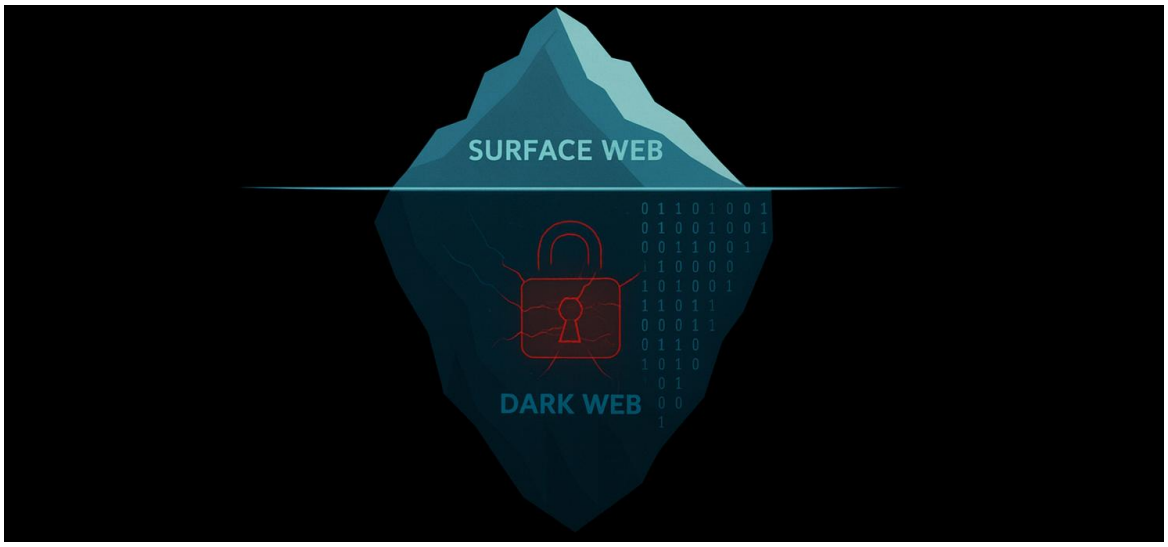
INTERNATIONAL HELLENIC UNIVERSITY

SCHOOL OF ENGINEERING

DEPARTMENT OF INFORMATION AND ELECTRONIC ENGINEERING

THESIS

« DISCLOSURE OF INFORMATION ON THE DARK WEB»



Of Konstantinos Terzis
Registration Number: 2020168

Supervisor
Full name: Prof. Christos Ilioudis
Rank: Lecturer

Date 24/08/2025

Title of Thesis: Disclosure of information on the Dark Web
Thesis Code: 25142
Name of student: Konstantinos Terzis
Name of the professor: Christos Ilioudis
Date of assumption of the Thesis: 06/03/2025
Date of completion of the Thesis: 24/08/2025

I certify that I am the author of this work and that any assistance I received in its preparation is fully acknowledged and referred to in the work. I have also listed any sources from which I used data, ideas, images and text, whether they are cited exactly or paraphrased. Furthermore, I certify that this work was prepared by me personally, specifically as a diploma thesis, at the Department of Computer Engineering and Electronic Systems of the Institute of Electrical and Electronics Engineering of the Hellenic University of Athens.

The approval of the thesis by the Department of Computer Engineering and Electronic Systems of the International Hellenic University does not necessarily imply acceptance of the views of the author by the Department.

«To all people who envision a safer digital world»

Prologue

The choice of this diploma thesis topic was initiated by a real interest in the cybersecurity field and, in particular, by the investigation of the data breaches and the dark web linkage. From the research I conducted, I recognized the fact that the leakage of secret data has been the most problematic phenomenon of the Information Age, causing massive impacts on individuals, organizations, and society as a whole. The reason why I was motivated was not only to investigate the theoretical foundation of the data leakage but also to contribute practically by proposing and implementing a tool that is capable of identifying such leakages.

The experience of developing this thesis provided me with great academic and professional value. In the first place, this widened my understanding of the dark web infrastructures, anonymizing technologies, and the nature of compromised data. Second, it provided me with the possibility of gaining advanced technical skills via the educative process of the development of a software product, connecting the research aspect to the practical development aspect. Overall, this experience constitutes the marriage of academic research and practical development, which enriched my knowledge of cybersecurity and my preparedness to confront the professional challenge of the field.

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται τη διακίνηση πληροφοριών στον σκοτεινό ιστό, δίνοντας έμφαση στον τρόπο με τον οποίο οι παραβιάσεις δεδομένων οδηγούν στην εμφάνιση ευαίσθητων πληροφοριών σε κρυφά διαδικτυακά περιβάλλοντα. Οι διαρροές και οι παραβιάσεις δεδομένων, που μπορεί να κυμαίνονται από προσωπικά διαπιστευτήρια έως αρχεία οργανισμών, αποτελούν ένα από τα μεγαλύτερα προβλήματα της ψηφιακής εποχής. Ως εκ τούτου, η εργασία ξεκινά διερευνώντας πώς συμβαίνουν οι παραβιάσεις δεδομένων, συμπεριλαμβανομένων των μεθόδων και των συνθηκών που οδηγούν στη διαρροή πληροφοριών, με τη βοήθεια γνωστών ιστορικών παραδειγμάτων που καταδεικνύουν την κλίμακα και τον αντίκτυπο των παραβιάσεων. Στη συνέχεια, εξετάζει πώς τα υποκλαπέντα δεδομένα διανέμονται μέσω των υποδομών και των τεχνολογιών ανωνυμοποίησης του σκοτεινού ιστού, όπως το Tor και το I2P, οι οποίες υποστηρίζουν κρυφές αγορές και συμβάλλουν στο παράνομο, υπόγειο εμπόριο κλεμμένων δεδομένων.

Βασιζόμενο σε αυτές τις διαπιστώσεις, το πρακτικό μέρος της εργασίας παρουσιάζει το BreakChecker, ένα εξειδικευμένο εργαλείο που σχεδιάστηκε για την ανίχνευση ψηφιακών εκθέσεων. Το εργαλείο ενσωματώνει ανίχνευση τομέων, απαρίθμηση υποτομέων και αναζήτηση παραβιάσεων, με σκοπό τον εντοπισμό διαρροών διαπιστευτηρίων και άλλων ευαίσθητων δεδομένων ενός οργανισμού. Συνδυάζοντας την αναγνώριση και τους ελέγχους έκθεσης, το BreakChecker καταδεικνύει πώς η ακαδημαϊκή έρευνα μπορεί να εφαρμοστεί στην πρακτική ανάπτυξη εργαλείων για την παρακολούθηση κινδύνων.

Τα ευρήματα αυτής της μελέτης μπορούν να χαρακτηριστούν ως πολλά υποσχόμενα, καθώς το εργαλείο προσφέρει μια πρακτική και ελαφριά μέθοδο για την ανίχνευση παραβιάσεων, ενώ η ίδια η μελέτη συμβάλλει στην ακαδημαϊκή εξοικείωση με τη διαρροή δεδομένων καθώς και με τους παράγοντες του οικοσυστήματος του σκοτεινού ιστού. Συνολικά, αυτές οι συνεισφορές υπογραμμίζουν την ανάγκη για σύζευξη της θεωρητικής έρευνας με την πρακτική ανάπτυξη, προκειμένου να αντιμετωπιστούν οι σύγχρονες προκλήσεις της κυβερνοασφάλειας.

«Disclosure of information on the Dark Web»

«Konstantinos Terzis»

Abstract

This thesis addresses the disclosure of information on the dark web, emphasizing how data breaches are resulting in sensitive information emerging in hidden online environments. Data leaks and breaches, which can range from personal credentials to organization records, are one of the biggest problems of the digital age. Therefore, the thesis begins by exploring how data breaches happen, including the methods and situations resulting in the leaking of information, with the help of well-known examples from history that illustrate the scale and impact of breaches. It then examines how the stolen data is distributed through infrastructures and anonymity technologies of the dark web, like Tor and I2P, which support hidden markets and contribute to the illicit underground trade of stolen data.

Building on these insights, the practical part of the thesis introduces BreakChecker, a custom tool designed to detect digital exposures. The tool integrates domain crawling, subdomain enumeration, and breach search to find leaked credentials and other sensitive data of an organization. By combining reconnaissance and exposure checks, BreakChecker shows how academic work can be applied to the practical development of risk monitoring.

The findings of this study can be discussed as promising, with the tool offering a practical and lightweight method of breach detection, and the study itself contributing to the scholarly familiarity with data leakage as well as dark web ecosystem factors. Collectively, these contributions point to the need to couple theoretical research with practical development to meet recent challenges of cybersecurity.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Professor Christos Ilioudis, for his invaluable guidance, constructive feedback, and continuous support throughout the course of this thesis. I am also grateful to the Department of Information and Electronic Engineering for providing a stimulating academic environment and the resources necessary for the completion of this work. Special thanks are extended to my family and friends for their patience, encouragement, and understanding during this demanding journey. Their support has been an essential source of strength.

Table of Contents

List of Figures

List of Tables

Abbreviations

2FA	Two-Factor Authentication
AES	Advanced Encryption Standard
API	Application Programming Interface
BEC	Business Email Compromise
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CI/CD	Continuous Integration / Continuous Delivery
CLI	Command-Line Interface
CSV	Comma-Separated Values
DHT	Distributed Hash Table
DMCA	Digital Millennium Copyright Act
DNS	Domain Name System
DoB	Date of Birth
DOM	Document Object Model
FERPA	Family Educational Rights and Privacy Act
GDPR	General Data Protection Regulation
HIBP	Have I Been Pwned
HIPAA	Health Insurance Portability and Accountability Act
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I/O	Input/Output
I2P	Invisible Internet Project
IBM	International Business Machines
IP	Internet Protocol
JSON	JavaScript Object Notation
KYC	Know Your Customer
LotL	Living off the Land
M&A	Mergers and Acquisitions

MaaS	Malware as a Service
MFA	Multi-Factor Authentication
MX	Mail Exchange
N-S-D	Name/SSN/DoB bundle
NGO	Non-Governmental Organization
NLP	Natural Language Processing
OSINT	Open-Source Intelligence
OTP	One-Time Password
OTR	Off-the-Record Messaging
PGP	Pretty Good Privacy
PII	Personally Identifiable Information
POS	Point of Sale
REST API	Representational State Transfer Application Programming Interface
RFC	Request for Comments
SFTP	SSH File Transfer Protocol
SHA-1	Secure Hash Algorithm 1
SIM	Subscriber Identity Module
SMS	Short Message Service
SOC	Security Operations Center
SSN	Social Security Number
TLD	Top-Level Domain
TLS	Transport Layer Security
Tor	The Onion Router
URL	Uniform Resource Locator
VPN	Virtual Private Network
WSGI	Web Server Gateway Interface
XMPP	Extensible Messaging and Presence Protocol

Chapter 1: Introduction

1.1 Definition

This thesis explores the phenomenon of information exposure of data breaches and its relationship with the dark web's underground economy. It analyzes the lifecycle of breached information from its original incursion to its eventual trading on dark marketplaces. In addition to studying previous breaches and dark web infrastructure, the thesis also details the development and evaluation of a custom tool—BreakChecker—able to detect exposure of personal data. Bringing technical insight, case-study analysis, and practical implementation together, the thesis attempts to present detailed information on threats and facts of information commodification in cyberspace.

The dark web is a part of the internet that does not appear on standard search engines like Google or Bing. It can be reached using specialty software such as Tor (The Onion Router), which allows users to browse anonymously. This anonymity can be a tool for privacy-conscious users, journalists, and whistleblowers, but it also creates a safe space for criminals [1]. Coincidentally, the dark web became a marketplace for illegal activities, such as drug trade, weapons trade, fake document trade, and the most dangerous is stolen personal data or corporate data.

One of the most significant risks posed by the dark web is the ease with which it allows for the trade of stolen information. Personal information such as credit card numbers, Social Security numbers, medical records, passwords, and even full profiles of identities are traded on underground forums [2]. Cybercriminals can use this data for identity theft, financial fraud, and even blackmail. However, over the years, data breaches have multiplied, creating a multibillion-dollar economy for stolen information. For individuals and businesses, their private information may have made its way into the wild without their knowledge.

As we move toward digital-centric lives, the volume of personal and sensitive data we put online increases. From banking and shopping to social media and work, we use the internet for nearly everything. However, that also makes us more vulnerable. Hackers take advantage of weak security systems to access databases and retrieve private information, which is eventually sold to the dark web [3]. What exacerbates this is the use of cyber currencies like Bitcoin and Monero that enable such cybercriminals to sell the stolen datasets without leaving a clear audit trail for any law enforcement to follow [4].

For an individual, if their data gets leaked on the dark web, then it can have serious repercussions. This leads to unauthorized transactions and financial losses if thieves have stolen credit card details and bank information. It can wreck a person’s credit score so badly that it is hard to get a loan – or even a job. And in some other situations, cybercriminals use this personal information to run scams, extortion, or blackmail. Dealing with the consequences of identity theft can be emotionally draining, especially since it can take victims years to restore their names and fix any financial harm that has been caused [3].

Companies also have significant risks when their data is leaked on the dark web. A single security breach could expose thousands, even millions, of customer records. This not only costs them money but can also cause them legal problems, regulatory fines, and a loss of customer trust. So, if a serious data breach happens, many companies cannot recover. Even major organizations, such Yahoo, Equifax and Facebook, have been the victims of massive cyberattacks, making it clear that no company should

consider itself entirely safe [5]. What's worse, some dark web marketplaces are even selling stolen trade secrets, making corporate espionage more accessible than ever before [3].

Governments and law enforcement agencies worldwide are attempting to combat cybercrime on the dark web. They've rolled out tougher laws on cybercrimes, stricter data protection policies, and international cooperation to allow the prosecution of cyber criminals. However, closing illegal enterprises on the dark web is extremely challenging. The dark web is not like the regular internet, and is decentralized, meaning websites often change their addresses and servers to avoid detection. Even when authorities succeed in shuttering one illicit marketplace, another pops up in its stead [6].

Cybersecurity is an area where experts say both consumers and businesses need to make it a priority. Using strong passwords, multi-factor authentication and regular software updates can help prevent cyberattacks. Cybersecurity training and regular vulnerability audits should be included by businesses. For people, using a password manager, watching credit reports and being careful of phishing scams can lessen the chances of having personal information exposed [7].

One big problem is that a lot of people don't even know if their data has been exposed. Most stolen information is traded on the dark web without victims' knowledge, unlike high-profile hacks that become news [8]. That means a person's private information could be held by criminals for several months – or years, in some cases – before anyone even realizes that fraudulent use is taking place. That's why cybersecurity experts say people should regularly check whether their personal information has been exposed in past breaches and take action to secure their accounts [5].

The dark web is still growing as a marketplace for stolen data, despite concerted efforts to detect and prevent cybercriminal activities [9]. The issue is not simply that there is a technology challenge – it's that there is an awareness and an education challenge. If people can explain how risky these things are, then they should know how to protect themselves. Cybersecurity isn't only a concern for IT professionals anymore – it's a topic everyone should consider in today's digital landscape. Though it may never be possible to wipe cybercrime completely off the map, taking security seriously is the best way to start protecting sensitive personal and business information.

1.2 Objectives

The subject of this thesis is to investigate the issue of information exposure due to data breaches and analyze leaked organizational and personal data on dark web sites. Its aim is to investigate breach causes, to identify which kind of sensitive data is most frequently leaked, and to determine the risks generated by this kind of data on obscured marketplaces on dark web sites.

Aside from a paper on breach mechanisms and effects, the thesis also entails what constitutes the core of dark web infrastructure to support thriving underground markets of illicit data. Special emphasis is on such marketplaces as well as anonymizing protocols through which such marketplaces can exist outside mainstream regulatory and law enforcement scope.

In addition to the research, this thesis presents BreakChecker as a usable software to help users and organizations identify their vulnerabilities after breaches. Instead of offering a purely theoretical analysis, this implementation demonstrates how verification and detection are achievable in controlled and privacy-minded environments.

More broadly, the thesis seeks to synthesize data-disclosure and dark-web-structuring investigations with the construction of an empirically informed mode of evaluation. It seeks thereby to contribute to

learned understanding of data leakage as well as to provide an actionable approach that will be able to improve resilience and security practice in live use.

1.3 Achievements

This thesis delivers both theoretical knowledge and a practical solution to the challenge of information disclosure. From a research angle, it offers a detailed examination of data breaches, their lifecycle, and their effects on individuals, organizations, and states. It discusses past case studies, combines technical and human reasons why such breaches happen, and examines channels through which illegal information emerges on the dark web. With a combination of both academic research and industry findings, the thesis offers a clear understanding of data leakage mechanisms alongside the general socio-economic systems underlying the criminal underground market of stolen information.

At a technical level, the biggest contribution is the development of BreakChecker. The tool merges host subdomain scanning, real-time reconnaissance, contact detection, and breach verification in a lightweight, extensible framework. Developed in Python with support from modern libraries for automation and parsing, like Playwright, tools like Subfinder to assist with domain discovery, and APIs like Leakcheck and HaveIBeenPwned for breach validation, it combines functionality with privacy by design, verifying identifiers without implicit exposure. Extremely modular and portable, one can utilize it reliably in a wide range of environments, providing researchers and practitioners with a real-world tool to monitor digital exposure.

The technical tool, BreakChecker, and the academic research together form a complementary framework towards comprehending and mitigating modern data leaks. The academic part places the risks of data exposure into perspective within wider dark-web scenarios, and the technical part delivers a practical toolset for detection and prevention. As a combined effort, they represent both cybersecurity research work and a measure towards increased robustness and personal safety from emerging cyber threats.

1.4 Contents Overview

The thesis comprises seven chapters, each addressing how sensitive information is exposed through data breaches and subsequently disseminated and exploited on the dark web.

The first chapter introduces the topic and explains the motivation for the research. It outlines the main objectives, expected contributions, and provides an overview of the chapters that follow.

Chapter two focuses on data breaches, what they are, how they occur, and the techniques attackers use to obtain private information. It also reviews major real-world cases, showing how these incidents have affected individuals and organizations, and how the compromised data often ends up in illicit trading networks.

Chapter three examines the dark web itself. It describes how it operates, how it differs from the surface and deep web, and the types of content typically hosted there. Special attention is given to the underlying structure and anonymizing technologies, such as Tor, I2P, and Freenet, which sustain hidden platforms and enable the exchange of leaked data.

Chapter four presents the tool developed for this thesis: BreakChecker. It explains the tool’s design, the external components it integrates, and how they work together to allow users to verify whether their data appears in breach datasets. The chapter also discusses the development choices made with regard to privacy and usability.

Chapter five provides an evaluation of BreakChecker in practice. It details testing results, introduces performance metrics, compares the tool with existing alternatives, and assesses its overall effectiveness.

Chapter six concludes the thesis by summarizing the key findings and reflecting on the contributions of both academic research and technical implementation to the broader field of cybersecurity. It also highlights the tool's limitations, provides recommendations for improvement, and suggests directions for future work.

Finally, chapter seven contains the reference list, including all sources used in the thesis, formatted according to IEEE standards.

Chapter 2: Data Breaches

2.1 What Is a Data Breach and How It Happens

In today's digital environment, in which almost all individual, financial, and business transactions are done online, the term data breach has become an issue of paramount importance. A data breach means any occasion on which confidential, sensitive, or protected data is accessed, disclosed, or stolen by an unauthorized entity. This can include the theft of usernames and passwords, credit card information, email addresses, social security numbers, medical data, or business confidential information [10]. The repercussions of such breaches extend far and wide to reach individuals, organizations, and even governments. For some, harm done by a breach is not only financial in nature—it is reputational, operational, and highly personal [11]. This section examines the nature of data breaches, their lifecycle, and the mechanisms through which they occur.

Breaches rarely occur in a single night. In all too many cases, however, breaches are the culmination of well-organized attacks or long-dormant vulnerabilities that never received attention. At an abstract level, nearly all breaches go through multi-step lifecycles that start with an intruder getting in and concluding with the distribution or reselling of stolen information [12]. Some of these intruders include cybercriminals who are acting out of greed, hacktivists who are acting out of desire to issue political statements, or even nation-states acting out of espionage. No matter what their motivation, their tactics are frequently sophisticated and dynamic [13].

The attack process tends to start with the intruder recognizing a weakness or vulnerable spot in a system. This can be as straightforward as an insecurely configured server or as sophisticated as taking advantage of a zero-day exploit—a previously unknown software weakness that has yet to be fixed [14]. More frequently in many situations, the point of entry is much less heroic: a phishing email. The email is designed to appear legitimate and is intended to get the user to click on malicious links or submit their login credentials [15]. With access well-established, the attacker can upgrade their privilege level, from a low-level user to the administrator level of permissions, with greater control over the platform [16].

From there, attackers can begin to locate and extract valuable data, often over a period of weeks or even months. This methodical exfiltration ensures that large volumes of information can be taken without triggering alarms [12]. Attackers may also establish long-term access through the installation of backdoors or remote access tools, allowing them to return to the compromised system even if the initial vulnerability is eventually discovered and patched. In some cases, attackers use malware designed specifically to remain hidden and transmit data silently [17].

In order to gain a better idea of the chain of activities that comprise a data breach, one can visualize the breach lifecycle. The lifecycle describes the five generic stages that a typical breach goes through: exploitation, escalation, exfiltration, persistence, and distribution [12], [18].

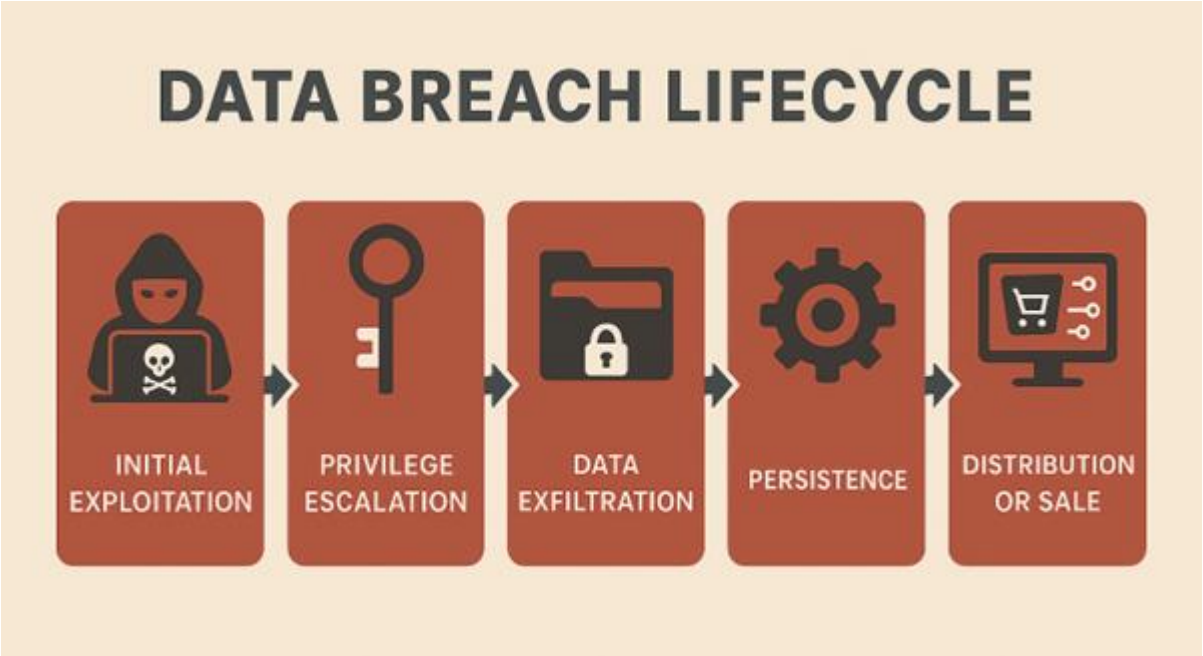


Figure 2.1: Breach Lifecycle – From Infiltration to Dark Web Sale

Figure 2.1 illustrates the steps from initial compromise to the last stage of data distribution. It is one of the typical processes utilized by threat actors to extract and sell stolen data [12], [18].

In the stage of Initial Exploitation, attackers gain entry to the system. This is done through social engineering, credential stuffing, brute-force attacks, or unpatched vulnerabilities. Having entered the system, they proceed to Privilege Escalation, a process in which they seek to acquire high-level access to the system and extend their reach to highly valuable targets. Next is Data Exfiltration, in which they exfiltrate sensitive data in a covert manner, encrypting the data before exporting it to remote servers. At the stage of Persistence, they ensure their long-term entry even if their activities are partly discovered. Lastly, the stolen data moves to the Distribution or Sale phase, in which it is leaked, ransomed, or marketed on dark web forums or marketplaces [12], [18].

The lifecycle described above isn't fictional. That is what modern attackers do in the real world. Many high-profile breaches—Equifax to Marriott—are all done this way. They also demonstrate how tricky it might be to detect and end a breach in its initial stages. Exploitation could be done in minutes or even a few hours, but detection can take months. Indeed, 277 days were the mean time to detect and contain a breach during the IBM 2023 Cost of a Data Breach Report. Attackers might steal hundreds of gigabytes of corporate and individual data in the meantime, unnoticed [19].

The emergence of remote working environments has only added to this. Staff logging into highly sensitive systems in their homes on unencrypted personal equipment or over weakly secured networks has introduced new attack vectors. Most organizations were not ready for this transition and had security measures in place that were not designed to process this type of distributed threat. Cybercriminals are aware of such changes and have adjusted their methods in response [20].

However, not all data breaches are the result of external attacks. Sometimes the cause is much simpler—human error or insider negligence. An employee might send sensitive data to the wrong email address, lose a device with unencrypted files, or misconfigure a cloud storage bucket, leaving it exposed to the public. Insider threats, where a staff member abuses legitimate access, are also a growing concern. These

types of breaches are often harder to detect and prevent, since the individuals involved already have some level of access to the system [21].

As soon as data has been exfiltrated, it frequently is readied to be sold. Cybercriminals are aware of the worth of varying data categories and structure it appropriately. Login credentials, passport scans, medical reports, financial data—all are monetizable. The dark net serves as one of its key distribution channels wherein this data is auctioned off, sold or packaged with other stolen items. All of this is done in cryptocurrency to ensure anonymity to the buyer and seller [4].

At its root, a data breach is not an isolated incident but a process developing over time that perhaps goes unnoticed by the victim until the damage is done. It's not so much a matter of technical failure but that of organizational resilience, employee vigilance, and continuous monitoring. Knowing the process of a data breach step by step is key to preventing future occurrences and serves as the basis on which dark web monitoring and breach detection software—such as the one discussed later in this thesis—is becoming irreplaceable in cybersecurity [12].

2.2 Historical Cases of Major Breaches

Understanding the anatomy of a breach—what causes it, how it occurs—is only half the story. To get a better understanding of just how far and wide-reaching their impact can be, one must examine actual breaches that have not only invaded digital infrastructure but have affected industries, destroyed reputations, and touched lives on a million scale, and which have been watershed moments for cybersecurity. There have been several high-profile breaches during the last two decades which have been pivotal moments for cybersecurity, and which illustrate how susceptible even the biggest and most technologically advanced corporations can be. This section explores major real-world breaches to illustrate their causes, scale, and long-lasting impact on organizations and individuals.

Among the most disastrous is Yahoo's data breach, which is said to be the most significant known breach to date. Between 2013 and 2017, Yahoo revealed that more than 3 billion accounts of users had been affected by cyber-attacks going as far back as 2013 [22]. Affected data included names, email addresses, phone numbers, and hashed passwords, as well as encrypted and unencrypted security questions and answers. The hackers allegedly used fake cookies to evade authentication and achieve persistent access to users. A complicated attack that demanded deep knowledge of Yahoo's internal code and architecture. The breach, the news of which was delayed for years, also had severe financial consequences—it lowered the sale of Yahoo to Verizon by $350 million and disclosed critical vulnerabilities of the company's cybersecurity response to incidents [23].

A second key moment was the 2017 Equifax breach where the personal data of almost 148 million people was stolen. The breach occurred because Equifax neglected to patch a publicly known bug for a high-severity vulnerability in the Apache Struts framework, which is an open-source web application framework. Attackers were able to exploit the unpatched vulnerability to gain access to backend systems without any authorization. After gaining entry, they traversed through poorly segregated databases and stole enormous quantities of sensitive information, such as Social Security numbers, birth dates, home addresses, and driver's license numbers [24]. The incident became a classic example of poor patch management and system segregation, resulting in extreme regulatory, legal, and reputational penalties.

The 2012 LinkedIn breach also holds valuable lessons. Originally reported to have hit around 6.5 million user passwords, the complete extent of the breach was only understood in 2016 when over 117 million email-password combinations were discovered on the dark web. LinkedIn had been keeping these passwords in outdated SHA-1 hashing with no salting, which meant that it was simple for attackers to

decrypt the information once stolen [25]. Although the initial penetration method is yet to be confirmed, the attacker, who is identified as Yevgeniy Nikulin, achieved the penetration by stealing an employee's credentials from LinkedIn, who then used them to penetrate the company's network. The attack serves to highlight the vulnerability of poor password policies as well as the absence of a second authenticator factor such as two-factor authentication [26].

Adobe Systems suffered a major breach in 2013 that exposed data from more than 38 million active user accounts and may have affected 150 million users. Hackers breached Adobe's internal network using outdated software and accessed source code repositories, encrypted passwords, and user data [27]. The breach was notable not only because of the volume of data stolen but also because intellectual property was stolen, indicating that hackers had an interest in both proprietary programs and user data [28].

The Marriott International attack is notable for its persistence and scope. Discovered as early as 2018, the attack, started as early as 2014, and had its target as the Starwood Hotels guest reservation database, purchased by Marriott in 2016. The attackers gained access through the use of trojan malware, probably distributed through phishing, and remained undetected for four years, exposing sensitive information such as passport numbers, encrypted payment card information, as well as email addresses of an estimated 500 million customers [29]. Another attack soon followed in 2020 where attackers used stolen login credentials of two franchise staff members to access an internal application, compromising the personal information of an estimated 5.2 million visitors [30]. Combined, both incidents highlight the risks of credential abuse, monitoring deficiency, and integration issues of systems after acquisition, emphasizing the need for tight access controls with complementing real-time threat discovery mechanisms.

While not a breach in the classical meaning, the Facebook–Cambridge Analytica debacle in 2018 exposed profound vulnerabilities in data governance. Cambridge Analytica obtained unauthorized access to the personal data of 87 million Facebook users through a harmless personality quiz app. The real issue was Facebook's overly lenient API controls that allowed the app to collect not only information from direct app users but from all their friend networks as well. While no technical exploitation was attempted, the scandal unleashed Facebook's failure to anticipate data misuse and set a new worldwide benchmark for privacy concerns and platform accountability [31].

The Target breach in 2013 compromised credit and debit card information for more than 40 million shoppers, along with personal information for another 70 million people. The breaches began via a third-party heating and air-conditioning contractor who had remote access to the Target network. Point-of-sale (POS) malware was utilized by the attackers, after gaining access, to steal data from checkout lanes. This breach highlights the risks of vendor management failure and poor network segmentation that permitted attackers to move laterally from a low-trust third-party account to the core infrastructure [32].

Finally, the 2014 hack of Sony Pictures Entertainment was a politically motivated hack by state-sponsored North Korean hackers. The hackers applied wiper malware to render data on over 3,000 computers unusable and stole around 10 Terabytes of sensitive employee data, unreleased movies and internal emails [33]. The breach was said to have been initiated using spear-phishing emails along with custom-made malware that was tailored to target Sony's networks. The hack was an act of revenge because it followed the forthcoming release of the movie The Interview, which described a fictional plot to assassinate the existing North Korean leadership. The hack is representative of one of the geopolitical motivations fueling the frequency and scale of cyberattacks [34].

Table 2.1 below illustrates the major attributes of eight of the most influential data breaches across various industries and time frames. Each row outlines the timeline of the attack, the number of records taken, methods of exploitation, types of stolen data, and the resulting consequences. Several vulnerabilities such as poor patch management, weak authentication, and lack of segmentation are highlighted side by side. Together, the examples reveal how everyone from individual hackers to nation-state groups has exploited vulnerabilities for financial, political, or strategic gain. By providing the samples in a unifying template, the table illustrates the true long-term cost of breaches and highlights the need for improved organizational resilience and real-time threat detection.

Table 2.1: Summary of Major Historical Data Breaches

Breach	Timeline	Records Affected	Method of Attack	Data Compromised	Consequences
Yahoo	2013–2017	3 billion	Fake cookies, credential reuse	Names, emails, phone numbers, passwords, security Q&As	Acquisition impact, reputational damage
Equifax	2017	148 million	Unpatched Apache Struts vulnerability	SSNs, birth dates, addresses, DL numbers	Lawsuits, regulatory penalties
LinkedIn	2012–2016	117 million	Credential theft, poor hashing	Emails, passwords	Dark web resale, password reuse risk
Adobe	2013	38–150 million	Outdated software, credential theft	Passwords, usernames, source code	IP theft, dark web sale
Marriott	2014–2020	500M + 5.2M	Phishing, credential theft	Passport data, payment info, emails	Repeated breach, integration challenges
Facebook-Cambridge Analytica	2018	87 million	Weak API governance	User profiles, friend network data	Global regulatory scrutiny
Target	2013	110 million	3rd party access, POS malware	Credit card info, addresses	Retail losses, compliance changes
Sony Pictures	2014	10 TB of data	Spear phishing,	Emails, unreleased	Nation-state retaliation,

			wiper malware	films, employee data	industry wake-up

When these breaches are placed side-by-side, distinct patterns appear across industries, organizations, and attack vectors. In almost all cases, the breaches could have been avoided—or entirely prevented—by better security hygiene. Was it Yahoo's failure to notice spoofed cookies, Equifax's slowness to apply a widely known security patch, or LinkedIn's use of outdated hashing functions? A string of preventable errors exists in all of them [22], [24], [25]. Even in more advanced, state-sponsored attacks like Sony's, the initial breach vector was a phishing email, a technique reliant more on social engineering than technical sophistication [34].

One of the common takeaways from these breaches is the necessity for basic cybersecurity fundamentals: prompt patching, strong password practices, network segmentation, multi-factor authentication, and real-time monitoring. Most of the systems that were breached lacked proper alerting capabilities or incident detection, which allowed attackers to remain hidden for months or even years. Marriott's multiyear breach and Adobe's source code exfiltration both illustrate how crippling a long-term, stealthy intrusion can be when companies do not have sufficient visibility into their networks [27–30].

Further, the nature of the data exfiltrated in each breach paints a picture of evolving attacker agendas. Older breaches tended to compromise account credentials or credit card information [22], [25], [27]. In contrast, newer breaches now encompass sensitive personal identifiers (i.e., passport or Social Security numbers), corporate intellectual property (i.e., source code), and even political influence data (such as with Cambridge Analytica) [24], [28], [31]. These developments reflect how information has not just become a commodity, but also a weapon—usable for economic sabotage, identity theft, corporate sabotage, and even electoral manipulation.

What happens to this information after it is stolen is of equal concern. A great deal of this stolen information eventually ends up on the dark web, where it fuels an underground economy of resale, extortion, and credential stuffing. Stolen credentials from LinkedIn, Yahoo, and Adobe have been found in bundled "combo lists" that are utilized to automate login attempts at other services [22], [25], [27]. Target and Marriott payment card data have been sold on dark web marketplaces, and Equifax and Facebook personally identifiable information profiles have enabled fraud activities and social engineering campaigns [24], [29], [31], [32]. That this breached data is still circulating with no foreseeable endpoints to the flows underscores the tactical imperative of dark web monitoring and post-breach risk mitigation strategies, such as consumer notification, identity theft protection, and credential rotation.

Lastly, these breaches had impacts that went well beyond technical remediation. They resulted in executive resignations, regulatory inquiries, lawsuits, and new legislation. The Equifax and Facebook incidents, especially, were part of the worldwide momentum that resulted in the passage of the General Data Protection Regulation (GDPR) in the European Union and heightened demands for equivalent legislation everywhere [24], [31], [35]. In the longer term, it forces businesses to think differently not only about how they protect and secure information but also about how they are made accountable for its misuse.

In total, these historical examples are not one-off occurrences; they are warnings. They show us how vulnerabilities—technical, procedural, or organizational—can be exploited with catastrophic

consequences. They also demonstrate the importance of anticipatory defense, user awareness, and transparency. As we transition to the next section, we'll examine the attack vectors and propagation channels that drive these breaches in the background and enable stolen information to flow into dark web markets.

2.3 Methods of Attack and Data Distribution

Data breach cyberattacks are layered and planned, reflecting a level of strategy far above technical intrusion. Modern adversaries have a strategic intent, coupling together social engineering, exploitation of software, and silent exfiltration to meet goals. Perhaps most critically, success isn't only dependent upon initial access establishment but also upon the ability to steal compromised data quietly and securely. This section looks at how attackers gain access to systems, as well as how data are secured during its first transfer and preparation for black marketplace distribution.

One of the most successful means of exploiting a point of entry has been through social engineering. Phishing has been a favorite point of entry for many attackers. Using spoofed emails presenting as a trusted organization, attackers manipulate individuals into sharing credentials or downloading payloads. Phishing has been a favorite amongst cybercriminals because of its ease of deployment along with versatility. Spear-phishing, where people are targeted using content specifically designed to appeal to that individual, and business email compromise (BEC), where corporate chain-of-trusts are employed to enable bogus activities, have been particularly catastrophic [36]. Even high companies like Marriott, along with LinkedIn, have been victims of being targeted through such points of entry [25], [29]. Such attacks are complemented by data obtained through open-source intelligence (OSINT), allowing attackers to tailor messages to enhance success rates.

Malware serves as a second critical enabler of modern breaches. Remote Access Trojans like NetWire and NanoCore enable extended unauthorized access to compromised networks [37]. Infostealers like RedLine capture credentials, session cookies, and e-wallets that are stored in browsers, making them highly effective at account takeover and stealing credentials [38]. More recently, double-extortion modes have been adopted by ransomware groups, where sensitive-type data get exfiltrated before files are encrypted. Conti and Maze have employed these methods to devastating success, dumping stolen data on specialized leak sites when ransoms are not paid [39]. The recent 2021 attack on Colonial Pipeline demonstrates how ransomware can escalate to a national-level crisis, affecting not only private entities but also critical infrastructure [40].

Exploitation also ranks among the primary reasons why initial compromise occurs. Vulnerabilities reported in mainstream software by public disclosure are exploited by attackers. Vulnerability scanners like Shodan make it possible for cybercriminals to identify open systems containing outdated services, making exploitation more efficient [41]. The 2017 Equifax breach, caused by an unpatched vulnerability in an old Apache Struts, is still considered among the most significant examples of late patch management [24]. More advanced threats include zero-days, where exploitation comes from previously undisclosed vulnerabilities. Zero-days are highly longed for by nation-states as well as organized crime groups, since these open access to silent intrusion without triggering legacy defensive mechanisms [14].

Credential-based attacks such as brute-force and credential stuffing are common as well. They utilize lists of credentials gathered from past compromises. Automatic tools like Sentry MBA or OpenBullet are used by adversaries to attack multiple services by submitting numerous usernames and passwords, exploiting like that the password reuse across websites [42]. This was a successful strategy during the case of the LinkedIn breach, where hashed passwords were weakly hashed, allowing decrypted

credentials to spread far and wide and be reused by third-party attacks [26]. The Marget Genesis specialized in brokering such compromised credentials and browser fingerprint information, allowing ease of login without triggering an alarm [43].

Insider threats represent another vector through which information is compromised. Resentful staff and contractors can intentionally publish sensitive information, while benevolent insiders can unintentionally publish data through improper configurations or lack of vigilance [44]. The Cambridge Analytica–Facebook situation demonstrated how internal access, though properly authorized, can be leveraged to carry out bulk data scraping without typical "hacking" [31]. It has become more challenging to ensure protection from insider threats, requiring companies to set up monitoring of activity and implement the principle of least privilege across all access levels [44].

Supply chain attacks have risen significantly in recent years. Instead of attacking a victim directly, attackers gain access to third-party vendors or service providers already having trusted access to a victim organization's network. The compromise of SolarWinds, during which malware was implanted in software updates, illustrated the wide reach of this kind of strategy. This technique allows attackers to bypass even robust perimeter defenses, affecting both government agencies and multinationals. As software supply chains are so deep-seated in modern IT infrastructure, this vector remains difficult to neutralize [45].

Watering hole attacks and "living off the land" (LotL) are also used to evade detection. In watering hole attacks, attackers compromise regularly visited websites of potential targets and inject malicious scripts in hopes of taking advantage of visitor browsers [46]. LotL methods, by contrast, utilize native system tools like PowerShell or WMI to conduct malicious functions without writing additional files to a system [47]. This makes detection by antivirus and endpoint software much more challenging.

After accessing data to be stolen, attackers proceed to exfiltrate data. Protection of data, along with stealth, is a top priority at this point. Sophisticated attackers first encrypt data on-premises—before exfiltrating from the victim's environment. 7-Zip or WinRAR tools are used in target environments for compression and encryption using robust algorithms like AES-256 [48]. Public-key encryption, mostly PGP, is employed in more sophisticated attacks to ensure data can only be decrypted by an attacker possessing the correlated private key [49]. It’s a measure to ensure that once data has been intercepted, or recovered from seized attacker infrastructure, investigators are unable to make any use of it.

After on-premises encryption, data are exfiltrated over secure, anonymized channels. HTTPS/TLS tunnels make data transfer indistinguishable from legitimate HTTP/HTTPS, bypassing most network security measures even those that perform deep packet inspection [50]. Some attackers also make use of VPN tunnels to exfiltrate data across multiple geographic zones, adding extra layers of anonymity. More sophisticated attackers install clients of Tor on a victim machine and exfiltrate stolen data using the Tor network, using onion routing to hide the source, as well as destination [6]. When secret communication is critical, attackers may opt to use DNS tunneling, an encapsulation technique of exfiltrated data in DNS queries [51]. It is difficult to detect, especially in configurations where outbound traffic monitoring is lacking.

Besides encryption and anonymity, attackers also utilize various means by which they evade data loss prevention tools and intrusion detection mechanisms. Among these means, reducing data transfer rate to mimic legitimate user behavior, splitting file contents into chunks, as well as concealing data in images through steganography, are included [51–53]. Through these methods, attackers can exfiltrate bulk data secretly, undetected, over days or weeks, yet still achieve what they set out to do.

After successful exfiltration, secure data dissemination takes precedence. The attackers split up the stolen data into various encrypted chunks and upload them to several cloud storage websites like Dropbox or Mega, or private servers through SFTP [54], [55]. Only trusted collaborators or technically affiliated groups are provided with decryption keys [56]. These data repositories are typically hosted on anonymized infrastructure, often within the Tor network, and configured with authentication controls to prevent external access and forensic examination [57]. In some instances, redundant upload paths are used to ensure persistent availability, even in case of disruption.

Sophisticated attackers extend their operational security further by introducing deception and obfuscation strategies to mask the origin of stolen data. They may blend real data with fake records, shuffle entries from multiple breaches, or embed decoys to disrupt attribution efforts [58]. These methods not only complicate forensic timelines but also enable attackers to keep possession of clean datasets until further use or handoff is operationally viable. Similarly, coordination between different roles—such as credential harvesters and exfiltration specialists—is done via secure, encrypted messaging protocols and isolated communication platforms, ensuring role compartmentalization and anonymity [59].

Ultimately, these advanced breach operations reflect not only technical precision but also long-term planning and risk awareness. Adversaries do not merely seek access; they aim to control every phase of the data theft lifecycle with confidence. Through multilayered encryption, covert exfiltration, redundant dissemination, and structured compartmentalization, attackers maintain a high degree of stealth and resilience. This enables them to persist even against modern defensive strategies, laying the foundation for the eventual monetization and external exploitation of the stolen data, as will be examined in the following section. Section 2.4 will explore how the exfiltrated data transitions from a secured asset into a commercial product within dark-web marketplaces.

2.4 Role of Breached Data in Dark-Web Marketplaces

After successful cyber exploitation, the exfiltration of the information is just the beginning of an extended chain of exploitation. The seized information is not dormant but enters an underground marketplace where it is exchanged with incredible efficiency. In its current form as a digital asset, the exchange has become a worldwide business performed by actors with anonymity, technical skills, and financial motivations [60], [61]. It begins with illicit access and progresses toward a legal resale process with compromised information as a digital good. Cross-border commerce is enabled by cryptocurrencies, anonymizing tools, and escrow services in a scenario with no requirement for identification and reliance on one's identity [62]. This section explores how compromised information becomes part of this exchange system—how it is valued, posted for sale, sold, and distributed throughout dark web marketplaces.

Dark web markets serve as distribution channels for filtered, repackaged, and enriched information. Unlike dumping sites that appear indiscriminate or breach forums, markets provide productized offerings in a presentation that is nearly indistinguishable from legitimate internet retail [63], [64]. Merchants maintain storefronts with buyer reviews, product descriptions, refund policies, and account history. Listings are organized by type of breach, date, and locality of breach, with prices determined by demand and risk [60]. “Fullz” packages of names, addresses, Social Security numbers, birthdates, and financial information have high value. They enable the creation of synthetic identities, fraudulent loans, and account takeovers. Region and verification level determine the price of fullz between $30 and $480 [65].

The freshness of a dataset is also a matter of value. Buyers require recently obtained data as they presume that the same is not reported and hence it is more exploitable. Sellers label their dumps as “fresh dump” or “exclusive batch,” occasionally with timestamps and/or breach source markers to create a sense of urgency [68]. Freshness is traded with graduate releases. Sellers offer private buyers or subscribe to public members early access before wider resale is possible for a lower price. By the time the mitigation is revealed, and users have changed credentials or locked accounts, the value—and utility—of the data is reduced. Still, older datasets find their usefulness through high-frequency attacks such as credential stuffing or identity farming but with far lower value [67], [62], [65].

One of the main components of the operation of such underground markets is the establishment of trust. Most transactions are anonymous, and therefore vendor reputation becomes the de facto indicator of reliability. Markets grade vendors on transaction fulfillment, buyer satisfaction, and frequency of disputes [63]. Top-scoring vendors receive perks from the platform, including front-page listing, reduced fees, and faster payments. Established vendors employ consistent branding across sites, such as the same pseudonyms, support channels, and visual style [60]. Such consistency signals create loyalty among their customers, particularly when they exchange high-value or custom data.

To prevent fraud, most dark web markets use escrow services. Money—in the form of Bitcoin or Monero—is transferred by the buyer into an escrowed account controlled by the market until the buyer confirms receipt and legitimacy of the purchased dataset [62], [61]. In case of dispute, mods in the marketplace act as mediators, refunding or releasing funds based on evidence and vendor history. Escrow is most important in high-value transactions, e.g., corporate access credentials or zero-day exploits. In some marketplaces, escrow contracts allow multi-signature logic so that all parties involved must agree before funds are transferred [68].

Marketing practices are very sophisticated. Vendors usually sell offerings in tiered packages: raw data in entry tiers, metadata-rich packages at a premium, and fully supported “enterprise kits” with tools or manuals [63], [66]. Charm pricing, for example, USD $99.99 instead of $100, is practiced. Loyalty discounts, referral rewards, and promotional events are also practiced. Some vendors sell “sample packs,” which allow potential customers to verify the validity of a breach before purchasing in full. Such samples usually contain redacted information up to a point or one-time use logs. Some offer time-limited “drops” of highly targeted material, for example, credentials of senior executives or defense contractors, to buyers seeking targeted impact [61].

Segmenting buyers increases sales and presentation of the information as well. Different buyers have different kinds of information that they want and different objectives. Some of them are financially motivated attackers who buy credentials for conducting fraudulent activities or emptying accounts. Some buyers are attackers who buy access to company VPNs and email domains to move laterally. Others resell the information to different organizations, creating layered marketplaces and echo economies. Sellers categorize the offerings accordingly by type (i.e., education, health care), location, or level of access [60], [67]. For example, listings that offer business email logins usually have internal documents or login instructions attached.

One of the most significant changes of the last few years has been the bundling of exploitation tools with raw data. Vendors increasingly offer OTP bypass tools packaged with session cookies, bank account credentials, or even device fingerprints [66]. Corporate access credentials get sold with VPN profiles, email inbox pictures, or stolen documents. These add-ons make the datasets usable immediately, especially for less technical buyers [63]. Some sales even offer auto-login scripts, phishing templates, or how-to guides for credential testing. The outcome is a commodified breach chain—raw

data supplemented by automation and instructions—to enable less technologically proficient buyers to execute advanced breaches with minimal prep [69].

One of the distinguishing features of pricing is the element of exclusivity. While certain vendors make datasets publicly available and in wholesale quantities to various marketplaces, others create the impression of uniqueness by marketing that the data have never been advertised for sale before. These "exclusive batches" are marketed as untapped, providing the buyer exclusive access for a certain amount of time. Exclusivity is a high-price tag—often more than double the base price [60], [65]. In a few cases, the data even gets rented out with short-term access provided as a contract-like offer for special buyers. This lures the threat actors who don't want to share the exploitation paths with anyone and gain a competitive advantage when conducting targeted attacks.

Another trend on the rise is the integration of automation into resale. Sophisticated sellers now employ automated storefronts and chatbots, which respond to buying questions, generate invoices, and transmit data sets in a few seconds of payment [61], [66]. Overhead is removed from the seller, but convenience is provided for buyers operating across time zones. Automated breach testing functionality also is provided by some sellers—an automated script or dashboard that allows the buyer to query the information against live environments. These customer self-service features solidify the illicit trade's professionalism, effectively commoditizing what started as a manually managed, forum-based process as a streamlined consumer experience.

Payment is primarily crypto-focused. Bitcoin remains the most used due to widespread acceptance and exchange support. Monero is used when anonymity is desired, though, due to inherent transaction obfuscation [67]. Stablecoins such as USDT or DAI are used more for massive transaction volumes or by sellers that avoid volatility. Platforms prefer to display prices in USD or EUR and convert payments into crypto in real time. This keeps the seller's revenue isolated from exchange rate issues and offers predictability for buyers.

Marketplaces also evolve because of enforcement. After takedowns of the likes of AlphaBay, Wall Street Market, or Hansa, traffic quickly shifts to successor sites, or encrypted chat forums [63]. Telegram, Tox, and Matrix-based groups themselves turn into marketplaces. Some vendors disappear from platforms completely, operating private Tor-based shops or invite-only channels [61]. These shops maintain their own escrow, verification, and support and sometimes demand buyers undergo a vetting process or provide a prior transaction history. Sellers who expect takedowns to occur maintain mirrors and backup shops.

Operational adaptation extends even into listing tactics. Actors control supply in ways that influence prices, generally by releasing them in controlled quantities [60]. Some delay the public release of a breach in order to not alert the victims or draw the attention of law enforcement. Others schedule releases in periods of high demand—or high-profile attacks—or when something more geopolitically significant is underway.

Such dynamics affect listings themselves. Some vendors repackage aged data with falsified attributes in an attempt to provide added value. Others integrate breached information with enrichments that originate from OSINT with work history on LinkedIn or social media usernames [69]. Such enrichments make the information actionable and high-end prices are demanded from them.

Interestingly, nowadays most of the vendors offer the services of data-by-request. Instead of waiting for the breaches to occur, clients contract vendors to attack certain domains or organizations [66]. Commissions might be in the shape of account types, platform names, or job titles of the workers.

Meanwhile, vendors deploy credential harvesters, infostealers, or customized phishing campaigns according to the request. Data is resold when received but only to the original buyer.

As the competition increases, vendors gain differentiation based on infrastructure. Mature vendors provide support channels, carry out automated delivery robots, and provide real-time feedback channels. Some vendors provide buyers with dashboards for keeping track of the status of orders, messaging on safe channels, or re-downloading previous buys [63]. Such functionality brings vendor businesses up to the level of full-service providers and propels the dark web economy towards vertical integration and specialization.

Commoditization of stolen data is no longer a question of direct information exchange between buyer and attacker. It has become an established business process with inventory, marketing strategy, and post-sales support [61], [62]. The platforms on which this business is carried out have become robust infrastructures, not easily disrupted and supported by strong liquidity. Merchandise is not being sold based on content alone but on timing, risk, and useability. Buyers choose providers not only for availability but also for reputation, completeness, and add-ons.

The following is a summary table of the most common categories of data exchanged in dark web markets based on a recent in-depth scholarly survey. It uses approximate price ranges and the typical use cases of each category of information. The values were based on a systematic survey of active underground listings as described in the paper CODASPY 2024 [65]. The estimates capture not only the inherent sensitivity of each category of data but also the factors such as criminal demand, risk, and value that can be exploited in pricing. Overall, the table gives a glimpse of the dark web's business model of compromised information.

Table 2.2: Common Data Types Sold in Dark-Web Markets [65]

Item / Service	Price Range (USD)	Attacker Use-Case
Fullz (< 3 K balance)	$4 – $40	Online fraud, gift-card purchases, or low-value laundering
Fullz (> 3 K balance)	$30 – $480	Large-scale financial thefts and resale
Fullz/Account (random balance)	$1 – $30	“Lottery” purchases of second-hand bundles hoping for high-value finds
Account (< 3 K balance)	$1 – $33	Entry-level account takeovers for small transfers
Account (> 3 K balance)	$30 – $500	High-value laundering, crypto exchange, or direct cash-out fraud
Bank statement	$12 – $25	Stand-alone statements for KYC bypass or account hijacking
Outlook account	$110 – $590	Intercept password resets, bypass email MFA, and access linked services
Yahoo account	$110 – $590	Intercept password resets, bypass email MFA, and access linked services
Gmail account	$150 – $800	Intercept password resets, bypass email MFA, and access linked services

10 random emails + cookies	$1 – $10	Bulk email-account takeovers plus session cookies for automated attacks
SIM swapping service	$80 – $150	Intercept SMS-based 2FA for account hijacking and password resets
Data breach (< 6 months old)	$15 – $50	Bulk PII for phishing, credential-stuffing, and synthetic-identity creation
Data breach (> 6 months old)	$3 – $20	Bulk PII for phishing, credential-stuffing, and synthetic-identity creation (older, discounted)
N-S-D – address, phone, email	$8 – $10	Bypass PII-based authentication in credit/loan applications
N-S-D – credit report	$10 – $17	Bypass PII-based authentication in credit/loan applications
N-S-D (used)	$4 – $5	Second-hand Name/SSN/DoB bundles at bargain prices
Passport (scan)	$15 – $2,000	High-trust forged ID for international fraud and regulated-service enrollment
Driver license (scan)	$8 – $550	ID verification for new accounts and KYC checks
ID card (scan)	$8 – $550	KYC circumvention for account creation
Resident permit (scan)	$15 – $25	Alternative to passports/driver's licenses for identity fraud
SSN card (scan)	$5 – $25	Synthetic-identity creation and Social Security verification
Credit card (scan)	$7 – $10	Cloning physical cards or deep-fake KYC
Utility bill (fake)	$8 – $12	Proving fraudulent residential addresses for new accounts
Background report (fake)	$5 – $12	Employment and criminal-history dossiers to bolster synthetic identities

What we observe in these markets is an entire criminal economy founded on the organized recycling of stolen information. Each sale, transaction, and reputation factor is cycled back into a marketplace system that rewards efficiency and flexibility. The data theft is only the beginning. It gains value when the information is parsed, packaged, and used in commercial channels that create additional intrusiveness, profit, and exploitation.

Chapter 3: Dark Web

3.1 Surface, Deep, and Dark Web Comparison

The internet is often considered a single entity, but it is a multi-layered system consisting of distinct zones of operations. The most extensively documented of these is the surface web, a layer made up of any information sought by web searches and accessed by standard web browsers [70]. Although the most visible of the layers, it represents only a small portion of the internet. Subordinate to it are the deep web and the dark web—segments invisible to web searches and requiring special conditions or technologies to be browsed. Each layer performs a different role in the nature of how information is stored, accessed, and concealed, of particular interest to secrecy, institutional control, and cybercrime. This section examines the structure and purpose of each layer, highlighting the differences responsible for their role in online communication and digital security.

To aid in the visualization of these layers, the "iceberg" analogy in Figure 3.1 illustrates the small, above-water tip as the Surface Web (≈ 4%), the enormous underwater majority as the Deep Web (≈ 96%), and the deepest, darkest regions at the bottom as the intentionally hidden Dark Web. This figure not only represents the relative sizes of each layer but also the different degrees of accessibility and surveillance.

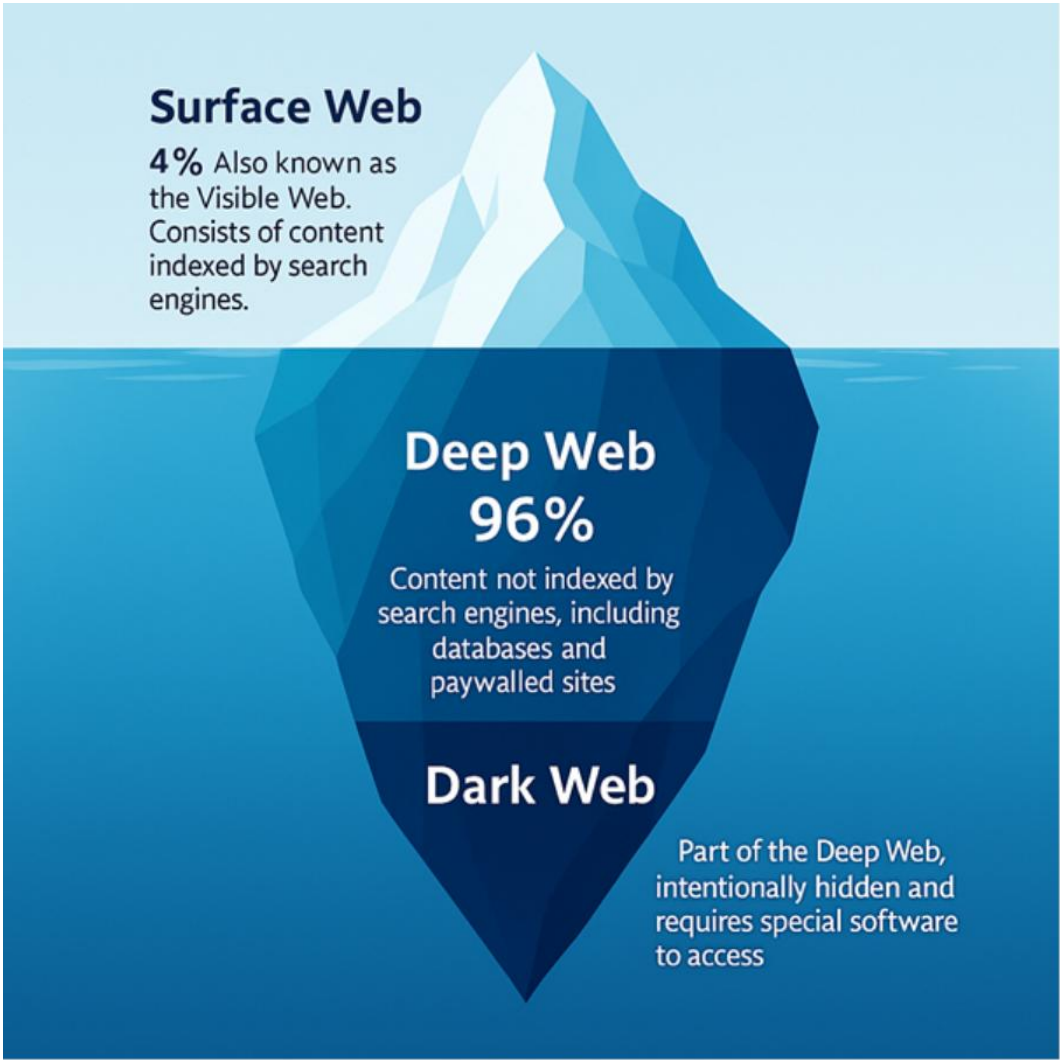


Figure 3.1: Iceberg infographic of Internet layers

The surface web consists of publicly available web pages. Search engines crawl them and are accessed through keyword searches. Blogs, news websites, social media profile pages, and retail web pages are some examples. The characteristic of openness—anybody can see these web pages anonymously without authenticating or using special software—defines it. Most of them run over HTTP or HTTPS and are served on traceable servers with exposed domain and IP information. All services in this layer are monitored and regulated. National laws, terms of service, and compliance frameworks such as GDPR or copyright regulations control the contents. Nevertheless, the surface web comprises not more than 5% of the total internet content [70], [71].

The deep web includes all the web content which isn't crawled by the search engines. It differs from the surface web by having pages that require login credentials or session-dependent querying. Webmail, bank pages, university portals, and paid libraries are some of the typical examples. The deep web contents aren't hidden for secrecy but for access. These systems make use of structured databases, back-end script, or API-dependent calls to make the contents dynamic. Authentication tokens and robots.txt exclusions are employed by them to disable the crawlers [72]. The deep web comprises a lot of institutions' inner workings and leads the charge in how personal information is controlled.

People use standard browsers to access the deep web. No special tools are used—only authorized credentials or institutional membership. This level of security relies on session authentication, role-based access control, and sometimes multi-factor authentication. All access attempts are logged, and most services are subject to legal compliance. It is not indexed, but can be internally monitored [71], [72]. Practically, this makes the deep web secure and fully part of everyday life despite the majority of the population being unaware.

On the other side, the dark web is a hidden component of the deep web. It has intentionally anonymized information and can be accessed only with the help of special software. The best-known one is the Tor browser, which routes the traffic of the user through a series of relays and applies multi-layered encryption to hide the server location as well as the identity of the user. Tor websites use the .onion domain suffix and are unreachable through standard browsers. Alternatives, for instance, I2P and Freenet, offer similar hidden services. These networks were initially developed to protect anonymity and resist censorship but now are used mostly to enable illegal markets and secure communication for criminals [70], [71].

The key selling point of the dark web is its invisibility. Dark websites aren't just not listed in searches—they're made to be undetectable. DNS resolution does not apply, IP addresses are hidden, and servers are often powered behind bulletproof hosting providers [71]. Advertisements for a few services are placed only in designated forums or directories. Invites, captchas, or time-limited access tokens are often required to gain access [73]. These security measures protect anonymity and guard operators from shutdown and tracking. Hidden services are transitory and decentralized, and URLs are rotated on a regular basis to avoid blacklisting or intrusion [74].

The dark web plays a dual role. On the one hand, it enables whistleblowers, human rights activists, and reporters to post information in repressive regimes. On the other, it hosts marketplaces, ransomware operations, and illegal file sharing. The same anonymity framework enabling freedom of expression is utilized to secure crime. Operations such as whistleblower websites are run over Tor by platforms like SecureDrop, while AlphaBay (defunct) used the same networks to exchange drugs, fake identity documents, or stolen data [71], [75]. It is a dual-use by nature, and the ecosystem presents serious challenges to law enforcement to distinguish legitimate from malicious traffic.

User behavior throughout all these levels differs significantly. Surface web users are consumers of public information and pay little attention to operations security. Their information is commoditized and monitored but rarely obscured. Users of the deep web operate under identity-verified controls, typically for functional reasons such as work or banking. Their activity is audited, authenticated, and monitored. On the dark web, anonymity is what the user prefers. Users exercise concerted efforts to keep their identity hidden—utilizing pseudonyms, encrypted channels, and security-hardened systems. Each user will possess various aliases, each distinguished by purpose [76]. Interaction takes place on the assumptions of mutual trust, typically built by reputation systems or cryptographical authentication.

Dark web websites utilize minimalist design and often put stability ahead of attractiveness. Slow and text-dense pages by design are common. CAPTCHA prompts, registration walls, and multi-step authentication are also used. Some pages can be seen only at certain times or after delay mechanisms [71]. Rather than utilizing simplified interfaces, survivability and resilience are favored by websites. Some are replicated to various URLs or supported by offline recovery copies [77]. If interrupted via law enforcement, backups can be restored quickly, or users can be re-directed to replacement domains. These operating patterns reflect the precedence of security at the expense of user experience.

Governance in these layers corresponds to different assumptions about the user's identity and compliance with regulations. On the surface web, content operates through public accountability. Domains are registered, hosts are traceable, and actions are subject to takedown notices, moderation, and lawsuits. Activity can legally be monitored by regulatory bodies, and the platforms are responsible for maintaining transparency, privacy, and security standards. On the deep web, it's restricted but still traceable. Logs are maintained, permission to enter is role-dependent, and security is defined by institutional policy. Access to the material is limited to legitimate participants only, but action in this layer remains legally responsible and frequently obliged to conform to industry-related regulations like HIPAA or FERPA [70], [71].

On the other hand, the dark web has no central government. Site administrators create the rules, and the enforcement mechanism is peer-to-peer rather than institutional. Sites are outside national standards, and services can disappear overnight or relocate. Conflict resolution occurs internally, most typically by administrators and moderators with no obligation to respond to questions of the law. Lack of legal jurisdiction creates a framework in which accountability becomes malleable. Trust is maintained by user reviews, pseudonymous reputation, or encrypted channels of authentication [70]. It is not an impossibility for law enforcement to reach the dark web, but it's hard to investigate and often it's hindered by the technological and cross-nation legal barriers [71].

A further disparity occurs in how information moves between these layers. Information in the deep web—such as an organization's internal databases or encrypted communications—becomes dark web information if it is exfiltrated by an attack. Stolen data gets repackaged, renamed, and distributed anonymously. It sometimes is sold again, sometimes leaked and sometimes used in larger-scale assaults. The surface web may only eventually hear of a breach after it has been partially exposed—through the media, dump indexing by a search engine, or notification from the authorities. Information's life cycle becomes a loop: secure in the deep web, extracted to the dark web, and potentially reappearing at the surface. This transfer illustrates the permeable character of cyber borders, with an incident ripple across all layers [70], [71].

Intent varies. Surface web users are end-users, generally ignorant of backend designs. Deep web users are authenticated users operating in controlled environments. Dark web users are anonymized participants who engage deliberately within concealed networks. Political, economic, ideological,

personal: these are perhaps their motivation. What they all have in common is a requirement to reduce exposure. The dark web requires technical knowledge to reach, and the risks are considerable—law enforcement surveillance, scam networks, and infection by malicious software. Therefore, its users are generally veterans of anonymity techniques. Their approach is a willing rejection of the mainstream web's dynamics. They operate in a digitally underground space not controlled by nation-states or corporate platforms but by interior logic, cryptography, and risk tolerance [70], [71].

Despite its smaller size, the dark web has an excessive impact. The deep and surface webs support the digital economy, scholarly communication, and institutional operations. The dark web however supports activities that conventional systems cannot accommodate legally. These include evading censorship, anonymous whistleblowing, and illegal trade [75]. It's an arena of experimentation, where new technologies, tools, and models of communication are tested outside the bounds of traditional control. This freedom to experiment unencumbered is both its strength and weakness. What begins as innovation can easily become abuse. While the surface web is an example of openness, and the deep web is an example of control of access, the dark web is an arena of obfuscation by design [70], [71].

The table below illustrates the major differences between the three layers. It determines differences in accessibility, visibility, usage scenarios, governance, and risk, framed within a paradigm that explains how the internet is operationally layered.

Table 3.1: Comparison of Surface Web, Deep Web, and Dark Web

Attribute	Surface Web	Deep Web	Dark Web
Indexed by Search Engines	Yes	No	No
Access Requirements	None	Authentication, Login	Specialized Software (e.g., Tor)
Typical Use Cases	Public Info, Media, Retail	Academic, Financial, Internal Systems	Censorship Evasion, Black Markets
Visibility	Full	Limited to Authorized Users	Intentionally Hidden
Legal Governance	Fully Regulated	Regulated, Restricted	Unregulated or Illicit
Hosting Infrastructure	Standard DNS, Clear IPs	Firewalled, Credentialed	Hidden Services, Obfuscated Routing
User Identity	Often Known or Tracked	Verified Internally	Anonymous or Pseudonymous
Risk Exposure	Low to Moderate	Moderate	High (Surveillance, Malware, Scams)

The internet's architecture is not defined solely by technical protocols but by the social and institutional purposes of its layers. Each level—surface, deep, and dark—prioritizes a different goal: discoverability, security, or secrecy. These not only define the way information is stored and accessed but also the way it's weaponized, concealed, or commodified. Information security professionals need to distinguish

between these layers. Confusing them risks mischaracterizing the threat, underestimating exposure, or overestimating control. Recognizing their structural and behavioral properties provides a more realistic map of the terrain—and of the vulnerabilities embedded in it.

3.2 Technologies Enabling Anonymity on the Dark Web: Tor, I2P, Freenet

Within the domain of anonymous online communication, there are specialized technologies upon which the infrastructure of the dark web rests. These technologies form the backbone of private, censorship-resistant communication systems by enabling users to remain hidden from normal surveillance authorities. Compared to normal internet services in which the location and identification of users may be easily determined by IP-level metadata, anonymity networks use complex routing topologies as well as cryptographic methods to hide such information. The three most widely adopted systems—Tor, I2P, and Freenet—represent different ways of anonymous networking. Each system utilizes a different model of data transmission, user privacy, and secure communication [70], [71]. This section describes how the systems work from a technical standpoint emphasizing architectural design, routing protocol, and operational nature.

Tor (The Onion Router) is the most developed and used anonymity network providing dark web infrastructure. Tor consists of a low-latency overlay network implemented via onion routing—a method where a message is encapsulated in a series of encryption layers per hop in a communication path. Once a user has selected a connection, a three-hop virtual connection is constructed by the Tor client by selecting a guard node (entry), a middle relay node, and an exit node. Each layer of the encryption is stripped by a relay in the path, which makes only the next destination in the path apparent and thereby prevents a single node from being able to associate the source with the destination [78].

One of the distinguishing features of Tor is being able to provide onion services, which make the servers and the clients anonymous in the network. Unlike normal web access is granted via exit nodes, onion services exist solely in the Tor network and have special .onion addresses. Onion services use a distributed protocol with introduction points and rendezvous points. To host an onion service, the server first selects a list of Tor relays as introduction points, signs their information with its private key and publishes the descriptor into the distributed hash table (DHT) of the Tor network. A client desiring to connect downloads the descriptor, establishes a connection over an independent rendezvous point and negotiates communication via circuits hiding the parties from each other [70].

Tor routing is coordinated by a network of trusted directory authorities. The authorities aggregate and publish information about all the active relays and their capabilities. Periodically, the client downloads the data to select suitable relays and establish circuits locally. Tor employs ephemeral session keys exchanged by a Diffie-Hellman exchange and thus offers forward secrecy for each session. All traffic is encapsulated in fixed-size 512-byte cells to prevent packet size analysis. With its layered encryption and random circuit routes, Tor provides strong unlinkability between the destination server and the user [79].

This process can be shown in Figure 3.2 where a user constructs a Tor circuit by encrypting a message in a sequence of three consecutive layers, one for each unique relay. The inner layer contains the destination and is encrypted for the exit node and encapsulated again for the middle and entry relays. While the message goes through the circuit, each node decrypts its layer, learning the next hop without being able to disclose either the source or the final destination. The layered approach achieves anonymity by separating knowledge across the path, ensuring that no single relay can link the sender with the receiver [80].

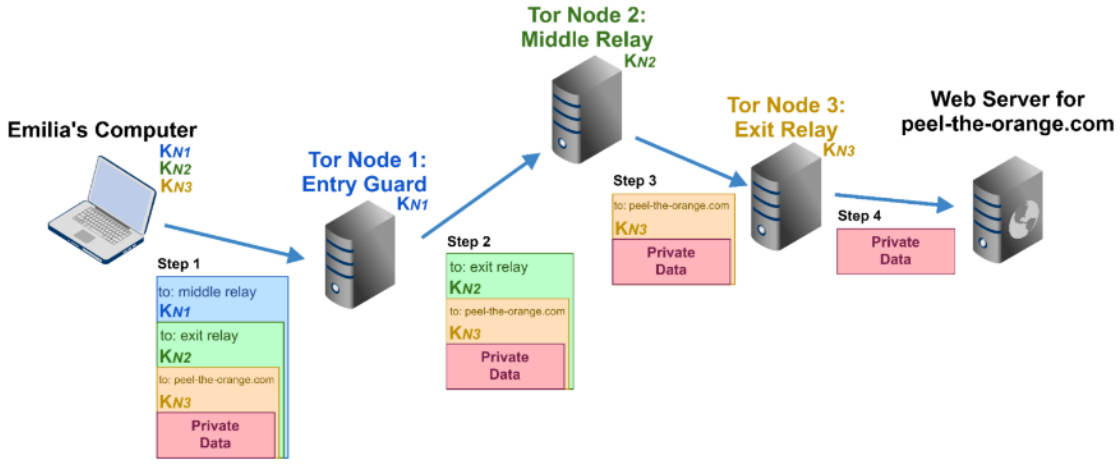


Figure 3.2: Tor Circuit Construction and Onion Routing [80]

While Tor can be used to access both the public internet and internal services anonymously, its design is particularly well-suited for interactive, real-time communication. Whistleblower websites, secure chat services, anonymous forums, and darknet marketplaces typically operate as onion services, offering anonymity to both servers and network clients. Tor supports low-latency traffic, making it ideal for browsing, chatting, and other applications that require nearly instantaneous communication. Its modular client architecture and integration into the Tor Browser make it simple to use and widely adopted, allowing individuals to benefit from anonymity without requiring extensive technical knowledge. Despite utilizing a semi-centralized directory infrastructure for coordination among relays, Tor has emerged as one of the most resilient and widely used tools for anonymized service usage and anonymous content publication [70].

The Invisible Internet Project (I2P) also serves as an alternative architecture to Tor in the guise of a peer-to-peer anonymity network that sustains itself. While Tor offers anonymous access to onion services and the Clearnet, I2P consists of an internal anonymous network of peers known as “eepsites” for the sake of communication. I2P has no exit nodes and consists entirely of its own routing domain. Such a design avoids potential leaks to the surface web as well as enhances resistance to destination-level traffic correlation [78].

I2P uses a one-way tunnel approach, meaning that communication between two peers happens through two separate tunnels—one for receiving messages (inbound) and another for sending messages (outbound). Each tunnel consists of a sequence of peers in the I2P network acting as a one-way router forwarding data in one direction. A full exchange of messages between sender and receiver involves four separate channels: two for each party’s incoming and outgoing communication. This design makes a single router unable to trace both the source and destination of a message even if it has been compromised [79].

Figure 3.3 displays how I2P separates inbound and outbound tunnels for communication to introduce high levels of anonymity. The figure portrays a client creating two separate tunnels, one output tunnel (green) and the other input tunnel (blue). There are several intermediary peers—named participants—used in each tunnel through which encrypted messages pass in one direction. No single peer knows the full route or again the parties communicating. The outbound tunnel directs the encrypted requests toward the destination (an eepsite), and replies get sent in the opposite direction via another distinct inbound

tunnel. By isolating these paths, I2P minimizes the correlation between where messages enter and exit and achieves high levels of anonymity for the receiver and the sender [81].

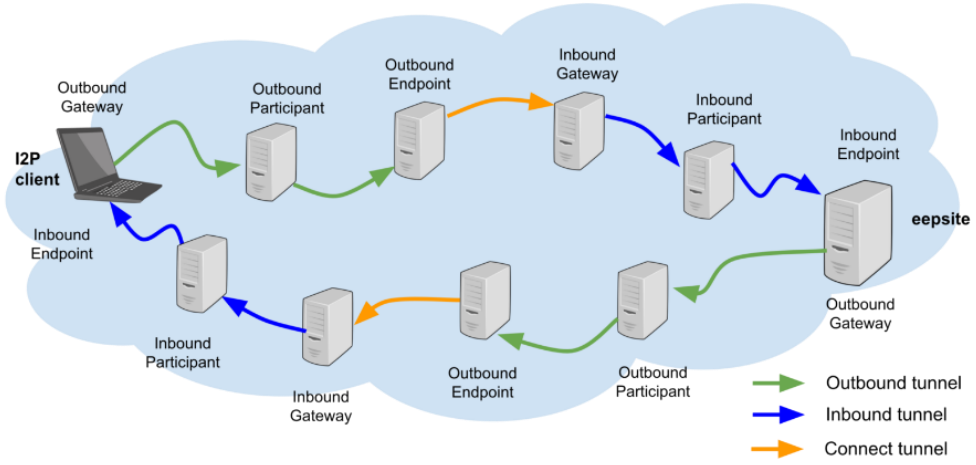


Figure 3.3: I2P Tunnel Segmentation and Peer Routing [81]

For peer routing and discovery, I2P utilizes a distributed database called NetDb based on the Kademlia distributed hash table (DHT) algorithm. Routers advertise cryptographic addresses and availability in the NetDb so peers can dynamically create tunnels without any need for central coordination. Routers efficiently perform routing by taking advantage of garlic routing, an onion routing-like system but with the ability to encapsulate a sequence of messages into one encryptable packet. This not only maximizes bandwidth usage but also hides inter-message relationships, thereby maximizing anonymity [70], [78].

Encryption in I2P is multi-layered and applied at multiple stages of routing. Tunnel segments are negotiated separately with each peer along the path and are encrypted using individual session keys. Garlic messages are further enveloped with an extra layer of encryption [70]. I2P also provides numerous internal services that anonymously mirror common internet functions, such as I2PTunnel (a SOCKS proxy-like facility), Susimail (a web-based anonymous email client), and I2PSnark (an embedded anonymous BitTorrent client). These services are components of the I2P router console, which runs locally on the user's computer and offers a web-based interface for configuration and monitoring [82].

While technologically advanced, I2P has expanded into use for anonymous publishing, chat, decentralized markets, and surveillance-resistant software. With its closed-loop design, external visibility is kept to a minimum, making I2P particularly resilient to filtering and surveillance. However, usability remains somewhat limited compared to Tor, as web technology integration often requires explicit modification. Nevertheless, I2P's strict internalization of end-to-end communications provides a high level of protocol-layer anonymity and makes it a feasible option for decentralized anonymous interaction within its network [78].

Freenet differs from Tor and I2P in being focused on anonymity based on data persistence and publication of content rather than real-time communication. It's a peer-to-peer decentralized data store in which users anonymously publish and consume information. Instead of facilitating direct conversations between two endpoints, Freenet serves as a distributed cache for documents and files, and it is appropriate for censorship-resistant publishing, whistleblowing, and permanent storage of protected content [70].

Freenet peers operate in either opennet or darknet mode. Under opennet mode, peers establish connections to a group of other peers in the network automatically. Under darknet mode, users establish trusted connections to known peers with an extra layer of protection against network penetration. Under either mode, each node in Freenet provides storage to store encrypted fragments of data, which are passed around the network by key-based content addressing. Each file added to Freenet is divided into many blocks, encrypted and assigned a unique key. The blocks are redundantly spread across a group of nodes following a heuristic request success rate routing algorithm so that nodes learn which peers will most likely possess specific information [70], [79].

Unlike Tor or I2P, Freenet does not use or require layer-by-layer encryption per hop. Instead, anonymity is achieved by routing forwarded requests and returned responses across random routes so that it's very hard to know which node forwarded a request or provided a particular file. Every node serves as a client and a server—forwarding and replying to content queries with no node in the chain knowing the data's starting or ending points [9]. The local caching of information on each node also achieves anonymity by buffering blocks of previously requested content so that requests for the same content can be satisfied out of local cache without network dissemination [70], [79].

Figure 3.4 illustrates a decentralized data routing model used by Freenet in which each node relays content requests throughout the network based on local routing tables and the outcome of previous attempts. Node A makes an original request in the diagram, which is forwarded by a sequence of peers before reaching a node with desired content (node D) or reaching a dead end. After reaching a dead end or failing to pass the request on, a node returns a “request failed” message upstream along the path and a routing strategy can be modified by higher-level nodes for future optimization. After being located, data returns via a possibly different path, like in the case D → E → B → A, such that a node will be unable to link source and destination [83].

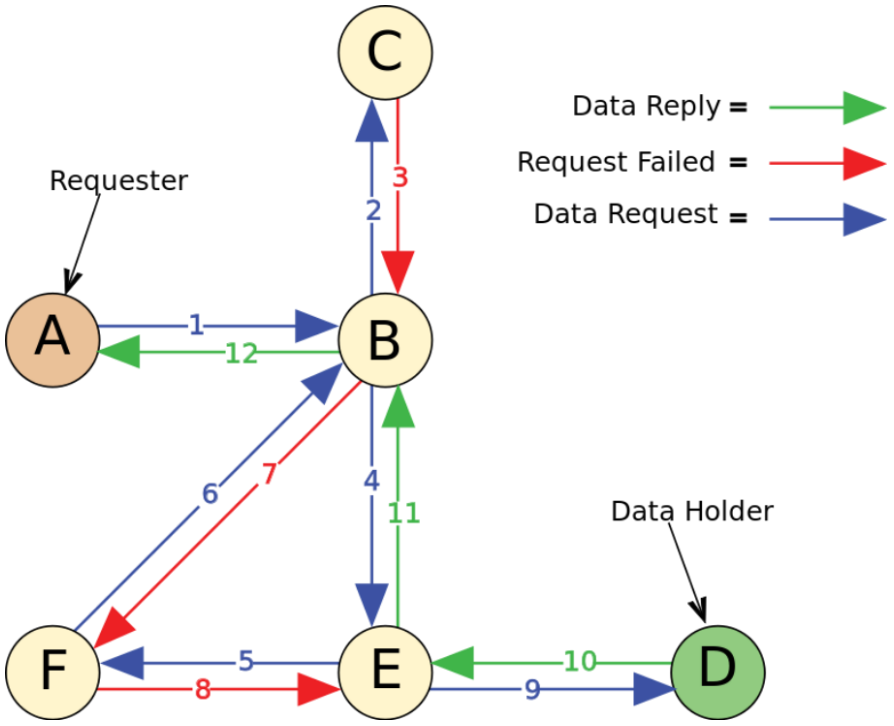


Figure 3.4: Freenet Decentralized Request-Response Routing [83]

Each of the three systems—Tor, I2P, and Freenet—are the technology basis of dark-web anonymous publishing and communication. Each of them embodies a different design priority: usability and latency

of traffic in Tor, tunnel-based segmentation of messages in I2P's internal-only mode of operation, and persistence and decentralized storage in Freenet. The different methods illustrate the diversity of models used in anonymity and how each one of them serves a different user need and operational restriction.

3.3 Types of Platforms and Services Hosted on the Dark Web

The dark web offers a sanctuary to a diverse and changing set of platforms that enable illicit commerce, cybercrime, anonymity, and digital dissidence. The platforms depend on privacy-seeking networks such as Tor and I2P and are engineered to support clandestine interaction, identity obfuscation, and evasion of legal discovery. Unlike the surface web and its centralized, law-abiding services, dark web platforms are typically decentralized, pseudonymous, and designed for takedown resistance. The set of platforms is extremely diverse in purpose, ranging from criminal commerce to shielded whistleblowing websites, and each is an essential element of the information leakage ecosystem, cybercrime organization, and subversive communication. This section outlines the most significant categories of platforms on the dark web, drawing on recent research and practical examples to highlight their contributions to the underground economy.

Illicit marketplaces are the most visible and economically important platforms on the dark web. The marketplaces are structurally similar and function in ways indistinguishable from surface web e-commerce platforms, offering interfaces for finding, buying, and rating products and vendors. These kinds of goods exchanged, however, are for the most part criminal or illegal in nature, ranging from narcotics, counterfeit currency, and malware bundles to fake documents and stolen credentials. The markets are designed to limit risk for both buyers and sellers, utilizing pseudonymous vendor profiles, cryptocurrency payment systems, and escrow services. Silk Road, recognized as the first successful dark web marketplace, established the foundational model for subsequent underground trading platforms and remains the principal benchmark for market scale and operational design [84]. Successor markets, such as Vice City and Incognito Markets, have adopted similar models of operation, including the mandatory use of PGP communication and Tor-based access controls [85], [86]. The business maturation of these platforms challenges traditional law enforcement abilities, with some marketplaces even providing customer support, quality ratings for vendors, as well as affiliate programs. Since the last few years, the number of transactions on dark web marketplaces has been on the scale of billions, with Hydra accommodating over $5.2 billion in transactions to date, including $1.3 billion in revenue alone in 2020—75% of all dark web marketplace revenue for the year [84].

Closely related to marketplaces are cybercrime forums, which are hubs of interaction, recruitment, and technical sharing between criminals. Less transactional and more community-oriented than marketplaces, users on these websites write threads, share hacking guides, promote or request criminal services, and establish reputational legitimacy. Entry onto these websites is typically gated by enrollment fees or invitation tokens to avoid law enforcement penetration [67], [84]. Dread, a Reddit-style forum available only on Tor, has discussion forums for market reviews, security tips, and buyer-seller complaints [87]. Another, Exploit.in, is a Russian-language forum where vendors sell zero-day exploits, rent malware, and trade intrusion detection system evasion techniques [88]. These forums make up the social layer of the cybercrime economy, allowing actors to network, cooperate, and achieve trust in a high-risk in an anonymous space.

Ransomware leak sites are a specialized and now widespread type of platform. They are maintained by ransomware-as-a-service (RaaS) groups and serve as coercive pressure in extortion campaigns. When victims fail to pay the ransom, they dump exfiltrated data on stages on these sites to impose public and

reputational pressure [89]. Groups like LockBit, Conti, and ALPHV maintain stand-alone leak sites on Tor, where compromised files are categorized by organization, leak date, and volume. The sites themselves may feature countdown clocks, sample file previews, and even chat portals for ransom negotiations. The LockBit leak blog, for instance, has a news website interface where information about high-profile corporate breaches is publicly searchable [90], [91]. These sites are located at the center of the business model of current ransomware gangs, positioning them as organized syndicates with penetration, encryption, negotiation, and publication departments.

Another important category of platforms includes whistleblower submission sites, differentiated from their criminal counterparts by an emphasis on ethical disclosure, transparency, and public accountability. They are used by journalists, NGOs, and citizens to obtain sensitive documents securely and anonymously [71]. SecureDrop is likely the most well-known of these platforms, which seeks to allow whistleblowers to contact media platforms via Tor with very little exposure of metadata. It supports anonymous two-way communication between journalists and sources and is now used by many major media outlets around the globe [75]. Another project, GlobaLeaks, is an open-source, customizable platform that facilitates anonymous reporting of institutional and governmental content. It includes client-side encryption, fine-grained access control, and metadata leakage protection [92]. While such platforms are sometimes manipulated for politically driven data leakage, they remain a crucial infrastructure for investigative journalism and civil society.

Another essential component of the dark web environment is the infrastructure and anonymity service providers. Such sites do not directly facilitate transactions but facilitate the hosting, protection, and persistence of criminal services. Some of the most fundamental services include bulletproof hosting providers, which provide server infrastructure in low-cooperation jurisdictions. Such hosting environments are designed specifically to ignore takedown requests, DMCA notices, or legal summonses, enabling darknet markets and ransomware sites to remain beyond the reach of traditional regulation [89]. Traditional examples include the now-defunct Daniel's Hosting, which hosted more than 7,000 hidden services at one stage before it was taken down [93]. Providers such as FlokiNET continue to provide high-resilience hosting, Tor exit nodes, and privacy-focused VPS services, specifically targeting political dissidents and controversial content creators [94]. Such services are necessary to provide uptime and anonymity over the most active nodes of the dark web.

Malware-as-a-Service (MaaS) platforms form another specialized category, allowing technically unsophisticated actors to conduct advanced attacks. The platforms offer ransomware builder tools, phishing toolkits, cryptojacking scripts, and other malware through an easy-to-use web interface. Payment is usually made in cryptocurrency, and service packages may include technical assistance, customization, and campaign metrics [89]. The most studied platform in the category is Genesis Market, which sold hijacked system browser fingerprints and access tokens to aid in account takeover attacks. Buyers were able to sort hijacked profiles by location, site, or session status, effectively automating identity theft [95]. Another trend widely observed is the pairing of MaaS offerings with access to botnets or infected endpoints to allow customers to spread malware at scale without owning their own hosting infrastructure [89]. These platforms blur the lines between service provision and organized cybercrime, adopting marketing campaigns, affiliate programs, and multi-level price plans.

Concurrently, encrypted communication services are the basis for coordination across these types of platforms. Cybercriminals require private channels through which they negotiate, operate, and avoid being watched [67]. In response to these needs, forums and leak sites incorporate secure messaging capabilities such as XMPP (Jabber) servers accompanied by Off-the-Record (OTR) encryption, or

instant messaging platforms such as Ricochet Refresh and Tox based on Tor [96 - 98]. Decentralized by nature, these resist surveillance by defying centralized storage and metadata gathering. Ransomware groups, for example, typically include Jabber IDs in ransom demands to enable direct victim-extortionist communication, while also providing fallback channels through .onion-based Webchat interfaces [99]. These platforms also enable vendor support across darknet markets, from conflict resolution through package tracking and transaction verification. Encrypted messaging is considered mandatory in operational security and is deeply ingrained in the entire scope of activity across the dark web.

Finally, data breach and exposure indexing sites offer operational and reputational services. Such sites collect, categorize, and publish enormous amounts of leaked data, typically stolen in ransomware attacks or platform breaches. Unlike attacker-controlled leak blogs, these indexing sites intend to make breach data searchable and monetizable [84]. IntelX, for instance, is an example of a search engine for darknet dumps containing password dumps, documents, and individuals' records [100]. Others offer services posing as legitimate breach notification sites but hosted on hidden services, offering subscription-based access to fresh data leaks. Both threat actors and researchers utilize such sites: attackers to monitor data availability and measure impact; researchers and defenders to track exposures and incident extent [61]. The utility of such sites lies not only in the short-term exploitation of the leaked data but in their long-term archiving capability, solidifying breach history past the lifespan of the original leak sites.

The table below gives an overview of the common types of platforms found on the dark web, their common uses, and at least one actual service of each category. The typology represents the structural heterogeneity of the ecosystem and is designed to facilitate comparative analysis of platform purpose, usage, and cybercriminal utility. All platforms included were mentioned in the above-mentioned scholarly literature or confirmed by official service documentation or Tor-accessible websites.

Table 3.2: Major Platform Categories on the Dark Web

Platform Category	Function	Representative Services
Illicit Marketplaces	Trade in contraband, digital goods, stolen data	Incognito Market, Vice City
Cybercriminal Forums	Community collaboration, knowledge exchange	Dread, Exploit.in
Ransomware Leak Sites	Data leaks, coercion via breach exposure	LockBit Leak site, Conti Leak blog, ALPHV Leak Blog
Whistleblower Platforms	Secure information disclosure from journalists, activists	SecureDrop, GlobaLeaks
Bulletproof Hosting Platforms	Resilient, takedown-resistant hosting	FlokiNET, Daniel’s Hosting
Malware-as-a-Service Platforms	Sale/rental of malware, infostealers for attacks	Genesis Market
Encrypted Messaging Platforms	Anonymous, encrypted communications	Tox, Jabber/XMPP OTR, Ricochet

Data Leak Indexing Platforms	Publicly searchable databases from past breaches	IntelX

The diversity of the underlying platforms reveals the complex architecture of the dark web. The dark web isn't a monolithic entity, it is a layered environment of mutually dependent services, each with a unique technical or economic purpose. Sites that enable commerce, data disclosure, secure messaging, malware distribution, or whistleblowing each contribute to specific operational needs within this covert economy. Their deployment across independent stacks and anonymizing networks maximizes their resilience to disruption and takedown. What is established is a compartmentalized but cooperative architecture—one that supports loosely affiliated actors to acquire services independently or in coordination, thereby maximizing specialization and the system's overall survivability.

Chapter 4: Tool Development (BreakChecker)

4.1 Design Goals and Use Case

In an era defined by the reliance on digital infrastructures supporting virtually all organizational operations, the cybersecurity discipline has evolved from an emphasis on defending discrete assets to an integrated understanding of how organizational infrastructures facilitate the exchange of information. Even with broad-based adoption of endpoint security and centralized monitoring systems, organizational security breach incidents remain rampant, and this isn't mostly because of new vulnerabilities, but because of the abandonment of publicly available information. The capability to identify openly accessible contact identifiers like telephone or email addresses—derived from an organization's web presence—often isn't sufficient. This type of passive exposure becomes an entry point for social engineering, impersonation, or phishing attacks. To deal with this active problem, BreakChecker was conceived not as a vulnerability scanner, but as a passive intelligence-gathering tool built to identify what is already visible, already accessible, and potentially already breached. This section analyzes the strategic motivation behind BreakChecker's development, its guiding constraints, and the practical environments in which it is deployed.

BreakChecker was created to address a common problem yet underaddressed: organizations all too often remain unaware of the data being published externally by their digital assets. The majority of standard breach studies mention technical vulnerabilities or examples of unauthorized access, while overlooking the likelihood that perpetrators will target data openly available on publicly published web pages. In most instances, inactive subdomains, expired contact portals, and embedded scripts remain for an extended period of time after active management of these components has stopped. Left undetected, these neglected items may include explicit references to employee email addresses, personal or departmental telephone numbers, or other easily vulnerable contact data that can be potentially abused by perpetrators as the basis of an extensive attack. BreakChecker was created to provide defenders with a low-risk, ethically scoped way to discover and map this exposure, starting not from leaked credentials, but from surface-level information accessible to anyone on the open web.

The BreakChecker approach is based on an exposure-based framework and places less emphasis on penetration testing. While contrasting most products, which scan systems to determine vulnerability, this one only uses published and easily available data and seeks to emulate the perspective of an outsider observer—namely, an attacker, an auditor, or an analyst—to gather intelligence without the need for authentication. The essential assumption is that exposure can be an existing security threat indicator regardless of access control circumvention. By analyzing published telephone numbers and email addresses openly exposed within an organization's online footprint associated with its registered domain name and correlating this data against known breach data, the technique allows one to develop an exposure overview for organizational entities and, concurrently, examine the correlation between one's exposures published and breaches elsewhere.

Another goal in the design was to enable analysts to carry out various professional responsibilities, achieving this goal without the need for intrusive tactics or higher access permission levels. Security engineers can utilize the software as part of internal risk assessments to determine if departmental websites or documentation portals unintentionally provide contact information. Incident responders can utilize it to identify associations between compromised data, like email addresses contained within breached collections of credentials and original sources, and thus better understand how attackers

originally obtained the data. Vendor risk assessment groups can integrate it with assessments of third-party services, particularly in situations where public exposure must be evaluated regarding aspects outside one's immediate control. Since BreakChecker's technique relies solely on passively observable and readily available web-based content, the use of the software can be extended across various scenarios without any contractual or regulatory limitations.

Among the reasons for making the tool's double-interface design—CLI and REST API—was the fact that various people and scenarios require various levels of automation and integration. A red teamer may like to always execute the tool from the command line as part of an extensive reconnaissance toolkit. A security operations center (SOC), on the other hand, may wish to integrate the tool as part of a system that automatically scans vendor domains for exposure or monitors changes over time. To manage both, BreakChecker was made modular at its fundamentals to share usage of the same scanning logic at both interfaces. This separation of logic from the interface enables diverse deployments of the tool—whether running locally, through CI/CD pipelines, or as a supporting microservice in the backend—without necessitating architectural rewrites.

Unlike traditional breach-checking tools requiring an existing email address or phone number as the foundation for analysis, BreakChecker follows an opposite approach. It starts with a root domain and performs methodical discovery to enumerate subdomains, crawl linked pages, extract visible contact identifiers, and determine whether those identifiers appear in third-party breach databases. The key change here is that this tool addresses an often-overlooked aspect of digital security: it broadens the focus from what has been compromised during an attack to include what has already been made public. By providing organizations with a tool to outline and verify such exposures, BreakChecker allows for proactive security management. It enables security teams to identify overly exposed contact information before it can be exploited as potential breach entry points.

At every stage, BreakChecker has uniformly made ethical considerations its primary concern. A carefully passive approach was employed to avoid the triggering of alarms, terms of service violations, or placing unnecessary loads on target systems. The program does not participate in authentication protocols, pull sensitive data, or mirror browser operations beyond what would be expected from an ordinary visitor accessing publicly visible websites. This makes it safe to use on organizational infrastructure, external vendor domains, or in controlled testing environments. This program's design adheres to professional standards of ethical reconnaissance and thus minimizes the chances of being inadvertently identified as malicious activity. This ethical framing is not only a legal safeguard but also a design principle that guided decisions throughout the tool's development.

Another real-world usage is noticed considering data protection and regulatory audit. In many legal cases, enterprises are compelled to carefully trace the various mechanisms by which personal data could be collected or retrieved through their websites. BreakChecker facilitates this procedure by providing structured results that identify sources of personal contact data and determine if these sources correlate to recorded security breaches. This capability supports both internal audits and external reporting obligations required for compliance. The system also provides supporting proof to show remediation steps took place to identify and address unnecessary exposure of personally identifiable information (PII).

BreakChecker is also useful in the context of due diligence and M&A (mergers and acquisitions). In order to determine the cybersecurity posture of an acquiring target, an organization must consider using tools that provide immediate and concrete feedback on the public visibility of the target being considered. BreakChecker enables expedited scanning to determine whether personally identifiable

information is publicly exposed to a prospective partner or acquisition target, thereby helping to mitigate reputational and regulatory risk. By feeding a domain into the tool and reviewing its output, analysts can flag risky patterns early in the process. This specific use case highlights the value of the tool both for technical security teams as well as for legal, governance, and strategic purposes.

In many threat intelligence methodologies, analysts would begin their assessments using compromised data as a method to determine an origin or prospective victim. In contrast, BreakChecker allows for the analysis of an existing domain in the reverse direction and, as such, sheds light upon whether an organization's presence online is growing or interfacing with compromised data. This reverse ancillary methodology helps with attribution and timely alerts regarding potential leaks that may be indirectly linked to an organization through publicly available contact points. In the case of breach records where specificity or anonymization isn't being undertaken, this contextual enhancement becomes essential to helping separate false positives from genuine risk. In turn, analysts can utilize this improved capability to systematically and completely correlate exposure with past breaches.

Design methodology takes interpretability as an important consideration. Despite a saturated market with numerous probabilistic assessments, BreakChecker aims to offer accurate and reproducible results. Each email address or phone number found can be traced back to its original source page, while each breach analysis is linked to a known third-party source. This amount of visibility allows end-users to confirm findings, question inconsistencies, and maintain trust in their evaluative processes. Whether used for proactive defense or post-breach analysis, the tool does not obscure its methods behind opaque metrics or heuristic weights. The results it produces are straightforward, appealing to those placing an emphasis upon traceability and accuracy. BreakChecker complements—not replaces—other security testing solutions by addressing a unique gap between passive domain enumeration and breach correlation, drawing attention to what is visible, accessible, and already at risk.

4.2 Technologies, Tools, and Dependencies

BreakChecker is developed upon a lightweight yet well-documented series of external libraries and configuration settings. Rather than running off a massive plugin-dependent system or machine-specific binaries, it takes advantage of Python-native packages with a dash of modern web automation frameworks to ensure portability and reliability. Through this design, BreakChecker ensures that it runs consistently across a myriad of environments -- from command-line running on individual workstations to headless microservice running within containerized infrastructure-- with no compromises towards accuracy. Through a series of asynchronous libraries, browser-emulation libraries, and specialized validation modules, the tool is highly responsive without compromising accuracy. This section provides an overview of its core dependencies, details how configuration is managed, and outlines how these elements contribute to its modular and reproducible nature.

BreakChecker's run-time dependencies are divided into two general categories. First are the core libraries for the scanning engine used for collecting, parsing, checking, and correlating exposed identifiers across sites. These libraries can be used to run the tool end-to-end with no support infrastructure. Second are application and deployment libraries needed to run BreakChecker as a long-term API microservice within Django using Gunicorn. While the core modules allow for one-time scans from the CLI, the API dependencies enable the use of BreakChecker in automated or real-time breach-checking workflows. Separating the dependencies shows that the tool differentiates its deployability interface from its scanning interface, while still being modular and testable.

Among libraries most fundamental utilized in BreakChecker is requests, a highly popular HTTP client for the Python language. Synchronous HTTP requests required for subdomain checking tasks, along with breach-checking requests, are provided with the dependency. For working with external APIs such as LeakCheck and HaveIBeenPwned, requests offers predictable handling for HTTP responses, network timeouts, along with JSON returns. Its combination of readability and reliability makes it a wise choice for interactions where concurrency is not critical, but complete and accurate data return is essential. Additionally, it is well-documented and widely used in the industry, making it an ideal choice for workflows centered around security, where reliability is essential.

For managing asynchronous data collection, especially in web crawling phases, BreakChecker uses aiohttp. aiohttp supports multiple HTTP requests concurrently using an extremely tight asyncio integration in Python. For web page crawling of potential contact information in huge numbers, aiohttp enables the application to have hundreds of open network connections concurrently, which improves speed and throughput rate greatly. Its design encourages scalability as well as responsiveness regardless of limited system resources or enormous data. In comparison to traditional synchronous approaches, aiohttp avoids I/O-bound activities like page retrieval from causing overall operation freeze, enabling faster, more dependable scanning action.

JavaScript-rendered pages pose problems for regular crawlers, and BreakChecker addresses this with its playwright support. This library enables headless browser automation, simulating real user interactions with full page rendering. Playwright is employed in capturing content that would be inaccessible otherwise behind client-side scripts to facilitate proper detection of embedded emails and phone numbers in dynamic content. The framework supports various browser engines and can execute in headless as well as full browser environments with huge flexibility when rendering. Its browser context isolation and strong automation controls make it especially well-adapted to simulating real user sessions while guaranteeing operational safety.

For parsing static and dynamically generated HTML content in Playwright, BreakChecker employs beautifulsoup4. This library allows for efficient traversal of document trees, searching for hyperlinks, and parsing out visible elements. Its lenient parsing semantics and decoding out for several encodings qualify BeautifulSoup for dealing with the kinds of noisy or mangled HTML for public websites. BeautifulSoup comes with several parser backends and integrates fully with regular expressions used copiously in BreakChecker for searching for patterns for email and phone numbers. Its convenience and adaptive API lead to a faster rate of development progress, with the readability of code maintained at a high level even after parsing poorly structured nested DOMs.

To validate that the phone numbers that are being retrieved are always consistently formatted and valid, BreakChecker integrates the phonenumbers library. A Google-maintained library built with an emphasis on parsing, validation, and formatting of national phone numbers, the library allows all matched numbers to be converted into a consistent local form based on regional conventions. This assists in preventing duplication in addition to enhancing the readability of outputs. Normalization must be done for dealing with varied forms of input—like “+44 7911 123456” versus “07911 123456”—that refer to the same number but appear differently across pages. Its complete country support and error handling make the library a suitable choice for validation and cleaning phases.

To accurately determine whether an extracted email belongs to the target organization, BreakChecker employs tldextract. This library cleanly separates a domain into its subdomain, registered domain, and public suffix, using the Public Suffix List for accuracy. It is essential for organization-scoped filtering, allowing BreakChecker to discard emails that do not belong to the intended registered domain while

avoiding false positives caused by shared suffixes or unusual domain structures. Its consistent parsing across a wide variety of domain formats improves the reliability of contact data correlation, especially in environments where subdomains and multi-level TLDs are common.

In order to validate email address structure and deliverability, BreakChecker utilizes the email-validator package. The package correctly validates syntax with respect to the standards of the RFC along with performing DNS-level lookups in an attempt at figuring out mail exchange (MX) records' presence at the same time. This two-step validation ensures that only legitimate and possibly reachable email addresses end up being considered for the process of correlation for breaches. False positive elimination, like superficially similar garbage strings that aren't real accounts, is ensured with the use of email-validator. Its lightweight design and ability to function without a network enhance the tool’s goal of remaining useful even in partially connected environments.

BreakChecker's application layer runs on top of the Django web framework, which allows the tool to expose its core functionality through a REST API. Django was chosen for being scalable, for naturally having security features built-in, and for an out-of-the-box integration with modular Python modules. Django in the BreakChecker application exists solely for microservice running reasons, with the user being able to start scans and get outputs via HTTP endpoints. Its routing structure and view handling provide a lightweight interface without altering the required scanning logic. Abstraction of this type keeps the backend logic clean, reusable, and independently testable.

To serve production-ready HTTP traffic for containerized deployments, Gunicorn comes packaged with BreakChecker for containerized deployments because Gunicorn is a WSGI HTTP server for running applications built with Python. While application logic is handled by the Django application package, Gunicorn is used for handling web requests concurrently and efficiently under load. It can scale horizontally with pre-fork worker models and is therefore adequate for bundling within an orchestration like Docker or Kubernetes. Packaging Gunicorn is necessary only for deployments where the API is exposed over the web and requires functionality beyond running in a CLI context. Its multiple-user operation at once capability enables integration for institutional or enterprise environments.

Besides Python packages, BreakChecker also incorporates high-performance external binaries for added capability augmentation within endpoint mapping with subdomain enumeration. The first one is Subfinder, a popular command-line tool developed by ProjectDiscovery in Go. It combines subdomains from numerous passive DNS sources and certificate transparency logs for a quick, all-encompassing snapshot of domain exposure. Its inclusion allows BreakChecker to initiate any type of scan from a more informed, accurate map of potential targets, providing a boost for follow-up analysis depth and accuracy. The architecture treats Subfinder as a plug-in component that enhances, rather than replaces, the tool’s built-in discovery logic. This approach ensures that core operations remain modular and maintainable, regardless of the level of integration.

Apart from these domain-specific utilities, BreakChecker utilizes the built-in libraries of Python for the system-level tasks at a foundational level. Modules such as os, re, argparse, and logging handle everything from path resolution and command-line parsing to regular expression matching and debug output. These libraries do not show up in install-time dependencies because they come bundled with the interpreter for the Python language but become an essential part of internal reasoning. This enables the tool to be light with minimal installation overhead but maintain the clear and modular flow. These modules, although non-user-facing, contribute significantly towards tool reliability and operational clarity.

To very briefly outline the structure of streamlined deployment planning and structured dependency, the following table provides a listing of all high-level needs by category. It specifies core libraries for validation and scanning, web deployment components used for the API version, and optional utilities who improve capability but can be absent at times. Version restrictions mirror corresponding lines in requirement.txt for compatibility across environments and reproducibility. This explicit perspective allows for a thorough description of external needs for BreakChecker for maintainability and transparency support.

Table 4.1: Core Dependencies and Tools Used in BreakChecker

Category	Dependency/Tool	Version	Purpose
Core Library	requests	2.32.5	Synchronous HTTP for API queries and certificate lookups
Core Library	aiohttp	3.12.15	Asynchronous page fetching and crawling
Core Library	playwright	1.54.0	Headless browser for JavaScript-rendered content
Core Library	beautifulsoup4	4.13.5	HTML parsing and element extraction
Core Library	phonenumbers	9.0.12	International phone number validation and formatting
Core Library	email-validator	2.3.0	Email syntax and DNS validation
Core Library	tldextract	5.3.0	Accurate email domain parsing for organization-scope filtering
Deployment Dependency	django	5.2.5	API server logic and endpoint management
Deployment Dependency	djangorestframework	3.16.1	Serialization and REST API scaffolding
Deployment Dependency	gunicorn	23.0.0	Production WSGI HTTP server for Django deployments
Optional Tool	subfinder	2.8	Subdomain enumeration from passive sources

These dependences as a whole form the technological foundation found for BreakChecker fulfilling its design requirement of being modular, reproducible, and transparent. The inclusion of optional tools such as Subfinder demonstrates a flexible approach to extensibility without imposing mandatory setup burdens. By maintaining clear boundaries between its scanning logic, deployment interface, and enhancement modules, BreakChecker achieves a streamlined architecture suitable for both individual analysts and large-scale automated workflows.

4.3 Architecture and Functional Flow

To ensure both functional usability and easy deployment, BreakChecker has been designed around a modular architecture that emphasizes adaptability and clarity. As before indicated, the tool itself isn't

designed as a vulnerability scanner but as a passive visibility enhancer—intended to shine light on what an organizational entity willingly reveals through its publicly available web assets. Towards that end, the system design avoids unnecessary complexity at every turn, opting instead for transparency, reproducibility, and concerns for ethics. Each component in the architecture serves a specific role within a linear yet flexible execution pipeline. Used either as part of private test environments or as part of complex automation configurations, the tool demonstrates its entire capabilities outright with minimal reconfiguration. This section provides a complete overview of the inner workings of BreakChecker, aided by one diagram that shows the system's core stages and the logical decisions that dictate its functioning.

Before performing an in-depth analysis of the tool's functionalities, it proves efficient to provide a graphical representation of its structure. The figure 4.1 presents a high-level flowchart of the tool's operational stages, starting from domain input through to breach-enhanced output, essentially highlighting branching logic, fallback plans, as well as verification procedures of contact phases. This picture acts as a supplementary guideline of the forthcoming discussion, linking every design phase of the system to some particular element of the visual model, hence providing an evident, traceable, and understandable system functioning overview.

The functioning of BreakChecker begins from a single root domain provided as input from the user. Such input may either be provided through the command line or its REST API, depending on the user's integration preferences. For purposes of supporting differently scaled deployments, the tool supports configuration of parameters through a config.json file or environment variables. Crawl depth, concurrency, as well as API keys, may be provided through either interface, hence ensuring that the tool performs well under interactive, containerized, or automated environments. This initial phase of setting up ensures that the tool runs based on the particular requirements of its environment while ensuring adequate versatility, thus eliminating the need for changes to the codebase.

Upon startup, the system proceeds from the subdomain enumeration phase. The Subfinder utility, as suggested, aggregates data from numerous public as well as passive sources of data. When Subfinder runs, it is used to extract all subdomains of the root domain. In cases where Subfinder isn't available or functional, by design, BreakChecker falls back to querying publicly available resources like crt.sh (Certificate Transparency), HackerTarget, as well as AnubisDB. This redundancy guarantees that the tool always has functionality, even in environments that are limited, while offering consistent user experience across deployments with different tooling availability.

Live subdomains are then filtered by probing for the existence of HTTP or HTTPS URLs, attempting HTTPS first and falling back to HTTP if necessary. Unreachable hosts are discarded from future analysis, improving efficiency overall and avoiding misapplication of resources at later crawl stages. This filtering step acts as a kind of quality control step that lies between the discovery and the collection phases, keeping only subdomains that are capable of yielding web content at the crawl phase. In addition, it ensures that later tasks—like endpoint extraction as well as Playwright rendering—are applied only where a meaningful user-facing interface is likely to exist, optimizing execution as well as improving results fidelity.

After filtering, BreakChecker uses its internal crawler to enumerate available URLs on each reachable subdomain. The crawler traverses pages within the configured crawl depth, discovering anchor links, embedded resources, and in-scope JavaScript asset URLs from each rendered page. By including both HTML and JavaScript references, it ensures coverage across static and dynamic site structures. The

results from this stage feed into the same pipeline for content rendering and identifier extraction, hence maintaining architectural coherence.

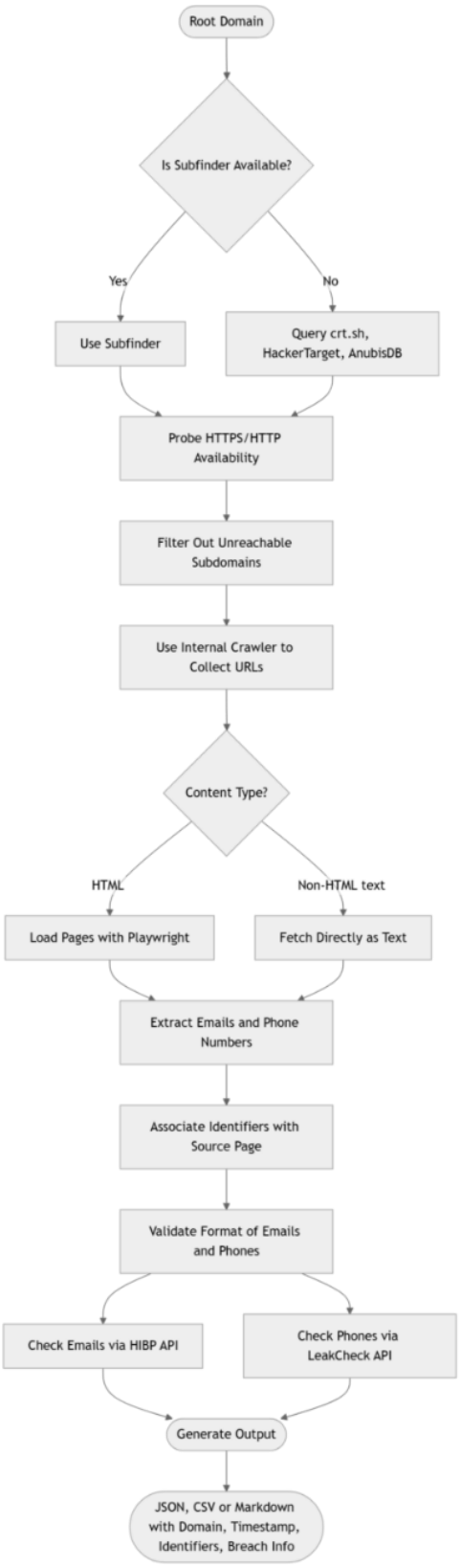


Figure 4.1: High-Level Flowchart of BreakChecker

All of the URLs gathered at the crawl stage go through a rendering stage based on Playwright for HTML content. Playwright functions by mimicking a headless browser environment, which is competent at navigating web pages that have content that is dynamically generated through JavaScript. It ensures obtaining contact identifiers from static HTML as well as page content that has been dynamically generated. Non-HTML resources are fetched directly when relevant. The system mimics actual web user browsing of the site, although without script injection or authentication, or any kind of privileged manipulation. This manner ensures that gathered data always accurately represents what is seen by an unauthenticated external observer over a real session.

For each of the rendered pages, the tool uses regular expression patterns as well as normalization methods to extract the observable email addresses and telephone numbers. Each identified contact info includes sources of URL from which they have been extracted, providing a traceable pointer back to the source. This pointer proves necessary to help provide context and validation, as security analysts are capable of effectively evaluating particular locations of sensitive data elements of the web design. Additionally, the extracting engine performs format validation to remove instances of false positives and poorly formed strings, which ultimately boosts the quality of the later analyses.

Upon verification and collection of identifiers, BreakChecker proceeds to the breach correlation stage. Email addresses are cross-checked against the HaveIBeenPwned (HIBP) API, while phone numbers are cross-checked against the LeakCheck API. Both of these third-party services offer breach-linked metadata, which includes breach names, dates, and source information. This phase links publicly visible contact data to known security incidents, enabling analysts to assess whether exposure is not only theoretical but already exploited.

The final stage of the architectural process refers to output generation. BreakChecker produces structured reports, which are available in various formats like JSON, CSV, or Markdown, depending on user requirements. Each output file includes key metadata such as the scanned domain, the timestamp of execution, the identifiers found, the breach records returned, and the source URLs of each match. This export capability in multiple formats aids compatibility integration with third-party systems used for ticketing, dashboards, or specialized IR platforms. The uniquely formatted outputs offer clarity as well as traceability, suitable for technical as well as non-technical stakeholders, thereby ensuring that the output easily integrates into rigorous compliance, risk, or investigative procedures.

In this holistic multi-stage pipeline, the design of BreakChecker skillfully balances completeness, efficiency, and ethical transparency. Every design decision—ranging from modular enumeration logic to passive rendering—has been made to ensure that people are able to use the tool responsibly on a range of applications. Used either as a standalone scan or as part of a CI/CD pipeline, BreakChecker produces predictable as well as repeatable results about domain-level exposure. This design structure is the technical backbone of its functionality, enabling organizations to gain actionable intelligence about possible leaks from public-facing infrastructure to potential attackers.

4.4 Usage Examples with Screenshots and Output Files

Use of BreakChecker in practice is provided through two main interfaces: a command-line interface (CLI) and a RESTful API endpoint. Both interfaces share the same internal scanning engine and were created to address the diverse needs of users in different environments. The CLI version is designed for

researchers/analysts working in terminal-based environments, offering support for user-customized runtime parameters and output format options. The API version is designed for use within automated installations, providing easy, programmatic access to the scan results without requiring direct user intervention. Both interfaces provide a JSON-structured report with all extracted domain metadata, contact points, as well as the relevant breach records (in the event of a match). The following section illustrates both interface usage within practical implementations, with an extensive description of the execution sequence, result structure, as well as sample outputs.

Prior to using both interfaces, there is a one-time setup to be run to activate the tool for complete functionality. That is, you must install all the Python dependencies listed in the requirements.txt file with "pip install -r requirements.txt". Users would also need to include a config.json file in the tool's root directory or set up appropriate environmental variables with the proper keys for the LeakCheck service and the HaveIBeenPwned service. Those keys will be utilized to activate breach validation on the emails and the phone numbers. Without them, the contact information would still be federated by the tool, but it would not do breach correlation and would flag those columns in the output as not available. Once these setup activities are run, the tool is functional within the CLI mode as well as the API mode with all functionalities enabled.

The CLI version is invoked by the break_checker.py script and accepts a list of switches that alter its run-time operation. The -d or --depth option defines crawl depth during page discovery, with three as a default when not provided. The -c or --concurrency flag defines how many simultaneous workers there are, and the user has the liberty to tweak performance and resource utilization at will. The -v or --verbose option specifies debug-level console output, exposing more of the scan activity. As far as output formats are concerned, the --json option instructs the tool to present results in JSON format; however, if no format flag option is provided, then JSON becomes the default. Other options like --csv or --md exist within the codebase of the tool to support different use cases. Once a scan is launched using a command like "python3 break_checker.py -d 1 k3ylabs.com", the tool performs domain validation, subdomain enumeration, HTTP probing, contact scraping, and breach correlation as discussed in section 4.3 before saving the result in a timestamped JSON report.

During runtime, the terminal prints live logs displaying the result and runtime of the tool. The logs provide the counts of subdomains detected, accessible hosts, emails, and telephony numbers detected on rendered webpages. Upon the completion of the installation of the API keys, the logs also display how many such contact points were detected within the public breach collections. The runtime execution is shown, for instance, in the terminal session for the scan on k3ylabs.com, presented in Figure 4.2. The output, as presented, provides a step-by-step account of the innermost process of the scan, concluding in a line displaying the subdomains, emails, phones, and breach hits total.

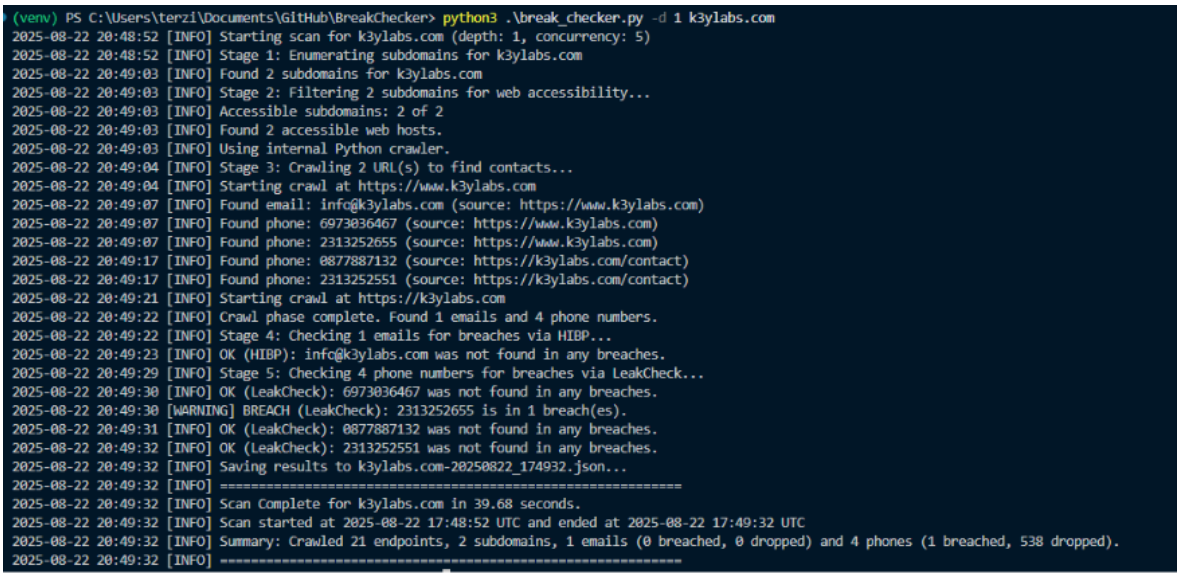


Figure 4.2: Command-line scan and real-time logs from BreakChecker

At the end, the utility outputs to a file with the name being the domain it has scanned plus the date and time, for example, in our instance k3ylabs.com-20250808_122649.json. The file consists of structured parts consisting of the normalized domain name, a list of subdomains extracted, emails, phone numbers, and breach details if known. Each contact item contains a reference back to the source URL along with a list of breach names where the information has been leaked. The format remains simple to import into dashboards, scripts, or SIEM systems to load the findings for further examination or to act on. Some of this output is shown within Figure 4.3, revealing the structure of the JSON with exemplary subdomains, extracted emails, and breach mapping.

The API version of BreakChecker offers an alternative method for initiating scans by accepting HTTP POST requests to a dedicated endpoint. This approach is designed for integration with automated systems, dashboards, or browser-based interfaces. It allows external applications to request full domain scans programmatically and to receive structured JSON responses that mirror the output of the CLI version. This mode is particularly useful in environments where analyst workflows are already centralized or when the tool must be embedded into a larger orchestration framework.

To start a scan via the API, a POST to the /api/scan/ endpoint is made, with the target domain and optional crawl depth being the JSON payload. For instance, submitting a request with the body {"domain": "k3ylabs.com", "depth": 1} initiates a scan of the domain k3ylabs.com with a maximum crawl depth of one. Without the depth, the system will default to an internally generated value, hopefully provided within the environment variables or the config.json file. The server first validates the domain syntax and format before it continues on with the scan action via the underlying functionality, identical to the CLI version.

```
{
    "domain": "k3ylabs.com",
    "scan_start": "2025-08-22 12:48:32 UTC",
    "scan_end": "2025-08-22 12:49:12 UTC",
    "scan_duration": 40.78951644897461,
    "summary": {
        "num_subdomains": 2,
        "num_endpoints": 21,
        "num_emails": 1,
        "num_phones": 4,
        "num_breached_emails": 0,
        "num_breached_phones": 1,
        "emails_dropped": 0,
        "phones_dropped": 536,
        "cpu_avg_percent": 5.51,
        "cpu_peak_percent": 38,
        "mem_avg_mb": 81.62,
        "mem_peak_mb": 100.75
    },
    "subdomains": [
        "k3ylabs.com",
        "www.k3ylabs.com"
    ],
    "emails": [
        {
            "address": "info@k3ylabs.com",
            "source": "https://k3ylabs.com",
            "breaches": []
        }
    ],
    "phones": [
        {
            "number": "0877887132",
            "source": "https://k3ylabs.com/contact",
            "breaches": []
        },
        {
            "number": "2313252551",
            "source": "https://k3ylabs.com/contact",
            "breaches": []
        },
        {
            "number": "2313252655",
            "source": "https://k3ylabs.com",
            "breaches": [
                "Romwe.com"
            ]
        },
        {
            "number": "6973036467",
            "source": "https://k3ylabs.com",
            "breaches": []
        }
    ]
}
```


Figure 4.3: JSON output generated by BreakChecker CLI scan

API service is installed locally using Docker for consistency and portability. The API is initiated by the user by going to the directory of the project on the command line and executing the command "docker compose up --build". The command initiates the Django-hosted backend, initiates the REST interface, and simply waits for incoming requests for the specified port to start the scan. The system is accessible from the outside right after deployment.

Figure 4.4 shows a standard HTTP POST request submitted through Postman to initiate a scan via the API. The request body includes the target domain and crawl depth, reflecting the same input structure used by the CLI tool to ensure consistency across usage modes.

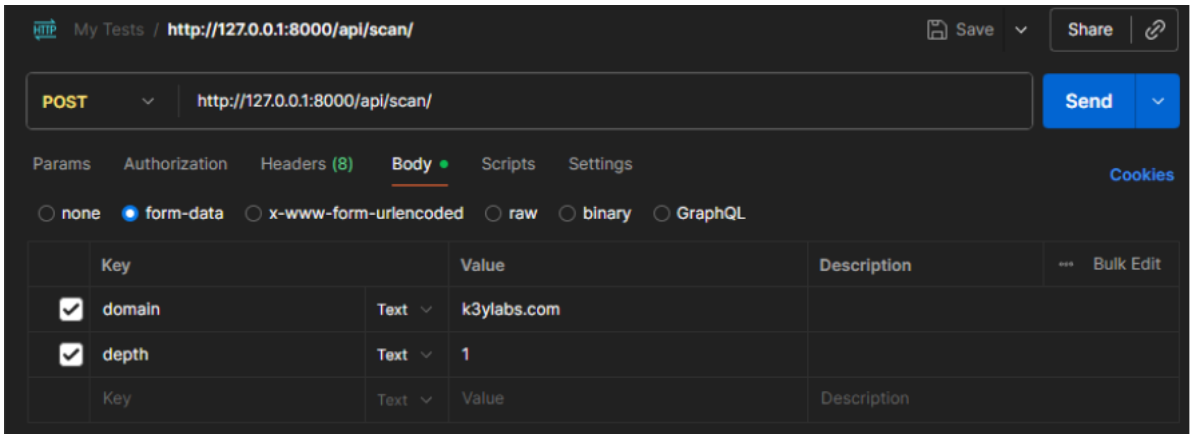


Figure 4.4: HTTP POST request triggering BreakChecker scan via REST API

The returned JSON payload includes both a high-level summary and detailed lists of the scan results. The summary field contains counts for subdomains, emails, phone numbers, and the number of breached items found in each category. The subdomain list is followed by a listing of all hosts that were found, while the emails and phones sections provide structured entries. Each entry contains the data point (e-mail address or phone number), the URL where it was found, and a list of breach names in which this item is exposed. That allows the API's consumers to process results in a structured, machine-friendly fashion.

An instance of a genuine JSON reply is displayed within Figure 4.5, showing how breach information gets correlated for certain contact records. From the figure, we obtain the source-domain email addresses along with breach metadata, so we can instantly gain insight into what assets require remediation or deeper analysis. Phone numbers get correlated similarly to the pages on which we observed them, along with breach records on which they happen to appear.

Both the CLI and the API interfaces call the same backend, so the format, the logic, and the scope of the scanning remain mostly the same across modes. The goal is to ensure that the reporting strategy remains consistent, making it easier to integrate BreakChecker into various workflows. Users can interact with the program either programmatically through the API or the command line. Regardless of the interface used, the program provides a clear, structured summary of the discovered contact information, along with any associated data breach occurrences.

```
{
  "scan_domain": "k3ylabs.com",
  "scan_start": "2025-08-18 14:11:14 UTC",
  "scan_end": "2025-08-18 14:11:54 UTC",
  "scan_duration": 39.89562726020813,
  "summary": {
    "num_subdomains": 2,
    "num_endpoints": 21,
    "num_emails": 1,
    "num_phones": 4,
    "num_breached_emails": 0,
    "num_breached_phones": 1,
    "emails_dropped": 12,
    "phones_dropped": 532,
    "cpu_avg_percent": 7.35,
    "cpu_peak_percent": 63.2,
    "mem_avg_mb": 82.95,
    "mem_peak_mb": 109.64
  },
  "subdomains": [
    "k3ylabs.com",
    "www.k3ylabs.com"
  ],
  "emails": [
    {
      "email": "info@k3ylabs.com",
      "source": "https://www.k3ylabs.com",
      "breaches": []
    }
  ],
  "phones": [
    {
      "phone": "0877887132",
      "source": "https://k3ylabs.com/contact",
      "breaches": []
    },
    {
      "phone": "2313252551",
      "source": "https://k3ylabs.com/contact",
      "breaches": []
    },
    {
      "phone": "2313252655",
      "source": "https://www.k3ylabs.com",
      "breaches": [
        "Romwe.com"
      ]
    },
    {
      "phone": "6973036467",
      "source": "https://www.k3ylabs.com",
      "breaches": []
    }
  ]
}
```


Figure 4.5: BreakChecker API response – email discovery and breach data in JSON format

Chapter 5: Tool Evaluation

5.1 Evaluation Criteria & Methodology

The evaluation of a security reconnaissance tool like BreakChecker requires a systematic approach that not only examines the critical functionalities of the tool but also explains the importance of each metric obtained in terms of its role in information security assessment. The objective of such an evaluation is to determine a systematic and reproducible basis for evaluating the effectiveness of the tool for different operational methods and scan results. By clearly connecting each measure with its related application in the extensive reconnaissance process, the evaluation ensures that the reader understands both the significance of the results and the reason for their choice. As a result, this section outlines the evaluation methodology, test domain selection criteria, implementation approach, and the used performance measures—explaining the reasoning for the suitability of each criterion in measuring the operational effectiveness of BreakChecker.

The chosen targets for testing were intentionally selected to demonstrate a range of complexity, size, and potential runtime impact. The least complicated target, k3ylabs.com, can be considered a simple test case, having minimal publicly visible content; as such, it is adequate for baseline testing where interference by unrelated endpoints may be minimized. In contrast, the second target, hackerone.com, presents a more complete and dynamic site with multiple subdomains as well as regularly updated content, thus presenting a more considerable challenge for BreakChecker's ability to handle redirection, dynamic rendering, and multiple HTML structures. The most complicated target, owasp.org, has a large number of subdomains, mailing lists, and documentation pages, thus providing a rich environment for endpoint discovery and identifier extraction. The categorization of these sites, from least to most complicated, adequately demonstrates the relationship between runtime and information quantity as it pertains to the target's size and complexity, while remaining within ethical scanning guidelines.

For each of the domains, two operational modes were tested: (1) the Command Line Interface (CLI) mode, running natively on the host machine, and (2) the Application Programming Interface (API) mode, running in a containerized environment with Docker Compose. Both operational modes had all relevant dependencies installed; however, the CLI version ran subdomain enumeration entirely with BreakChecker’s built-in API-based fallback mechanism, while the API mode allowed for the use of Subfinder for additional passive enumeration support. The following subsequent phases of the process—URL collection, rendering with Playwright, extraction of the identifiers, and breach correlation—were identical for both modes. The crawl depth and concurrency parameters were kept as controlled variables such that the impact of each on coverage and performance could be evaluated without any of the remaining settings changing.

All tests were run using the same hardware to prevent variations in the test environment. The tested configuration included a Windows 11 machine that had an AMD Ryzen 7 5800X central processing unit, 32 GB of DDR4 random-access memory, and an AMD Radeon RX 6800 XT graphics processing unit. Command-line interface executions were run natively, and application programming interface mode tests were run in Docker containers to efficiently mimic the remote deployment of services. Performance measurement was implemented using natively available system monitoring tools to measure command-line interface executions and via Docker system statistics to measure application programming interface tests. For all tests, the API keys were supplied, thus ensuring breach verification

was included in all relevant tests. All scan configurations were run three times, and the results were averaged to cover any effects of temporary network variability.

The performance evaluation focused on six major metrics: (1) the overall number of passive subdomains found, intended to measure the scope of discovery at the reconnaissance stage; (2) the accuracy of host reachability classification, used to measure the effectiveness of filtering by the tool in terms of reachable and unreachable hosts; (3) the overall number of unique URLs extracted, determining the scope of endpoint discovery; (4) the number of unique email addresses and phone numbers found, expressing the tool's effectiveness in identifying sensitive contact information; (5) the cumulative time in seconds from the start to the end of the scan, measuring operational efficiency; and (6) the accuracy of identifier extraction, verified by manually inspecting all results for smaller domains and a representative sample for OWASP.org due to its scale. Each metric was selected according to its ability to clarify a particular aspect of operational performance and collectively to present a full appreciation of the completeness, correctness, and efficiency of BreakChecker's operation.

Through the application of the evaluation under standard conditions, well-defined criteria, and differential levels of intricacy, the evaluation approach ensures that the resultant analysis is both meaningful and reproducible. By maintaining consistency in the test environment and isolating the effect of each variable, the process enables direct attribution of performance differences to specific factors in the tool's operation. In addition to strengthening the validity of the findings, this approach makes it easier for others to replicate the tests, verify the results, and adapt the methodology to their own security assessment environments.

5.2 Empirical Test Results

The criteria defined in Section 5.1 provided a guide on how the BreakChecker would be evaluated on different organizational domains. Three different targets have been chosen, those being K3yLabs, HackerOne, and OWASP, to offer a variety of scales and complexities when it comes to web infrastructure and overall exposure. Both the command-line interface (CLI) and the API-enabled deployment were tested at depths of one and two. The priority was given to subdomains, web endpoints, and contact identifiers (like email addresses and phone numbers) and their successful breach verification results. Besides that, the time of execution was measured as a measure of operational efficiency and evaluation of whether or not the correct identifiers were obtained, via manual review of all results in the smaller domains and a representative subset in OWASP. It is important to note that the tables list only reachable subdomains as a result of DNS and HTTP probing, leaving inactive entries out. This section shows the results obtained in raw format and explains the empirical significance of the results, not just in terms of absolute values but also in terms of the difference between API execution and CLI execution, as well as constraints faced during testing.

The first domain that was evaluated was that of K3yLabs, which is a smaller organizational target with not a lot of surface area. The results demonstrate the efficiency with which BreakChecker depletes very shallow site structures with relatively high costs of correctness preservation on identifier extraction. Table 5.1 summarises the discovery results at depths one and two involving the results of the measured outputs. In both settings, the crawler yielded the same discovery and breach-correlation results, and this observation supports the argument that enumeration and verification logic are stable across execution modes when external factors are similar.

Table 5.1: Discovery results for K3yLabs

Mode	Depth	Subdomains	Endpoints	Emails	Phones	Breached Emails	Breached Phones	Runtime (s)
API	1	2	21	1	4	0	1	29.25
CLI	1	2	21	1	4	0	1	37.45
API	2	2	250	1	4	0	1	187.63
CLI	2	2	250	1	4	0	1	212.02

Manual validation of K3yLabs showed that the single email and four phone numbers were extracted successfully, and appropriately normalized, with breach checks behaving as expected. Executions also scaled proportionally with depth, rising from an average of about thirty seconds to just over three minutes as the number of reachable endpoints grew from twenty-one to two hundred fifty. The similarity in subdomain and endpoint frequencies between API and CLI shows that execution context was not a factor in coverage on a small target with a simple structure. These measurements provide a baseline, showing that when the domain is limited in scope, BreakChecker rapidly converges to complete discovery while maintaining consistent identifier accuracy and stable verification performance.

The second domain, HackerOne, represents a medium-sized and publicly active platform, characterized by multiple subdomains and a broader exposure of contact information. This setup enables a more selective evaluation of identifier discovery and breach verification at two crawl depths, while also allowing a direct comparison of execution dynamics between CLI and API runs. The outputs analyzed are presented in Table 5.2. As depth increases, both the number of endpoints and the number of extracted identifiers expand, and the scale of breach-correlation results reflects the heightened likelihood of previously exposed organizational emails in a widely accessed, public-facing platform.

Table 5.2: Discovery results for HackerOne

Mode	Depth	Subdomains	Endpoints	Emails	Phones	Breached Emails	Breached Phones	Runtime (s)
API	1	5	114	8	1	2	0	118.06
CLI	1	5	114	8	1	2	0	126.94
API	2	5	347	22	1	3	0	284.76
CLI	2	5	347	22	1	3	0	308.72

At HackerOne on depth level one, eight emails and one phone number were found, two of which were identified as present in breach data sets. At depth two, the number of endpoints grew to three hundred and forty-seven, with the number of emails and phone numbers being twenty-two and one, with three breach email records. Manual validation confirmed that the automatically extracted identifiers were valid organizational contacts, and the hits of the breaches were consistent with API responses during validation. As projected, runtimes scaled with the size of traversal and post-processing at both depths, with API runs completing slightly ahead of CLI runs by a modest but consistent margin.

The only change that made a noticeable difference occurred during subdomain enumeration. Using Subfinder on the API environment returned an initial set of nineteen subdomain candidates, whereas the built-in mechanism in the CLI reported sixteen. Following reachability filtering, there were only five live subdomains for both contexts, so no observed differences were spotted in the final coverage or downstream output. This demonstrates the power of passive enumeration to extend outwards on sets of initial candidates, yet in this scenario, the final validated results remained unchanged.

The third and most complicated domain, OWASP, offers a community-level target with several subdomains, project documentation, and open mailing lists. The result of this diversity is an expansive surface area of endpoint exposure and a high volume of contact identifiers, creating a rigorous test of both extraction reliability and the tool's performance under constraints imposed by third-party services. Table 5.3 shows the results of what was measured at the depths of two crawls. The absolute values are even more staggering than the former domains since, on top of the effect of the wider reach, external throttling plays a significant part in the total run time.

Table 5.3: Discovery results for OWASP

Mode	Depth	Subdomains	Endpoints	Emails	Phones	Breached Emails	Breached Phones	Runtime (s)
API	1	41	1624	104	3	66	1	1834.38
CLI	1	40	1616	104	3	66	1	2148.99
API	2	41	3958	828	17	209	7	7661.86
CLI	2	40	3941	828	17	209	7	7774.36

On OWASP at depth one, the crawler discovered more than sixteen hundred endpoints and retrieved one hundred four email addresses and three phone numbers. Sixty-six of the email addresses and one of the phone numbers were confirmed as previous disclosures in breaches. At depth two, enumeration expanded to nearly four thousand endpoints, yielding eight hundred twenty-eight email addresses and seventeen phone numbers, with two hundred nine of the emails and seven of the phone numbers validated by breach checks. A manual validation sample of this domain was also reviewed because of its size, and the identifiers that were sampled were properly formed, contextually acceptable to the organizational structure of the domain, and consistent with the breach-verification responses, with no differences being noted in the reviewed sample. Such findings show that BreakChecker is capable of retaining extraction accuracy in the high-volume condition, where both heterogeneous content and archival pages are prevalent.

The difference at OWASP when enumerating raw subdomains in API and CLI was greater than the difference at HackerOne. The fallback mechanism in the CLI environment yielded up to fifty-nine indicative results, but Subfinder yielded sixty-six in the API one. Following the investigation, the validated numbers narrowed down to forty-one and forty live subdomains each. This indicates that Subfinder gave a broader original scope, whereas the CLI method was not always consistent in enumeration. But the ultimate confirmed coverage was almost the same here, supporting the argument that the candidate stage difference did not significantly alter the results upon reachability tests.

Execution behavior on OWASP explicitly documents the intrinsic cost of deeper traversal and the extrinsic delays of throttling on breach verification. The Have I Been Pwned service uses a per-request

pause, and on depth two, where hundreds of emails have been tested, these delays outweigh the total runtime. This fact explains why it took depth one approximately thirty minutes and depth two a few hours, with the two settings producing almost similar discovery results. These results suggest that external verification constraints are the primary determinant of runtime at larger scales, and BreakChecker's extraction and correlation quality are not dependent on the environment.

When the findings are combined in the three areas, they reveal a common pattern. Both modes exhaust fast and with one hundred percent coverage identity in small targets of shallow structure, and the identifiers are perfectly verified. On middle targets with richer exposure, the number of identifiers and matches on breaches increases as expected, yet both environments converge to the same results, and runtimes increase predictably with depth. On large community targets, the quantity of identifiers and boundaries of breach-verification services dictates total time, although API and CLI variants are converging in the outcomes of discovery and correlation. In each of the experiments, it was observed that manual validation of the extracted emails and phone numbers—performed fully in the smaller domains and on a representative sample in OWASP—verified the stability of the extraction pipeline. The empirical evidence, therefore, indicates that BreakChecker provides consistent coverage and accurate correlation across execution modes, with the main influences on performance being domain complexity as an internal factor and verification limits as an external factor affecting execution time.

5.3 Comparison with Existing Solutions

Comparing BreakChecker against present tools is necessary to understand its technological standing within the comprehensive security and reconnaissance solution set. Individual utilities are stronger in specialized areas such as subdomain enumeration, breach validation, or in the OSINT collection, but fewer combine such capabilities in the end-to-end process, bringing them all together in harmony. BreakChecker fills the void effectively by integrating discovery, verification, and reporting within a single, lightweight framework, making it comparable to both single-function tools and larger solutions. This section, therefore, compares BreakChecker against representative solutions, where it overlaps, where it deviates, and where it provides additional value.

Tools such as Have I Been Pwned (HIBP), DeHashed, and LeakCheck offer useful services by offering support for single identifier lookups in conjunction with domain-level querying [101 - 103]. Such domain lookups offer historic exposures relevant to an organisation's namespace, providing companies with knowledge about compromised accounts. These, though, are passive and database-centric, simply returning indexed records. BreakChecker is different in this part because it first conducts live reconnaissance. It crawls, discovers, and enumerates web infrastructure to extract emails and phone numbers, and only then submits these identifiers to breach APIs. Such a sequence flips the process over from static querying towards active discovery and validation, which lets exposures be discovered that wouldn't otherwise be instantly available under direct domain lookups. That's why BreakChecker complements tools such as HIBP, DeHashed, and LeakCheck in incorporating them in a process of dynamic reconnaissance.

BreakChecker is similar to stand-alone subdomain enumeration tools. Tools like Subfinder, Amass, and Sublist3r provide subdomain discovery pathways, where Subfinder combines the passive sources, Amass combines passive sources and active probing, and Sublist3r offers plain text lists by querying the search engines [104 - 106]. BreakChecker combines Subfinder natively at install time but otherwise falls back to querying other APIs if needed. Unlike the stand-alones, it also checks host accessibility in parallel, filtering out the unreachable and irrelevant results before beginning the work of the crawl. That

way, redundant processing is prevented—a feature not natively present in Subfinder, Sublist3r, or Amass—since achieving this level of functionality with those tools would require an enumeration utility, manual filtering, and a separate crawler.

Beyond domain enumeration, BreakChecker's workflow overlaps with multi-purpose OSINT frameworks. Tools such as theHarvester, SpiderFoot, and Maltego can collect identifiers, infrastructure data, and metadata from a wide variety of sources [107 - 109]. theHarvester, for example, can harvest emails and hosts from search engines, while SpiderFoot has an enormous collection of plugins for domains, IPs, and leaks, and Maltego offers a graph-based interface that integrates both free and commercial data sources. BreakChecker adopts a narrower focus, targeting only emails and phone numbers with direct relevance to breach exposure rather than broad-spectrum OSINT. It complements this with integrated breach-checking, something that frameworks like theHarvester or SpiderFoot only support indirectly through third-party modules or external integrations. By constraining scope, BreakChecker reduces noise and emphasizes efficiency while remaining lightweight and easier to deploy than larger OSINT suites.

Operational and privacy protections also set BreakChecker apart from full discovery tools. High-speed scanners, such as Masscan or ZMap, can scan large address spaces, but they generate perceptible network traffic and immediate detection or ethical problems [110], [111]. Tools like SpiderFoot, when operating in sweeping modes, can generate a large volume of irrelevant information, requiring analysts to filter it [108]. BreakChecker avoids such problems through the use of crawl-depth limiting, removing duplicates, and capping concurrency, thus giving it a measured footprint that aligns itself with operational common sense. This places it in practice more among tools like Nmap when run with strict rate limiting, but it provides the added aspect of querying identifiers against breach datasets [112]. Such built-in throttling places BreakChecker in a better posture in test environments where network security and reduced exposure are of main interest.

Versatility of deployment is one area where BreakChecker shines. Sublist3r is only accessible by command-line invocation, Maltego uses a graphical client, and Shodan's CLI primarily outputs JSON [106], [109], [113]. BreakChecker can be directly invoked in the CLI and controlled through the API, making it quite useful enough for such automation uses such as continuous integration pipelines or monitoring tools. It's Docker-ready, enabling predictable deployment across environments, similar in conception, although less in overhead, to the portability provided in the containerized versions of larger frameworks. Light and open-source, BreakChecker avoids the licensing expense and support requirements that can make more full-featured OSINT platforms complicated and envisions use also from individuals and small teams of researchers just as easily as it will through professional users.

Extensibility is also a consideration in determining how BreakChecker compares against what already exists. SpiderFoot attains breadth through the possession of a large set of plugins, and Maltego uses transforms in a bid to attain higher scope [108], [109]. Although strong, such systems pay in terms of steep learning curves and steep maintenance burdens. BreakChecker uses the light model: a fixed set workflow that is nonetheless modular enough that it can be extended with new APIs or parsing code. This lets it be maintained in the long term in deployments without giving up the ability to add functionality where necessary. It trades general-purpose breadth for task-specific accuracy, balancing adaptability with stability.

Lastly, output management gives a clear point of distinction. Nmap generates primarily structured text or XML, Subfinder generates mostly text lists or JSON, and Sublist3r generates plain text only [104], [106], [112]. BreakChecker makes it easier by having JSON, CSV, and Markdown exports out of the

box. JSON exports are directly compatible with SIEM environments like Splunk or ELK, CSV exports can be easily viewed in spreadsheet applications such as Excel or LibreOffice, while Markdown creates human-readable reports compatible with tools like Dradis or Serpico. All exports include metadata like timestamps and source URLs, so results can readily be sent straight into technical pipelines or official reports without reformatting. By handling multi-format output natively within the core tool, BreakChecker reduces the need for conversions and provides immediate flexibility to various audiences and workflows.

While the previous text discusses the unique aspects of comparison, providing a general overview offers a clearer perspective on BreakChecker's position relative to other tools. The following table compares various sample solutions based on characteristics such as subdomain enumeration, identifier extraction, breach validation, extensibility, deployment, and output formats. By placing BreakChecker in relation to more broadly recognized tools, the table gives the function of pointing out its distinction as a lightweight, open-source tool that accumulates various functionalities in one, unified pipeline.

Table 5.4: Comparative Overview of BreakChecker and Existing Security Tools

Tool / Platform	Core Focus	Identifier Discovery	Breach Verification	Deployment Model	Output Formats
HIBP	Breach database	No	Yes (email/domain)	Web/API	JSON, web UI
DeHashed	Breach database	No	Yes (email/domain)	Web/API	JSON, web UI
LeakCheck	Breach database	No	Yes (email/domain)	Web/API	JSON, web UI
Subfinder	Subdomain enumeration	Yes	No	CLI	JSON, text
Amass	Subdomain enumeration	Yes	No	CLI	Text, JSON, Graphviz
Sublist3r	Subdomain enumeration	Yes	No	CLI	Text
theHarvester	OSINT framework	Yes	Limited (indirect)	CLI	Text, XML
SpiderFoot	OSINT framework	Yes	Limited (via modules)	CLI/Web GUI	CSV, JSON, GEXF, HTML
Maltego	OSINT/graph analysis	Yes	Limited (via transforms)	Desktop GUI	Graphs, CSV, JSON
Masscan	Network scanning	No	No	CLI	Text
ZMap	Network scanning	No	No	CLI	JSON, text

Nmap	Network scanning	No	No	CLI	Text, XML
BreakChecker	Unified enumeration + breach checking	Emails, phone numbers	Yes (via APIs)	CLI, API, Docker	JSON, CSV, Markdown

This comparative perspective highlights BreakChecker's use case as a midpoint alternative between one-off tools and high-volume OSINT platforms. In comparison against breach-only tools like HIBP, DeHashed, and LeakCheck, it combines verification with real-time recon, so exposures appear in context and not in a vacuum. In relation to Subfinder, Amass, or Sublist3r, it doesn't simply enumerate subdomains but filters and scrapes them in bulk to feed into identifier collection and breach scanning. When compared against theHarvester, SpiderFoot, or Maltego, it doesn't try to do high-level OSINT but targets identifiers such as email addresses and phone numbers that are readily usable for breach analysis, prioritizing efficiency and simplicity. Lastly, when compared against network scanning tools such as Masscan, ZMap, or Nmap, it offers results at a minimum of operational risk and maps directly to breach data sets. By encapsulating all these capabilities in a single light, Docker-deployable, open-source package, BreakChecker offers a real-world alternative that favors ease of access over depth of analysis.

Chapter 6: Conclusions & Future Work

This thesis aimed to explore the disclosure of information on the dark web, focusing on the life cycle of compromised data and the technological environments that facilitate its migration. On one hand, it offered an academic analysis of data breaches and the migration of information into illicit marketplaces. On the other hand, it made a practical contribution by creating BreakChecker, an open-source, lightweight tool for reconnaissance, subdomain exploration, and compromise validation. In doing so, the thesis applied theoretical concepts to practical situations, situating the investigation within both academic literature and the realm of cybersecurity practice. This section concludes the study by an analysis of its contributions, discussing key findings, and outlining promising future development plans.

The first of the primary findings of this study is the illustration of how data breaches continue to be among the most persistent and most crippling threats in the information security environment. Examining the leading breaches in Yahoo, LinkedIn, and Equifax, it was clear that flaws in authentication mechanisms, poor configurations, and bulk credential reuse continue to be leading drivers for unauthorized disclosures. The case studies also demonstrated that breaches are rarely isolated mistakes, but rather systemic issues intensified by poor security and response policies. Their long-term consequences—financial and reputational—highlight the value of continuous oversight of compromised identifiers outside of corporate infrastructure.

The second is in examining the dark web as a unique space for the monetization, distribution, and tradability of breached information. Unlike traditional layers of the internet, the dark web is characterized by anonymity and decentralization, making it a rich terrain for credential stealers. The thesis brought to light the mechanisms by which leak sites, markets, and forums are intermediaries that convert breaches to economic value. From this perspective, it is necessary to track not only traditional security media but dark services where the breached information first arises. By placing data leakage within this context, the work underscores the urgent need for technical tools that can bridge the gap between breach disclosure and real-time operational intelligence.

A third central achievement was creating and executing BreakChecker, an OSINT tool that deals with such problems in a lightweight, focused manner. Unlike more general OSINT designs that create large quantities of datasets, BreakChecker narrows its scope to emails and phone numbers associated with an organization. It combines reconnaissance, crawling, and breach checking within one process where accuracy and operational prudence are prioritized. Employing API services such as Have I Been Pwned and LeakCheck, the tool ensures it works with reliable breach databases while maintaining a modular design for future enhancements. The inclusion of active infrastructure discovery alongside breach checking makes it unique from tools based on passive datasets alone.

In addition to its design philosophy, BreakChecker is extremely versatile. It can be run directly from the command line, invoked from an API for scripting, or run from containers with Docker, and therefore is extremely useful within many varied environments. Its low-footprint design, open-source availability, and multiple forms of output format (JSON, CSV, Markdown) make it extremely useful within both large-scale enterprises and more constricted research environments. This blend of technological expertise and usability is one of the wider thesis points: effective solutions to problems of cybersecurity never need to make themselves complicated but always need to be accurate and complete.

The real-world utility of BreakChecker outweighs its own cutting-edge technological advancements. By bundling together reconnaissance and breach validation, the tool offers an approach that can be pursued

by organisations to keep control of their online exposure in check continuously. Security teams are most often challenged with the problem of fragmented datasets, while using various tools to gather info for public and accessible subdomains, as well as breached identifiers, and present the results in a suitable format. BreakChecker addresses this issue by offering a cohesive solution that reduces redundant efforts and minimizes the risk of omissions. In addition, the integration of traffic control and deduplication makes it very well adapted towards managed deployments where traffic volume needs to be kept low to prevent alarm triggering or operation failure.

The evaluation phase of the thesis validated that BreakChecker stands its ground in this competitive market. The tool does not intend to outperform huge frameworks such as SpiderFoot in coverage depth or Nmap in low-level network scanning, but it does offer an identifiable edge in the area of accuracy and usability. The tool white executed extracted precisely actionable identifiers using minimal footprint, aligning it with those situations where stealth and speed are of maximum importance. This finding further verifies the overall thesis argument: that a smart balance, tipping towards concentrated intel over breadth of information gathering, can produce results both actionable and practical in practice.

Nonetheless, the investigation also uncovered limitations that define the boundaries of future optimization. While the focus of BreakChecker's is on phone numbers and email addresses was intentional, this choice overlooks other important identifiers that often appear in breach scenarios, such as API tokens, session cookies, and even cryptocurrency wallets. Its third-party API component-based architecture also adds a dependent element; if they restrict access or change their formats, the software would need an update promptly. Lastly, while running on Docker allows for reproducibility and lightweight operations, additional investigation, perhaps, can be conducted on scaling BreakChecker on multi-node deployments and deploying it to scan enterprise-scale distributed environments. Defining such constraints is important, since they allow future fine-tuning routes.

Future development should thus aim at several areas of expansion. Firstly, expansion of the identifier extraction capacity beyond email and phone number would enhance the tool's usefulness in more general forensic and incident response scenarios. Use of NLP models with the ability to rate leaked data could also simplify triage through automated identification of high-value exposure from noise. Secondly, the modularity of the architecture may be used to incorporate additional breach intelligence APIs, providing more redundancy and resilience. Thirdly, more powerful visualization and reporting feature inclusion would enable analysts to compare results more effectively, possibly correlating subdomain exposure against certain breach incidents in one location, like a dashboard.

Beyond technical capabilities, BreakChecker could also support active defense practices. By being plugged into normal monitoring pipelines, it would be possible for organizations to monitor trends in their online footprint over time, converting the present snapshot into a continuous analysis. In this way, BreakChecker could be both an early warning system and a compliance tool, providing verifiable records during audits or regulatory reporting. Researchers could also gain from the lightweight and open-source nature of the tool, as it allows the creation of a reproducible baseline against which to investigate breach disclosure dynamics and compare methods across outputs from different scanned domains. By reducing barriers to entry, the project is open to contributions from professional practitioners as well as the research community.

In the future, research on disclosure of information on the dark web should also broaden into wider academic directions. One key area is regulation and governance. Regulations like HIPAA in the United States or GDPR in Europe were designed for traditional data environments and are thus not well-suited for anonymized, transnational markets. Comparative studies could investigate how different

jurisdictions try to remedy illicit leaks, where enforcement would be most successful, and if coordinated international action would provide better deterrence against decentralized economies designed around leaked data.

Equally important is considering the human impact that disclosure may have. While technical analysis focuses on datasets and identifiers, much less is known about individuals and organizations that are directly impacted when their data is leaked to the dark web. Victim-centric studies could focus on the psychological, social, and economic implications of exposure, such as financial damage, loss of trust in digital systems, or long-term changes in security behavior. A combination of these perspectives would deepen the academic study of disclosure and bring it closer to the concrete impact it has on society.

A study of the comparative regional approaches may also be useful. Disclosure is not the same everywhere, but it depends on cultural norms of expectations of privacy, regulatory structures, and economic incentives. Further exploration of why, for example, some types of data or markets are more prominent in some geographies than others would yield a more subtle understanding of disclosure as a worldwide phenomenon. Such work would enrich the study of illicit economies by uncovering how local conditions interact with wider transnational processes in the flow of violated knowledge.

On a broader level, the thesis highlights the ongoing need to connect academic research with practical applications. Examinations of data breaches and dark web activity identified patterns that cannot be remedied solely by technology-focused solutions. Factors like organisational culture, regulatory frameworks, and economic imperatives all contribute to persisting threats. BreakChecker exemplifies this approach by showing how practical tools can bring theoretical insights to life, transforming conceptual frameworks into effective defense mechanisms. This intersection anticipates that the most effective cybersecurity solutions will be constructed not from a single domain, but from projects of combined theory, practice, and innovation.

Overall, this work has revealed the prevalence of data breaches, the dark web's enabling role in their exploitation, and the need for light on this area in the form of tools. By creating BreakChecker, it has shown that lightweight, focused, and open-source tools have the potential to play a valuable role in serving specialists and general frameworks. By striking a balance between precision and usability, innovation and availability, the tool shows the wider thesis assertion: that effective cybersecurity must be operationally workable and technically sound. In the future, the project will continue to build on its foundations and ensure that BreakChecker remains relevant in addressing the threats it originally identified, while making an academic and professional contribution to data disclosure in the digital domain.

References

[1] V. Bidv and A. S. Waghmare, “Beyond the onion routing: Unmasking illicit activities on the dark web,” *Int. J. Innov. Sci. Res. Technol.*, 2024.

[2] G. Pantelis, S. Karagiorgou, P. Petrou, and D. Alexandrou, “On strengthening SMEs and MEs threat intelligence and awareness by identifying data breaches, stolen credentials and illegal activities on the dark web,” in *Proc. 16th Int. Conf. Availability, Reliability and Security (ARES ’21)*, New York, NY, USA: ACM, 2021, Art. 156, pp. 1–7.

[3] S.-L. Hung, C.-K. Chen, K. Furumoto, T. Takahashi, and H.-M. Sun, “Dark watchdog: A novel RAG-driven system for real-time detection and analysis of data leaks on dark web forums,” in *Proc. Annu. Comput. Secur. Appl. Conf. Workshops (ACSAC Workshops)*, Honolulu, HI, USA, 2024, pp. 11–19.

[4] B. R. Jung, K.-S. Choi, and C. S. Lee, “Dynamics of dark web financial marketplaces: An exploratory study of underground fraud and scam business,” *Int. J. Cybersecurity Intell. Cybercrime*, vol. 5, no. 2, pp. 4–24, Aug. 2022.

[5] A. Skebaite, “The 20 biggest data breaches in history,” NordVPN, Aug. 21, 2024. [Online]. Available: https://nordvpn.com/blog/biggest-data-breaches/.

[6] E. Jardine, *The Dark Web Dilemma: Tor, Anonymity and Online Policing*. Waterloo, Canada: Centre for Int. Governance Innovation and Chatham House, Sept. 2015.

[7] U.S. Small Business Administration, “Strengthen your cybersecurity,” Jul. 2, 2024. [Online]. Available: https://www.sba.gov/business-guide/manage-your-business/strengthen-your-cybersecurity.

[8] H. Xiang, Y. Wu, X. Wang, J. Shi, C. Zhao, and J. Zhao, “Identifying data breaches in dark web through prompt active learning,” in *Proc. IEEE 9th Int. Conf. Data Sci. Cyberspace (DSC)*, 2024, pp. 61–68.

[9] K. Connolly, A. Klempay, M. McCann, and P. Brenner, “Dark web marketplaces: Data for collaborative threat intelligence,” *ACM Trans. Digit. Threats: Res. Pract.*, vol. 4, no. 4, Art. 49, pp. 1–12, Oct. 2023.

[10] F. Ullah, M. Edwards, R. Ramdhany, and R. Chitchyan, “Data exfiltration: A review of external attack vectors and countermeasures,” *J. Netw. Comput. Appl.*, vol. 112, pp. 97–112, 2018.

[11] M.-H. Chung, Y. Yang, L. Wang, G. Cento, K. Jerath, A. Raman, D. Lie, and M. H. Chignell, “Implementing data exfiltration defense in situ: A survey of countermeasures and human involvement,” *ACM Comput. Surv.*, vol. 1, no. 1, Art. 37, pp. 1–37, Jan. 2023.

[12] B. Sabir, F. Ullah, M. A. Babar, and R. Gaire, “Machine learning for detecting data exfiltration: A review,” *ACM Comput. Surv.*, vol. 54, no. 3, Art. 50, pp. 1–47, Apr. 2022.

[13] N. Sfetcu, *Advanced Persistent Threats in Cybersecurity – Cyber Warfare*. MultiMedia Publishing, 2024. ISBN 978-606-033-853-6.

[14] A. Waheed, B. Seegolam, M. F. Jowaheer, C. L. X. Sze, E. T. F. Hua, and S. R. Sindiramutty, “Zero-day exploits in cybersecurity: Case studies and countermeasure,” *Preprints*, 2024.

[15] S. P. Singh and N. Afzal, “The MESA security model 2.0: A dynamic framework for mitigating stealth data exfiltration,” *arXiv preprint*, arXiv:2405.10880, 2024. [Online]. Available: https://arxiv.org/abs/2405.10880.

[16] M. Mundt and H. Baier, “Towards mitigation of data exfiltration techniques using the MITRE ATT&CK framework,” in *Proc. 12th Int. Conf. Digital Forensics and Cyber Crime (ICDF2C 2021)*, Virtual Event, Singapore, Dec. 6–9, 2021, pp. 149–164.

[17] C. Beretas, *Information Systems Security, Detection and Recovery from Cyber Attacks*, vol. 1, no. 1. Universal Library of Engineering Technology, 2024.

[18] Z. Tan, S. P. Parambath, C. Anagnostopoulos, J. Singer, and A. K. Marnerides, “Advanced persistent threats based on supply chain vulnerabilities: Challenges, solutions, and future directions,” *IEEE Internet Things J.*, vol. 12, no. 6, pp. 6371–6395, Mar. 2025.

[19] IBM Security, “Cost of a Data Breach Report 2023.” IBM Corp., 2023. [Online]. Available: https://d110erj175o600.cloudfront.net/wp-content/uploads/2023/07/25111651/Cost-of-a-Data-Breach-Report-2023.pdf.

[20] S. Rohatgi, *Understanding and Mitigating Advanced Persistent Threats in a Dynamic Cyber Landscape*. Hoboken, NJ: Wiley, 2025.

[21] N. Essilfie-Conduah, “A systems analysis of insider data exfiltration,” M.S. thesis, Massachusetts Institute of Technology, Cambridge, MA, 2019.

[22] J. Finkle and S. Nellis, “Yahoo now says all 3 billion accounts were affected by 2013 hack,” Reuters, Oct. 3, 2017. [Online]. Available: https://www.reuters.com/article/us-yahoo-cyber/yahoo-now-says-all-3-billion-accounts-were-affected-by-2013-hack-idUSKCN1C82O1.

[23] L. J. Trautman and P. C. Ormerod, “Corporate directors’ and officers’ cybersecurity standard of care: The Yahoo data breach,” *Am. Univ. Law Rev.*, vol. 66, no. 5, pp. 1231–1268, 2017.

[24] U.S. House of Representatives, Committee on Oversight and Government Reform, “The Equifax data breach.” 115th Congress, Dec. 2018.

[25] A. Gune, “The cryptographic implications of the LinkedIn data breach,” *arXiv preprint*, arXiv:1703.06586, 2017.

[26] B. Catalin, “Man who lived luxury lifestyle after hacking LinkedIn and Dropbox is found guilty,” Bitdefender Hot for Security, Jul. 15, 2020. [Online]. Available: https://www.bitdefender.com/en-us/blog/hotforsecurity/man-who-lived-luxury-lifestyle-after-hacking-linkedin-and-dropbox-is-found-guilty.

[27] B. Krebs, “Adobe to announce source code, customer data breach,” Krebs on Security, Oct. 3, 2013. [Online]. Available: https://krebsonsecurity.com/2013/10/adobe-to-announce-source-code-customer-data-breach/.

[28] D. B. Kawushika de Zoysa, “Adobe Cyberattack 2013 case study,” B.Sc. thesis, Faculty of Computing and Technology, Univ. of Kelaniya, Sri Lanka, Jan. 2025. [Online]. Available: https://www.researchgate.net/publication/388499862_Adobe_Cyberattack_2013_Case_study.

[29] M. Manglani, “Compromised systems, compromised data: A technical analysis of the Marriott data breach,” *Int. J. Sci. Res. (IJSR)*, vol. 13, no. 4, pp. 176–180, Apr. 2024.

[30] O. Lawal, “A cybersecurity data breach: The Marriott International hotel incidence,” *Comput. Inf. Syst. Dev. Informatics Allied Res. J.*, vol. 15, no. 4, pp. 25–28, 2024.

[31] I. Kozlowska, “Facebook and data privacy in the age of Cambridge Analytica,” Henry M. Jackson School of International Studies, Univ. of Washington, Apr. 30, 2018. [Online]. Available: https://jsis.washington.edu/news/facebook-data-privacy-age-cambridge-analytica/.

[32] U.S. Senate Committee on Commerce, Science, and Transportation, “A ‘Kill Chain’ analysis of the 2013 Target data breach.” Majority Staff Report for Chairman Rockefeller, Mar. 26, 2014.

[33] M. Seal, “An exclusive look at Sony’s hacking saga,” *Vanity Fair*, Feb. 4, 2015. [Online]. Available: https://www.vanityfair.com/hollywood/2015/02/sony-hacking-seth-rogen-evan-goldberg.

[34] S. Steinberg, A. Stepan, and K. Neary, *The Hacking of Sony Pictures: A Columbia University Case Study*. Columbia Univ. School of International and Public Affairs, SIPA-21-0023.1, 2021.

[35] European Commission, “Statement by Vice-President Ansip and Commissioner Jourová ahead of Data Protection Day,” European Commission Press Corner, Jan. 25, 2019. [Online]. Available: https://ec.europa.eu/commission/presscorner/detail/en/STATEMENT_19_662.

[36] A. Kajave and S. A. H. Nismy, “How cyber criminals use social engineering to target organizations,” *arXiv preprint*, arXiv:2212.12309, 2022.

[37] V. Valeros and S. Garcia, “Growth and commoditization of remote access Trojans,” in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops (EuroS&PW)*, Genoa, Italy, 2020, pp. 454–462.

[38] M. Neagu, “RedLine stealer analysis,” Bitdefender Whitepaper, Apr. 2022.

[39] S. Duraibi, C. Kaur, and A. B. Pawar, “Cyber extortion unveiled: The evolution, tactics, challenges, and future of ransomware,” in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Las Vegas, NV, USA, 2023, pp. 861–867.

[40] J. Beerman, D. Berent, Z. Falter, and S. Bhunia, “A review of Colonial Pipeline ransomware attack,” in *Proc. IEEE/ACM 23rd Int. Symp. Cluster, Cloud Internet Comput. Workshops (CCGridW)*, Bangalore, India, 2023, pp. 8–15.

[41] A. Alabdulatif and N. N. Thilakarathne, “Hacking exposed: Leveraging Google Dorks, Shodan, and Censys for cyber attacks and the defense against them,” *Comput.*, vol. 14, p. 24, 2025.

[42] Team SpyCloud, “The new cracking tools that automate credential stuffing & account takeover,” SpyCloud Blog, Jul. 29, 2020. [Online]. Available: https://spycloud.com/blog/new-cracking-tools-automate-credential-stuffing-account-takeover-openbullet-sentry-mba/.

[43] D. Woods, S. Boddy, and S. Backer, “Genesis marketplace – A digital fingerprint darknet store,” *F5 Labs Newsletter*, Nov. 19, 2020. [Online]. Available: https://www.f5.com/labs/articles/threat-intelligence/genesis-marketplace--a-digital-fingerprint-darknet-store.

[44] A. P. Singh and A. Sharma, “A systematic literature review on insider threats,” *arXiv preprint*, arXiv:2212.05347, Dec. 2022.

[45] B. Gokkaya, L. Aniello, and B. Halak, “A systematic literature review on software supply chain attacks, risk assessment strategies, and security controls,” *arXiv preprint*, arXiv:2305.14157, May 2023.

[46] S. Alrwais, K. Yuan, E. Alowaisheq, X. Liao, A. Oprea, X. Wang, and Z. Li, "Catching predators at watering holes: Finding and understanding strategically compromised websites," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl. (ACSAC)*, Los Angeles, CA, USA, Dec. 2016.

[47] F. Barr-Smith, X. Ugarte-Pedrero, M. Graziano, R. Spolaor, and I. Martinovic, "Survivalism: Systematic analysis of Windows malware living-off-the-land," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, 2021, pp. 1557–1574.

[48] R. Allen, "The dangers of 7-Zip and WinRAR," ThreatLocker, Jun. 12, 2024. [Online]. Available: https://www.threatlocker.com/blog/the-dangers-of-7-zip-and-winrar.

[49] S. Ozarslan, "The most common ransomware TTP – MITRE ATT&CK T1486 data encrypted for impact," Picus Security, Oct. 23, 2024. [Online]. Available: https://www.picussecurity.com/resource/the-most-common-ransomware-ttp-mitre-attck-t1486-data-encrypted-for-impact.

[50] S. A. Samarakoon, "Bypassing content-based internet packages with an SSL/TLS tunnel, SNI spoofing, and DNS spoofing," *arXiv preprint*, arXiv:2212.05447, 2022.

[51] A. Nadler, A. Aminov, and A. Shabtai, "Detection of malicious and low throughput data exfiltration over the DNS protocol," *arXiv preprint*, arXiv:1709.08395, 2018.

[52] MITRE ATT&CK, "Data transfer size limits," MITRE ATT&CK, 2025. [Online]. Available: https://attack.mitre.org/techniques/T1030/.

[53] MITRE ATT&CK, "Data obfuscation: Steganography," MITRE ATT&CK, 2025. [Online]. Available: https://attack.mitre.org/techniques/T1001/002/.

[54] MITRE ATT&CK, "Exfiltration to cloud storage," MITRE ATT&CK, 2025. [Online]. Available: https://attack.mitre.org/techniques/T1567/002/.

[55] A. A. R. Team, "Attack graph response to US CERT AA22-152A: Karakurt data extortion group," AttackIQ, Jun. 3, 2022. [Online]. Available: https://www.attackiq.com/2022/06/03/attack-graph-response-to-us-cert-aa22-152a-karakurt-data-extortion-group/.

[56] Federal Bureau of Investigation, "Ransomware prevention and response for CISOs." Washington, DC: FBI, 2016.

[57] P. Tippe, "Hardening Tor hidden services," M.S. thesis, Dept. of Computing, Univ. of Turku, Turku, Finland, Dec. 2022.

[58] R. O'Neill, "Cyber-criminals boost sales through 'data laundering'," ZDNet, Mar. 16, 2015. [Online]. Available: https://www.zdnet.com/article/cyber-criminals-boost-sales-through-data-laundering/.

[59] L. Terrelonge III and M. Aliapoulios, "Cybercrime economy: An analysis of criminal communications strategies," Flashpoint, 2017.

[60] C. M. S. Steel, "Stolen identity valuation and market evolution on the dark web," *Int. J. Cyber Criminol.*, vol. 13, no. 1, pp. 61–77, 2019.

[61] D. Georgoulias, *BED: Botnet Economic Disruption, via Darkweb Marketplace Infiltration*, Ph.D. dissertation, Dept. of Electronic Systems, Aalborg Univ., Aalborg, Denmark, 2024.

[62] P. Spagnoletti, F. Ceci, and B. Bygstad, “Online black-markets: An investigation of a digital infrastructure in the dark,” *Inf. Syst. Front.*, vol. 24, no. 6, pp. 1811–1826, Dec. 2022.

[63] M. van Loosen, “Telegram as a new cybercrime marketplace,” M.S. thesis, Eindhoven Univ. of Technology, 2024.

[64] L. Polan, “When privacy goes private: Technological stewardship in post-Snowden Silicon Valley,” M.A. thesis, Univ. of Chicago, Mar. 2023.

[65] M. Zaeifi, F. Kalantari, A. Oest, Z. Sun, G.-J. Ahn, Y. Shoshitaishvili, T. Bao, R. Wang, and A. Doupé, “Nothing personal: Understanding the spread and use of personally identifiable information in the financial ecosystem,” in *Proc. 14th ACM Conf. Data Appl. Secur. Privacy (CODASPY)*, 2024, pp. 55–65.

[66] J. Robertson, A. Diab, E. Marin, and E. Nunes, “Darkweb cyber threat intelligence mining,” Cambridge Univ. Press, 2017.

[67] Y. Bayoumy, “Cybercrime economy – A netnographic study on the dark net ecosystem for ransomware,” M.S. thesis, Norwegian Univ. of Sci. and Technol. (NTNU), 2018.

[68] U. Pesch and G. Ishmaev, “Fictions and frictions: Promises, transaction costs and the innovation of network technologies,” *Social Studies of Science*, Mar. 2019.

[69] M. Magnante, “Exploring the interactional theory: A theoretical exploration of the dark web’s impact on delinquent behaviour,” M.A. thesis, Ontario Tech Univ., Aug. 2021.

[70] J. Saleem, R. Islam, and M. A. Kabir, “The anonymity of the dark web: A survey,” *IEEE Access*, vol. 10, pp. 33628–33660, 2022.

[71] B. Akhgar, M. Gercke, S. Vrochidis, and H. Gibson, *Dark Web Investigation*. Springer, 2021.

[72] V. Ciancaglini, M. Balduzzi, M. Goncharov, and R. McArdle, *Deepweb and Cybercrime: It’s Not All About TOR*. Trend Micro Research, 2013.

[73] R. Basheer and B. Alkhatib, “Threats from the dark: A review over dark web investigation research for cyber threat intelligence,” *J. Comput. Netw. Commun.*, vol. 2021, no. 1, p. 1302999, 2021.

[74] A. Sanatinia, J. Park, E.-O. Blass, A. Mohaisen, and G. Noubir, “A privacy-preserving longevity study of Tor’s hidden services,” *arXiv preprint*, arXiv:1909.03576, 2019.

[75] SecureDrop Documentation, “What is SecureDrop?” Freedom of the Press Foundation. [Online]. Available: https://docs.securedrop.org/en/stable/what_is_securedrop/.

[76] M. Madouh and K. H. Kwon, “Evolving in the shadows: A media ecology study of dark web social networks,” *J. Commun. Inquiry*, Nov. 2023.

[77] A. R. Barredo-Valenzuela, S. P. Portillo, and G. Suarez-Tangil, “Snorkeling in dark waters: A longitudinal surface exploration of unique Tor hidden services (extended version),” *arXiv preprint*, arXiv:2504.16836, Apr. 2025.

[78] M. Hosseini Shirvani and A. Akbarifar, “A comparative study on anonymizing networks: TOR, I2P, and Riffle networks comparison,” *J. Elect. Comput. Eng. Innov. (JECEI)*, vol. 10, no. 2, pp. 259–272, Jul. 2022.

[79] D. Chao, D. Xu, F. Gao, C. Zhang, W. Zhang, and L. Zhu, “A systematic survey on security in anonymity networks: Vulnerabilities, attacks, defenses, and formalization,” *IEEE Commun. Surveys Tuts.*, vol. 26, no. 3, pp. 1775–1829, 2024.

[80] J. Szurdi, “Tor 101: How Tor works and its risks to the enterprise,” Unit 42. [Online]. Available: https://unit42.paloaltonetworks.com/tor-traffic-enterprise-networks/.

[81] S. Damaye, “Chapter 3: I2P,” in *Navigating the Dark Web: A Comprehensive Guide to Darknets, Tools, and Intelligence*. [Online]. Available: https://navigating-the-darkweb.readthedocs.io/en/latest/chapter3_i2p.html.

[82] I2P Project, “Technical introduction,” I2P Anonymous Network, Jun. 2025. [Online]. Available: https://geti2p.net/en/docs/how/tech-intro.

[83] NetworxSecurity, “Freenet,” NetworxSecurity, Jun. 17, 2025. [Online]. Available: https://www.networxsecurity.org/members-area/glossary/f/freenet.html.

[84] D. Georgoulias, J. M. Pedersen, M. Falch, and E. Vasilomanolakis, “A qualitative mapping of darkweb marketplaces,” in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, Dec. 2021, pp. 1–15.

[85] Vice City Market, “Vice City Market darknet – Secure onion links, mirror URLs, and darknet resources.” Accessed: Jun. 30, 2025. [Online]. Available: https://vicecity-market.io/index.html.

[86] Incognito Market, “Incognito darknet market – Official clearnet site.” [Online]. Available: https://incognitodarknet.net/.

[87] Dread Forum, “Dread forum | Darknet community.” [Online]. Available: https://forumdread.com/.

[88] Exploit.IN, “Exploit.in forum.” [Online]. Available: https://exploit.in/.

[89] C. Patsakis, D. Arroyo, and F. Casino, “The malware as a service ecosystem,” *arXiv preprint*, arXiv:2405.04109, May 2024.

[90] Information Age, “Exposed: How ransom gang Lockbit negotiates payments,” Information Age, 2025. [Online]. Available: https://ia.acs.org.au/article/2025/exposed--how-ransom-gang-lockbit-negotiates-payments.html.

[91] A. Elsad, J. R. Gumarin, and A. Barr, “LockBit 2.0: How this RaaS operates and how to protect against it,” Unit 42. [Online]. Available: https://unit42.paloaltonetworks.com/lockbit-2-ransomware/.

[92] GlobaLeaks, “Documentation.” [Online]. Available: https://docs.globaleaks.org/en/stable/.

[93] D. Winder, “Hack attack takes down dark web host: 7,595 websites confirmed deleted,” *Forbes*, Mar. 30, 2020. [Online]. Available: https://www.forbes.com/sites/daveywinder/2020/03/30/hack-attack-takes-down-dark-web-7595-websites-confirmed-deleted/.

[94] FlokiNET, “FlokiNET.” [Online]. Available: https://flokinet.is/.

[95] U.S. Department of Justice, Office of Public Affairs, “Criminal marketplace disrupted in international cyber operation,” U.S. DOJ, 2020. [Online]. Available: https://www.justice.gov/archives/opa/pr/criminal-marketplace-disrupted-international-cyber-operation.

[96] Darknet Index, “Guide to using XMPP (Jabber).” [Online]. Available: https://darknetindex.com/categories/guidestutorials/guide-to-using-xmpp-jabber.

[97] Ricochet Refresh, “Ricochet Refresh.” [Online]. Available: https://www.ricochetrefresh.net/.

[98] Project Tox, “A new kind of instant messaging.” [Online]. Available: https://tox.chat.

[99] U.S. Dept. Health and Human Services, Health Sector Cybersecurity Coordination Center, “HC3: Analyst note – Venus ransomware targets publicly exposed remote desktop services,” Nov. 9, 2022.

[100] Intelligence X, “About – Intelligence X.” [Online]. Available: https://intelx.io/about.

[101] T. Hunt, “Have I been pwned: Check if your email address has been exposed in a data breach,” Have I Been Pwned. [Online]. Available: https://haveibeenpwned.com/.

[102] DeHashed, “DeHashed — #FreeThePassword.” [Online]. Available: https://dehashed.com/.

[103] LeakCheck, “LeakCheck – Find out if your credentials have been compromised.” [Online]. Available: https://leakcheck.io/.

[104] ProjectDiscovery, “Subfinder overview,” ProjectDiscovery Documentation. [Online]. Available: https://docs.projectdiscovery.io/tools/subfinder/overview.

[105] OWASP Foundation, “OWASP Amass.” [Online]. Available: https://owasp.org/www-project-amass/.

[106] Aboul3la, “Sublist3r: Fast subdomains enumeration tool.” GitHub. [Online]. Available: https://github.com/aboul3la/Sublist3r.

[107] Laramies, “theHarvester: E-mails, subdomains and names harvester.” GitHub. [Online]. Available: https://github.com/laramies/theHarvester.

[108] Smicallef, “SpiderFoot: Automates OSINT for threat intelligence and attack surface mapping.” GitHub. [Online]. Available: https://github.com/smicallef/spiderfoot.

[109] Maltego, “Maltego: OSINT and cyber investigations platform.” Maltego. [Online]. Available: https://www.maltego.com/.

[110] R. Graham, “Masscan: TCP port scanner.” GitHub. [Online]. Available: https://github.com/robertdavidgraham/masscan.

[111] ZMap Project, “The ZMap project.” [Online]. Available: https://zmap.io/.

[112] Nmap, “The network mapper – Free security scanner.” [Online]. Available: https://nmap.org/.

[113] Shodan, “Installation – Shodan command-line interface,” Shodan Help Center. [Online]. Available: https://help.shodan.io/command-line-interface/0-installation.

APPENDIX A: CORE ORCHESTRATION FUNCTION

This appendix provides the main orchestration function of BreakChecker, which integrates all core operations of the tool, including subdomain enumeration, accessibility filtering, crawling, data extraction, breach verification, and structured result generation. This function represents the central logic flow of the system, tying together its supporting components. It is the primary orchestration point used by both the command-line interface (CLI) and the API layers of the tool. The full codebase, including helper modules, configuration files, and API implementations, is openly available at the public repository: https://github.com/terzikk/BreakChecker.

```
async def scan_domain(
    domain: str,
    depth: int = 3,
    hibp_key: Optional[str] = None,
    leakcheck_key: Optional[str] = None,
    *,
    verbose: bool = False,
    concurrency: int = 5,
    save: bool = False,
    fmt: str = "json",
    output_path: Optional[str] = None,
) -> dict:
    """Crawl a domain and optionally check contacts against breach data.

    Args:
        domain: Validated target domain (ASCII/IDNA normalized).
        depth: Maximum crawl depth.
        hibp_key: API key for HaveIBeenPwned proxy.
        leakcheck_key: API key for LeakCheck.
        verbose: Enable debug logging for more detail.
        concurrency: Number of concurrent workers for fetching/crawling.
        save: When True, persist results via :func:`save_results`.
        fmt: Output format for saving ("json", "csv", or "md").
        output_path: Optional explicit output file path.

    Returns:
        Dictionary with subdomains, emails, phones, breach mappings,
        per-item sources, and scan timing/summary.
    """
    start_time = time.time()
    start_dt = datetime.datetime.now(datetime.timezone.utc)
    logger.info("Starting scan for %s (depth: %d, concurrency: %d)",
                domain, depth, concurrency)

    stage = 1
    logger.info("Stage %d: Enumerating subdomains for %s", stage, domain)
    subs = enumerate_subdomains(domain)

    stage += 1
    logger.info(
        "Stage %d: Filtering %d subdomains for web accessibility...", stage, len(subs))
    subdomain_schemes = await filter_accessible_subdomains(subs, concurrency=concurrency, retries
=2)
```

```
    logger.info("Found %d accessible web hosts.", len(subdomain_schemes))

    crawler = Crawler(domain, max_depth=depth, concurrency=concurrency)
    await _ensure_pw_started()

    stage += 1
    logger.info("Stage %d: Crawling %d URL(s) to find contacts...",
                stage, len(subdomain_schemes))
    for sub, scheme in sorted(subdomain_schemes.items()):
        start_url = f"{scheme}://{sub}"
        await crawler.crawl(start_url)

    stage += 1
    breached_emails: Dict[str, List[str]] = {}
    for email in crawler.emails.values():
        breaches = check_hibp(email, hibp_key)
        if breaches:
            breached_emails[email] = breaches

    stage += 1
    breached_phones: Dict[str, List[str]] = {}
    for phone in crawler.phones.values():
        breaches = check_leakcheck_phone(phone, leakcheck_key)
        if breaches:
            breached_phones[phone] = breaches

    await _shutdown_pw()

    results = {
        "subdomains": set(subdomain_schemes.keys()),
        "emails": set(crawler.emails.values()),
        "phones": set(crawler.phones.values()),
        "breached_emails": breached_emails,
        "breached_phones": breached_phones,
        "email_sources": crawler.email_sources,
        "phone_sources": crawler.phone_sources,
        "num_endpoints": len(crawler.visited),
        "emails_dropped": getattr(crawler, "_emails_dropped", 0),
        "phones_dropped": getattr(crawler, "_phones_dropped", 0),
    }

    end_dt = datetime.datetime.now(datetime.timezone.utc)
    results["scan_duration"] = time.time() - start_time
    results["scan_start"] = start_dt.strftime("%Y-%m-%d %H:%M:%S %Z")
    results["scan_end"] = end_dt.strftime("%Y-%m-%d %H:%M:%S %Z")

    if save:
        saved_to = save_results(results, domain, fmt=fmt, output_path=output_path)
        logger.info("Saved results to %s", saved_to)

    return results
```

This function highlights how BreakChecker ties together discovery, validation, crawling, and breach correlation in a single scanning workflow. It serves as the **main orchestration function** for both CLI and API execution, ensuring unified logic across different usage contexts. For further details, implementation modules, and extensions, refer to the complete repository linked above.