked WWW server Safarikovej Univ...   
.. ...   
95 ps-2 and kni own j00 hahahahahaha....... we're...   
96 ::(defaced by the dhc):: defaced by the dhc (n...   
97 Multiple Domains! Multiple domains were hit wi...   
98 another spammer bites the dust closed by for s...   
99 RSH: oh Visit we're not only\nabout defacing. ...   
  
 linklist \  
0 ['http://envirolink.org/arrs/faqtop.html', 'ht...   
1 ['#10', '#0', '#1', '#2', '#3', '#4', '#5', '#...   
2 ['povodny.html', 'modifikovany.html', 'clanok....   
3 ['povodny.html', 'hacked.html']   
4 ['http://www.upjs.sk', 'hack.html']   
.. ...   
95 []   
96 ['http://dhcorp.cjb.net', 'mailto:auto45040@hu...   
97 ['mirror.html']   
98 ['mailto:utreg@surimaribo.com?subject=Good Job...   
99 ['http://rsh.defacements.com', 'mailto:piffyco...   
  
 processed\_text \  
0 former kriegsman fur page whoa site hit times ...   
1 server tlacovej agentury sr bol napadnuty tejt...   
2 wwwarmycz hacked server armady ceskej republik...   
3 wwwnetlabsk hacked server internet providera n...   
4 wwwarmycz hacked server safarikovej univerzity...   
.. ...   
95 kni hahahahahaha antihackerlink greetz hc ch...   
96 defaced dhc defaced dhc nemesystm visit questi...   
97 hit web page defacementdomainswwwtribunacommxw...   
98 another spammer bites dust closed sending unso...   
99 rsh oh visit onlyabout defacing long time sinc...   
  
 lowercase \  
0 the \*former\* kriegsman fur page whoa!! this si...   
1 www server tlacovej agentury sr bol napadnuty ...   
2 www.army.cz hacked www server armady ceskej re...   
3 www.netlab.sk hacked www server internet provi...   
4 www.army.cz hacked www server safarikovej univ...   
.. ...   
95 ps-2 and kni own j00 hahahahahaha....... we're...   
96 ::(defaced by the dhc):: defaced by the dhc (n...   
97 multiple domains! multiple domains were hit wi...   
98 another spammer bites the dust closed by for s...   
99 rsh: oh visit we're not only\nabout defacing. ...   
  
 ner\_ent   
0 [kriegsman inc., today, today, millions, today...   
1 [www, agentury, sr bol, tejto, 1996, skupine s...   
2 [www.army.cz, www, armady ceskej, bol hacknuty...   
3 [www.netlab.sk, www, netlab s.r.o., kopia doma...   
4 [www.army.cz, www, univerzity, kosiciach, kopi...   
.. ...   
95 [kni, hahahahahaha, antihackerlink greetz, chi...   
96 [the night, one, nemesystm]   
97 []   
98 [ksoze, theregister.co.uk, domineevisser, carl...   
99 [hitler]   
  
[100 rows x 10 columns]

df.to\_csv("file\_lda\_all\_ner.csv") #saving with NER results

The result is shown in the added ner\_ent column in the data frame. As can be seen, the results are varied, based on the variety of language used in the defacements. On some occasions, the default Spacy package produces good results, i.e. the extracted entities reflect the content of the defaced page well. In other cases, too much or too little information is extracted. As Spacy allows users to build custom NER models, this approach offers potential to be extended into a custom-build NER models for web defacements. Yet even at this early stage, default NER packages can produce usable results for analysis of the archive.

The described approach can be extended through the use of topic modeling to produce a more comprehensive overview of the corpus. The goal at this stage is to be able to identify themes and topics within the corpus. Since the playback of the produced WARC files is provided by SolrWayback, all identified topics or combination of topics can be used as search terms. As in the case of the Named Entity Recognition analysis step, this step is somewhat hardware-intensive and performance may vary based on the installation and hardware. For the context of this paper, a smaller subset of the corpus will be used to demonstrate the topic modeling approach using Latent Dirichlet allocation (lda). Lda is a standard approach for topic modeling and can be used to extend the described workflow () . The configuration of the model parameters is largely kept to defaults, and only stopwords are only filtered out in cases where they obviously interfere with the analysis. To increase accessibility and reduce barriers of access, this topic modeling approach can also be undertaken through existing software solutions such as the [DARIAH Topics Explorer](https://github.com/DARIAH-DE/TopicsExplorer).

df\_small = df.iloc[:1000,:] #create a smaller dataframe to quickly run the code in the notebook. For full analysis, it is better to save and load the model instead or re-creating it.

df\_small['processed\_text'] = 'NaN' #add empty column for processed text  
df\_small.head()

id path date \  
0 0 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-12   
1 1 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-06   
2 2 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-25   
3 3 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-27   
4 4 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-20   
  
 url tld text \  
0 www.kriegsman.com/ com The \*Former\* Kriegsman Fur Page Whoa!! this si...   
1 www.tasr.sk/ sk WWW server Tlacovej Agentury SR bol napadnuty ...   
2 www.army.cz/ cz www.army.cz hacked WWW server Armady Ceskej Re...   
3 www.netlab.sk/ sk www.netlab.sk hacked WWW server Internet Provi...   
4 www.upjs.sk/ sk www.army.cz hacked WWW server Safarikovej Univ...   
  
 linklist processed\_text \  
0 ['http://envirolink.org/arrs/faqtop.html', 'ht... NaN   
1 ['#10', '#0', '#1', '#2', '#3', '#4', '#5', '#... NaN   
2 ['povodny.html', 'modifikovany.html', 'clanok.... NaN   
3 ['povodny.html', 'hacked.html'] NaN   
4 ['http://www.upjs.sk', 'hack.html'] NaN   
  
 lowercase \  
0 the \*former\* kriegsman fur page whoa!! this si...   
1 www server tlacovej agentury sr bol napadnuty ...   
2 www.army.cz hacked www server armady ceskej re...   
3 www.netlab.sk hacked www server internet provi...   
4 www.army.cz hacked www server safarikovej univ...   
  
 ner\_ent   
0 [kriegsman inc., today, today, millions, today...   
1 [www, agentury, sr bol, tejto, 1996, skupine s...   
2 [www.army.cz, www, armady ceskej, bol hacknuty...   
3 [www.netlab.sk, www, netlab s.r.o., kopia doma...   
4 [www.army.cz, www, univerzity, kosiciach, kopi...

#more imports are needed  
import nltk  
from nltk.stem import PorterStemmer  
from nltk.stem.wordnet import WordNetLemmatizer  
from nltk.corpus import stopwords  
nltk.download('stopwords')   
import spacy  
import string  
import gensim  
from gensim.utils import simple\_preprocess  
import pyLDAvis.gensim\_models  
stop\_words = stopwords.words('english') #select English stopwords here  
#import nlp libraries for further text processing

[nltk\_data] Downloading package stopwords to /home/mk/nltk\_data...  
[nltk\_data] Package stopwords is already up-to-date!

# Load spacy  
nlp = spacy.load('en\_core\_web\_lg')  
  
def process\_text(text, stem="None"): #text cleaning and pre processing  
  
 final\_string = ""  
  
 # Make lower  
 text = text.lower()  
  
 # Remove line breaks  
 # Note: that this line can be augmented and used over  
 # to replace any characters with nothing or a space  
 text = re.sub(r'\n', '', text)  
  
 # Remove punctuation  
 translator = str.maketrans('', '', string.punctuation)  
 text = text.translate(translator)  
  
 # Remove stop words  
 text = text.split()  
 useless\_words = nltk.corpus.stopwords.words("english")  
 useless\_words = useless\_words + ['hi', 'im', 'from', 'subject', 're', 'edu', 'use', 'xd', 'xd', 'mih', 'xc', 'xa', 'com', 'xe', 'gif', 'n' ,'www' ,'xb','jpg', 'oz', 'directory', 'file', 'bytes', 'inet', 'pub', 'htm', 'wwwroot', 'mix', 'html,', 'www', 'multiple','domains'] #adjust here  
  
 text\_filtered = [word for word in text if not word in useless\_words]  
  
 # Remove numbers  
 text\_filtered = [re.sub(r'\w\*\d\w\*', '', w) for w in text\_filtered]  
  
 # Stem or Lemmatize  
 if stem == 'Stem':  
 stemmer = PorterStemmer()   
 text\_stemmed = [stemmer.stem(y) for y in text\_filtered]  
 elif stem == 'Lem':  
 lem = WordNetLemmatizer()  
 text\_stemmed = [lem.lemmatize(y) for y in text\_filtered]  
 elif stem == 'Spacy':  
 text\_filtered = nlp(' '.join(text\_filtered))  
 text\_stemmed = [y.lemma\_ for y in text\_filtered]  
 else:  
 text\_stemmed = text\_filtered  
  
 final\_string = ' '.join(text\_stemmed)  
  
 return final\_string

df\_small['processed\_text'] = df\_small['text'].astype(str).apply(lambda x:process\_text(x)) #use astype(str) to only apply to str values

df\_small['data\_words'] = df\_small['processed\_text'].astype(str).values.tolist()

def sent\_to\_words(sentences):  
 for sentence in sentences:  
 # deacc=True removes punctuations  
 yield(gensim.utils.simple\_preprocess(str(sentence), deacc=True)) #using gensim simple preprocessing

df\_small['data\_words\_clean'] = df\_small['data\_words'].apply(lambda x: gensim.parsing.preprocessing.remove\_stopwords("".join(x))) #custom stopwords can be removed here

data\_words = list(sent\_to\_words(df\_small['data\_words\_clean'])) #select source here. In our case, use the preproceesed data\_words\_clean

df\_small.head(100)

id path date \  
0 0 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-12   
1 1 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-06   
2 2 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-25   
3 3 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-27   
4 4 /home/michael/Desktop/attrition\_analysis/1996/... 1996-11-20   
.. .. ... ...   
95 95 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
96 96 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
97 97 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
98 98 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
99 99 /home/michael/Desktop/attrition\_analysis/2000/... 2000-11-19   
  
 url tld \  
0 www.kriegsman.com/ com   
1 www.tasr.sk/ sk   
2 www.army.cz/ cz   
3 www.netlab.sk/ sk   
4 www.upjs.sk/ sk   
.. ... ...   
95 www.kalsel.go.id/ go.id   
96 www.tribuna.com.mx/ com.mx   
97 www.tribuna.com.mx/ com.mx   
98 www.i-mg.com/ com   
99 www.alaskayellowpages.com/ com   
  
 text \  
0 The \*Former\* Kriegsman Fur Page Whoa!! this si...   
1 WWW server Tlacovej Agentury SR bol napadnuty ...   
2 www.army.cz hacked WWW server Armady Ceskej Re...   
3 www.netlab.sk hacked WWW server Internet Provi...   
4 www.army.cz hacked WWW server Safarikovej Univ...   
.. ...   
95 ps-2 and kni own j00 hahahahahaha....... we're...   
96 ::(defaced by the dhc):: defaced by the dhc (n...   
97 Multiple Domains! Multiple domains were hit wi...   
98 another spammer bites the dust closed by for s...   
99 RSH: oh Visit we're not only\nabout defacing. ...   
  
 linklist \  
0 ['http://envirolink.org/arrs/faqtop.html', 'ht...   
1 ['#10', '#0', '#1', '#2', '#3', '#4', '#5', '#...   
2 ['povodny.html', 'modifikovany.html', 'clanok....   
3 ['povodny.html', 'hacked.html']   
4 ['http://www.upjs.sk', 'hack.html']   
.. ...   
95 []   
96 ['http://dhcorp.cjb.net', 'mailto:auto45040@hu...   
97 ['mirror.html']   
98 ['mailto:utreg@surimaribo.com?subject=Good Job...   
99 ['http://rsh.defacements.com', 'mailto:piffyco...   
  
 processed\_text \  
0 former kriegsman fur page whoa site hit times ...   
1 server tlacovej agentury sr bol napadnuty tejt...   
2 wwwarmycz hacked server armady ceskej republik...   
3 wwwnetlabsk hacked server internet providera n...   
4 wwwarmycz hacked server safarikovej univerzity...   
.. ...   
95 kni hahahahahaha antihackerlink greetz hc ch...   
96 defaced dhc defaced dhc nemesystm visit questi...   
97 hit web page defacementdomainswwwtribunacommxw...   
98 another spammer bites dust closed sending unso...   
99 rsh oh visit onlyabout defacing long time sinc...   
  
 lowercase \  
0 the \*former\* kriegsman fur page whoa!! this si...   
1 www server tlacovej agentury sr bol napadnuty ...   
2 www.army.cz hacked www server armady ceskej re...   
3 www.netlab.sk hacked www server internet provi...   
4 www.army.cz hacked www server safarikovej univ...   
.. ...   
95 ps-2 and kni own j00 hahahahahaha....... we're...   
96 ::(defaced by the dhc):: defaced by the dhc (n...   
97 multiple domains! multiple domains were hit wi...   
98 another spammer bites the dust closed by for s...   
99 rsh: oh visit we're not only\nabout defacing. ...   
  
 ner\_ent \  
0 [kriegsman inc., today, today, millions, today...   
1 [www, agentury, sr bol, tejto, 1996, skupine s...   
2 [www.army.cz, www, armady ceskej, bol hacknuty...   
3 [www.netlab.sk, www, netlab s.r.o., kopia doma...   
4 [www.army.cz, www, univerzity, kosiciach, kopi...   
.. ...   
95 [kni, hahahahahaha, antihackerlink greetz, chi...   
96 [the night, one, nemesystm]   
97 []   
98 [ksoze, theregister.co.uk, domineevisser, carl...   
99 [hitler]   
  
 data\_words \  
0 former kriegsman fur page whoa site hit times ...   
1 server tlacovej agentury sr bol napadnuty tejt...   
2 wwwarmycz hacked server armady ceskej republik...   
3 wwwnetlabsk hacked server internet providera n...   
4 wwwarmycz hacked server safarikovej univerzity...   
.. ...   
95 kni hahahahahaha antihackerlink greetz hc ch...   
96 defaced dhc defaced dhc nemesystm visit questi...   
97 hit web page defacementdomainswwwtribunacommxw...   
98 another spammer bites dust closed sending unso...   
99 rsh oh visit onlyabout defacing long time sinc...   
  
 data\_words\_clean   
0 kriegsman fur page whoa site hit times hack si...   
1 server tlacovej agentury sr bol napadnuty tejt...   
2 wwwarmycz hacked server armady ceskej republik...   
3 wwwnetlabsk hacked server internet providera n...   
4 wwwarmycz hacked server safarikovej univerzity...   
.. ...   
95 kni hahahahahaha antihackerlink greetz hc chik...   
96 defaced dhc defaced dhc nemesystm visit questi...   
97 hit web page defacementdomainswwwtribunacommxw...   
98 spammer bites dust closed sending unsollicited...   
99 rsh oh visit onlyabout defacing long time peop...   
  
[100 rows x 12 columns]

import gensim.corpora as corpora# Create Dictionary  
id2word = corpora.Dictionary(data\_words)# Create Corpus  
texts = data\_words# Term Document Frequency   
corpus = [id2word.doc2bow(text) for text in texts]# View  
print(corpus[:1][0][:30])

[(0, 1), (1, 1), (2, 1), (3, 2), (4, 1), (5, 5), (6, 2), (7, 1), (8, 1), (9, 1), (10, 1), (11, 2), (12, 1), (13, 1), (14, 1), (15, 1), (16, 1), (17, 1), (18, 1), (19, 1), (20, 1), (21, 2), (22, 1), (23, 1), (24, 1), (25, 1), (26, 1), (27, 1), (28, 1), (29, 1)]

#more extensive lda model  
lda\_model = gensim.models.ldamodel.LdaModel(corpus=corpus,  
 id2word=id2word,  
 num\_topics=10,  
 random\_state=100,  
 update\_every=1,  
 chunksize=10,  
 passes=10,  
 alpha='symmetric',  
 iterations=100,  
 per\_word\_topics=True,  
 )  
  
print(lda\_model.print\_topics())

[(0, '0.063\*"hacked" + 0.050\*"el" + 0.028\*"school" + 0.027\*"los" + 0.027\*"en" + 0.020\*"que" + 0.015\*"su" + 0.015\*"se" + 0.013\*"es" + 0.011\*"si"'), (1, '0.000\*"rhsck" + 0.000\*"iua" + 0.000\*"blow" + 0.000\*"pampers" + 0.000\*"cespi" + 0.000\*"trant" + 0.000\*"skynny" + 0.000\*"cheetah" + 0.000\*"semen" + 0.000\*"subzero"'), (2, '0.013\*"fluffy" + 0.013\*"dont" + 0.012\*"bunny" + 0.010\*"admin" + 0.008\*"know" + 0.008\*"like" + 0.008\*"site" + 0.008\*"pm" + 0.008\*"people" + 0.007\*"email"'), (3, '0.000\*"rhsck" + 0.000\*"iua" + 0.000\*"blow" + 0.000\*"pampers" + 0.000\*"cespi" + 0.000\*"trant" + 0.000\*"skynny" + 0.000\*"cheetah" + 0.000\*"semen" + 0.000\*"subzero"'), (4, '0.049\*"hit" + 0.014\*"defacement" + 0.000\*"dart" + 0.000\*"darts" + 0.000\*"cricket" + 0.000\*"verrier" + 0.000\*"scorer" + 0.000\*"accepted" + 0.000\*"auctions" + 0.000\*"wendys"'), (5, '0.000\*"rhsck" + 0.000\*"iua" + 0.000\*"blow" + 0.000\*"pampers" + 0.000\*"cespi" + 0.000\*"trant" + 0.000\*"skynny" + 0.000\*"cheetah" + 0.000\*"semen" + 0.000\*"subzero"'), (6, '0.062\*"page" + 0.052\*"web" + 0.051\*"city" + 0.021\*"need" + 0.019\*"south" + 0.016\*"deleted" + 0.015\*"service" + 0.013\*"themagic" + 0.009\*"pell" + 0.008\*"grades"'), (7, '0.000\*"rhsck" + 0.000\*"iua" + 0.000\*"blow" + 0.000\*"pampers" + 0.000\*"cespi" + 0.000\*"trant" + 0.000\*"skynny" + 0.000\*"cheetah" + 0.000\*"semen" + 0.000\*"subzero"'), (8, '0.029\*"para" + 0.028\*"que" + 0.023\*"suspectz" + 0.022\*"prime" + 0.015\*"esta" + 0.015\*"ownz" + 0.014\*"da" + 0.013\*"server" + 0.012\*"zine" + 0.012\*"insanity"'), (9, '0.000\*"rhsck" + 0.000\*"iua" + 0.000\*"blow" + 0.000\*"pampers" + 0.000\*"cespi" + 0.000\*"trant" + 0.000\*"skynny" + 0.000\*"cheetah" + 0.000\*"semen" + 0.000\*"subzero"')]

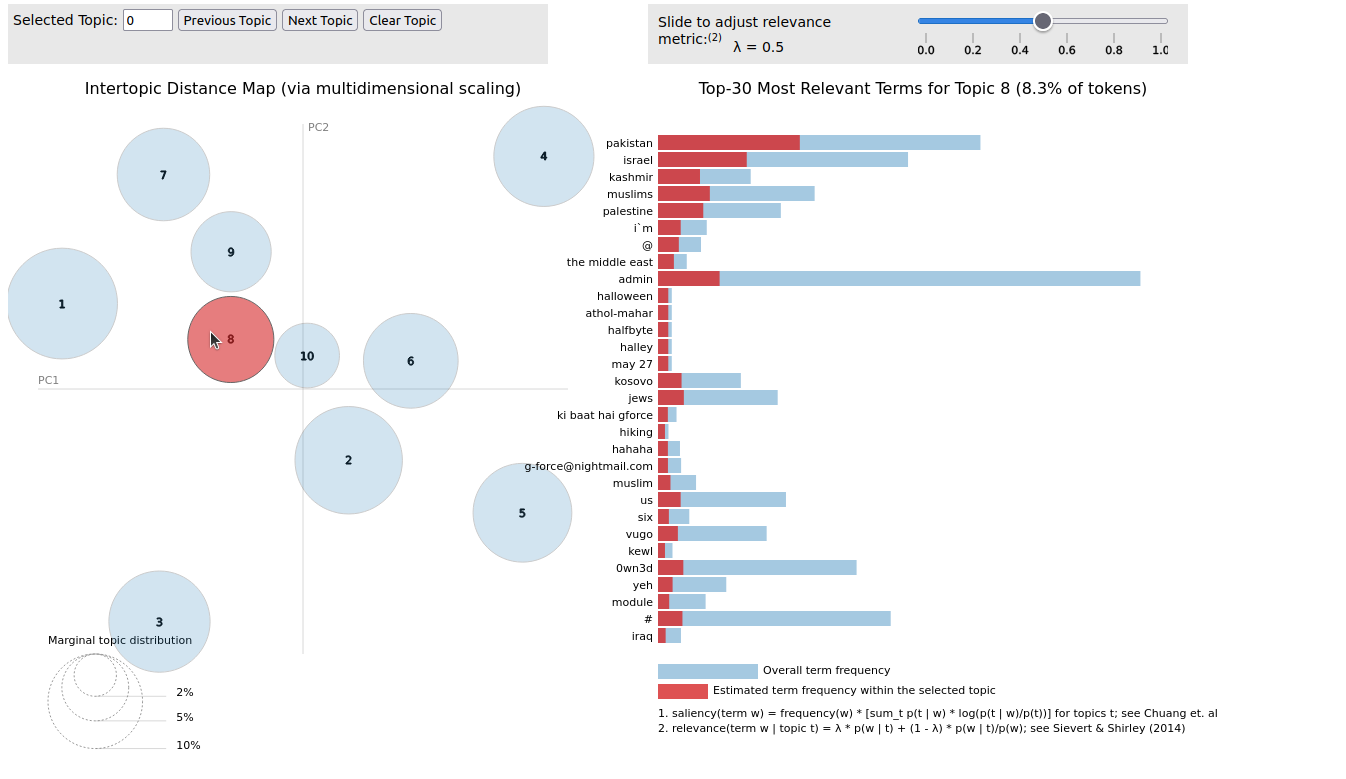
#shows an interactive notebook to explore the model  
pyLDAvis.enable\_notebook()  
vis = pyLDAvis.gensim\_models.prepare(lda\_model, corpus, dictionary=lda\_model.id2word,mds='mmds')  
vis

/home/mk/anaconda3/lib/python3.9/site-packages/pyLDAvis/\_prepare.py:246: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.  
 default\_term\_info = default\_term\_info.sort\_values(  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/past/builtins/misc.py:45: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses  
 from imp import reload  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)

/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)  
/home/mk/anaconda3/lib/python3.9/site-packages/joblib/backports.py:36: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 if LooseVersion(np.\_\_version\_\_) < '1.13':  
/home/mk/anaconda3/lib/python3.9/site-packages/setuptools/\_distutils/version.py:351: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.  
 other = LooseVersion(other)

PreparedData(topic\_coordinates= x y topics cluster Freq  
topic   
2 -0.270168 0.310235 1 1 61.574393  
8 0.214427 -0.248603 2 1 17.769980  
0 0.185758 0.182297 3 1 11.199576  
6 -0.157448 -0.134337 4 1 7.657002  
4 -0.009334 -0.017769 5 1 0.663751  
9 0.007354 -0.018362 6 1 0.227060  
3 0.007354 -0.018362 7 1 0.227060  
1 0.007351 -0.018371 8 1 0.227060  
7 0.007353 -0.018363 9 1 0.227060  
5 0.007353 -0.018366 10 1 0.227060, topic\_info= Term Freq Total Category logprob loglift  
1969 hacked 395.000000 395.000000 Default 30.000 30.0000  
88 page 265.000000 265.000000 Default 29.000 29.0000  
5120 el 311.000000 311.000000 Default 28.000 28.0000  
2285 web 223.000000 223.000000 Default 27.000 27.0000  
4060 city 220.000000 220.000000 Default 26.000 26.0000  
... ... ... ... ... ... ...  
15984 logon 0.007676 0.789514 Topic10 -9.714 1.4544  
15962 aaarg 0.007676 0.892380 Topic10 -9.714 1.3320  
15780 mirdonamys 0.007676 0.939889 Topic10 -9.714 1.2801  
15781 museum 0.007676 0.939889 Topic10 -9.714 1.2801  
15145 linong 0.007676 0.740782 Topic10 -9.714 1.5181  
  
[478 rows x 6 columns], token\_table= Topic Freq Term  
term   
3507 1 0.999961 admin  
5088 3 0.997869 al  
4452 3 0.997541 ass  
8 4 0.930204 aware  
12463 3 0.989860 beeeee  
... ... ... ...  
2305 4 0.991147 war  
2285 4 0.996344 web  
1870 1 1.000230 world  
4454 3 0.985721 ya  
3213 2 0.995141 zine  
  
[150 rows x 3 columns], R=30, lambda\_step=0.01, plot\_opts={'xlab': 'PC1', 'ylab': 'PC2'}, topic\_order=[3, 9, 1, 7, 5, 10, 4, 2, 8, 6])

from IPython.display import Image   
metadata={  
 "jdh": {  
 "module": "object",  
 "object": {  
 "type":"image",  
 "source": [  
 "Figure 11: lda model with 10 topics based on the whole corpus"  
 ]  
 }  
 }  
}  
display(Image("media/lda10.png", width=1000), metadata=metadata)



[Link to full model notebook](https://github.com/mkrzmr/jdh/blob/main/script/ldavis_prepared_10.html) The lda model with 10 topics created using standard parameters shows further insight into the topic clusters found in the Attrition.org defacement archive. The results confirm the presence of defacements mentioning the Kashmir conflict (see the selected topic 8 in Figure 11), and also indicate further topics such as "israel", "muslims" and "palestine". While the model was created with mostly default parameters, it becomes apparent that the multi-layered communication found in defacements is also reflected in the most relevant terms; while "israel", "kosovo" and "iraq" indicate political context, terms like "admin" and "own3d" indicate messages relating to hacking itself. Web defacements, including political web defacements, often feature separate elements to introduce the hacker, send a message and explain relations to other hacker groups. A model such as the one described in Figure 11 can be a finding help for scholars looking to gain an overview of the topic distribution in a larger collection of web defacements. The model can also help justify the selection of single or collections of pages in derived datasets for further research. Finally, the model provides a significant extension of the search and analysis functionality provided by SolrWayback and thus complements the overall workflow.

## Conclusion

Web defacements are an important resource for the study of the history of the web, online communication of political expression, and hacktivist practices. Due to their illicit and unwanted nature, web defacements attract groups and individuals who feel their political opinions are not adequately covered in mainstream media. Additionally, the unexpected appearance of a web defacement communicates its message on a dual layer: By breaking the expected flow of information, the hack exposes vulnerabilities in the compromised system and demonstrates the attacker’s capabilities. Through the embedded content, defacements can deliver messages to previously unreachable audiences. It is also because of this mentioned nature that web defacements usually are short-lived and generally unlikely to become part of large, generalized web archives. While a number of community archives exists, barriers to academic use of web defacements as a resource are the lack of a stable archive and the lack of a conceptual approach. Contextualizing and making accessible archives of web defacements can contribute to a more diverse and complete historical record of the web and fill in gaps. Writing about the digitization process, Milligan attests:

A scholar taking a peripheral glance at a relatively unfamiliar period or country may not realize the sheer expanse of what has not been digitized in that domain. Their lack of deep knowledge of a field prevents them from being self-reflective about archival silences. This means that when- ever we carry out a database search for primary sources we must always ask: what is and is not here? ()

Similarly, the defacements described in this paper can help answer this question about what is *not* included in web archives. They can complement existing web archives and provide into new communities.

This article has described the workflow of enriching and converting the given material from Attrition.org into a collection of WARC files indexed on a SolrWayback installation. This workflow includes the creation of appropriate metadata for each defacement, including date and target URL. To increase discoverability of relevant material, a link network was also prepared and visualized. This workflow in its first stage produced a stable web archive which allowed for the creation of derived data sets for further research.

In a second step, the available content was described through cross-referencing political events (in this case, the Kashmir conflict) and known defacers or defacer groups. This first discovery and description of material was aided by SolWayback’s data visualization tools. The created network visualization also is a useful finding aid for important sites (high degree of outgoing links) and indicators of various hacker communities (high degree of incoming links from defaced pages). In a similar manner, defacers can be traced through their contact information left behind on defaced pages.

Web defacements stand, as the article has shown, in a special relationship with the original page. While one can not be where the other is, the relationship between original and defaced page can range from completely unrelated to very targeted parody. Whatever the relationship is for a given defacement, integration of other sources can help to contextualize and understand the defacement. In the case of the labour.org.uk website, a cross-reference with material held in the Internet Archive confirmed the defacement was in fact a targeted parody of the original page, mimicking not only the structure but also the style of writing found on the original 1996 Labour website. Regarding possible future research, the availability of an API for SolrWayback opens up the possibility of automatically cross-referencing a range of material. The section on cross-referencing material serves as a bridge between the technical side of the approach to stabilizing and making accessible web defacements for research and the conceptual framework necessary for analysis of and engagement with web defacements.

That such an approach can serve as the basis for further research is shown in the final section, outlining the basis of an NLP-based automated processing of the corpus. As during the initial processing, text was extracted from the HTML elements, Named Entities can be extracted as the basis for further discovery and description of the corpus. Some of the special elements commonly found in web defacements, such as the claim “no harm done” could be used as custom NER elements for the identification of defacements.

In addition to the NER stage, a lda model was created to further describe and analyze the corpus. This model helped confirm clusters of defaced pages in the corpus and suggested new relevant terms. Even in a largely out-of-the-box configuration, the lda model was able to produce relevant results which can be used for further analysis.

Through the workflow and conceptual approach presented in this article, web defacements can be stabilized, enriched with metadata and indexed in a web archive. This enables easy access to these materials and opens them up to further research, while also providing a stable, referenceable source. By cross-referencing material from other web archives, a relational framework is created which helps understand type and context of web defacements. Material embedded in such a way can, as the examples have shown, be a valuable resource for the study of the history of the web and early moments of political protest on the web.

## Bibliography