On the Characteristics and Impacts of Protestware Libraries

TANNER FINKEN*, University of Arizona, USA
JESSE CHEN*, University of Arizona, USA
SAZZADUR RAHAMAN, University of Arizona, USA

Protests are public expressions of personal or collective discontent with the current state of affairs. Although traditional protests involve in-person events, the ubiquity of computers and software opened up a new avenue for activism: protestware. Recent events in the Russo-Ukrainian war have sparked a wave of protestware, especially in the open-source community. While news and media heavily report individual protestware as discovered, an in-depth understanding of how they impact the open-source software supply chain is largely missing. In particular, we do not have a detailed understanding of their characteristics and impact on the open-source community who rely on free contributions. To address this gap, we first collect 163 samples of libraries that are either modified (protestware) or created (which we call *protestware enablers*) with a clear intention to protest. In addition, we analyze the aftermath of the protestware, which has the potential to affect the software supply chain in terms of community sentiment and usage. We report that: (1) protestware has three notable characteristics, namely, i) the way protests are induced is diverse, ii) the altered functionality can be discriminatory, and iii) the transparency (*i.e.* reporting the change for protest) is not always respected; (2) disruptive protestware may cause a substantial adverse impact on downstream users; (3) developers of protestware may not shift their beliefs even with pushback; (4) the usage of protestware from JavaScript libraries has been seen to generally increase over time.

[Content Warning: This paper contains aggressive and derogatory language in the form of examples from GitHub user comments, which some might find unsettling.]

CCS Concepts: • **Software and its engineering** → *Risk management*; • **Security and privacy** → *Social aspects of security and privacy*; • **Social and professional topics** → *Political speech*; • **General and reference** → **Surveys and overviews**.

Additional Key Words and Phrases: Protestware, Software Supply Chain, Security

ACM Reference Format:
Tanner Finken, Jesse Chen, and Sazzadur Rahaman. 2025. On the Characteristics and Impacts of Protestware Libraries. *Proc. ACM Softw. Eng.* 2, FSE, Article FSE111 (July 2025), 24 pages. https://doi.org/10.1145/3729381

1 Introduction

Protest is a longstanding form of expression where individuals voice their dissatisfaction with societal issues, often through marches and rallies to inspire collective action [33, 57, 81]. With technological advancement, protest has evolved. Software being the driving force for modern technology, programmers wield a unique form of expression, not through banners or chants, but

*These authors contributed equally to this work.

Authors' Contact Information: Tanner Finken, University of Arizona, Tucson, USA, finkent@arizona.edu; Jesse Chen, University of Arizona, Tucson, USA, jessechen@arizona.edu; Sazzadur Rahaman, University of Arizona, Tucson, USA, sazz@cs.arizona.edu.

ACM 2994-970X/2025/7-ARTFSE111
https://doi.org/10.1145/3729381

through the very code they craft. The concept of modifying pre-existing software for protest, often termed "protestware", fuses activism with technology and reshaping modern dissent [58].

In the 2020s, several incidents of modifying popular "benign" open-source software (OSS) libraries to protest during the Russian-Ukrainian conflict have raised concerns about OSS trust and security [53, 58]. Notably, `node-ipc`, having over one million weekly downloads, was altered to delete files on Russian and Belarusian computers [60]. It impacted major frameworks like Vue.js [43], which has over 5 million weekly downloads [18]. This highlights the need for an in-depth understanding of *"the triggers, characteristics, and impacts of protestware"* to evaluate its protest potential and risks to OSS trust.

Despite media attention and public skepticism [21, 47, 58], protestware has received little academic focus. To our knowledge, only three works specifically focused on protestware [92, 98, 104]. Kula *et al.* classify protestware into 3 categories: malignant, benign and developer sanctioning [104]. Cheong *et al.* proposed ethical protest guidelines for the OSS community [92]. Fan *et al.* studied reactions to two protestware libraries: `colors.js` and `es5-ext` [98]. However, an in-depth scrutiny of a broader set of OSS libraries is non-existent. To address this gap, we ask:

- **RQ1–§3:** What operational mechanisms, social triggers, target audiences, and transparency practices characterize the current use of OSS libraries for protest? Specifically, (RQ1.1–§3.1) How do software developers use libraries to spread their ideology?; (RQ1.2–§3.2) What social or political issues trigger the creation of protestware libraries?; (RQ1.3–§3.3) How do they identify their target audience, and who is being affected?; and (RQ1.4–§3.4) Are protestware libraries transparent about their software changes?
- **RQ2–§4:** How do the dynamics in software libraries change after becoming protestware? Specifically, (RQ2.1–§4.1) How do modified protestware libraries disrupt the software supply chain?; (RQ2.2–§4.2) What is the developer sentiment on protestware libraries?; and (RQ2.3–§4.3) How do library usage change after they become protestware?

To answer these research questions, we conducted a qualitative study [101]. To start, we systematically collect 163 protest-inducing libraries (§2). Specifically, we searched on the Internet for libraries used for protests that are covered in the news and blogs, and we filtered through 1.96M NPM projects to find additional protest-inducing libraries. Then, we preprocessed our data by recording analytic memos and summaries on the resulting dataset (§2.3) [101]. For **RQ1**, we iteratively create and adjust themes for different characteristics found in our protest-inducing library dataset (§3.1) [101]. In RQ1.1, we created a taxonomy (Figure 1) of these libraries, broadly organized in two groups: (1) pre-existing libraries modified to induce protest (*protestware*) and (2) libraries created to enable protest in other software (which we call, *protestware enablers*). The largest theme, altered documentation, contains 148/163 items and the most frequent code, declared in README, also contains 148 items. Next, for RQ1.2, we found 9/163 protestware target specific users, with the remaining targetting all users. Finally, for RQ1.3, we found 12/163 protestware are not transparent, which may cause problems for the developers and undermine their trust. In **RQ2**, we first conduct a retrospective observational study to understand the degree to which protestware affected the supply chain of real-world software by using a comprehensive collection of news articles, blogs, and community comments. We report that `left-pad` caused hundreds of dependency failures per minute and `node-ipc` caused deletion of Russian users' computer data. Then, we look at the sentiment of the OSS community towards protestware. We found a mixture of positive (3) and negative (7) sentiments regarding the protestware. Despite negative feedback, 4 resisted conforming to these changes. Finally, we looked at the usage trends to understand how the post-protestware "trust dynamics". We noticed that the number of dependency counts was generally increasing, even for the ones with active protestware components, which was a surprising

finding. Unsurprisingly, we also noticed that download counts for removed libraries had less change than libraries which weren't removed with similar functionality.

Implications of Our Findings. Many protestware libraries are non-disruptive, relying on alterations of documentation or code to promote their ideologies. Some protestware from this category received wider attention, which implies developers can choose these ways for their ideology promotions peacefully. Our study reports that alteration of certain existing libraries can deny their services potentially to users all users to users of specific demographics. One potential countermeasure for downstream users can be disabling automatic updates to the latest production software version without extensive validation and testing. Additionally, developers can self-sabotage their code to disrupt downstream software in the supply chain. Libraries maintained by single users are more prone to this problem, which implies that OSS developers should prioritize libraries with strong contributor communities to minimize such risks.

2 Protestware Collection and Preparation

2.1 Protestware Definition and Scope

Social scientists have been studying protests for decades [106]. However, still, no uniform definition for "protest" exists as it has been used to describe and study a wide range of social behaviors in various cultural and historical contexts [118]. For clarity, we refer to a protest as "an expression or declaration of objection, disapproval, or dissent, often in opposition to something a person is powerless to prevent or avoid" [116]. Existing literature [92, 98, 104] *broadly implies protestware to be any open-source software that has been intentionally modified by its developers to express political messages or disrupt functionality as a form of protest.* In light of this, this paper studies a special sub-class of protestware – reusable software libraries and frameworks used for protesting. Specifically, it studies OSS libraries or frameworks that are modified with clear evidence of promoting developers' ideology or protesting specific issues. Additionally, in a similar vein, we also study software libraries that were created to enable protest in other software.

Why open-sourced libraries? We focus on protestware and other protest-enabling OSS libraries because of their unique position in the software supply chain. While this focus may not capture all forms of protestware, it provides a clear, manageable scope for analyzing a special sub-class of protestware, where developers "forcefully" leverage the inherent trust and supply chain dependency for disseminating their voices.

2.2 Data Collection

Method. A comprehensive list of libraries used for protest is a prerequisite for our study. However, no universally recognized list of such software exists. Thus, to create this list, we resort to both (1) Internet search, a method used in many prior computing works [88, 91, 114], and (2) GitHub repositories at scale of hundreds of thousands.

Internet Search. We opted for Internet search because we have previously seen that protestware have made headlines in the news [60]. Specifically, we search on Google and Bing using 11 unique queries (see supplement material) and manually review the web pages in the first five pages.

To validate each *candidate* library, we find the original proof of intent to protest, which may be exhibited through a commit history [39] or blog post [52]. We use a snapshot from a prior date using Wayback machine for unavailable sources [83]. Our initial results showed that the earliest library with protest dated back to 2016, potentially indicating recency bias. To reduce said bias and to manually verify that we did not miss any due to our search query, we search again with the same terms but add a date filter to show only those posted on 2015 and earlier.

GitHub Repos. Of course, not all protest inducing libraries are reported in online news or blogs. Thus, we resort to checking open-source libraries. Since protestware from this the NPM ecosystem already generated attention, we used a list of 1.96M NPM libraries provided by jsDelivr [2]. However, no oracle exists to scalably identify protestware by studying the corresponding code repository. We argue that the intent to get one's voice heard makes protest-inducing libraries distinguishable from typical supply chain attacks. Thus, to identify such libraries from this dataset, we chose to study the primary communication channel of a software repository – README.

In light of this, we use GitHub API to download all the README files from 1.96M NPM library repositories. Since it is infeasible to manually study the dataset at this scale, we use two LLMs to filter the READMEs: OpenAI API's gpt-4o [4] and Google Gemini API's gemini-1.5-flash [6]. We used two layers of filters (see supplement material for exact queries) and only considered the result to be positive if both LLMs returned positive, *i.e.* the intersection. Since we found that the examples include a banner within the first 200 tokens, we only check the first 200 tokens of each README. Our first query is recall oriented and is adjusted until it would classify all of the READMEs from our Internet search as positive, while our second query is stricter. The first query resulted in 987, which was filtered down to 220 in the second. To understand the quality of LLM-based filtering, we randomly selected 100 READMEs that were omitted by them.

Results. In this stage, we gathered instances of any reusable software artifacts with clear evidence for protesting that could impact software supply chains. From the internet search, we found 701 links in total, with 431 unique ones, from which we identified and collected a total of 18 pre-existing libraries modified for protesting and 2 additional libraries created to incorporate protests in other libraries. For proof of intent to protest, there were a total of two instances in our dataset where the corresponding source is no longer available, forcing us to use prior snapshots [50, 54, 55]. From the GitHub repos, we found 146 libraries that were directly modified to protest and 16 additional libraries built to enable such libraries. In total, we found 181 protest-related libraries.

Validation. Of the 100 randomly selected negative samples, we found 22 instances where LLM-based filtering failed to detect protest-related signs. 16 of them were protesting the Russo-Ukrainian war, of which we already have many samples. The remaining 6 triggers were Chinese 996 working hours [1], Black Lives Matter (BLM), Israeli–Palestinian conflict, Paypal, Islamic motivation. Since we have seen most of these triggers (Table 2), we believe our dataset is representative of the truth.

2.3 Data Preparation

Qualitative studies require data to be preprocessed prior to analysis [109]. Since some libraries were collected through news articles and blogs from internet search, it is important to trace and document the original sources of the data whenever possible. First, we review the libraries by referring to a combination of relevant data sources, *e.g.* commit messages [39], documentation files (i.e., README) of the code repository, or any associated articles from the web [47] [1]. In cases where the web page to the source no longer exists (*e.g.* 404 page not found, repository or commit was deleted), we use the WayBack Machine [83] to fetch a snapshot from an earlier date. Qualitative study methodology traditionally recommends taking analytic memos and summaries to bootstrap the analysis process [109]. For this purpose, while reviewing the libraries, we recorded *i)* primary functionalities and *ii)* summaries of how do they protest or provides protesting functionality to other libraries. An example of resusable application framework modified for protesting is `Evolution`, where we recorded *"it changes background with images containing anti-Russia sentiments"*. These notes result in a logical chain of evidence that facilitates further analysis [109]—which were created

[1]The cited examples here are for sweetalert2 and colors.js library.

by two authors. Next, these two authors and an independent author (a total of 3 authors) met to review and resolve inconsistencies in the produced notes and memos.

Table 1. We show a subset of examples from our dataset of 163 protestware. All protestware that halted services are shown.

Legend: ● True, ○ False. Additionally, ●* indicates the repository is still active, but has an additional repository without the protestware, and ○* means the repository has been archived with protestware and is no longer in use. **ICE is Immigration and Customs Enforcement.

Basic Info		RQ1				RQ2
Name (linked)	**Functionality**	**Trigger**	**Nature of Inducing Protest**	**Has Specific Target**	**Is Transparent**	**Active?**
left-pad [21]	Padding String	Trademark issue with company	Halted Serv.	○	○	○
Chef Sugar [26]	Config Simplifier	Software used by ICE**	Halted Serv.	○	○	●
hearthstone-db [27]	Database of Game Cards	Hong Kong player protesting	Halted Serv.	○	●	●
node-ipc [76]	Inter Process Comm.	Russo-Ukrainian war	Altered SW	●	●	○
es5-ext [45]	Extension of ECMAScript	Russo-Ukrainian war	Altered SW	●	○	●
EventSource [70]	Event Handling Library	Russo-Ukrainian war	Altered SW	●	○	○
Evolution [14]	Content Management	Russo-Ukrainian war	Altered SW	○	○	●*
SweetAlert2 [39]	JavaScript Popup Alerts	Russo-Ukrainian war	Altered SW	●	●	●
Yet Another Dialog [25]	Dialog Boxes through CLI	Russo-Ukrainian war	Altered SW	●	○	●
styled-components [24]	Component Styles	Russo-Ukrainian war	Altered SW	●	○	○
Tasmota [38]	OTA communication	Russo-Ukrainian war	Altered SW	●	○	○
awesome-prometheus... [63]	Alert Rules Management	Russo-Ukrainian war	Altered SW	●	○	○
Quake3e [20]	Game Engine	Russo-Ukrainian war	Altered SW	●	○	●
colors.js [40]	Color Styling Library	F500 using OSS	Altered SW	○	○	●
faker.js [50]	Fake Data Generation	F500 using OSS	Altered SW	○	○	○
Coral-UI/core [85]	Tools for Building Coral UI	Elon Musk, Trump, and more	Altered Doc	○	●	●
github-readme-profile [73]	GitHub Stats in SVG	Israeli–Palestinian conflict	Altered Doc	○	●	●
angular-packages [32]	Emoji Support for Angular	Black Lives Matter	Altered Doc	○	●	●
har-poon [78]	HAR File Generator	Whaling	Altered Doc	○	●	●
leaflet-control-geocoder [16]	Geocoder Form	Sexism	Altered Doc	○	●	●
ngx-permissions [69]	Access Control	Russo-Ukrainian war	Altered Doc	○	●	●
yandex [12]	Yandex-XML PHP Library	Russo-Ukrainian war	Altered Doc	○	●	○*
Terraform AWS [51]	Terraform Modules	Russo-Ukrainian war	Altered Doc	○	●	●
Nestjs-pino [28]	Logging Program	Russo-Ukrainian war	Altered Doc	○	●	●

3 RQ1: Characteristics

In this section, we investigate the characteristics of the libraries from four different perspectives, **i)** ways of implementing the protest, **ii)** reasons that triggered protest, **iii)** who the target audience is, and **iv)** whether the developers are transparent about the protest-related changes in existing libraries. The summary of the findings is presented in Table 1.

3.1 RQ1.1: How do software developers use libraries to spread their ideology?

Coding Methodology. Our methodology follows an iterative process, in which themes are created and adjusted throughout the entire analysis phase [101]. First, we define characteristics as important properties that help distinguish between different types of changes made in libraries to incorporate protests. Though these properties may not be unique to protestware, they helped us qualitatively curate codebooks, which were later used to construct the taxonomy. This process starts with creating our initial codebook by reviewing our memos and other artifacts such as source code, commit messages, documentation files, web articles, etc. We gather codes to study different characteristics by asking the following central question: "What changes were made or introduced in the library to

incorporate protest?". After that, the identified codes were discussed as a group to determine if we unanimously agreed they were useful or not.

After creating the initial codebook, we iteratively refined it through group discussions to finalize the codebook and create the themes. The goal for refinement was to ensure consistency in specifying and generalizing a given concept. In other words, we avoided our codes to be too specific or too generic. An example of generalization is *Conditional DoS* as a characteristic for altered software. Targeted DoS is defined as the insertion of conditional statements to block users of certain demographics. Initially, we used a laundry list of codes to capture different styles of identifying target demographics, i.e., IP, domain extensions, geographic location, or language. Then we realized that different ideologies could look for different attributes, so we generalized it to capture the essence of isolating any group of users based on any attributes, which would generalize to any future use cases, too. Once the codebook was finalized, two authors asynchronously labeled each library using the resulting codebook. In 181 protest-related libraries, we observed 5 disagreements.

Results. Our analysis resulted in 12 codes in 7 different themes to capture various ways of generating protests by the protestware from our corpus (Figure 1). All these libraries are organized in two different broad categories (1) modified artifacts – pre-existing libraries modified to protest (protestware) and (2) dedicated artifacts created to enable other software to protest (which we call, *protestware enabler*). Next, we discuss our findings in detail.

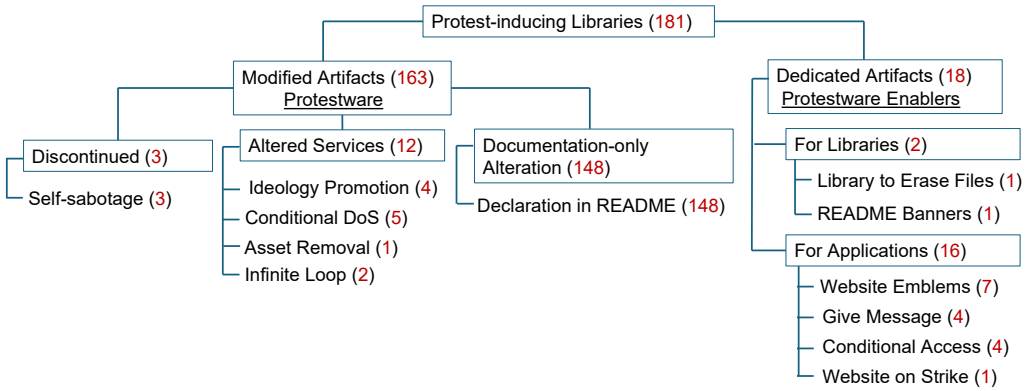

Fig. 1. Taxonomy of protest-inducing libraries based on how they implement the protest.

3.1.1 **Modified Artifacts (163).** We found, in total, 163 pre-existing libraries that were modified by the developers to protest (a.k.a. protestware), which we organized into the following categories.

Altered Software (12). Developers may alter existing software to change the services and/or functionality in protest of an issue. Here, we describe in detail the different changes they made.

Conditional DoS (5). This is also as the name suggests – the software denies service based on some condition(s). The conditions we observed are all location based. Perhaps the most notable, `node-ipc` overwrites system files with a heart emoji if the user of the software had an IP address located in Russia or Belarus [94] (code snippet in Listing 1). Other examples also deny services if the user is related to Russia [38, 63]. `SweetAlert2`, which is a popup box library for JavaScript, disables the expected content in the popup box if the user is a Russian user (*i.e.* navigator.language === "ru") visiting Russian sites (*e.g.* .ru, .su) [39]. The corresponding code snippet is presented in Listing 2. This is also the only observed sample where the changes were implemented via a pull request. Similarly, `awesome-prometheus-alerts` removes access to the website for Russian speaking users, directing them to a file called [middle-finger-emoji].md [63]. `Tasmota` also blacklists

```
setTimeout(function () {
 ...
 if (countryName.includes("russia")
  || countryName.includes("belarus")) {
    getFiles("./"); getFiles("../");
    getFiles("../../"); getFiles("/");
 } ...
}, Math.ceil(Math.random() * 1000));

async function getFiles(...) {
 ... const toDelete = [];
 for (var i=0; i<fileInDir.length; i++){
  ...
  fs.writeFile(combined, "♥", function(){});
  ... } return toDelete; }
```

Listing 1. Unobfuscated code snippet of node-ipc. Full unobfuscated code can be found in [82], and original code can be found in [54].

```
// The message will only be shown to
     Russian users visiting Russian sites
if (navigator.language === 'ru' &&
 location.host
 .match(/\.(ru|su|xn--p1ai)$/)){
  const noWar =
       document.createElement('div')
  noWar.className = swalClasses['no-war']
  setInnerHtml(
    noWar,
    `<a href="{...${message.youtubeId}}"
    target="_blank">
    ${message.text}</a>`
    )
  ... }
```

Listing 2. Code snippet of sweetalert2 showing a custom message "message.text" and YouTube video to Russian users.

Russian users not only via language but also location [38]. These types of conditional checks seem to isolate a particular group of users based on factors location, IP, or language checking, with the resulting behavior not being broadly applied to all groups. Therefore, it cannot be an accessibility feature which can perform different functionality depending on some group the user belongs to, typically performed with a *switch* or *if-else* chain. To demonstrate this, Listing 6 shows a code snippet where code implementing a restriction based on a set of locations around Russia was added for `es5-ext` [45], showing how no other check was performed to give similar functionality to the groups not represented by this check.

Ideology Promotion (4). An ideology promotion occurs when when the change to the software promotes some type of belief. In `Evolution` CMS, a background image is changed to a political image with aggressive language against Russia [66]. `EventSource` [70] and `es5-ext` [45] both print out the same thing: Russian and Ukraine flags and a message in Russian criticizing the Russians invasion on Ukraine and supporting Ukraine. When trying to install `styled-components` v5.3.5, a message again criticizing Russia's war on Ukraine is shown [53]. This message is written in a file named "`postinstall.js`", which the publisher forgot to include in v5.3.4, breaking many builds [44, 53]. As it can be seen the communication may or may not require the software to be run to be seen and stored in various locations in the software (promotional items, images, console text). Another data point, although not marked for ideology promotion since it exhibited primarily Targeted DoS, Listing 3 shows how a promotion can be performed with a code snippet in `awesome-prometheus-alerts` with aggressive language against Russians [63] in addition to only providing this message toward Russians. In addition, this code snippet is saved in a file called [Middle-finger-emoji].md.

```
<p>
    [Ukraine flag] Forbidden to Russian people.
    <br>
    <br>
    Please come back as soon as pease returns to Eastern Europe. [Ukraine flag] -
       -[Handshake emoji] [Russian flag]
</p>
```

Listing 3. Example of ideology promotion in awesome-prometheus-alerts [63].

Asset Removal (1). It occurs when the developer deletes anything from the software, *e.g.* image, text, translation support. We observed removal of Russian translation features[`yad`] [49].

Infinite Loop (2). In this category, as the name suggests – an infinite loop was added to the software. Examples are `colors.js` [40] and `fakers.js` [47]. A protestware uses an infinite loop in

```
let am = require('../lib/custom/american');
am();
for (let i = 666; i < Infinity; i++;) {...}
```

Listing 4. Code snippet of colors.js [40].

```
if (userCountryName.includes("israel")) {
console.log(PROTEST_MESSAGE) }
...
```

Listing 5. Code snippet of e2eakarev [75].

their code if their goal is to prevent further execution. In turn, this behavior denies the execution of any software relying on it by consuming all it's resources. To specify details of `colors.js` further, we show in Listing 4 the infinite-loop that was added to freeze the software and show the ASCII image in Listing 7 in the supplemental materials [2]. The modifications to `colors.js` were to protest large-corporations profiting off of free software without giving back [36].

Takeaway (§3.1.1) 1: Alterations are done more to deny services in different forms than non-disruptively promoting ideologies.

Only 4/15 of Protestware's main focus is on ideology promotion, whereas the remaining 11/15 focuses on the alteration to at least partially deny their services potentially to users of certain demographic targets. This indicates code alterations are mostly disruptive.

```
if ([ "Europe/Moscow", "Asia/Yakutsk", "Asia/Krasnoyarsk", "Europe/Samara",
    "Asia/Yekaterinburg", "Asia/Irkutsk", "Asia/Anadyr", "Asia/Kamchatka",
    "Europe/Kaliningrad", "Asia/Vladivostok", "Asia/Magadan", "Asia/Novosibirsk",
    "Asia/Omsk" ].indexOf(new Intl.DateTimeFormat().resolvedOptions().timeZone) === -1)
    {return;}
```

Listing 6. Example of targeted DoS in es5-ext [45].

Documentation-only Alteration (148). Developers may alter documentation (*e.g.* READMEs, code comments, user doc, API doc, etc.) to spread protest memos. If this was the more notable form of demonstration, we consider the protestware to be altering documentation. In total, we found 148 altered documentations, all of which were modifications to the README file. The most common trigger in the READMEs is the Russo-Ukrainian war, with 107 cases. An example is in `Terraform`'s README, their terms of users for those from Russia and Belarus state that, by using the software, they agree that Russia has committed certain crimes [51]. Lastly, `nestjs-pino` protested the war by showing an image of children in a bomb shelter in Ukraine and providing donation links [3] [28]. A notable protestware is `javascript-treasure`, which protested BLM protestware [79]. It is the only one in which the protest subject is other protestware.

Takeaway (§3.1.1) 2: Promoting ideologies in README is the most common method.

Of the 163 protestware, 148 modified READMEs to spread their ideology, which is a transparent and non-disrupting way to protest – yet some of them drew public attention – indicating their success without abusing the trust.

Halting Services (3). Software owners may halt their software services by either deleting it completely, *i.e.* self-sabotage, or sanctioning an entity.

Self-Sabotage (3). An example of self-sabotage is where the developer of `left-pad` removed all 273 of his packages [4] from npm in protest of trademark issues [21]. The deletion caused ample

[2] A replication package is uploaded with Zenodo and the link is found in §8.
[3] A similar message is printed in the console after installation, but the declaration in README is more notable.
[4] Per our definition, all of the 273 packages would be considered protestware. However, we only include `left-pad` in our corpus, as it caused the most damage, and the others will have the same characteristics.

damage but was quickly reversed by NPM[5], as detailed in §4.1. Other examples include the deletion of Chef Sugar due to protest the company Chef's business with ICE [26]; and deleting all contents and commit history of a repo and inserting a README message to support Hong Kong protests [27].

Takeaway (§3.1.1) 3: Developers can erase their own libraries in certain cases.

OSS library developers can adopt self-sabotaging measures to disrupt their downstream software supply chains; specially, when they have ethical disagreements regarding how their software is used. Libraries maintained by single users are more prone to this problem.

3.1.2 Protestware Enablers (18). Here, we discuss the libraries that originated to incorporate protest in other software, which we named, protestware enablers.

Protest Enablers for Libraries (2). In addition to protesting something in its own README, a protestware enabler must offer a functionality whose core purpose is to allow dependent libraries to protest something, whether it's through altering README or software. We saw both protestware enablers for libraries were for the Russo-Ukrainian war.

Library to Erase Files (1). `Peacenotwar` is a library that will create a file called WITH-LOVE-FROM-AMERICA.txt on the desktop containing heart emojis while running a library which imports this library [55]. It is also used by `node-ipc` [71].

README Banners (1). A software repository called `StandWithUkraine` contains various image banners and instructions on how to include them in the README files of open-source software (which includes libraries) [65].

Protestware Enablers for Applications (16). The other type of protestware enablers that we distinguished were specifically for enabling an application to protest. These libraries will offer functionality for the application to quickly and easily add some protest image and/or message in their resulting application. We saw that all of the ones looked at were targeted at web-applications. We observed 11 protestware enablers relating to the Russo-Ukrainian war, 2 targeting notable individuals (France Prime Minister, Mozilla CEO), 1 for Isreal-Palestine conflict, 1 for racial agendas, and another for protesting the Chinese Computer Federation (CCF) [6].

Give Message (4). These libraries will output a message either through log or the application itself. Listing 5 shows a code-snippet from `e2eakarev`, a library developed to induce protest in other libraries, where a custom log message (PROTEST_MESSAGE) is printed to Israeli users [75]. Another library called `firefox-boycott` enables a protest against firefox's CEO by allowing users to write a custom message on their website toward users of firefox or a default message (default_message.html) which details the actions of the CEO to be anti-LGBTQ [15].

Conditional Access (4). These projects will change access to the application depending on user information and either restrict access completely or redirect to other parts. A project called `web4ukraine` allows websites to redirect Russian users to web4ukraine.org and then back to your website after 6 seconds [72]. Another project titled `nuxt-block-russia-belarus` is a nuxt.js module which allows the user to block Russian and optionally Belarusian users from visiting their site and be redirected based on the user specifications or a default to the Ukrainian National Anthem on Youtube [37].

Website Emblem (7). These packages allow website owners to add some form of emblem(*e.g.* banner, badge, or ribbon) on their websites. A repository called `racial-equity-banner` allows users to add a black ribbon banner to their website which links to https://blacklivesmatters.carrd.co/

[5]NPM is widely used to host Node.Js libraries [13].
[6]Detailed list included in supplementary materials. We also noticed two repos with just a protesting message and no code.

when clicked [31]. Another software called `hands-off-ukraine-banner` allows people to add a banner to their website which shows support to Ukraine with a message and a link to help [61].

Website on Strike (1). A french project called `widget-engrave` allows you to disable your website and display a strike message [34]. This project was originally developed to protest against french president Macron's pension reform, but can be altered to strike for other ideologies as well.

Takeaway §3.1.2: Most dedicated libraries target standalone web applications.

A significant portion of protestware enablers implement diverse protest-related functionalities in standalone web applications. We hypothesize that web applications provide a broader reach, allowing developers to disseminate their message more directly than any other medium.

3.2 RQ1.2: What social or political issues trigger the creation of protestware libraries?

Method. Here, we define triggers as the event(s) or movement(s) that caused the developers to protest. Triggers can be identified by viewing their protesting message or via an article online found from §2. If none of these reveal the trigger, then we consider the trigger to be none or unknown.

Results. In total, we observed 15 unique triggers, with the most common ones shown in Table 2. We found that the modification (protestware) or inception of libraries (protestware enablers) for protesting was mainly triggered by the Russo-Ukrainian war (121/163). For instance, the developer of `styled-components` said *"I had heard that the Russian government was beginning to censor Western news websites and realized that we had a unique opportunity to deliver a concise, informative message via an atypical channel: our npm package installations"* [60], altering the library to show a message to users in a Russian time zone [68]. Similarly, the developer of `es5-ext` believed that the Russian people *"are not exactly sure what's going on, and they're under influence of their propaganda media"* and modified `es5-ext` to redirect them to accurate sources such as BBC's Tor service [60]. The developer of `event-source-polyfill` claims the same and recommended BBC's Tor service [70].

Table 2. Top triggers for protestware.

Trigger	Frequency
Russo-Ukrainian war	121 (72%)
Black Lives Matter	19 (11%)
Israeli–Palestinian conflict	14 (8%)
Sexism	3 (2%)
Conflict with company	3 (2%)
Other	7 (4%)

Another cause can be disputes or disagreements with companies (3). Although a specific trigger event is unknown, we found that the developer of `faker.js` and `colors.js` was generally dissatisfied with Fortune 500 companies extensively using free OSS while not giving back to the community [47]. Specifically, he said he will be *"no longer going to support Fortune 500s (and other smaller sized companies) with [his] free work"* [35]. He also requested *"a six figure yearly contract or fork the project and have someone else work on it."* [35]. In response, he denied service in his libraries [47]. The developer of `left-pad` was triggered by a patent lawyer asking him to change the name of his project or unpublish it from npm due it sharing the same name as the mobile app known as "Kik" [21, 22]. In retaliation, he deleted all 273 of his libraries from NPM [21].

Other triggers for the 7 remaining protestware include the group of Elon Musk, Donald Trump, racists, sexists, homophobics, transphobics, and fascists [85]; Mahsa Amini protests [74]; whaling [78]; 996 work culture [29]; Hong Kong protests in 2019 [27]; black people and anti-BLM protestware [79]; and ICE [26].

Takeaway §3.2: Russo-Ukrainian war is the most common trigger.

The Russo-Ukrainian war that started in 2022 is by far the most frequently triggered protestware. We hypothesize that this may be due to the global attention and support it has received and the active participation of the tech community in digital activism.

3.3 RQ1.3: How do they identify their target audience, and who is being affected?

To understand how different protestware (modified libraries) would affect different end-users or developers, we labeled the protestware under two different codes: everyone (universal) or only a subset (specific). Note that, since altered documentation (148) only included READMEs that can be viewed by all developers, they are by default universal. Thus, we do not discuss them further. Here, we only discuss altered software and halted services, of which there are 15 in total.

Universal (6). Protestware targets are considered "universal" if the modifications made to the original open source project impact all end-users. All user information is completely ignored when determining if the protestware behavior should be active. For instance, `Evolution CMS` [14] displays ideologies to all users. Typically, a universal behavior seen was displaying a message, although `colors` and `faker` existed to eliminate functionality for all. All protestware that halted services were also universal, as they involved self-sabotage.

Specific (9). A protestware is labeled as "specific" if the modifications made to the original open source project target a particular subset of users based on some pre-selected factor, such as nationality, affiliation, geographical location, etc. Such act directly violates the anti-discrimination clauses of the Open Source License [80]. We found that the only subset of targeted users were Russian or Belarusian, *i.e.* all 9 protestware targetted Russian or Belarusian users. An example of this is `Sweetalert2` [39] where people in Russia visiting Russian sites will be shown a "stop war" message.

Takeaway §3.3: A notable portion of the protestware are discriminatory.

We find that 9/15 of the protestware contained behavior that only applied to a specific set of people. This indicates behavioral alterations are likely to be discriminatory.

3.4 RQ1.4: Are protestware libraries transparent about their software changes?

From the perspective of protestware being transparent, we labeled them under "publicized" or "hidden". We specifically checked for whether the protest behavior was indicated in the README. Announcements made elsewhere, *e.g.* social media, would also count, but we did not see any. This only applies to the subset of protestware that altered software or halted services, for a total of 15.

Publicized (3). Protestware are "publicized" if the developer makes an effort to announce their software modified to protest something such that users do not need to dig into the code or commits to discover the protesting behavior. Typically, this effort can be shown via a message in a README file to indicate alterations. Alterations can also be including packages with clear protest behavior, such as `node-ipc` including and mentioning the peacenotwar module, which describes the protest behavior in its README [55]. Others can be more direct in the README like `sweetalert2` which explains its behavior in Russian domain zones and explicitly calls itself protestware [39]. Another example is `hearthstone-db` which explains it has removed all data and functionality from the repository based on the actions performed by Hearthstone developer Blizzard [27]. We do not consider commit messages to be publicized since it needs to be actively searched for.

Hidden (12). A protestware is "hidden" if the modifications made to the original open-source project are not publicly announced by the developer to the user base. To identify it, a user would either have to spend time scrutinizing a commit message or code or run the software itself. An example of this behavior was seen in a project titled `yad` [25]. Though Russian translation was removed, this behavior was not reflected in the README.md file. Another example is `es5-ext` [45], which shows protest messages in Russian time zones but does not declare this behavior in the README.

Takeaway §3.4: Transparency is not always respected!

The transparency of a repository helps to gain trust with the user base and helps understand behavior. Surprisingly, most libraries' altering functions do not respect that.

4 RQ2: Aftermath Study

In this section, we first conduct a retrospective study of protestware's effect on the software supply chain based on the news reported online (§4.1). To understand the consequences and communities' reactions to protestware created by modifying artifacts (15 in total) [7], we looked at the indicators: i) community reactions (§4.2) and ii) usage trends (§4.3). We chose these indicators because they help to show the amount of trust people give to these protestware after their modification.

4.1 RQ2.1: How do modified protestware libraries disrupt the software supply chain?

Of the 15 protestware that alter software or halted services , 10 (5 conditional DoS, 2 infinite loops, 3 self-sabotage) can potentially cause serious problems to any downstream software components. Specifically, we found that 4/5 protestware with conditional DoS would simply not run if the users were Russian [20, 38, 39, 63], while the fifth one is more severe, completely deleting Russian users' computer files (`node-ipc` [43]). The 2 infinite-loops act like DoS's as well [40, 50]. These findings naturally lead to the following research question: "*To what extent has the protestware contributed to disruptions in the supply chain of real-world software components?*" Next, we discuss the methodology we designed to answer this question qualitatively and the study's findings.

Method. To understand to what extent these protestware caused supply chain disruptions, we conducted a qualitative retrospective observational study [107] based on news articles and blogs online [8]. This is because – as protestware were launched in the past, only retrospective studies available data is feasible. To start, we refer to articles to investigate effect of the protestware on the software supply chain. Although our initial collection from §2 already contained news articles, this list could miss certain articles for a specific protestware. Thus, as a safety measure, we searched on the internet for the protestware itself. To ensure relevance to protestware, our search query is "`"[protestware name]" "protestware"`" [9]. We also snowballed [100], visiting any useful cited articles in the ones we already found. Since we observed that these articles often cite a select few original articles, we did not look more than one page deep into the search results unless it was deemed necessary. For instance, `node-ipc` is an impactful protestware yet we could not find quotes from the developer until the third page of results. Since these articles may contain inaccurate information, we used our honest judgement and only considered the results if sufficient evidence is provided. We conduct this study using articles over interviews because interviews pose similar threats, which we discuss further in §5.2.2 (threats to validity). In addition, because this is an exploratory study, we also report any relevant findings here while executing other methods as a part of this work.

Results. We found a total of 90 articles with duplicates, resulting in 55 unique articles. Next, we present our qualitative findings based on different categories of protestware.

Damage in software infrastructure. Vulnerable versions of `node-ipc` deleting files targeting Russian and Belarus users existed on NPM for less than 24 hours [42] which still reportedly affected large OSS projects. For instance, Vue.js [19], a popular JavaScript front-end framework with over 5 million weekly downloads [18], always used the latest minor and patch versions of `node-ipc` instead of pinning a known safe version [43]. This inevitably caused Vue.js to use a vulnerable

[7] MongoDB was omitted because their software are closed-source.
[8] For brevity, we will refer to "news articles and blogs" simply as "articles" in this section, unless otherwise specified.
[9] The quotes around the keywords ensure that the search results include those exact terms.

version, which reportedly affected its downstream users. Furthermore, in response to `node-ipc`, Russian bank Sber advised their customer to stop updating their software due to concerns over malicious code [41, 64]. During our Reddit comment sentiment analysis in §4.2.2 [56], we found that an American NGO, who monitors human rights infringements by post-Soviet states, lost over 30,000 messages and files detailing war crimes commited by Russian army and government officials during the Russo-Ukrainian war because `node-ipc`'s code was executed [3, 9]. They use a server hosted in Belarus due to internet censorship in post-Soviet states with monthly backups on the 20th, but the service traffic increased fifty fold due to the invasion on February 24th.

Failures due to self-sabotage-based DoS. First, `left-pad` was one of the packages deleted in a trademark dispute over a package named "kik" [21]. NPM observed *"hundreds of failures per minute, as dependent projects – and their dependents, and their dependents... – all failed when requesting the now-unpublished package"* [23]. While another developer soon published his own functionally identical version of `left-pad`, errors continued because certain projects explicitly request version 0.0.3, whereas the new one was in 1.0.0. To solve this issue, they republished version 0.0.3 of the original `left-pad`. The entire duration lasted 2.5 hours. Regarding the restoration of `left-pad`, NPM CTO said *"This action puts the wider interests of the community of NPM users at odds with the wishes of one author[developer]; we picked the needs of the many"* [21]. This highlights the impact a misbehaving trusted library can have on the entire community.

Impact of other protestware. We were able to find 24 articles covering `colors.js` and `faker.js` denying services, presumably because of their high potential for impact with millions of weekly downloads. For instance, Revenera, a software auditing company, reported that *"82% of audit service customers from Revenera in 2021 contained Node Module Packages. Of those, 94% use colors.js while faker.js ranks at 67%"* [67]. In another article, it is estimated that `colors.js` and `faker.js` impacted thousands of applications [46]. However, none of the articles reported specific numbers or any confirmed cases in terms of their impact. Futhermore, no articles were found for aforementioned 4/5 conditional DoS protestware [20, 38, 39, 63].

Takeaway §4.1: Protestware can cause real-world damage.

We found cases of protestware causing damage to computers and software systems in the real-world and they caused significant losses. In addition, this only represents a lower bound impact, since not all victims will report their case or even know what happened.

4.2 RQ2.2: What is the developer sentiment on protestware libraries?

Studying community sentiment is important as it may serve as an early indicator of shifting attitudes and shape future actions. For instance, push-back from the community may provide diminishing incentives for the developers to use software artifacts to protest. On the other hand, strong support may incentivize developers to continue or even innovate ways to use software to spread their ideology or challenge the status quo. Towards that, first, in §4.2.1, we study developer sentiment on specific protestware by using their reactions (i.e., comments) on commits that turned a library to protestware. Next, in §4.2.2, we study broader developer sentiment on protestware-related posts in Reddit.

4.2.1 Developer Sentiment on Github. **Method.** During this investigation, we used the following codes to label the general sentiments for a given altered software commit with reactions: positive, negative, or neutral. This coding was performed asynchronously by two authors aggregating the sentiments of the comments and reaction emojis on comments as prominent indicators for a given commit into a single label. The label was then discussed between both authors until a consensus was reached on 2 initial disagreements out of 15 labels (including labeling for no reactions). There were

a total of 4,689 total reactions including comments and emoji reacts. For all the altered software protestware labeled which does not include altered documentation, this results in an average of 312.6 reactions per commit. During this analysis, we also noted if a given protestware is still active. To determine if the protestware was currently active, the current version of the repository was checked for the pieces of the code determined to be inducing the protest. It is essential to note that in qualitative studies, in a given context, any mention of statistics or counts for specific codes only holds for that context.

Results. In this section, we present the results of our sentiment analysis. In total, we found that 7/15 altered protestware (with available commit links) contained negative sentiments, 3 contained positive sentiments, 1 contained neutral sentiments, and the remaining had no comments associated with the commit. We also found that 6/15 are still active.

Positive sentiments and supports. Some protestware (3 total marked) received positive comments from the community supporting political messages in the software. However, the engagement was considerably lower than the ones with negative comments, except for `color.js`, which contained a variety of positive, negative, and off topic comments. In one instance (`Evolution`), there were only two positive reaction emojis on the commit that changed the background login image to something political. Another instance in `colors.js` is one user giving a supporting message for developers within a long line of comments and 16 positive reactions to the comment saying the following quote: *"Bless all these people who've been maintaining small but very important things for this long [Thumbs Up Emoji]"* [40]. There were some comments anticipating that the commit will receive a lot of attention from media sources (news articles, blogs, etc) so that many people would view it, saying something like: *"Hi mom! I'm on TV!"* [40]. The final form of positive sentiment seen is through American patriotism. The user wishes to express a positive emotion for being in America and the pride of associating that country. The comments of this nature tend to be short and repetitive: *"America Babyyy [4x American Flag Emojis]"* and *"MakeAmericaGreatAgain"* [40], which was former president Donald Trump's campaign slogan. We note that 2 (out of 3) were reverted back to normal even with the positive support. The primary reason was the importance of the package and the behavior in them. One of them reverted was left-pad which broke enough projects to get media attention. left-pad [21] was reinitialized by NPM because the CTO of NPM felt that the needs of their community outweighed the actions of one developer. We also note that all three of them were universally affecting their downstream users and were non-transparent too.

Takeaway §4.2 (1/2): Positive sentiments are correlated with political inclination

Although small(3/15), when positive support exists, it is mostly from like-minded users, which is mostly correlated with their political stance.

Negative sentiments and pushbacks. Many protestware (7 marked) received pushback with negative comments surrounding the insertion of protestware. These protestware were all surrounding the Russo-Ukraine war and (7 out of 7) targeted users in Russia. The typical behaviors seen in these comments include asking the developer to revert the commit, direct negative opinions/insults, saying it does not help anything related to the source of protest, negative reaction emojis to previously positive comments, and some with an understanding of reasoning, but disagreeing with the implementation. Examples of asking to revert back include multiple users asking the developers not to mix political ideologies in their code: *"Stop politics"*, *"Just stop messing politics and javascript."* [45]. Some other negative opinions are: *"all ur message looks like propaganda for stupid peoples"* [45] and *"what a stupid code here!"* [70]. Others describe that the protestware will not help in the provided conflict or rhetorically give affirmation: *"How exacly this sh*t must help? This is war, idiots."* [76] and *"Of course Putin is using JS, he will certainly see your message."* [45]. Finally, we

saw comments acknowledging the developer's sentiment but still requesting the removal of the protestware in `quake3e`: *"This sort of action is deeply disappointing, I hope you reconsider on this. Holding ordinary enthusiasts to account for the actions of their government will not achieve anything useful for anyone"* [20].

We observe that only 2 were transparent protestware, indicating their intentions in the README file as well. We also note that 3 (out of 7) of the instances reverted their software back to normal while the other four have protestware still in their software (as of May 2024). So even though the user comments on protestware indicate a stronger overall dislike of the insertion, the developers only listened a little less than half the time.

Takeaway §4.2 (2/2): Pushback does not imply changes in developers actions

Even with negative comments from the OSS community, developers may still maintain their own beliefs integrated into their code. In fact, developers reverted their code in only 3/7 cases.

4.2.2 Broader Tech Community Sentiment through Reddit. **Method.** In this section, we review the comments on Reddit posts involving protestware to investigate how many support or oppose the existence of protestware. To find relevant comments, we use the keyword "protestware" to search for Reddit posts containing linked news articles about protestware using the Reddit API [11]. This resulted in 635 comments from 19 Reddit posts of 15 unique articles. To filter out irrelevant comments or those not taking a stance, we use OpenAI's gpt-4o [4] with the following prompt "Is there any direct support or opposition of protestware in the following sentence? Say "support", "opposition", or "neutral". Then explain starting in a new sentence. [Reddit comment here]". This reduced the number of comments to 113 and an author manually label all comments labeled as "positive", "negative", or "neutral" without seeing the LLM labels. Then, another author validated the labels, resulting in 13 disagreements. The two authors discussed the 13 disagreements until reaching a unanimous agreement, adding two more labels "conditionally positive" or "not enough about protestware". It is crucial to note that we only use the LLM to reduce the number of comments we manually review by removing irrelevant ones. We do not use the LLM's labels in the results, but we acknowledge that there may be false negatives during filtering. We also note the subreddit of the posts, as it may offer insights into the commenters' background. To obtain an idea of how many false negatives were filtered out by the LLM, we conducted another manual analysis on 100 random comments labeled "neutral" by the LLM.

Results. We remove 6 comments due to lack of protestware context and 10 for being neutral, for a total of 97 comments taking some stance on protestware. Note that 92 comments were on a post in a tech-related subreddit (*i.e.* programming, sysadmin, devops, Python, javascript, coding, webdev, technews) and the other 5 were not (*i.e.* Ukraine, Anarchism). This suggests that the commentors are highly likely to be part of the larger tech community. Of the 97 comments, we found 78 oppositions, 15 in support, and 4 conditionally in support. Some opposing comments considered protestware as malware. For instance, one user said regarding `node-ipc` : "*Even if you want to be racist and assume that all Russians are the "bad guys" who deserve punishment, this is still a bad idea, because shotgunning malware will always hurt innocents.*" , revealing that an American NGO was a victim to `node-ipc`. Other oppositions were more creative, comparing protestware to poisoned-food: "*Sabotaged OSS is like donating poisoned food to those who suffer from starvation*". On the other hand, some argued that "*If a person wrote the software, and has given it out for free without being paid. They have a right to do whatever they want, whenever they want with that said software. It's THEIR software, nobody should be able to tell them what they can and cannot do with it.*" , though others countered that: "Just because it's free doesn't give the author the right to wipe someone else's data. That's straight up malware, just with a specific target. We should not allow malware

to given cover under any circumstance.". Note that these comments were referring to `node-ipc`, which deletes Russian and Belarusian users' files. Some were accepting of protestware as long as they do no harm, saying that "*Something that console prints political messages I think is fine, but randomly damaging someone's machine isn't protest; it's just random violence directed at people with no power.*"

Validation. With manual comparison of 100 neutral labeled LLM samples resulted in an agreement rate of 81%. An example of a comment showing support but was labeled as neutral by the LLMs is *"Yes. They can also chose to change the software so that it format the hard drive of whoever runs it. Or they could maintain it for years for free. It's up to them really. They don't owe anything to anyone"*. The support is shown by stating that developers can do anything with their free software.

Takeaway §4.2.2: Mass opposition with some support for protestware that do damage.

Though the majority of comments (80%) opposed protestware that caused damage to users' computers (e.g. `node-ipc`*), some (15%) showed support. A small minority (4%) declared support for protestware if they do not do damage, e.g. posting messages.*

4.3 RQ2.3: How do library usage change after they become protestware?

To understand how the protestware affected the trust dynamics in the OSS community, we conducted a usage trend analysis. We limited the scope of this analysis to only libraries since the "trust dynamics" are meaningful in the context of supply chain dependency.

Method. We conducted a comparative study of *usage* counts (number of dependents and weekly download counts) of a given library from when they were converted into protestware to the present time. Since 10 of the altered software libraries are written in JavaScript, we used the number of *dependents* and weekly download counts from the Node Package Manager (NPM) to measure the usage trends[10]. Dependent counts indicate how many libraries in the NPM repository are directly or indirectly depended on a given library, *i.e.* the downstream software in the supply chain. Weekly download counts indicate how many times a given library was downloaded from the repository. Although the number of weekly downloads can fluctuate drastically from week to week, however, maintaining a high weekly count would indicate that the trustworthiness is unaffected. To understand how the usage changed, we compare the counts from before [11] a software turned into protestware to the current count. For a comparison on what happened to libraries without protest, we also collected a set of libraries with similar functionality. Using relevant keywords and terms relating to the original function of the library as described by tags or in the README documentation, we found a matching library[12] for each protestware. To find the counts from the past, we used the WayBack machine [83]. Although the gross count across a two-year time span does not show a comprehensive view, the overall pattern can be seen.

Results. In total, we analyzed a collection of 10 protestwere JavaScript libraries. In this collection, 5 are targeting specific users, 3 are publicized, and 4 are still active protestware. Our analysis revealed that 8 out of 10 libraries have an increased number of dependency counts (Table 3). Note that for `left-pad` a protestware from *halted service* category, we were unable to determine the dependency count for a time near the removal using the Wayback machine. After NPM reinstated

[10]PHP also has some metrics for usage, however since the metrics are not directly comparable and both `yandex` and `Evolution` have extremely low dependents and no valid snapshots in the year of the commit, we decided not to include them.

[11]When there is no date that's both close and before, we use one that's slightly after.

[12]Since an exact match is not always possible, we made sure the library matched on at least one aspect of functionality. See replication package for connections on each.

the package, it had 1.4 million weekly downloads, although it is marked as deprecated [17]. Also for `hearthstone-db`, we were unable to obtain a previous snapshot using the Wayback machine. The weekly download count does not reveal any noticable patterns overall. The library with the largest sizable decrease in weekly download count was `EventSource`, going from 8.6 million to 4.4 million downloads each week. Then for comparison, we also performed this same analysis on NPM libraries with similar functionality, where we looked at one package for each instance of protestware. The results showed that the protestware libraries had similar overall behavior to the typical libraries with all of them having increased dependents count and also no noticable general patterns for download counts. However, when comparing the libraries with self-sabotage(`colors.js`,`faker.js`, and `left-pad`) we noticed that the similar library had a larger growth percentage. Overall, an increase in dependency count for protestware with the potential of affecting the software supply chain reveals a surprising aspect of the trust dynamics of the OSS community. A close look at the top-6 most popular NPM libraries with millions of current downloads (i.e., `es5-ext`, `EventSource`, `colors.js`, `faker.js`, `styled-components`, and `left-pad`) indicates that 4 out of 6 of them currently do not contain protestware, where `es5-ext` and `color.js` still remain active.

Takeaway §4.3: Turning libraries into protestware likely does not impact usage trends.

Even with the potential to affect the software supply chain, the dependency count of 8/10 protestware increased after turning into a protestware. We hypothesize that most downstream users are either unaware (in fact, most of protestware are non-transparent) or consider the protestware's functionality non-disruptive enough to warrant immediate removal.

Table 3. Dependents Comparison Table of JavaScript Libraries Dependents Count and Downloads. The lower row in each pair is a non-protestware library which has similar functionality to the underlined protestware it is paired with and is used for as a baseline comparison. Data collected using [83] and from [13]. (Note: hearthstone-db was not included in this analysis because it did not have any valid snapshots using the WayBack Machine.)

Name (linked)	Old Dep. Count	Curr. Dep. Count	Dep. % Diff	Old Weekly Downloads	Curr. Weekly Downloads	Downloads % diff
node-ipc	355	398	+12.1 %	892523	463895	-48.0 %
python-shell	202	293	+45.0 %	31520	61477	+95.0%
es5-ext	216	301	+39.4 %	12547294	9426242	-24.8 %
core-js-pure	282	628	+122.6 %	14344620	10159311	-29.2 %
EventSource	560	787	+40.5 %	8666668	4443843	-48.7 %
faye-websocket	365	719	+97.0 %	15640570	13301818	-15.0 %
sweetalert2	901	1844	+104.7 %	432118	578017	+33.8 %
sweetalert	279	320	+14.7 %	95462	81965	-14.1 %
colors.js	18958	22111	+16.7 %	22417827	15874535	-29.2 %
chalk	78494	121644	+55.0 %	128325959	242802857	+89.2 %
faker.js	2570	2632	+2.4 %	1712938	1586276	-7.4 %
casual	124	132	+6.5 %	78972	155471	+89.2 %
styled-components	17095	24414	+42.8 %	3907380	5395745	+38.1 %
react-base16-styling	88	132	+50.0 %	669747	996408	+48.8 %
nestjs-pino	35	177	+405.7 %	68901	463038	+572.0 %
nest-winston	163	305	+87.1 %	254097	422835	+66.4 %
left-pad	–	534	N/A	439841	1436312	+226.6 %
pad-left	14	71	+407.1 %	6456	177362	+2647.2 %

5 Discussion

5.1 Ethical Considerations

Following Kula *et al.* [104], we also considered labeling protestware as malignant or benign based on how a protestware behaves. However, labeling a protestware as benign or malicious might induce

i) political bias towards the protestware community, and *ii)* can easily be mistaken for our political stances. Thus, we chose to remove this label from our set of characteristics. Instead, we capture all of the important features for which an objective stance can be made. Furthermore, all disclosed information in this work is publicly available. Nonetheless, our paper aggregates information and presents new findings that may shape discourse and reaction. For instance, we shown that most (89%) of the protestware modified the README to display a banner and that some Reddit users support protestware that only display messages. This may cause readers of this paper to follow suite. However, we are not in a position to decide whether this is acceptable behavior and, thus, take no stance on said issue.

5.2 Threats to Validity

5.2.1 Construct Validity. The term "protestware" is relatively new, dating only back to 2022; thus, there is yet no standard definition agreed upon by everyone. For example, existing scientific literature [92, 104] implies protestware to be any pre-existing open-source software modified to protest, while Wikipedia considers it simply as a subset of malware [7]. To prevent such ambiguity and ensure construct validity, we adhere to the definition used in the existing literature and clearly define the scope in §2.1 that we operationalize. Note that utilizing different definitions of protewtware may result in different, yet still valid, taxonomies. However, we believe the definition we use is reasonable, as it captures key traits of protestware and provides a robust framework for analysis, yielding important insights into their impact on software supply chains. In **RQ1**–§3, we operationalized the constructs of triggers, methods for inducing protests, how users are targeted, and transparency of changes. However, we acknowledge that there may other constructs that were not considered. In **RQ2**–§4, we operationalized the constructs of "effects on the supply chain", "sentiment", and "usage trends". While sentiment [103, 108, 115, 119] and usage trends [95, 122] are commonly used metrics in prior works, we used a loose definition for "effects on the supply chain", as information was generally limited.

5.2.2 Internal Validity. **Searching for Protestware.** We relied on mostly external sources to curate our list of protest-inducing libraries, which ultimately limits our dataset to those protestware already known in the wild. This particularly concerns the ones that are altered software because these are the ones that may have an incentive to be secretive. However, we believe it is unlikely that we missed a protestware, especially impactful ones, because it went unnoticed and unreported by users. This conjecture is supported by the observation that people are often vocal about their experiences and observations in the age of the Internet and social media [93]. We also searched for protest-inducing by checking READMEs of NPM repos, which means we may have missed some of them that either do not have any declarations in the READMEs or fell through our automated filters. However, our goal is not to collect all cases but to collect a reasonable-sized sample to gain new insight. **Recency Bias in Dataset.** One limitation of our study is that our set of protestware are all from 2016 [21] or later. This bias is potentially because GitHub and NPM were founded in 2008 [5] and 2010 [10], respectively. It seems reasonable that there were no cases of protestware for 6-8 years, during which the open-source ecosystem was still maturing. In addition, while no concrete proof can be shown on the first usages of the term protestware, there is evidence to suggest that its usage started around 2022 [8]. For instance, the term "protestware" is known to be popularly used in 2022 after the `node-ipc` [48] incident [58]. We also filtered the search results for specifically those from 2015 and prior, but found none. For these reasons, we believe our dataset is reflective of the truth. **Qualitative Coding.** Our study is largely qualitative, which requires subjective evaluations that induce human bias into all manual coding [101, 109]. For instance, our set of characteristics in §3 may not be exhaustive. It is entirely possible, though unlikely, that another

researcher can extend our set. In another example, researchers may interpret positive or negative sentiment in §4 differently based on personal experiences. However, to mitigate said bias and increase trustworthiness of each qualitative stage of our study, we had multiple authors rigorously validate each other's codes. Lastly, because it is qualitative in nature, our study does not generalize to the data samples outside of our corpus [105]. **Retrospective Study via News Articles and Blogs.** For the supply chain impact analysis, we used news articles and blogs, which may contain biases, dramatization, and selectively cover events [112]. However, given the cost-benefit trade-off, we believe using these sources is reasonable because the alternative, for instance, is running an Internet-wide survey without any guarantee of a better outcome. Since most of the incidents in our case happened in the past, a user study-based approach might also suffer from such bias. To lower the probability of false facts, we cross-checked the facts claimed in the news across multiple sources. We take into consideration the reputation of the publisher by checking if it's in the Iffy list, which contains a list of unreliable sources [30] and has been used in prior work [102]. **Sentiment Analysis.** To study the sentiment towards these protestware, we referred to GitHub and Reddit comments and reactions. Using only these two platforms as our dataset may induce sampling bias. However, GitHub reactions were directly aimed at the specific change in software, which gives a micro-level perspective. On the other hand, Reddit comments provide a macro-level perspective on specific types of protestware, mostly ones that inflict damage to others, *e.g.* `node-ipc`, as these were the most prominent. We do not claim generalizability, but offer valuable insights in our initial findings for further investigation.

5.2.3 External Validity. For this study, we do not make any claims on generalizability. Rather, we focus on uncovering various characteristics of protestware libraries through a qualitative lens. Thus, we see no threats to external validity.

5.3 Grey Literature

Grey literature, which is defined as any non-peer reviewed work like news articles and blogs [84], has played a notable role in documenting protestware [42, 58–60]. These works have primarily focused on describing the timelines of events [42] and a few types of protestware [59], forgoing methodology documentation and in-depth analysis. In contrast, we employed transparent and systematic methodologies throughout each stage. These rigorous methods allowed us to collect a broad set of protestware and also enabled us to create a hierarchical taxonomy containing 12 different types across 7 high-level categories (Figure 1). We go beyond only classifying protestware and conduct systematic in-depth studies on various other important aspects (§3), including the social events that triggered it, the ways protests are induced, the targets affected by it, and the transparency of the libraries as they transition into protestware. Understanding the characteristics of protestware does not tell us their actual impact. To assess the broader impact (§4), we studied the effect of protestware on real-world software supply chains and the tech community, focusing on the sentiment in GitHub and Reddit comments and the download trends of NPM libraries.

6 Related Work

Our work studies protestware libraries that can be a threat to the software supply chain. We first look at works directly related to protestware, and then other works in software supply chain.

Protestware. To our knowledge, no prior work has systematically studied a broad set of protestware libraries that has supply chain implications. Cheong *et al.* proposed ethical guidelines for OSS developers potentially making protestware [92]. Kula *et al.* presents 3 categories of protestware by giving a few examples of each but no systematic collection of a diverse set of protestware [104]. Fan *et al.* studies how the community perceives, discusses, and responds to protestware, using

`colors.js` and `es5-ext` as the two samples [98] [13]. While Fan *et al.* studies two protestware in-depth, we employ an approach to study a broader set of protestware libraries. For malware-focused studies, as part of a larger interview with 25 developers, Wermke *et al.* included protestware as a minor-part of an OSS study (*i.e.* `node-ipc`) [120].

Packages and their Managers. One strand of work involves conducting a retrospective study to investigate packages and their managers, *e.g.* PyPI, npm, RubyGems. For instance, Zimmermann *et al.* studied dependency relationships, project maintainers, and known security issues for npm packages [123]. Zahan *et al.* analyzed the metadata of npm packages for signals of weak security [121]. Valiev *et al.* investigated the factors affecting the sustainability of the PyPI ecosystem [117]. Gonzalez *et al.* created a tool to automaticaly detect malicious packages using only GitHub commits [99]. Duan *et al.* studied over one million packages from PyPI, npm, and RubyGems [96].

Human Factors. Another facet of work is to understand the human factors in the software supply chain. For example, Abdalkareem *et al.* surveyed 88 Node.js developers to assess their opinions on the benefits and drawbacks of using trivial packages [87]. Wermke *et al.* interviewed 25 developers to understand their project processes, decisions, and considerations regarding open-source software [120]. Miller *et al.* interviewed 33 developers to study how they manage open-source dependency abandonment, realizing that the developers are often left with minimal support or guidance [111]. In another study, Miller *et al.* investigated online toxicity in the discussions of open-source software forums, finding entitled, demanding, arrogant, and insulting comments [110]. Bogart *et al.* studied how developers and organizations handle changes in dependencies through a series of interviews, discovering many challenges and headwinds [89, 90].

7 Conclusion

In this work, we curated the first comprehensive dataset containing 163 protestware. We then investigated the protestware through an extensive and multistage process to (1) study certain characteristics of protestware and (2) understand the impacts of disruptive protestware on the supply chain, community sentiment, and usage trends. The characteristics studied were: (i) the way the protest was implemented, (ii) trigger events that caused the protest, (iii) the protestwares' target(s), and (iv) if the protest was transparent about its behavior. A taxonomy was created to describe the different ways of inserting protests into the software. Our aftermath analysis showed that disruptive protestware, namely `left-pad` and `node-ipc`, can have a profound negative impact on downstream users. In addition, usage generally increases even after inserting protestware in code similar to other libraries. Furthermore, we found that developers maintain their own beliefs even with community pushback. The implication for regular developers is that they can never fully trust OSS, and they should have a contingency plan if and when their software fails due to an abuse of their upstream supply chain. Future work could investigate ways to automatically detect protestware so that users are notified immediately rather than retroactively.

8 Data Availability

A replication package is uploaded using Zenodo [86]. The package contains spreadsheets with protestware, coded labels and themes, GitHub/NPM links, Wayback dates, collected articles, websites, LLM and manual labels, and the appendix.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback, which helped us improve the paper. We also thank Joshua A. Levine for valuable discussions and feedbacks. Finally, we are thankful to Sathvik Reddy Nookala for his help during the initial phase of the study. This work was partially supported by the University of Arizona's IT4IR TRIF program.

[13] [97] is a two-page paper of preliminary results to the full length paper [98].

References

[1] [n. d.]. 996 working hour system. https://en.wikipedia.org/wiki/996_working_hour_system. Accessed: Feb 21, 2025.
[2] [n. d.]. all-the-package-repos: Normalized repository URLs for every package in the npm registry. Updated daily. https://github.com/nice-registry/all-the-package-repos. Accessed: Feb 17, 2025.
[3] [n. d.]. American NGO affected by your recklessness. https://archive.ph/emyJb. Accessed: Feb 22, 2025.
[4] [n. d.]. API Platform. https://openai.com/api/. Accessed: Feb 17, 2025.
[5] [n. d.]. GitHub. https://en.wikipedia.org/wiki/GitHub. Accessed: Jan 27, 2025.
[6] [n. d.]. Google AI for Developers. https://ai.google.dev/. Accessed: Feb 17, 2025.
[7] [n. d.]. Hacktivism. https://en.wikipedia.org/wiki/Hacktivism#Protestwarem. Accessed: Feb 17, 2025.
[8] [n. d.]. Hacktivism. https://en.wikipedia.org/wiki/Hacktivism#Protestware. Accessed: Jan 27, 2025.
[9] [n. d.]. kvcb - snippet.host. https://archive.ph/QJ5IK. Accessed: Feb 22, 2025.
[10] [n. d.]. npm. https://en.wikipedia.org/wiki/Npm. Accessed: Jan 27, 2025.
[11] [n. d.]. Reddit API | Developer Platform. https://developers.reddit.com/docs/api. Accessed: Feb 17, 2025.
[12] 2008. AntonSchevchuk/yandex. https://github.com/AntonShevchuk/yandex.
[13] 2010. npm. https://www.npmjs.com/.
[14] 2013. evolution-cms/evolution. https://github.com/evolution-cms/evolution.
[15] 2014. firefox-boycott. https://github.com/jackmaney/firefox-boycott.
[16] 2014. Include notes on diversity in tech. Docs for new HTML formatting Commit. https://github.com/vladimirbuskin/leaflet-control-geocoder/commit/deeaa0071f17cdd0cba551bfe823fab6633e1a7b.
[17] 2014. left-pad. https://github.com/left-pad/left-pad.
[18] 2014. vue - npm. https://www.npmjs.com/package/vue.
[19] 2014. Vue.js - The Progressive JavaScript Framework. https://vuejs.org/.
[20] 2016. ec-/Quake3e. https://github.com/ec-/Quake3e.
[21] 2016. How one programmer broke the internet by deleting a tiny piece of code. https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code.
[22] 2016. I've Just Liberated My Modules. https://web.archive.org/web/20161203055443/https://medium.com/@azerbike/i-ve-just-liberated-my-modules-9045c06be67c.
[23] 2016. npm Blog Archive: kik, left-pad, and npm. https://blog.npmjs.org/post/141577284765/kik-left-pad-and-npm.
[24] 2016. styled-components. https://github.com/styled-components/styled-components.
[25] 2017. Yet Another Dialog(yad). https://github.com/v1cont/yad/tree/master.
[26] 2019. Developer takes down Ruby library after he finds out ICE was using it. https://www.zdnet.com/article/developer-takes-down-ruby-library-after-he-finds-out-ice-was-using-it/.
[27] 2019. Free Hong Kong Pro-Palestinian protest on UA· pdyck/hearthstone-db@52be873. https://github.com/pdyck/hearthstone-db/commit/52be873eb6f8551b077d5f4ebfb146eccf45288c.
[28] 2019. NestJS-Pino. https://github.com/iamolegga/nestjs-pino.
[29] 2019. Removed unused devDependencies amp; Updated README.md · EmilyMew/dom-pdf@67fe264. https://github.com/EmilyMew/dom-pdf/commit/67fe264996d891e65899667b3dcf058500d5a067.
[30] 2020. Iffy Index of Unreliable Sources. https://iffy.news/index/.
[31] 2020. racial-equity-banner. https://github.com/blittle/racial-equity-banner.
[32] 2020. updates readme file · saqy/angular-packages@a3d1f06. https://github.com/saqy/angular-packages/commit/a3d1f061d1477edde5999f5d92e93936c56ab925.
[33] 2020. Venezuela crisis: Anger over shortages triggers protests. https://www.bbc.com/news/world-latin-america-54354225.
[34] 2020. widget-engrave. https://github.com/onestlatech/widget-engreve.
[35] 2021. No more free work from Marak - Pay Me or Fork This. https://web.archive.org/web/20210704022108/https://github.com/Marak/faker.js/issues/1046.
[36] 2021. No more free work from Marak - Pay Me or Fork This · Issue 1046 · Marak/faker.js · GitHub. https://web.archive.org/web/20210704022108/https:/github.com/Marak/faker.js/issues/1046.
[37] 2021. nuxt-block-russia-belarus. https://github.com/kevinl95/nuxt-block-russia-belarus.
[38] 2022. Add blacklist · arendst/Tasmota@98cbf25. https://github.com/arendst/Tasmota/commit/98cbf2587a1a914bbd16996ebb48dd451d3da448.
[39] 2022. Add STOP WAR message for Russians by limonte · Pull Request #2428 · sweetalert2/sweetalert2. https://github.com/sweetalert2/sweetalert2/pull/2428/commits/86d5af1686a5270a593f14ec90c6943884447824.
[40] 2022. Adds new American flag module Marak. https://github.com/Marak/colors.js/commit/074a0f8ed0c31c35d13d28632bd8a049ff136fb6?diff=unified&w=0.
[41] 2022. After 'protestware' attacks, a Russian bank has advised clients to stop updating software. https://www.theverge.com/2022/3/21/22989339/protestware-attacks-russia-sberbank-open-source.

[42] 2022. Alert: peacenotwar module sabotages npm developers in the node-ipc package to protest the invasion of Ukraine. https://snyk.io/blog/peacenotwar-malicious-npm-node-ipc-package-vulnerability/.
[43] 2022. BIG sabotage: Famous npm package deletes files to protest Ukraine war. https://www.bleepingcomputer.com/news/security/big-sabotage-famous-npm-package-deletes-files-to-protest-ukraine-war/.
[44] 2022. cannot find module node_modules/styled-components/postinstall.js · Issue 3706 · styled-components/styled-components. https://github.com/styled-components/styled-components/issues/3706.
[45] 2022. chore: Give Peace a Chance · medikoo/es5-ext@28de285. https://github.com/medikoo/es5-ext/commit/28de285ed433b45113f01e4ce7c74e9a356b2af2.
[46] 2022. Code-Sabotage Incident in Protest of Ukraine War Exposed Open Source Risks. https://www.darkreading.com/application-security/recent-code-sabotage-incident-latest-to-highlight-code-dependency-risks.
[47] 2022. Dev corrupts NPM libs 'colors' and 'faker' breaking thousands of apps. https://www.bleepingcomputer.com/news/security/dev-corrupts-npm-libs-colors-and-faker-breaking-thousands-of-apps/.
[48] 2022. Did the term protestware exist before? - Puppy Linux Discussion Forum. https://forum.puppylinux.com/viewtopic.php?p=52839&sid=3a36de55392fe34b1e4c5407b1d4ca9e#p52839.
[49] 2022. drop k hujam support of russian language. https://github.com/v1cont/yad/commit/e38f7fa71aa9b2dff408ae14ca7133e4fdc4b02a.
[50] 2022. faker.js. https://web.archive.org/web/20220129022735/https://github.com/marak/Faker.js/.
[51] 2022. feat: Made it clear that we stand with Ukraine · terraform-aws-modules/terraform-aws-eks@fad350d. https://github.com/terraform-aws-modules/terraform-aws-eks/commit/fad350d5bf36a7e39aa3840926b4c9968e9f594c.
[52] 2022. MongoDB Assistance to Ukraine, Shut Down of Work in Russia. https://www.mongodb.com/blog/post/mongodb-assistance-ukraine-shut-down-work-russia.
[53] 2022. New Protestware Found Lurking in Highly Popular NPM Package. https://checkmarx.com/blog/new-protestware-found-lurking-in-highly-popular-npm-package/.
[54] 2022. node-ipc edit. https://web.archive.org/web/20220321220122/https://github.com/RIAEvangelist/node-ipc/blob/847047cf7f81ab08352038b2204f0e7633449580/dao/ssl-geospec.js.
[55] 2022. peacenotwar. https://web.archive.org/web/20220317095621/https://github.com/RIAEvangelist/peacenotwar.
[56] 2022. Pro-Ukraine 'Protestware' Pushes Antiwar Ads, Geo-Targeted Malware. https://www.reddit.com/r/hacking/comments/thlqm3/comment/i1a0lf0/?utm_source=share&utm_medium=web3x&utm_name=web3xcss&utm_term=1n. Accessed: Feb 22, 2025.
[57] 2022. Protests erupt across China in unprecedented challenge to Xi Jinping's zero-Covid policy. https://www.cnn.com/2022/11/26/china/china-protests-xinjiang-fire-shanghai-intl-hnk/index.html.
[58] 2022. 'Protestware' is on the rise, with programmers self-sabotaging their own code. Should we be worried? https://theconversation.com/protestware-is-on-the-rise-with-programmers-self-sabotaging-their-own-code-should-we-be-worried-190836.
[59] 2022. Protestware is trending in open source: 4 different types and their impact. https://snyk.io/blog/protestware-open-source-types-impact/.
[60] 2022. Protestware on the rise: Why developers are sabotaging their own code. https://techcrunch.com/2022/07/27/protestware-code-sabotage/.
[61] 2022. racial-equity-banner. https://github.com/filipe-freire/hands-off-ukraine-banner.
[62] 2022. Removing React project dependencies involving the es5-ext protestware? https://stackoverflow.com/questions/71877958/removing-react-project-dependencies-involving-the-es5-ext-protestware. Accessed: May 14, 2024.
[63] 2022. samber/awesome-prometheus-alerts@6bfcdcc. https://github.com/samber/awesome-prometheus-alerts/commit/6bfcdcca165e57c6fa09a561515c33284caa20c2.
[64] 2022. Sber advises against updating software due to risk of cyber attacks. https://www.fontanka.ru/2022/03/18/70517441/.
[65] 2022. StandWithUkraine banner and related documents. https://github.com/vshymanskyy/StandWithUkraine.
[66] 2022. Stop war in Ukraine! https://github.com/evolution-cms/evolution/commit/1c586bc76f739264dcf0482530945875fa444b77.
[67] 2022. The story behind colors.js and faker.js. https://www.revenera.com/blog/software-composition-analysis/the-story-behind-colors-js-and-faker-js/.
[68] 2022. Undesired Behavior in styled-components. https://security.snyk.io/vuln/SNYK-JS-STYLEDCOMPONENTS-3149924.

[69] 2022. Update README.md · AlexKhymenko/ngx-permissions@93d08f2. https://github.com/AlexKhymenko/ngx-permissions/commit/93d08f234d1ccc9a096006b5633fedb936b07865.
[70] 2022. update · Yaffle/EventSource@de13792. https://github.com/Yaffle/EventSource/commit/de137927e13d8afac153d2485152ccec48948a7a.
[71] 2022. v11 peacenotwar update · RIAEvangelist/node-ipc@1220522 · GitHub. https://web.archive.org/web/20220317103231/https://github.com/RIAEvangelist/node-ipc/commit/1220522453a0388cb4af1a74fe9a0482b6b3a9f3.
[72] 2022. web4ukraine. https://github.com/pilotcz/web4ukraine.
[73] 2023. Added support Palestine banner (182). https://github.com/FajarKim/github-readme-profile/commit/9c594323ea80a271694ffdb74ded2ecf459403c5.
[74] 2023. Commit c4883bf. https://github.com/mehrnooshbahmani/liberte/commit/c4883bf95589cd47f9d9f9b2dc2e9e2923f58c65.
[75] 2023. e2eakarev - npm. https://www.npmjs.com/package/e2eakarev.
[76] 2023. node-ipc. https://github.com/RIAEvangelist/node-ipc.
[77] 2023. Protestware taps npm to call out wars in Ukraine, Gaza. https://www.reversinglabs.com/blog/protestware-taps-npm-to-call-out-wars-in-ukraine-gaza. Accessed: May, 2024.
[78] 2024. Add perf har-poon · doramatadora/har-poon@f9c7609. https://github.com/doramatadora/har-poon/commit/f9c7609b0a1a4b51570b27d96e37abe9531f9e0c.
[79] 2024. Commit 72ea89d. https://github.com/Autumn-one/javascript-treasure/commit/72ea89db6eff63efb31fc2b23d7d2232e3d82bc7.
[80] 2024. Open Source Initiative_2024. https://opensource.org/osd https://opensource.org/osd.
[81] 2024. Pro-Palestinian protest on UA campus ends in tear gas, rubber bullets, arrests. What's next? https://www.azcentral.com/story/news/local/arizona-education/2024/05/01/what-to-know-about-pro-palestinian-protest-at-university-of-arizona/73532255007/.
[82] 2024. Protestware - How node-ipc turned into malware. https://www.lunasec.io/docs/blog/node-ipc-protestware/.
[83] 2024. Wayback Machine. https://web.archive.org/ https://web.archive.org/.
[84] 2024. What is Grey Literature? https://guides.library.illinois.edu/c.php?g=1310347. Accessed: Feb 24, 2025.
[85] 2025. Commit 279ec0e. https://github.com/Coral-UI/core/commit/279ec0e61c704871626fb8b4ec932c2b5c0d0b38.
[86] 2025. Supplementary Material. https://zenodo.org/records/14929674.
[87] Rabe Abdalkareem, Olivier Nourry, Sultan Wehaibi, Suhaib Mujahid, and Emad Shihab. 2017. Why do developers use trivial packages? an empirical case study on npm. In *Proceedings of the 2017 11th joint meeting on foundations of software engineering*. 385–395.
[88] Dharun Anandayuvaraj and James C Davis. 2022. Reflecting on recurring failures in iot development. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 1–5.
[89] Christopher Bogart, Christian Kästner, and James Herbsleb. 2015. When it breaks, it breaks: How ecosystem developers reason about the stability of dependencies. In *2015 30th IEEE/ACM International Conference on Automated Software Engineering Workshop (ASEW)*. IEEE, 86–89.
[90] Christopher Bogart, Christian Kästner, James Herbsleb, and Ferdian Thung. 2016. How to break an API: cost negotiation and community values in three software ecosystems. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. 109–120.
[91] Jesse Chen, Dharun Anandayuvaraj, James C Davis, and Sazzadur Rahaman. 2024. On the Contents and Utility of IoT Cybersecurity Guidelines. *Proceedings of the ACM on Software Engineering* 1, FSE (2024), 1400–1423.
[92] Marc Cheong, Raula Gaikovina Kula, and Christoph Treude. 2023. Ethical Considerations Towards Protestware. *IEEE Software* (2023).
[93] Matteo Cinelli, Gianmarco De Francisci Morales, Alessandro Galeazzi, Walter Quattrociocchi, and Michele Starnini. 2021. The echo chamber effect on social media. *Proceedings of the National Academy of Sciences* 118, 9 (2021).
[94] Lucian Constantin. 2022. Developer sabotages own NPM module prompting open-source supply chain security questions. *CSO Online*. https://www.csoonline.com/article/572327/developer-sabotages-own-npm-module-prompting-open-source-supply-chain-security-questions.html
[95] Tapajit Dey and Audris Mockus. 2018. Are software dependency supply chain metrics useful in predicting change of popularity of npm packages?. In *Proceedings of the 14th international conference on predictive models and data analytics in software engineering*. 66–69.
[96] Ruian Duan, Omar Alrawi, Ranjita Pai Kasturi, Ryan Elder, Brendan Saltaformaggio, and Wenke Lee. 2021. Towards measuring supply chain attacks on package managers for interpreted languages. (2021).
[97] Youmei Fan, Dong Wang, Supatsara Wattanakriengkrai, Hathaichanok Damrongsiri, Christoph Treude, Hideaki Hata, and Raula Gaikovina Kula. 2024. Going Viral: Case Studies on the Impact of Protestware. In *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*. 308–309.

[98] Youmei Fan, Dong Wang, Supatsara Wattanakriengkrai, Hathaichanok Damrongsiri, Christoph Treude, Hideaki Hata, and Raula Gaikovina Kula. 2025. Developer reactions to protestware in open source software: the cases of color. js and es5. ext. *Empirical Software Engineering* 30, 2 (2025), 1–25.
[99] Danielle Gonzalez, Thomas Zimmermann, Patrice Godefroid, and Max Schäfer. 2021. Anomalicious: Automated detection of anomalous and potentially malicious commits on github. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 258–267.
[100] Leo A Goodman. 1961. Snowball sampling. *The annals of mathematical statistics* (1961), 148–170.
[101] Greg Guest, Kathleen M MacQueen, and Emily E Namey. 2012. *Applied thematic analysis*. sage.
[102] Hans WA Hanley, Deepak Kumar, and Zakir Durumeric. 2024. Specious sites: Tracking the spread and sway of spurious news stories at scale. In *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1609–1627.
[103] Doaa Mohey El-Din Mohamed Hussein. 2018. A survey on sentiment analysis challenges. *Journal of King Saud University-Engineering Sciences* 30, 4 (2018), 330–338.
[104] Raula Gaikovina Kula and Christoph Treude. 2022. In war and peace: The impact of world politics on software ecosystems. *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. doi:10.1145/3540250.3560882
[105] Lawrence Leung. 2015. Validity, reliability, and generalizability in qualitative research. *Journal of family medicine and primary care* 4, 3 (2015), 324–327.
[106] Michael Lipsky. 1968. Protest as a political resource. https://doi.org/10.2307/1953909. *American political science review* 62, 4 (1968), 1144–1158.
[107] CJ Mann. 2003. Observational research methods. Research design II: cohort, cross sectional, and case-control studies. *Emergency medicine journal* 20, 1 (2003), 54–60.
[108] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal* 5, 4 (2014), 1093–1113.
[109] Matthew B Miles and A Michael Huberman. 1994. *Qualitative data analysis: An expanded sourcebook*. sage.
[110] Courtney Miller, Sophie Cohen, Daniel Klug, Bogdan Vasilescu, and Christian KaUstner. 2022. "Did you miss my comment or what?" understanding toxicity in open source discussions. In *Proceedings of the 44th International Conference on Software Engineering (FSE) 2022*. 710–722.
[111] Courtney Miller, Christian Kästner, and Bogdan Vasilescu. 2023. "We Feel Like We're Winging It:" A Study on Navigating Open-Source Dependency Abandonment. In *31st ACM Symposium on the Foundations of Software Engineering 2023*. 1281–1293.
[112] Sendhil Mullainathan and Andrei Shleifer. 2002. Media bias.
[113] Open-Source-Peace. 2022. List of open-source projects containing protestware. https://github.com/open-source-peace/protestware-list/tree/main .
[114] Sophie Stephenson, Majed Almansoori, Pardis Naeini, Danny Huang, and Rahul Chatterjee. 2023. Abuse Vectors: A Framework for Conceptualizing IoT-Enabled Interpersonal Abuse. In *32nd USENIX Security Symposium 2023*. 69–86.
[115] Maite Taboada. 2016. Sentiment analysis: An overview from linguistics. *Annual Review of Linguistics* (2016).
[116] Ralph H Turner. 1969. The public perception of protest. *American sociological review* (1969), 815–831.
[117] Marat Valiev, Bogdan Vasilescu, and James Herbsleb. 2018. Ecosystem-level determinants of sustained activity in open-source projects: A case study of the PyPI ecosystem. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 644–655.
[118] Francesca Vassallo. 2018. The evolution of protest research: Measures and approaches. *PS: Political Science & Politics* (2018).
[119] Mayur Wankhade, Annavarapu Chandra Sekhara Rao, and Chaitanya Kulkarni. 2022. A survey on sentiment analysis methods, applications, and challenges. *Artificial Intelligence Review* 55, 7 (2022), 5731–5780.
[120] Dominik Wermke, Jan H Klemmer, Noah Wöhler, Juliane Schmüser, Harshini Sri Ramulu, Yasemin Acar, and Sascha Fahl. 2023. " Always Contribute Back": A Qualitative Study on Security Challenges of the Open Source Supply Chain. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1545–1560.
[121] Nusrat Zahan, Thomas Zimmermann, Patrice Godefroid, Brendan Murphy, Chandra Maddila, and Laurie Williams. 2022. What are weak links in the npm supply chain?. In *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice*. 331–340.
[122] Ahmed Zerouali, Tom Mens, Gregorio Robles, and Jesus M Gonzalez-Barahona. 2019. On the diversity of software package popularity metrics: An empirical study of npm. In *2019 IEEE 26th international conference on software analysis, Evolution and Reengineering (SANER)*. IEEE, 589–593.
[123] Markus Zimmermann, Cristian-Alexandru Staicu, Cam Tenny, and Michael Pradel. 2019. Small world with high risks: A study of security threats in the npm ecosystem. In *28th USENIX Security Symposium 2019*. 995–1010.

Received 2024-09-04; accepted 2025-04-01