Confidentiality Protection in Crowdsourcing

Student Name: Simran Saxena

IIIT-D-MTech-CS-GEN-18-2013104
April, 2018

Indraprastha Institute of Information Technology
New Delhi

Submitted in partial fulfillment of the requirements
for the Degree of M.Tech. in Computer Science,
in General Category

Keywords: Crowdsourcing, Confidentiality, Information Leak

Certificate

This is to certify that the thesis titled **“Confidentiality Protection in Crowdsourcing"** submitted by **Simran Saxena** for the partial fulfillment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by her under our guidance and supervision in the Security and Privacy group at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Ponnurangam Kumarguru

Indraprastha Institute of Information Technology, New Delhi

Abstract

Crowdsourcing is the practice of getting information or input for a task or project from a number of people by floating out the task at hand to a pool of people who are usually not full-time employees. Use of Crowdsourcing to get work done is on the rise due to the benefits that it offers. Trends and surveys show that the conventional workforce is moving towards gig-economy, and by 2020, 43% of the US workforce is expected to be on-demand workforce [9]. In our work, we focus on higher level software development/engineering tasks and how they could potentially lead to Confidentiality Loss in the process of giving the task and getting it done. We look at different stages and components of the crowdsourcing cycle to examine potential information leak sources. We conducted a survey to study how people perceive this problem and what is their level of understanding when it comes to sharing information online. We also analyzed a dataset of tasks posted online previously to get insight if the problem persists in the real world or not. Some conversations between the task posters and the workers were studied along with a dataset of reviews for workers to get a deeper insight into the problem and its detection. Based on the analysis, we propose NLP based techniques to detect such potential leaks and nudge the task poster before the information is dissipated. Having such an additional layer of scrutiny at the level of companies before a task is given for crowdsourcing out will keep in check any loss of confidential information.

Acknowledgments

It is my privilege to express my sincerest gratitude to my advisors, Dr. Ponnurangam Kumaraguru and Dr. Alpana Dubey, for giving me this opportunity to work on this thesis. I would also like to thank them for their valuable inputs, guidance, encouragement and wholehearted support throughout the thesis. I would like to thank my esteemed committee members, Dr. Arun Balaji and Dr. Niharika Sachdeva for agreeing to evaluate my thesis work. I am also grateful to all the members of my Precog family at IIIT Delhi who have consistently helped me with their inputs and suggestions on the work. Special thanks to Indira Sen for guiding me all along. Last but not the least, I would like to thank all my supportive family and friends who encouraged me and kept me motivated throughout the thesis.

Contents

List of Figures

Chapter 1

Research Motivation and Aim

1.1 Research Motivation

The term crowdsourcing was coined back in 2006 by Howe and Robinson [23, 30]. Since then, the use of crowdsourcing to get things done has been gaining a constant momentum. It is being used widely for tasks ranging from Image Annotations [3, 25, 41], Data Labeling [3, 6, 8] to complex tasks like getting Code Modules written or getting Wireframes or Prototypes made. Typically, crowdsourcing takes place over the Internet, but in many cases, it can be done physically as well. Crowdsourcing has proved to be a very effective way of getting repetitive tasks done where the intervention of human intelligence is required. In our work, we focus on the higher level software development/engineering tasks related Crowdsourcing tasks.

It is estimated that by 2020, the gig workforce will raise up to reach 43% of the workforce in USA [9]. With this rise, it is believed that the share of software development related tasks will also increase. The on-demand hiring of workers for shorter time periods has given much more flexibility in getting things done. Popular apps and services like Uber, Airbnb, Zomato function on the very same principle of crowdsourcing. Companies as old as 180 years such as P&G are also experimenting with the way their organizations function and are trying to add on-demand workers to step up their innovation game [2]. It has been observed in a pilot study that with crowdsourcing, the products were delivered faster and at a cheaper rate than conventional methods 60% of the times [2]. TopCoder, one of the popularly used platforms for software development crowdsourcing, claims that crowdsourced software development could reduce the costs by 30% - 80% as compared to using conventional ways [10].

Crowdsourcing As The Next Big Revolution

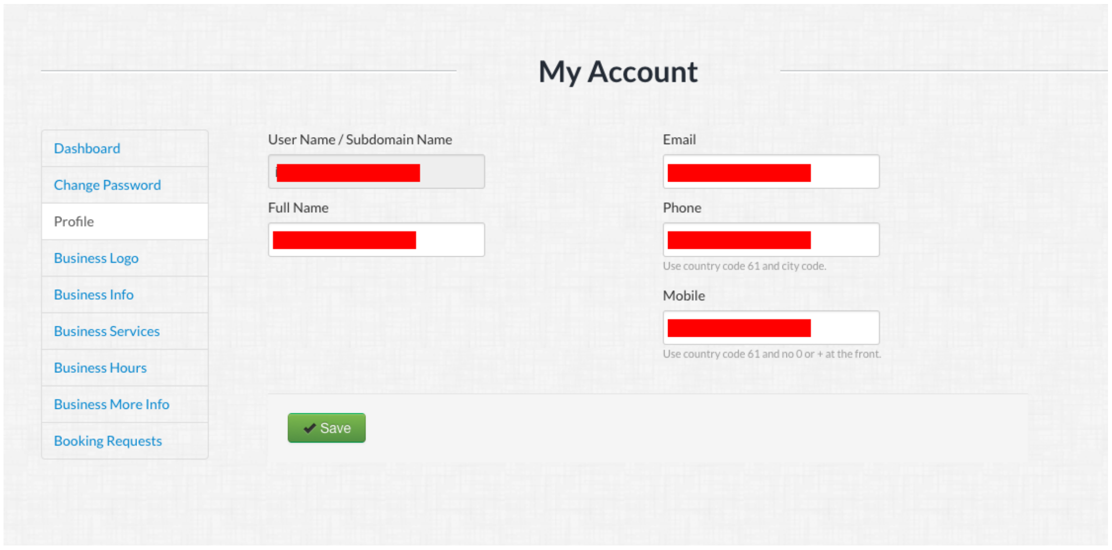
According to the 2017 Technology Vision document of the prominent consulting firm ***Accenture***, the world is moving towards "on-demand and online workforce" from "conventional full-time employees." They predict that the benefits and flexibility offered by on-demand workforce will outweigh the practice of old-school employment and worker system [2]. The world will eventually move towards gig economy. The 2016 Field Nation Freelancer Study: The Changing Face of the New Blended Workforce [1], studies how people perceive the advent of freelancing. As per the survey, 74% of the professional freelancers say that it's an ideal working situation for them. 95% said that they love, like or are satisfied with what they do on a daily basis. Seventy two percent say they make the same amount or more money than they did before. Statistics like this confirm that people are also positive about working in the freelancing environment. All these facts make us believe firmly that the paradigm shift to gig-economy is sure to happen over the coming decade. This shift in the type of employment will also be seen as a revolution in the way the economy functions.

Now that we know that we're about to step into the next revolution, it's always better to be prepared while stepping into it. While crowdsourcing has its own benefits, it also comes with many associated risks and problems. The trustworthiness of workers, the performance capabilities of workers, the reliability of workers, how to find the right match of workers for the tasks, how to prevent workers from cheating the system, etc. A lot of research is already being done into some of these problems. The problem that we'll focus on in this work is the ***Loss of Confidentiality in the Crowdsourcing Process***. We believe, that while giving away a task to a worker or in the process of getting a task done by workers, it is quite possible that the task poster is exposing some confidential information which the worker or the potential worker should not have had access to, actually had access to it.

On analysis of a data-set[1] of previously posted tasks, it was observed that by logging into a third party service using the credentials given in the Task Description field of the data-set we were able to access:

- Email ID, Phone Number, Address of the company
- Option to change password of the company's account on the platform
- The records of the customer of a company

[1]A dataset of 65,499 tasks pertaining to Software Development extracted from a leading Crowdsourcing Platform. The dataset cannot be shared due to data sharing policy



My Account

- Dashboard
- Change Password
- Profile
- Business Logo
- Business Info
- Business Services
- Business Hours
- Business More Info
- Booking Requests

User Name / Subdomain Name
[redacted]

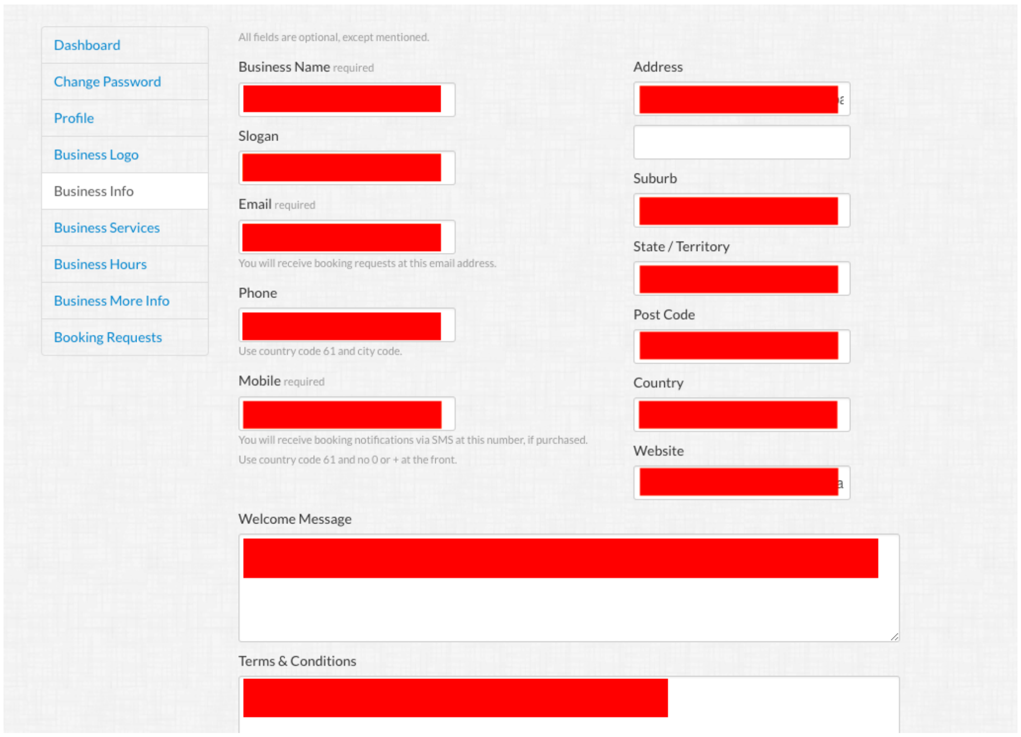
Full Name
[redacted]

Email
[redacted]

Phone
[redacted]
Use country code 61 and city code.

Mobile
[redacted]
Use country code 61 and no 0 or + at the front.

Figure 1.1: Example of exposed details of a company accessed on a third party website using a vulnerability found in a task description. Company's *name, email, and phone numbers* were exposed with an option to change them.



- Dashboard
- Change Password
- Profile
- Business Logo
- Business Info
- Business Services
- Business Hours
- Business More Info
- Booking Requests

All fields are optional, except mentioned.

Business Name required
[redacted]

Slogan
[redacted]

Email required
[redacted]
You will receive booking requests at this email address.

Phone
[redacted]
Use country code 61 and city code.

Mobile required
[redacted]
You will receive booking notifications via SMS at this number, if purchased.
Use country code 61 and no 0 or + at the front.

Address
[redacted]

Suburb
[redacted]

State / Territory
[redacted]

Post Code
[redacted]

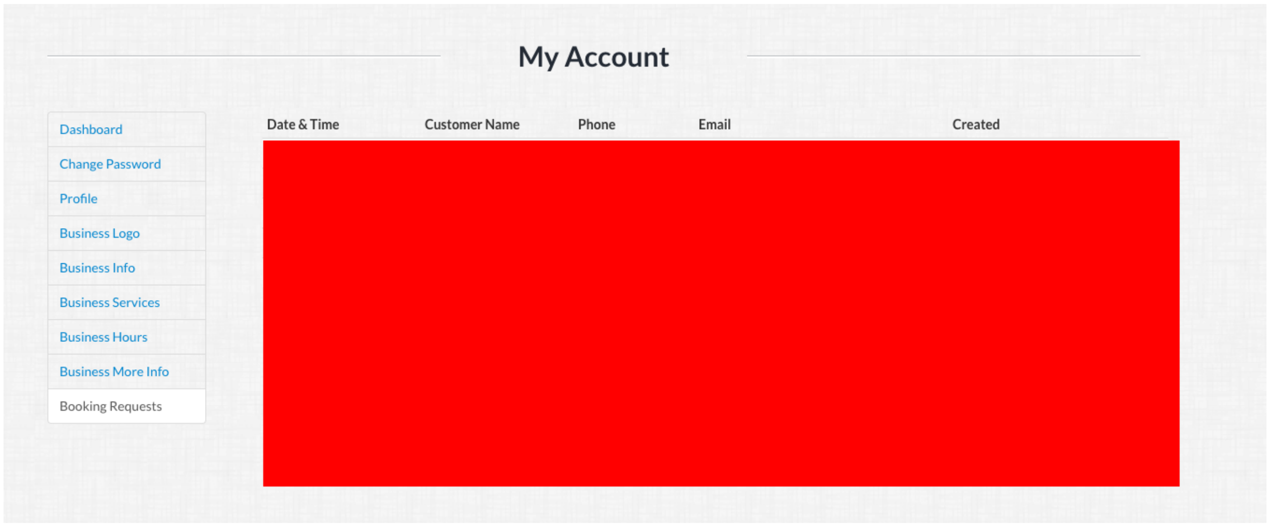
Country
[redacted]

Website
[redacted]

Welcome Message
[redacted]

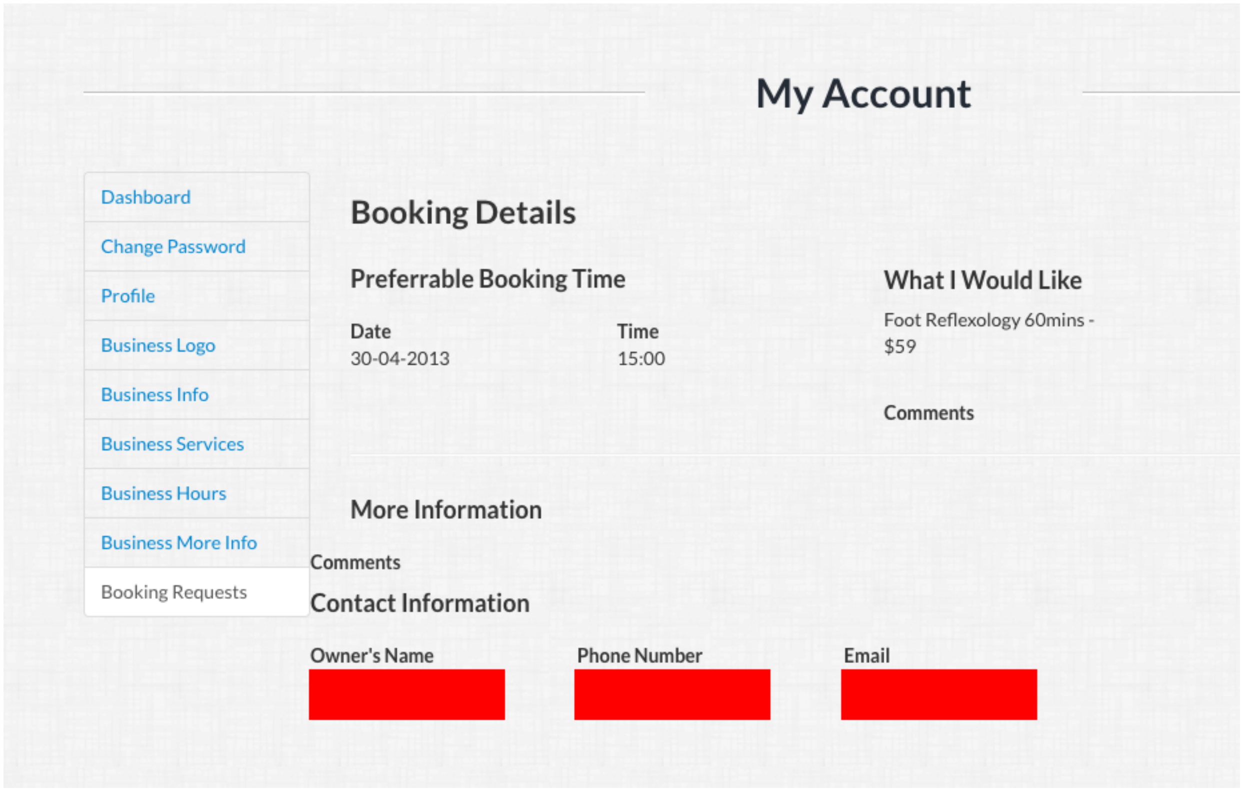
Terms & Conditions
[redacted]

Figure 1.2: Example of exposed details of a company accessed on a third party website using a vulnerability found in a task description. Company's *full address, email, and phone numbers* were exposed with an option to change them. It was also possible to make changes to company's brand related details such as the *welcome message, slogan, and terms conditions.*



Date & Time	Customer Name	Phone	Email	Created
[redacted]				

Figure 1.3: Example of exposed details of a company accessed on a third party website using a vulnerability found in a task description. The list of *customers of the company* along with with their *phone numbers and email addresses* was exposed.



Booking Details

Preferrable Booking Time

Date
30-04-2013

Time
15:00

What I Would Like
Foot Reflexology 60mins - $59

Comments

More Information

Comments

Contact Information

Owner's Name	Phone Number	Email
[redacted]	[redacted]	[redacted]

Figure 1.4: Example of exposed details of a company accessed on a third party website using a vulnerability found in a task description. *PII (personally identifiable information)* of customers along with the details of services availed by them was exposed.

1.2 Research Aim

In this work, we aim to address:

- In the crowdsourcing setting, how does confidentiality loss come into play? Which stages/processes in the crowdsourcing process are involved in confidentiality loss?
- What attributes in the crowdsourcing process lead to information leak?
- What type of data can be leaked in crowdsourcing?
- Can NLP/Rule Based techniques aid organizations in identifying sensitive information leak? How can such frameworks be incorporated at the organization level?

To the best of our knowledge, the aspect of confidentiality loss in the crowdsourcing setting has not been delved into. This research is an extension of the work done by Accenture Technology Labs, Bangalore, where they propose a Framework to Preserve Confidentiality in Crowdsourced Software Development [15]. We try to fill in some of the gaps and challenges that are discussed in this work apart from addressing the questions mentioned above.

- ***Which stages/processes in the crowdsourcing process are involved in confidentiality loss:*** We conducted a survey, looked at previously posted tasks and their various attributes, looked at the conversations between task posters and workers, analyzed the reviews received by workers and zeroed down on the stages where the information leak could occur. From our analysis, we were able to validate our claim about the stages that mainly entail loss of confidentiality.
- ***Which attributes in the Crowdsourcing process lead to information leak:*** Based on the survey, detailed analysis of a corpus of previously posted tasks and literature, we narrowed down the components of a task that could give away confidentiality. A task typically consists of 20 attributes such as task title, task description, task duration, etc. We see how each of these could lead to confidentiality loss and to what extent.
- ***What type of data can be leaked in crowdsourcing:*** Apart from the basic confidential data such as PII (Personally identifiable information), we also find out what other type of confidential data lies in the crowdsourcing setting when it comes to specific organizations and software development. We look in great detail how this varies at the level of an organization and seek the opinion of experts in organizations who are actively involved in crowdsourcing.
- ***Can NLP/Rule Based techniques aid organizations in identifying sensitive information leak? How can such frameworks be incorporated at the organization level:*** We study how various NLP and rule-based techniques can help in analyzing the task before it is posted online, how they function when it comes to a very specific domain such as crowdsourcing, software development, and a specific organization.

Chapter 2

Related Work

This is an extension of the work done by Dubey et al. [15] where they propose a Framework to Preserve Confidentiality in Crowdsourced Software Development. They briefly explain the challenges that are faced while getting software development work done via Crowdsourcing. Our work aims to delve deeper into those challenges, seek answers to some of them, and give an NLP and Rule Based approach to tackle the problem of Confidentiality Loss.

Ke Mao et al. [30] have conducted an extensive survey of 203 papers based on Software Development related Crowdsourcing tasks. They also give a comprehensive list of all the PhD and Master thesis in the domain of Software Development related Crowdsourcing. Extensive research has been done on other aspects of crowdsourcing such as, how crowdsourcing can be used to get tasks done efficiently [26, 28] and securely [14, 20, 45]. However, none of them cover the aspect of loss of Confidentiality. To the best of our knowledge, this is the first attempt to study this aspect of Crowdsourcing in detail.

Google's DLP (Data Loss Prevention) API [19] is available to check for data loss, but it only picks out RegEx (Regular Expression) based terms such as email ids, phone numbers, passport numbers, etc. Few other cloud-based DLP solutions exist such Amazon Macie [12] and End Point Protector [34], however, these services lack scanning for entities specific to software development (API Keys, OAuth Tokens, App Secrets, etc.), and even miss out on organization-specific context let alone miss out on the Crowdsourcing scenario. These services are available on a payment basis.

We also looked at work done on sensitivity and tried to see how they could be fit in the scenario of Crowdsourcing. These mainly focused on detecting sensitive components [18] in textual data and preventing data loss [13, 22]. Goyal et al. [21] have looked into enterprise-level data breaches and confidentiality loss when it comes to sharing data and built a classifier to prevent such breaches. The aspect of Differential Privacy [16] was explored and analyzed to fit into the scenario of Software Development related Crowdsourcing. Previous work done on data redaction and data sanitization [17] was also studied to see how it can be incorporated in our work.

Chapter 3

Contributions

In addition to the questions mentioned in section 1.2, we, in this thesis, address:

- We point out the aspect of confidentiality loss in the crowdsourcing process; this is something on which very little prior research has been done.
- We pinpoint the various stages in the process from where information could be leaked based on in depth analysis of the entire crowdsourcing process.
- We categorize different attributes of a crowdsourcing task into whether it could cause any confidentiality loss or not.
- We also take a holistic approach to see what all kind of confidential information can flow out from a posted task.
- Based on these findings, we propose an organization specific NLP and rule based algorithm which the task can be parsed through before posting online.

In our work, we look at the various possible sensitive components in a task, how they could be leaked, and what repercussions it could have. To understand people's take on crowdsourcing, sensitivity, and sharing data, we conducted a survey to validate some of the insights that we had. The motivation to continue researching into this domain was further corroborated when on examining some tasks posted previously, it was observed that the task poster had provided a username and password in the task description using which we were able to access some information about the company like the address, phone number, email id, list of customers, details of the customers (Name, Email ID, Phone Number, Date, Time, Cost of Services Availed, etc). This was a task which was posted back in 2013, and still, we were able to see the data.

Chapter 4

Understanding Crowdsourcing

4.1 The Stakeholders in Crowdsourcing

The Crowdsourcing Process has the following stakeholders:

1. **Task poster i.e.** $\boldsymbol{tp}$: This is the person who is posting the task.
2. **Worker i.e.** $\boldsymbol{w}$: This is the worker who works on the task.
3. **Company/Organization i.e.** $\boldsymbol{c}$: This is company or organization that $\boldsymbol{tp}$ is associated with and the task posted is of relevance to this company/organization.

4.2 Some other Terms and Annotations

Here's a list of terms and symbols we'll use across the thesis:

1. Task $\boldsymbol{t}$.
2. Task Title $\boldsymbol{tt}$.
3. Task Description $\boldsymbol{td}$.
4. Task Resources $\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, \ldots \mathbf{r_n}$

4.3 The Crowdsourcing Cycle

Here's what the entire crowdsourcing cycle looks like in general (the stages from which information leak could happen have been highlighted in bold letters):

1. The Task Poster ***tp*** has a task ***t*** at hand to post.
2. **The *tp* posts the task *t* on a suitable platform.**
3. Task ***t*** is discovered by workers on the platform.
4. Workers $\mathbf{w_1, w_2, w_3, \ldots w_n}$ apply for task ***t***.
5. Task Poster ***tp*** chooses a worker ***w*** from the pool of Workers $\mathbf{w_1, w_2, w_3, \ldots w_n}$.
6. **Task *t* is handed over to the worker *w* by task poster *tp*. Resources of task $\mathbf{r_1, r_2, r_3, \ldots r_n}$ are handed over to worker *w*.**
7. **If it's a large task and needs to be done in chunks with some review cycles in between, then the *tp* and *w* engage in some sort of correspondence.**
8. Worker ***w*** submits the task to ***tp***.
9. ***tp*** evaluates the task, verifies it, and processes the payment.

We looked at various stages of the crowdsourcing process and narrowed down to three stages (**steps 2, 6 and 7**, highlighted above in bold) from which any confidential information could be leaked. First is the posting of the task itself, second is the handing over of more task related resources to the worker, and the third is the correspondence stage in which messages, calls or resources are exchanged between the task poster and the worker. So, the need of the hour is to pay particular attention to these stages in the crowdsourcing cycle.

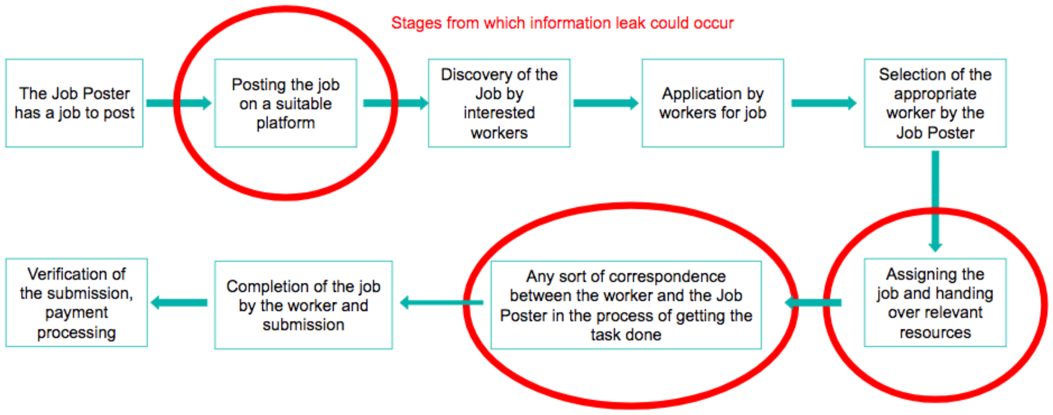

Stages from which information leak could occur
The Job Poster has a job to post
Posting the job on a suitable platform
Discovery of the Job by interested workers
Application by workers for job
Selection of the appropriate worker by the Job Poster
Assigning the job and handing over relevant resources
Any sort of correspondence between the worker and the Job Poster in the process of getting the task done
Completion of the job by the worker and submission
Verification of the submission, payment processing

Figure 4.1: Stages in which information leak is possible in the crowdsourcing cycle.

This conclusion about the stages prone to information leak is supported by the analysis that we performed on (i) a data-set[1] of 65,499 Tasks that were posted previously on a leading crowdsourcing platform, (ii) a data-set of 320,000 comments which were left for the workers on a leading crowdsourcing platform, and (iii) a data-set of conversations between the task poster and the worker.

4.4 Information Leak in the Crowdsourcing Scenario

Information leak in the crowdsourcing setting can happen in three ways:

1. Inadvertent Information Leak: This is the case when the data is accidentally/unknowingly leaked by the Task Poster ***tp***. He / She is unaware of the presence of any confidential data. A suitable example of this would be while sharing a document with the worker ***w***, a link to an old project's GitHub repository was somewhere hidden in the document, and the ***tp*** was unaware that it contained some confidential data which worker ***w*** should not have had access to ideally.

2. Information Leak during Intended Transfer of Data: This type of leak occurs when you know you're transferring data, and some traces of confidentiality are present in the data. For instance, you know that you're sending a *.py (python)* code file to a worker ***w*** which has an API Key which was present in the code file. So, here you were intentionally sharing the code file, you knew that it had an API key. Sharing of this key could have repercussions.

3. Malicious Information Leak: This type of leak is seen when the worker or an employee of a company, while giving away the task, intentionally gives away confidential data with malicious intent. This might be a way of the task poster ***tp*** colluding with the worker ***w*** for some spiteful activity. This type of leak would ideally be less than that of the other two types.

[1]All these there datasets are internal to the company and more details cannot be disclosed due to confidentiality.

Chapter 5

Survey to understand Confidentiality

A survey was conducted among the professionals of a business services company to get a deeper insight into how software development experts, some of them with prior experience with freelancing, perceived confidentiality and freelancing. We had some assumptions about these, which were validated by the survey. The survey was filled by 93 professionals.

5.1 Survey Setup

A survey[1] was created and floated. The survey consisted of questions of objective as well as subjective nature. The set of questions was carefully selected to verify some of our assumptions and get the view of experts.

Following are the assumptions that we wanted to validate through the Survey:

- Details like Phone Number, Address, National Identification Number, API Keys are sensitive in most cases.
- Sharing a database publicly may compromise with confidentiality.
- Sensitive information may be picked up from code: API Keys, variable names, usernames, passwords.
- People can also pick up sensitive information included in the comments of code.
- Wireframes and UX examples may give out information about the company posting the job: logo.
- Market of the end result of the task can be inferred given a task.
- Domain of the task can be identified from the task, given the type details it is asking for.

[1]Link to the survey: https://goo.gl/forms/jI8OkBIVJzvbx6kW2

5.2 Detailed Analysis of the Survey Questions and Responses

Some statistics from the survey:

- 93 people who took the survey ranged from work experience of 0 to 40 years. The average being 12.33 years.
- 35.6% of the people had some prior experience on working with Freelancers.

The survey takers were given the following scenario and were asked to rate the sensitivity levels of the following attributes:

> *Suppose you are a project manager in a multinational organization and you need to get some job done by an external freelancer. These jobs require sharing some details like sample code, documents, UX wire-frames, etc. and some information might be sensitive in these resources. You need to answer the following questions considering the above scenario.*

Likert scale based Questions:

The survey had some likert scale based questions to assess the level of sensitivity people associated with various things. The survey takers were asked to rate on a scale of 1 - 5.

1 being ***Not Sensitive at all*** and 5 being ***Very Sensitive***. By sensitive, we mean entities which may be confidential and sharing of which might be considered as a sensitive activity. By sensitive, we mean that if exposed to people with a trust level lower than a certain level, they could cause harm or could lead to identification of a person. PII (Personally Identifiable Information) is usually considered to be Confidential and Sensitive.

Question: How sensitive do you think the following information is when shared with someone. Rate on a scale of 1 - 5.

1. ***Phone Numbers***: ***33.3%*** of the people think that phone numbers are ***very sensitive*** to share with people. In all, ***80.6%*** believe that sharing phone numbers is indeed sensitive. Refer to Figure 5.1.

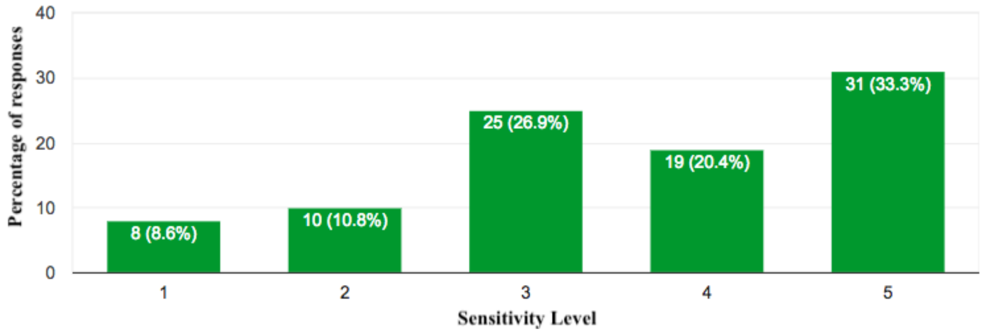


Figure 5.1: Sensitivity levels of sharing phone numbers. N = 93.

2. ***Address***: ***10.8%*** people think that address is not something sensitive to share with people and ***39.8%*** people believe that address is something very sensitive to share with people. In all, ***89.3%*** people agree that sharing of address is sensitive. Refer to Figure 5.2.

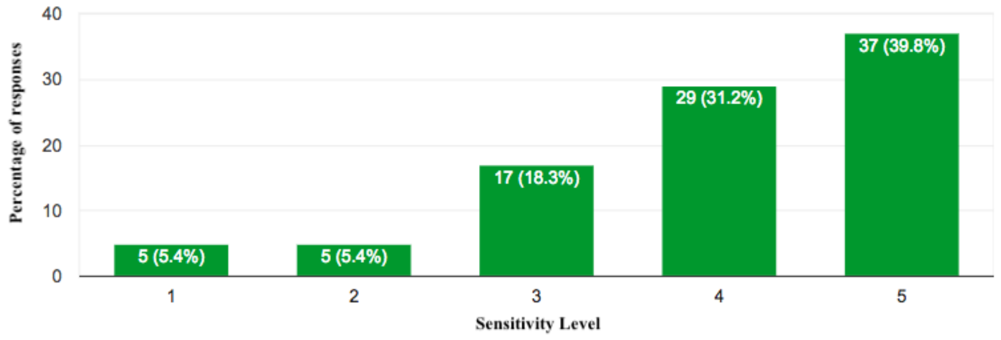


Figure 5.2: Sensitivity levels of sharing address. N = 93.

3. ***National Unique Identification Numbers like Aadhar or SSN***: ***100%*** people agree that sharing of such unique identification numbers is highly sensitive. ***89.2%*** people say it's Very Sensitive. This verifies our claim that such PII information is indeed very confidential and show not flow out while sharing tasks or talking to people of varied trust levels. Refer to Figure 5.3.

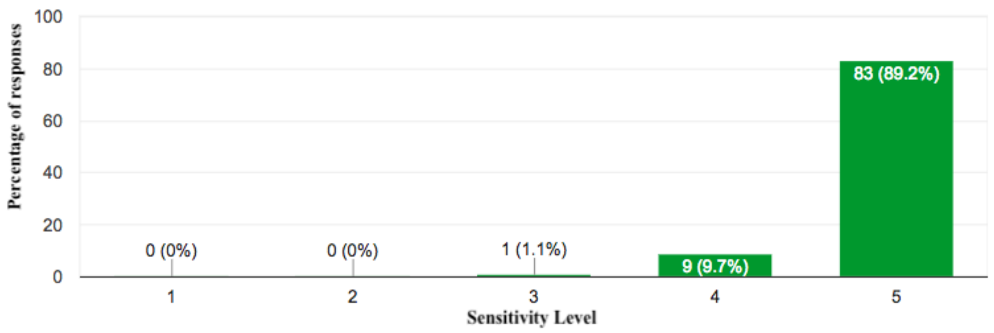


Figure 5.3: Sensitivity levels of sharing national unique identification numbers like Aadhar or SSN(Social Security Number). N = 93.

4. ***API Keys***: ***92.5%*** (refer to Figure 5.4.) people think that sharing of API keys is risky. ***2.2%*** people think that sharing API Keys is okay, here's where the problem lies. API keys can be reverse engineered to get into the user's account or even the misuse of API keys is a great concern. Many a times it has been observed that people upload their codes on the cloud/GitHub or share API Keys while sharing code files. Many of the past high profile internet security breaches took place because of some API security Vulnerability. Companies like Buffer, Pinterest, Facebook, and Snapchat dealt with API Security Vulnerabilities recently [46]. It was observed that hackers have deployed bots that consistently look for AWS (Amazon Web Service) API keys shared on GitHub. They make use of such exposed API keys to mine Bitcoins. With the reliability on cloud services, API keys face an even bigger threat. Both inexperienced and seasoned programmers have been observed to accidentally expose API Keys, OAuth Tokens, and App Secrets during the version control of codes. In the light of Crowdsourcing, with software development related tasks, version controlling plays a critical role. It's quite possible that API Keys, Cloud Credentials, App Secrets get shared. So, it's imperative that exchange of code with such possible secrets is done carefully. It is advised to keep any such API Keys or App Secrets in a separate configuration file and read them in the code from that file. Sharing API Keys accidentally has also seen to have had some monetary losses. Andrew Hoffman, in his blog "My $2375 Amazon EC2 Mistake" explains how he lost $2375 by accidentally pushing code on GitHub which contained his AWS API Keys. GitHub's policy on pushing any such secret to GitHub states that, "Once a secret is pushed, consider it compromised" and they advice to change passwords immediately or generate new API Keys if they have been accidentally pushed.

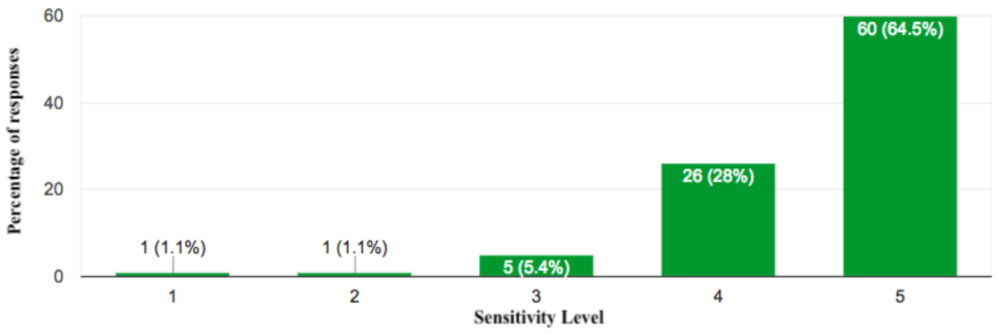
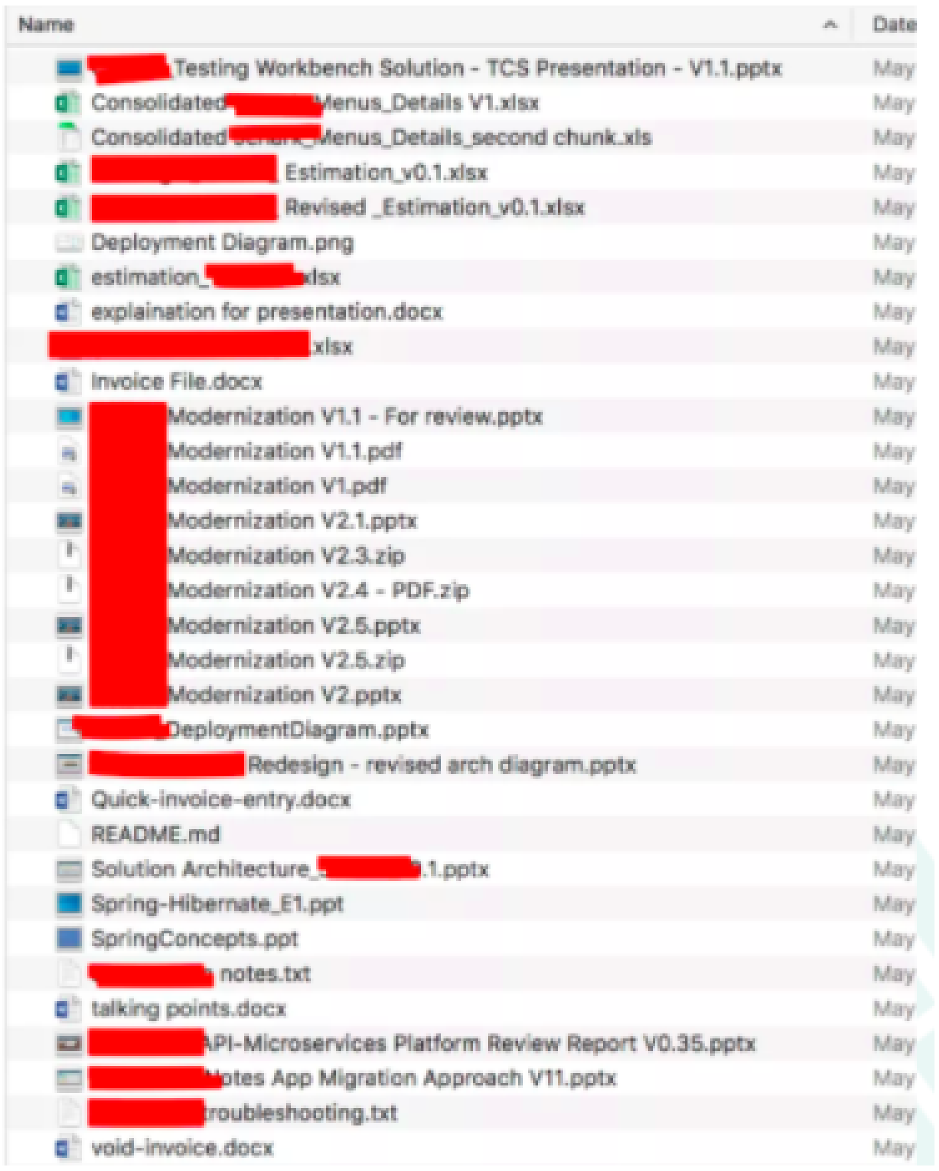


Figure 5.4: Sensitivity levels of sharing API keys. N = 93.

5. ***Source Code / Blueprints of a Project***: ***99%*** of the responders think that sharing of source code might leak some confidential information (refer to Figure 5.6). With the massive use of source control when it comes to software development, it is possible that some sensitive data might be leaked when code is pushed onto GitHub or other such version control services. It has been observed that code itself might contain a lot sensitive content and confidentiality can be compromised through variable names, comments, presence of usernames and passwords, presence of *API Keys, OAuth Tokens and App Secrets*. Companies and workers need to be very careful while sharing code or other project related details with each other. A data breach was seen recently in June, 2017, when a TCS Employee inadvertently leaked sensitive data of a banking project of at least 10 companies by uploading the code on GitHub [37]. It included the migration plans, estimates and presentations, of a number of companies. Given the crowdsourcing setting, with the frequent use of version control, it is possible that while pushing such data online, some sensitive data also gets pushed like seen in the case of the TCS data leak. This sort of a data breach calls for active training of people involved in freelancing so that they are careful about what they share with the workers.



Name	Date
[redacted]Testing Workbench Solution - TCS Presentation - V1.1.pptx	May
Consolidated[redacted]enus_Details V1.xlsx	May
Consolidated[redacted]enus_Details_second chunk.xls	May
[redacted] Estimation_v0.1.xlsx	May
[redacted] Revised _Estimation_v0.1.xlsx	May
Deployment Diagram.png	May
estimation_[redacted]xlsx	May
explaination for presentation.docx	May
[redacted]xlsx	May
Invoice File.docx	May
[redacted]Modernization V1.1 - For review.pptx	May
[redacted]Modernization V1.1.pdf	May
[redacted]Modernization V1.pdf	May
[redacted]Modernization V2.1.pptx	May
[redacted]Modernization V2.3.zip	May
[redacted]Modernization V2.4 - PDF.zip	May
[redacted]Modernization V2.5.pptx	May
[redacted]Modernization V2.5.zip	May
[redacted]Modernization V2.pptx	May
[redacted]DeploymentDiagram.pptx	May
[redacted]Redesign - revised arch diagram.pptx	May
Quick-invoice-entry.docx	May
README.md	May
Solution Architecture_[redacted].1.pptx	May
Spring-Hibernate_E1.ppt	May
SpringConcepts.ppt	May
[redacted] notes.txt	May
talking points.docx	May
[redacted]PI-Microservices Platform Review Report V0.35.pptx	May
[redacted]otes App Migration Approach V11.pptx	May
[redacted]roubleshooting.txt	May
void-invoice.docx	May

Figure 5.5: Example of leaked confidential data online.

Figure 5.5 shows the data which was leaked by the TCS Employee accidentally. It exposed data related to projects, migration plans, estimates, and presentations of some American, Canadian and, Japanese financial institutions.

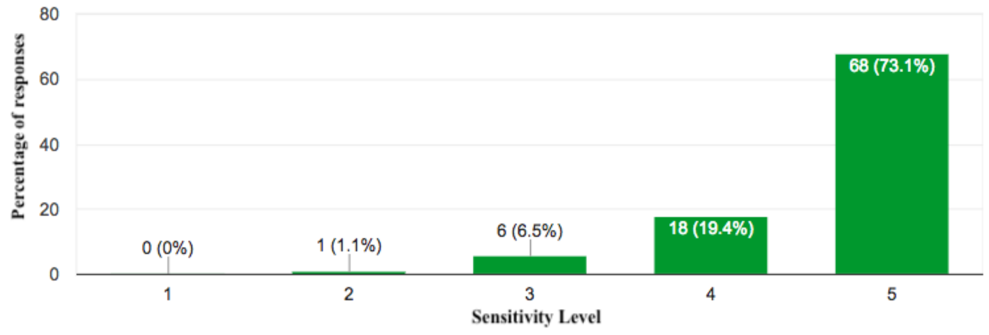


Figure 5.6: Sensitivity levels of sharing source code/blueprints of a project. N = 93.

6. ***Details of Pending Patents / Intellectual Property***: ***Almost all*** respondents believe that sharing of any such details is a point of loss of Confidentiality [24] (see Figure 5.7). While companies want to test out new products on crowdsourcing/crowdfunding websites, while explaining the product on these sites, one is at a risk of being copied by others. Even if not much information is shared in the initial stages, at later stages of getting views from the marketplace on the product/prototype/idea, it is advisable to get NDAs [Non-Disclosure Agreements] signed to protect Intellectual Property.

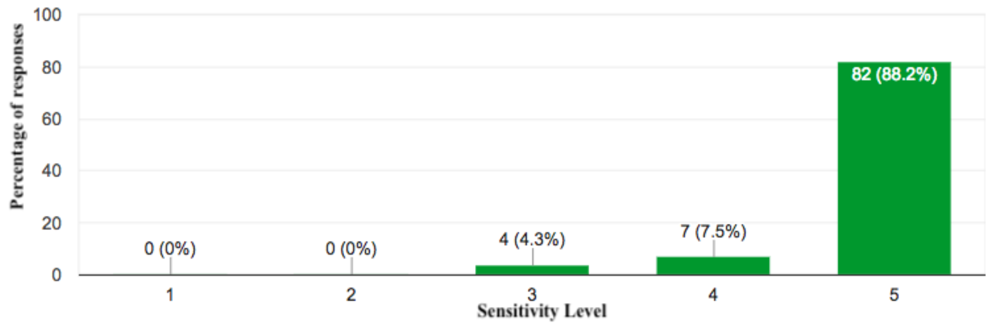


Figure 5.7: Sensitivity levels of sharing details of pending patents/intellectual property. N = 93.

7. ***Details of Customers of a Company***: The customer base of any company is considered to be Confidential and sharing of it is seen as a Sensitive activity by ∼ ***99%*** of the survey takers. ***78.5%*** of the people say it's *Very Sensitive* to share the details of the customers of a company (see Figure 5.8). If these details go in the wrong hands, it could even initiate legal troubles. The task posters need to be doubly sure that they are not passing on any such information. This also poses a problem from the other way round. Imagine a worker who is working for two big firms A and B, where A and B are rivals. A might not like the worker exposing to B, that it's working for A. In such cases, companies tend to avoid exposing their entities.

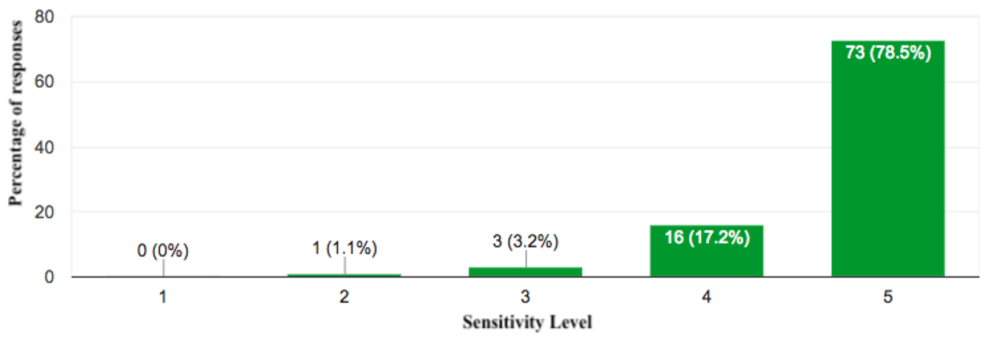


Figure 5.8: Sensitivity levels of sharing details of customers of a company. N = 93.

8. ***Publicly sharing a Database***: ***6.4%*** of the respondents believe that sharing a database publicly is not a sensitive activity (see Figure 5.9). Here's where the problem lies, while not all databases might be sensitive, but sharing them publicly might lead to confidentiality loss. In the crowdsourcing setting, databases should be shared with workers only after scrutiny. The databases must be run against privacy preserving practices like *K-Anonymity* [42], *L-Diversity* [29], *T-Closeness* [27] and must be shared only after proper Sanitization and Anonymization.

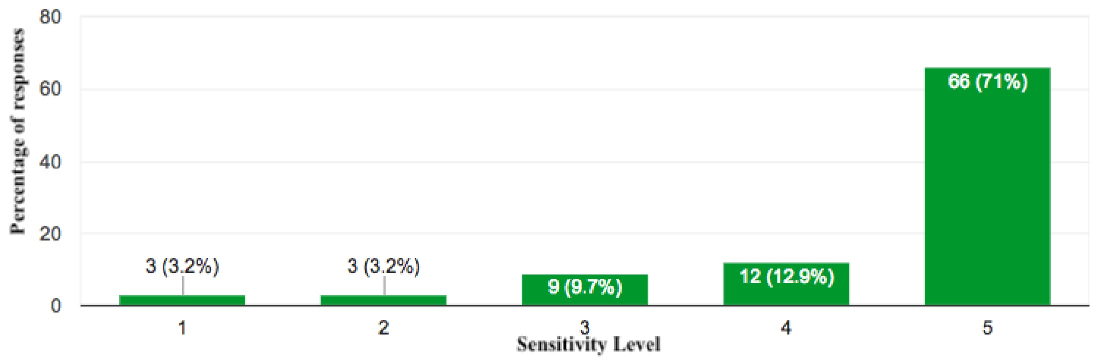


Figure 5.9: Sensitivity levels of sharing a database publicly.

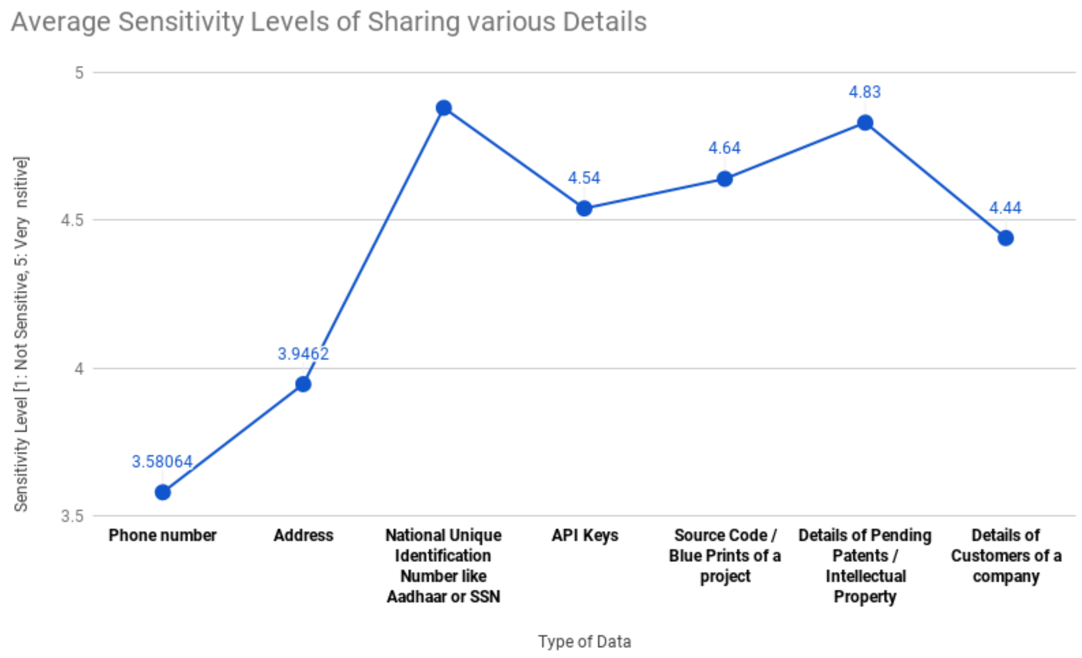


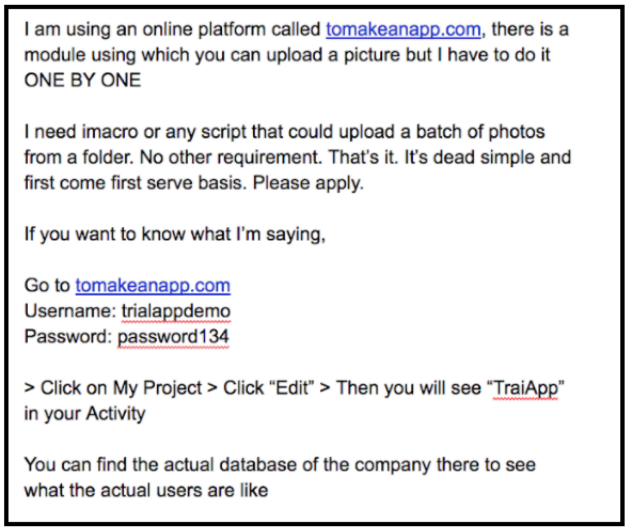
Figure 5.10: Relative sensitivity levels of confidential data.

It was observed from the responses that things like: national unique identification numbers such as Adhaar, SSN, API keys, source code of a project, details of patents of a company, and details of customers of a company are more sensitive as compared to details like address and phone numbers. This assisted us in backing up our classification of entities as low-risk, medium-risk and high-risk. Refer to Figure 5.10.

Next, some questions with sample tasks containing sensitive content were presented to the respondents, and they were supposed to identify the sensitive parts in it.

The first one was to validate our assumption that ***given a task description, people can pick up sensitive information from it***. For this, we showed a sample task description to them and asked them to pick up components that they find sensitive.

Question: *This is a sample task posted by a Task Poster, according to you, what all sensitive data does it contain?*



I am using an online platform called tomakeanapp.com, there is a module using which you can upload a picture but I have to do it ONE BY ONE

I need imacro or any script that could upload a batch of photos from a folder. No other requirement. That's it. It's dead simple and first come first serve basis. Please apply.

If you want to know what I'm saying,

Go to tomakeanapp.com
Username: trialappdemo
Password: password134

> Click on My Project > Click "Edit" > Then you will see "TraiApp" in your Activity

You can find the actual database of the company there to see what the actual users are like

Figure 5.11: Sample task description with sensitive content.

On presenting Figure 5.11 to the respondents, they were able to identify most of the sensitive content. ***98%*** felt that giving ***username and password*** to log-in to a different website was risky. Figure 5.12 shows a word cloud of the responses received from this question. Some of them even mentioned that this task description was giving away ***lots of confidential data related to the organization.*** Some of the people spotted giving away password in clear text as a problem. They even felt that giving or talking about the actual database of the company with freelancers is an issue. This sharing of data might be okay for smaller organizations, but then when it comes to larger organizations, giving such type of data might be a bad idea. Data is the new gold, and simply giving it away may have several backfiring results.

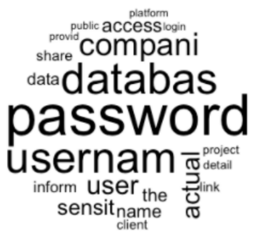
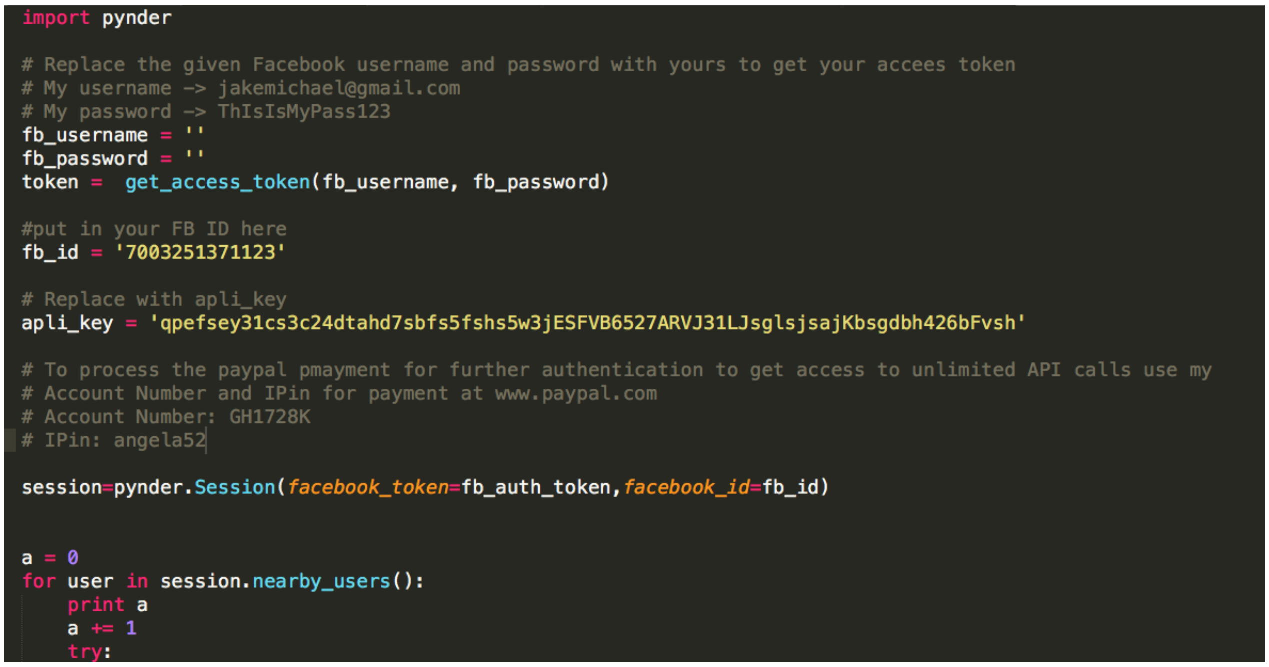


Figure 5.12: Word Cloud of responses from question in Figure 5.11. Keywords ***Password, Username and Database*** stand out in it.

Question: *Following is a sample code snippet, point out the parts in the code that you think are sensitive*



```
import pynder

# Replace the given Facebook username and password with yours to get your accees token
# My username -> jakemichael@gmail.com
# My password -> ThIsIsMyPass123
fb_username = ''
fb_password = ''
token =  get_access_token(fb_username, fb_password)

#put in your FB ID here
fb_id = '7003251371123'

# Replace with apli_key
apli_key = 'qpefsey31cs3c24dtahd7sbfs5fshs5w3jESFVB6527ARVJ31LJsglsjsajKbsgdbh426bFvsh'

# To process the paypal pmayment for further authentication to get access to unlimited API calls use my
# Account Number and IPin for payment at www.paypal.com
# Account Number: GH1728K
# IPin: angela52

session=pynder.Session(facebook_token=fb_auth_token,facebook_id=fb_id)


a = 0
for user in session.nearby_users():
    print a
    a += 1
    try:
```

Figure 5.13: Sample code snippet with sensitive content.

This snippet was made on similar lines to the code shared in some of the task descriptions.

People were able to point out things like ***fb_ id, apli_ key*** as sensitive. They were even able to identify sensitive content hidden in the comments of the code such as ***username, password, Account Number, Paypal IPin, etc.*** An interesting observation was, while someone claimed that they did not understand the code language of the example, still they were able to pick out the things which are sensitive. Exposing such App Secrets while sharing code has been observed to come with a risk of losing sensitive data. We were able to validate that what we perceive as sensitive when it comes to the software development setting in Crowdsourcing, is actually seen as a threat. Through this approach, we were even able to see the point of view of an adversary, where when all this information is presented to them, whether they are able to pick them up or not. As expected, people picked them up and our claims were validated. These examples might contain extreme examples of exposing sensitive data, while it comes to real world, we might see only a subset of these. Again, sharing of such data may have severe repercussions. Adversaries may be able to intrude into the system of the company and carry out malicious activities. Thus, while sharing any piece of code with freelancers, one has to be very careful.

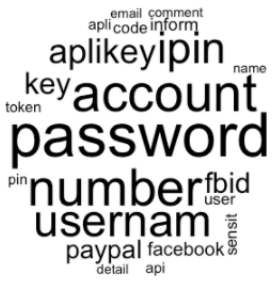
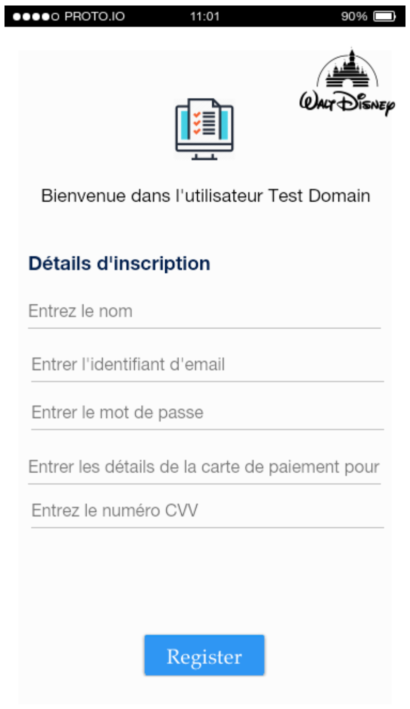


Figure 5.14: Word Cloud of responses from question in Figure 5.13. Keywords ***password, account, username, fbid, ipin, paypal, aplikey, etc*** stand out in it.

Question: *A wireframe of an App has been attached, what all can you infer from it?*


●●●●○ PROTO.IO 11:01 90%

Bienvenue dans l'utilisateur Test Domain

Détails d'inscription

Entrez le nom

Entrer l'identifiant d'email

Entrer le mot de passe

Entrer les détails de la carte de paiement pour

Entrez le numéro CVV

Register

Figure 5.15: Sample wireframe with sensitive content.

Many a times, freelancing involves getting some UI/UX work done, getting prototypes made or getting designs evaluated by people. While giving away such images/designs, it's possible that some information may be given away. Like, in Figure 5.15, the wireframe exposes the company this task is for (from the logo), the country the task is meant for, and the type of information it's seeking clearly gives away the type of the application. In this, we intended for the survey-takers to verify that:

- Having a logo of the company can give away the company the task is for
- Language, i.e. French in this example, can give away the intended audience of the application
- The type of things on the screen also give away more information

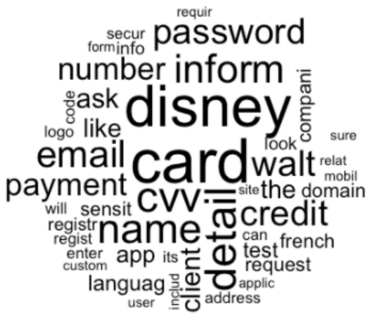


Figure 5.16: Word Cloud of responses from question in Figure 5.15. Keywords *disney, card, CVV, credit, password, email, payment, French* stand out in it. We wanted to see if given a wireframe, would people be able to infer the company or the type of application from the components on the screen.

The respondents were able to identify that this task is related to Disney. They claim that the logo gave away this detail. This is what we wanted to confirm from this question; the logo is sensitive when it comes to giving away tasks to Freelancers. While giving away any such UI/UX related tasks, task posters need to be careful to remove such details. Task related resources need to be carefully analyzed and sanitized before giving them away.

Question: *What can you infer about the type of industry that this app is for?*

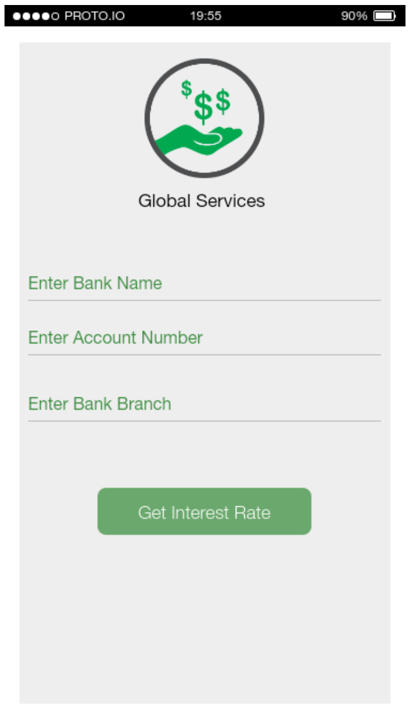


Figure 5.17: Sample domain based question. Given a wireframe and UI components, we wanted to test if people can infer the domain and type of task.

The purpose of this question was to confirm that given some wireframe, the domain of the task can be inferred even if the name of the company has not been explicitly stated. While giving away such wireframes to freelancers, it should be carefully analyzed if some information is being given inadvertently to the worker or not. The survey-takers were able to identify the domain the task was related to given the wireframe.

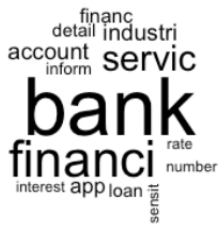


Figure 5.18: Word Cloud of responses from question in Figure 5.17. Keywords *bank, finance, account* stand out in it.

The last question of the survey was open-ended.

Question: *What other things could be sensitive while giving out a task on any freelancing website?*

Here are the responses that we got for this question:

1. Master Database of a company
2. Personal Information:
   (a) Name
   (b) Email ID
   (c) SSN / Unique Identification Number
   (d) Address
   (e) Date of Birth
   (f) Employee ID
   (g) Health related information
3. Credentials of any sort: Usernames and Passwords
4. Bank Account Number
5. Context for the given task
6. Company related details:
   (a) Details of the clients of a company
   (b) Details of a company
   (c) Details of a company's future projects

(d) Any sort of financial data

(e) Proprietary Data

The survey helped us in curating a list of items which could be sensitive in the setting of Crowdsourcing in an organization. Knowing what to look for exactly is a very crucial thing. The survey and talking to some experts in the organization who are involved in crowdsourcing gave us some very good insights into what to look for as sensitive when giving away a task. This helped us in getting answers to our third research question, *What type of data can be leaked in crowdsourcing?*

Chapter 6

Unboxing a Typical Crowdsourcing Task

Chapter 5 dealt with confirming what type of data is sensitive. Now, this chapter looks into a typical crowdsourcing task, unravels it and points out the instance which could contain sensitive information. This chapter helps us in answering the research question, ***Which attributes in the Crowdsourcing process lead to information leak?***

We look at a dataset of previously posted 65,499 Tasks on a leading Crowdsourcing Platform[1] and analyzed them to look for patterns and see if they contained sensitive information. Due to confidentiality of the database, exact details cannot be shared. These tasks were related to software development.

Some insights into the dataset:

- 4 unique task categories were identified in the given tasks:
  - Data Science Analytics
  - IT Networking
  - Web, Mobile Software Development
  - Writing
- Each of the tasks came around 20 fields such as:
  - Task ID
  - Task Title
  - Task Description
  - Task Category
  - Task Sub-Category

[1]This is a private dataset and more details cannot be disclosed due to confidentiality.

  - Date of creation of Task
  - Task Type
  - Task Workload
  - Task Duration
  - Task Budget
  - Preferred Location of the worker for the task. To match time-zones
  - Preferred range of feedback score of workers
  - Level of English Skills required for the task
  - Range of payment for the task
  - Range of working hours per day for the task
  - Requirement of Cover Letter from worker
  - Requirement of Portfolio of the worker
  - Candidates Registered for the task
  - Skills required for the task

All the 65,499 tasks were analyzed thoroughly, and it was observed that of these fields, the Task Title and the Task Description were the crucial points which could contain sensitive information. Some insights from the analysis of the data-set:

- Not many task descriptions had associations with companies as such
- Few descriptions had links in them, these links could potentially lead to giving away more information than what is intended. Eg. Link to an app on the play store, link to a data-set, link to documents, link to websites, etc
- In some instances of tasks, it was observed that credentials were provided to get into a third party website to access information. Of the 65,499 tasks, 1,331 tasks had some password mentioned in the task description. Using one of the passwords in a task, we could get into a company's customer database. This was a task which was posted back in 2013, and still, the data was accessible. This also provided a tremendous validation for the need to go ahead with the research that we are conducting. Maybe this data was not confidential, but in future, there might be a case where the data might have been confidential and had been inadvertently passed on to a worker
- Task Duration of the tasks: From Figure 6.1, it can be seen that 60.2% of tasks needed 30+ hours of input per week and 16.7% os the tasks required 10-30 hours of work per week. This indicates the level of complexity of the tasks. Such tasks need some sort of interaction between the task poster and the worker. Information leak can occur during this interaction

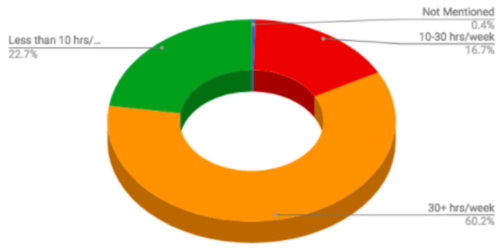


Figure 6.1: Distribution of task duration.

- Occurrence of certain terms in the data-set: A basic search into the data-set with different keywords was done to gain some insight into how different sensitive things might occur in tasks. This was a very preliminary investigation and the presence of the keyword does not necessarily signify it being sensitive for sure. For instance, as lot of tasks had the mention of password while not actually giving away the password.

Term	Occurence in Tasks	Percent
$	65332	99.74
.com	37107	56
http	15072	23.01
api	2328	3.55
password/password:	1445	2.20
username/username:	758	1.6
key	742	1.13
card	508	0.77
@<any text>.com	251	0.38
account number	17	0.025

Table 6.1: Occurrence of potentially sensitive terms in the data-set.

Analysis shows how many of the tasks had presence of sensitive entities such as password, API keys, account numbers, app secrets, account details, credit card numbers, presence of external links, etc. Presence of external links is also a critical point of information leak. Who knows what this link might point to? This analysis helped us in finding answers to questions: ***Which attributes in the Crowdsourcing process lead to information leak?*** and even ***What type of data can be leaked in crowdsourcing?*** Now, we have a fair idea of what to look for, and where, while giving a task away.

Chapter 7

Understanding the Conversations between Workers and Task Posters

A data-set of conversations between Workers and Task Posters[1] was analyzed to get an idea of how information leak could occur in the correspondence stage of the crowdsourcing cycle. However, due to the difficulty in getting this data due to its confidential nature, the size of the data-set was very small as compared to the data-set analyzed in Chapter 6.

It was observed, that in getting a typical Crowdsourcing task, 214 lines were exchanged between the task poster and the worker from the stage where the task was allocated to the stage where the final task was submitted and payment made. This, however, can vary as the type of task will vary. On an average, 30 files were exchanged between the task poster and the worker. Minimum being 0 and maximum being 92 files. These files could be a source of leak of confidential data. Such files should be carefully analyzed before sharing. 42% of the tasks involved Skype or any other type of phone calls. During such calls also it's possible that the task poster might accidentally spill out some confidential data. 14% of the tasks involved sharing files via other mediums like Google Drive. So, more than building another layer of scrutiny for tasks and related resources before it's posted, the need of the hour is to train the task posters about these critical places from which confidentiality loss could occur.

[1]This is a private dataset and more details cannot be disclosed due to confidentiality.

Chapter 8

Protecting Confidentiality Loss in Crowdsourcing

Now that we have some of the questions answered such as, in which stages does Confidentiality loss take place, which Attributes give out more sensitive information than others, what type of data can be lost in the Crowdsourcing process, we move on to use this data collected to propose an NLP based unsupervised algorithm which will scan the task.

We tackle the problem of confidentiality loss in crowdsourcing at the organization level. We take that there's an organization which uses Crowdsourcing to get some of its tasks done. In this setting, there are some full-time employees who act as the task posters and they post the tasks on an appropriate marketplace and hire workers. We propose an additional layer of scrutiny of the task before it is actually posted online.

The question that we want to answer here is: ***Can NLP/Rule Based techniques aid organizations in identifying information leak? How can such frameworks be incorporated at the organization level?***

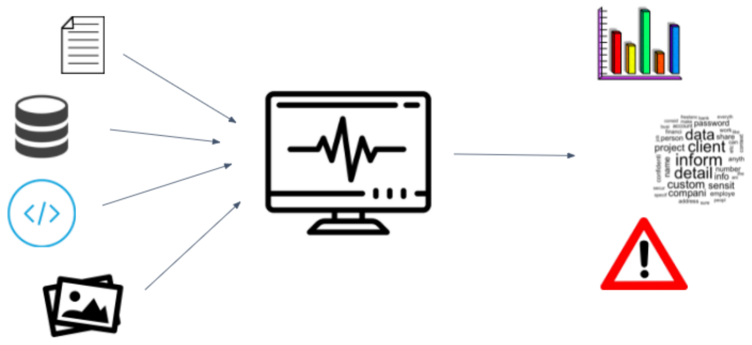


Figure 8.1: Framework to analyze tasks.

We propose a framework which takes as input the following components of a task ***t***:

- Task Title ***tt***
- Task Description ***td***
- Task Resources ***r1, r2, r3, ...***

The framework analyzes ***t, tt, td, r1, r2, etc*** and classifies the task's risk level as low, medium or high.
NLP (Natural Language Processing) and Rule Based approach is followed here for analysis. Each task title, task description and all the resources are first searched for RegEx based components such as phone numbers, email addresses, etc. Then they are tested against a list of carefully curated keywords. The list of keywords is formed based on: the name of the company, names of employees of the organization, list of ongoing projects of the organization, list of products of the organization, list of branches of the organization, and then the list of terms such as usernames, passwords which were picked up from the survey, analysis of tasks and literature.

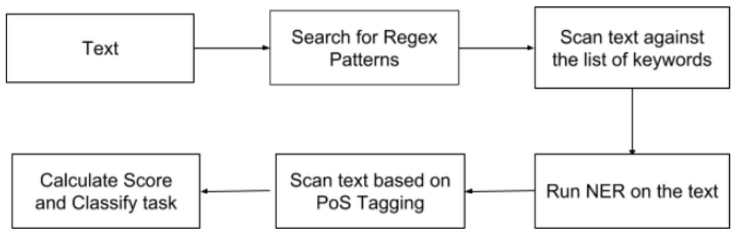

Text
Search for Regex Patterns
Scan text against the list of keywords
Run NER on the text
Scan text based on PoS Tagging
Calculate Score and Classify task

Figure 8.2: Pipeline to analyze task related textual content. Given the textual data, folowing procedures are applied on it to extract sensitive content.

Natural Language Processing is a widely used technique which is used to process and understand Natural Language [32]. We use NLP techniques such as Part-of-speech tagging [39,43], stemming [33], NER (Named-entity-recogniton) [38] to better understand the text component of the tasks. Part-of-Speech tagging was used to identify Proper Nouns, and Named-entity-recognition was used to pick out names of locations, organizations, and persons. Using stemming, we were able to match the text component of tasks in a more efficient way and cater to different forms of a word. Apart from this, many rules [40] were created like in Algorithm 2, to filter out sensitive content such as API keys, passwords, usernames, email addresses, etc.

Before the text was passed on for analysis, as per Algorithm 1, it was tokenized and stop-words were removed. Another attempt was made to curate a list of keywords from company related documents and websites such the Wikipedia Page of the Company, the Website of the Company, and the Privacy Policy of the Company. Techniques such as TF-IDF [35, 44], RAKE [36] and TextRank [31] were used to extract keywords from the above mentioned documents after tokenizing them, removing stop words and stemming. RAKE was seen to be performing the best, of the techniques tried. The keywords picked up this way were also added to the list of keywords against which the text is analyzed.

Next, the text can be analyzed for NER[1] (Named-entity recognition). In case of presence of any names entities, such as Person, Organization or Location. The text is also scanned for URLs. This can be done using regular expressions.

Algorithm 1 Analysis of a task

```
Result: Array of keywords found sensitive in $t$
Given $t$ and its components $tt$, $td$, $\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, \dots \mathbf{r_n}$;
 res = [ $tt$, $td$, $\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, \dots \mathbf{r_n}$];
for each item in res do
    tokenize(res)
    removeStopWords(res)
    detectRegEx(res)
    detectPasswordsAndAPIKeys(res)
    detectSensitiveKeywords(res)
    detectNER(res)
    add sensitive terms detected to result
end
```

[1]https://nlp.stanford.edu/software/CRF-NER.html

Algorithm 2 Detecting Password and API Key like instances

```
Result: Return a list of terms identified
list = []
for each term do
    if term is not an English Word then
        list.append(term)
    end
    if term is a digit and length(term) > 3 then
        list.append(term)
    end
    if term is alphanumeric then
        list.append(term)
    end
    if term contains special characters then
        list.append(term)
    end
    if term is in CamelCase and has >1 Capital Letters then
        list.append(term)
    end
    Return list
end
```

The Algorithm given above was run on sample tasks. The output is a set of terms, which were found to be sensitive. While saving the terms, it was also recorded from which rule was it flagged as sensitive. These rules were also classified as low risk, medium risk and high risk. Based on the occurrence of rules in categories, the task is classified as low risk, medium risk or high risk. For instance, the High Risk keywords would contain things like credentials, list of sensitive projects in a company, API Keys, presence of URLs, etc. The model proposed above can be implemented at the level of an organization as per their needs and rules. Such a layer for the scrutiny of tasks before it is posted can be implemented at the level of organizations. Based on the keywords detected, a nudge can be given to the task poster.

Accuracy of the Algorithm:
Due to the lack of presence of data with the level of sensitivity that we want/foresee, it was difficult to come up with any supervised classification techniques. Even if we got the data annotated, the existing tasks were not sensitive enough to get decent results. The lack of ground truth was a challenge to test the accuracy of the Algorithm. To overcome this, we decided to come up with our own synthetic data-set for testing. For this, we took the 65k data-set of previously posted tasks, induced sensitivity into 40% of the tasks. To induce sensitivity, we created a pool of sensitive terms and added 1 to 50 terms randomly in each of the 40% tasks which selected to induce sensitivity. While injecting these terms, we recorded them and treated them as the ground truth. 50 tasks from the ones in which sensitivity was induced were sampled randomly

and shown to two experts in the domain to verify that the data did not vary too much from what is seen in real instances of such data.

Apart from the above mentioned techniques, we also attempted to use clustering to establish a ground truth by classifying the tasks into various risk levels. However, due to the nature of our data, we were unable establish good results. Techniques such a topic modeling and domain ontology were also tried in an attempt to establish the ground truth for the data-set. But this methodology also did not give results. Finally, using the keyword based approach was seen to be giving the best results.

Curating a list of Sensitive Terms:
The list contained 3500 terms. Following entities were added to the list:

- A list of names of 513 companies [4]
- A list of 264 countries and cities [5]
- A list of 300 email addresses from Enron Data-set [7]
- A list of 1889 names [11]
- A list of randomly generated API Keys
- A list of randomly generated passwords
- A list of randomly generated usernames
- A dummy list of projects in a company [2]
- A dummy list of persons in a company
- A dummy list of projects in a company
- A list of URLs

After inducing sensitivity in the form of terms from the above mentioned list, we moved ahead to test the accuracy of our algorithm to detect information loss.

A recall of ***0.82***, precision of ***0.68***, and an F1 Score of ***0.74*** was recorded using this technique.

$$\text{Precision} = \frac{\text{Number of sensitive terms correctly identified}}{\text{Number of sensitive terms identified by the algorithm}} \tag{8.1}$$

$$\text{Recall} = \frac{\text{Number of sensitive terms correctly identified}}{\text{Number of sensitive terms in the ground truth}} \tag{8.2}$$

$$\text{F1 Score} = \frac{\text{2*Precision*Recall}}{\text{Precision + Recall}} \tag{8.3}$$

[2]A dummy list has been used to avoid any confidentiality loss for the company. While in use, the list will be replaced by a real data-set

In the given scenario, Recall is a good measure to quantify the accuracy of the algorithm. Recall reports the fraction of terms identified correctly as compared to the terms in the ground truth. In this case, we want at least all the terms in the ground truth to be reported correctly. A lower precision is due to the fact that our algorithm is conservatively reporting some terms which are not there in the ground truth. This is always a better option than missing out on some terms which are actually sensitive. So, over-reporting is okay in this context.

Chapter 9

Real World Impact

With this vision in mind, where, in the near future, companies would actively engage in crowdsourcing to hire workers for getting work done, it is of utmost importance to check the data being passed on to the workers for information loss. The framework proposed in chapter 8 caters to additional task-related resources such as code files, documents, etc. as well. Having such a mechanism in place removes the cumbersome and error-prone work of manual inspection of the task before giving it away. Manual inspection might be okay to do with simpler tasks, but as tasks get complex and code reaches a thousand lines, manual scrutiny is bound to have some errors.

The work presented in this thesis will be used at Accenture Technologies Lab, Bangalore, where they already have a tool in place which aids them in Crowdsourcing. Currently, the tool caters to posting tasks on multiple platforms, selecting workers based on a reputation system built by them, matching workers for tasks, verification of task, and processing payment, etc. This algorithm will be added as an API Call to the already existing page to post a task. An ***analyze*** button will be added to the existing page. The proposed framework can be expanded to other organizations as well by making some small changes such as the list of employees, projects, etc.

Chapter 10

Conclusions, Limitations, Future Work

10.1 Conclusions

In this study, we narrow down on the stages in the crowdsourcing cycle from where confidentiality loss could occur, list down the attributes in the Crowdsourcing tasks which could give away confidentiality, see what type of data can be lost, and then come up with an NLP and Rule based algorithm to detect confidentiality loss through a task by analyzing its various components. We reported a recall of 0.82 for our technique of identifying sensitive terms in a given task.

10.2 Limitation

We faced difficulty is in getting hold of real world data using which we could build supervised classification models. This limited us to try our hands on various Machine Learning techniques to better classify tasks. This also made testing the accuracy of the system tough for us. We had to come up with our own synthetic data-set and compute accuracy on it. It also made it tough for us to establish baselines.

10.3 Future Work

As a part of future work, we can improve the data-set and enhance it to go through even rigorous testing to validate the algorithm. The algorithm can be extended to handle more resources such as images and databases. Images can be searched for logos and the privacy preserving techniques such as K-Anonymity and L-Diversity can be implemented on the data-set. After detection of sensitive terms, various data sanitizaion techniques can also be applied before the task is given out.

Bibliography

[1] The 2016 field nation freelancer study: The changing face of the new blended workforce. Technical report, Field Nation, 2016.

[2] Accenture technology vision 2017. Accessed: April 12, 2018.

[3] Amazon mechnicalturk: Getting different human intelligence tasks done, https://www.mturk.com/, Accessed: April 12, 2018.

[4] Companies open dataset, http://www.opendata500.com/us/list/, Accessed: April 12, 2018.

[5] Countries dataset, http://www.countries-list.info/download-list, Accessed: April 12, 2018.

[6] Crowdtruth: Framework for crowdsourcing ground truth, http://crowdtruth.org/, Accessed: April 12, 2018.

[7] Enron dataset, https://www.cs.cmu.edu/ enron/, Accessed: April 12, 2018.

[8] Figureeight (crowdflower previpusly), https://www.figure-eight.com/platform/training-data/data-enrichment/, Accessed: April 12, 2018.

[9] Intuit forecast: 7.6 million people in on-demand economy by 2020, http://math.tntech.edu/rafal/cliff11/index.html. Accessed: April 12, 2018.

[10] Topcoder, a platform for innovation, Accessed: April 12, 2018.

[11] Us baby names dataset, https://catalog.data.gov/dataset/baby-names-from-social-security-card-applications-national-level-data, Accessed: April 12, 2018.

[12] Amazon. General data protection regulation (gdpr), Accessed: April 12, 2018.

[13] Vitor R. Carvalho and William W. Cohen. *Preventing Information Leaks in Email*, pages 68–77.

[14] L. Elisa Celis, Sai Praneeth Reddy, Ishaan Preet Singh, and Shailesh Vaya. Assignment techniques for crowdsourcing sensitive tasks. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, CSCW '16, pages 836–847, New York, NY, USA, 2016. ACM.

[15] A. Dubey, K. Abhinav, and G. Virdi. A framework to preserve confidentiality in crowdsourced software development. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 115–117, May 2017.

[16] Cynthia Dwork. Differential privacy. In *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, volume 4052, pages 1–12, Venice, Italy, July 2006. Springer Verlag.

[17] Dale Edgar. Data sanitization techniques, Accessed: April 12, 2018.

[18] Evgeniy Gabrilovich and Shaul Markovitch. Text categorization with many redundant features: Using aggressive feature selection to make svms competitive with c4.5. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 41–, New York, NY, USA, 2004. ACM.

[19] Google. Cloud data loss prevention api, https://cloud.google.com/dlp/, Accessed: April 12, 2018.

[20] Mitchell Gordon, Walter S. Lasecki, Winnie Leung, Ellen Lim, Steven Dow, and Jeffrey P. Bigham. Glance privacy: Obfuscating personal identity while coding behavioral video. In *HCOMP*, 2014.

[21] Tanya Goyal, Sanket Mehta, and Balaji Vasan Srinivasan. Preventing inadvertent information disclosures via automatic security policies. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon, editors, *Advances in Knowledge Discovery and Data Mining*, pages 173–185, Cham, 2017. Springer International Publishing.

[22] Michael Hart, Pratyusa Manadhata, and Rob Johnson. Text classification for data loss prevention. In Simone Fischer-Hübner and Nicholas Hopper, editors, *Privacy Enhancing Technologies*, pages 18–37, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[23] Jeff Howe. The rise of crowdsourcing, https://www.wired.com/2006/06/crowds/. (1).

[24] Aileene Koh. How to protect ideas during patent pending, Accessed: April 12, 2018.

[25] Adriana Kovashka, Olga Russakovsky, Li Fei-Fei, and Kristen Grauman. Crowdsourcing in computer vision. *CoRR*, abs/1611.02145, 2016.

[26] L. Layman and G. SigurÃřsson. Using amazon's mechanical turk for user studies: Eight things you need to know. In *2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 275–278, Oct 2013.

[27] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115, April 2007.

[28] Xuan Liu, Meiyu Lu, Beng Chin Ooi, Yanyan Shen, Sai Wu, and Meihui Zhang. CDAS: A crowdsourcing data analytics system. *CoRR*, abs/1207.0143, 2012.

[29] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[30] Ke Mao, Licia Capra, Mark Harman, and Yue Jia. A survey of the use of crowdsourcing in software engineering. 126, 09 2016.

[31] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[32] NLP. Natural language processing, Accessed: April 12, 2018.

[33] NLTK. Nltk stemming, http://www.nltk.org/api/nltk.stem.html, Accessed: April 12, 2018.

[34] End Point Protector. End point protector: Cross-platform data loss prevention, Accessed: April 12, 2018.

[35] Thomas Roelleke and Jun Wang. Tf-idf uncovered: A study of theories and probabilities. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 435–442, New York, NY, USA, 2008. ACM.

[36] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents, 03 2010.

[37] Sriram Sharma. Tcs employee accidentally leaks confidential data on github, gets roasted, Accessed: April 12, 2018.

[38] Stanford. Named entity recognition, https://nlp.stanford.edu/software/crf-ner.html, Accessed: April 12, 2018.

[39] Stanford. Part-of-speech tagging, https://nlp.stanford.edu/software/tagger.shtml, Accessed: April 12, 2018.

[40] Stanford. Rule based filtering, https://nlp.stanford.edu/software/tokensregex.html, Accessed: April 12, 2018.

[41] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection, 2012.

[42] Latanya Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002.

[43] PoS Tagging. Part-of-speech tagging, https://en.wikipedia.org/wiki/part-of-speech_tagging, Accessed: April 12, 2018.

[44] TF-IDF. Tf-idf, https://en.wikipedia.org/wiki/tf12, 2018.

[45] Sabrina De Capitani Di Vimercati, Sara Foresti, Stefano Paraboschi, Gerardo Pelosi, and Pierangela Samarati. Shuffle index: Efficient and private access to outsourced data. *ACM Trans. Storage*, 11(4):19:1–19:55, October 2015.

[46] Janet Wagner. Why exposed api keys and sensitive data are growing cause for concern, Accessed: April 12, 2018.