100 Popular Open-Source Infosec Tools

Rauli Kaksonen[0000−0001−8692−763X], Tommi Järvenpää, Jukka Pajukangas,
Mihai Mahalean, and Juha Röning[0000−0001−9993−8602]

University of Oulu, Oulu, Finland
first.last@oulu.fi
https://www.oulu.fi

Abstract. We examined the popularity of open-source tools used for information security analysis (infosec tools). This information would be useful, e.g. in security research, but it was not available. In our study, we created first a corpus of 423 tools from various sources. Then we collected source popularity metrics by Google search, tweets, GitHub stars, SecTools.org ranking, and cross-references between tools. We found a strong correlation between the metrics. We created an aggregate popularity metric from Google search, GitHub stars, and tool cross-reference source metrics using principal component analysis. The aggregate metric explains 70% of the variance in the source metrics. The three most popular tools are Metasploit, Nmap, and Wireshark. We estimated the impact of source metric errors and concluded that the aggregate metric gives an estimate of tool popularity, rather than an exact popularity rank. Furthermore, we divide the tools into overlapping categories by tool scope and type of activity. In the top 100, 51 tools are in the network scope, 27 in the host scope, 15 in the storage scope, 13 in the passwords scope, and 4 in the other tools scope.

Keywords: information security, infosec, security analysis, open-source tools, popularity

1 Introduction

Open-source tools are widely used for information security tasks like penetration testing, incident response, and digital forensics. A survey among forensics experts found that 69% of the responders used open-source tools at least sometimes while only 5% had never used them [29]. In another survey, one key requirement for cyber-forensics research was the requirement for “backing for and improvement of open-source tools“ [28]. The experience of the authors supports that open-source tools are widely used by the security community.

However, when we tried to find out which tools are popular, we could not find a lot of concrete information. Studies often refer to open-source tools to be used for a particular purpose [30] [33] [4]. However, usually the tools are chosen by the researchers or selected by limited surveys. This feels quite unsatisfactory. Objective data on tool popularity would help e.g. to choose a set of representative tools for research, to examine the properties of the popular tools, and to find alternative tools for a task.

As the security community and open-source community operate in the Internet, representative information about the security tools and open-source software should be discoverable on-line. We were able to identify the following sources of open-source tools used for information security tasks (*infosec* tools):

- **SecTools.org** lists 125 Network Security Tools ranked based on suggestions by the community [20]. The site excludes tools they maintain themselves.
- **GitHub** is a code hosting site [9]. It is very popular and hosting more than 50% of the tools which data we collected for this study. GitHub allows users to give stars for the project they like.
- **Google code archive** contains the projects from the discontinued Google code hosting site [11]. Most projects are moved to other sites, many to GitHub.
- **Sourceforge** is an older open-source project hosting site [22]. Roughly 20% of the projects we collected have a project page there, but many indicate that they have moved elsewhere, usually to GitHub.
- **GitLab** is a challenger for GitHub [10], but just a handful of the security tools we collected are using it. There is a star-system similar to GitHub.
- **Infosec-specific distributions and images** are available with preinstalled infosec tools, e.g. Linux distributions ArchStrike [1], BlackArch [2], CAINE [3], Fedora Security Lab [6], Kali [13], Penroo [17], Remnux [19], and SIFT[21]. For Microsoft Windows there is e.g. Flare VM [7].

We outlined the following research questions for this study:

1. What sources contain quantitative data on the popularity of open-source infosec tools?
2. Do the sources agree on the popularity of tools?
3. How can the tools be divided into relevant categories by task they perform?
4. What are the most popular tools in general and per category?

The rest of this document is organized as follows. In Section 2, we present the proposed infosec tool popularity metrics and the tool categorization scheme. In Section 3, we present the results and estimate the reliability of the results. Finally, in Section 4 we discuss the results and conclude this study.

2 Methods

The purpose of this study is to identify the most popular open-source infosec tools. These tools are used for a variety of tasks: network scanning, traffic analysis, traffic generation, web security, disk and memory forensics, file analysis, host analysis, password cracking, etc. For this study, we ruled out software that provides security-related functionality continuously, e.g., VPNs, and firewalls, and tools intended mainly for software development. We adapted the definition "popular" from Merriam-Webster [14], and define a *popular infosec tool* being a frequently encountered tool, commonly considered for a task, or widely approved as a tool to use.

2.1 Collecting a Corpus of Tools

To measure the popularity of infosec tools, we first collected a corpus of tools by the following methods.

- List the tools in the infosec Linux distributions and the Flare VM
- Add open-source tools from SecTools.org [20]
- Add tools identified in prior work, such as in CinCan project [4]
- Add infosec tools the authors know about
- Later, check URLs in the downloaded pages for yet unlisted tools

To help excluding tools that are not security-related we filtered out tools that are present in Debian and Ubuntu, i.e. "normal" Linux distributions [5] [23]. We stored the information of tools in a database. Each tool was given a unique name, which typically matches the package names used in the Linux distributions.

2.2 Google Tool Name Search

Popularity can be measured by web searches. Bagrow et al. found that the fame of scientists correlates with Google search counts [25]. Weiss measured open-source project success by counting web pages and project references [34]. Web searches have been used to find correlations between various terms and concepts [26], even using forensics terms [32].

A simple search using tool names would not produce good results for tool names like "Flare" or "Volatility", which have other more popular meanings. We need to adjust or filter out the meaningless search results. For this, we entered tool URLs, such as home and/or source code repository addresses, into our database. Next, we downloaded the first 100 returned pages by each tool name search and checked the pages for URLs that refer to infosec tools. This gives us information if the search returned relevant data. When checking for URLs we accept all sub-pages of the tool URL, e.g. URL `https://nmap.org/download.html` for `https://nmap.org`. We also ignored the schema difference of (`http`, `https`) and the domain name component `www` in the link assignment process.

Search hit counts follow an exponential distribution, which is normalized by taking the double logarithm of the search count. We calculate *name popularity* (P_n) for a tool as

$$P_n = \begin{cases} \log\log(\frac{S_t}{S_d}S), & \frac{S_t}{S_r} \geq 0.5 \textit{ and } \frac{S_r}{S_d} \geq 0.2 \\ \text{otherwise} & \text{undefined} \end{cases}, \qquad (1)$$

where S is the total search hit count, S_d is the number of downloaded pages, S_r is the count of pages with references to any infosec tools URLs, and S_t is the count of pages with references to the tool which name was used in the search. The condition $\frac{S_r}{S_d} \geq 0.2$ filter out pages not relevant for infosec. The condition $\frac{S_t}{S_r} \geq 0.5$ filter out cases where the tool name matches some generic term in the infosec domain. Figure 1 shows name popularity value distribution.

We used the Google Custom Search engine API to search for the names [12]. According to its documentation, the Google Custom Search is intended to create search engines for websites, blogs, etc. However, it allowed us to make scripted searches for a small cost. When we compared the search counts returned by the API to the counts we get got from the normal Google search, we noticed that the API returned significantly smaller numbers of results. It appeared that this is expected when using a custom search engine [18]. Nonetheless, the results acquired by a single custom search engine should be comparable with each other.

2.3 Google Tool URL Search

As we defined the relevant URLs for tools, we can perform the popularity search using the URLs themselves. Google Custom Search API parameter `linkSite` allows us to do so [12].

Unfortunately, we noticed that many pages returned by the URL search did not refer to the URL and we did not consider the data trustworthy. In the end, we did not use the URL search counts.

2.4 Twitter Tool Name Search

When using Twitter hits as a measurement of tool popularity, we could not just use the tool name as the search term, as with the Google searches. Thus, we used the tool categories we define later, and the term“infosec”, to narrow our search. A tweet query starts with a tool's name and the `AND` operator. Then follows the category names in braces separated by the `OR` operator. For example:

```
"nmap" AND ("infosec" OR "network scan" OR "traffic analysis" OR "packet injection" OR
"packet capture" OR "web security" OR "disk forensics" OR "file analysis" OR "host scan" OR
"memory forensics" OR "malware execution" OR "disassembly" OR "password cracking")
```

We then filtered away tweets where the search terms are found only in metainformation of the tweet. Next, we downloaded the five most recent tweets to see how many of them were related to the tool. We excluded the tool if less than two of the tweets were related. So, the tool *tweet popularity* is

$$P_t = \begin{cases} \log\log S_f,\ R'_5 \geq 2 \\ \text{otherwise}\ \ \text{undefined} \end{cases}, \tag{2}$$

where S_f is the number of filtered tweets, and R'_5 is the number of relevant tweets in the most recent five tweets. The data value distribution looks exponential, and as with name popularity, we take double logarithm of the search count. Figure 1 shows tweet popularity value distribution. The searches were performed by GetOldTweets3 library [8].

2.5 SecTools.org Ranking

The site SecTools.org provides page “Top 125 Network Security Tools” [20]. There are both commercial and open-source tools divided into various categories.

For this study, we use the rank of a tool in the Top 125 list as the *SecTools popularity*

$$P_s = -\text{SecTools.org rank}. \tag{3}$$

We negate the rank for the first tool to have the highest popularity value. Figure 1 shows SecTools popularity value distribution.

2.6 GitHub Stars

Source code repository GitHub is hosting source code for many infosec tools [9]. The repository allows users to give starts to projects. For this study, we use the star count of a tool project as *GitHub popularity*

$$P_g = \log(\text{GitHub star count}). \tag{4}$$

The star count distribution is exponential, and we use logarithm to get a normally distributed value, as shown in Figure 1. Many tools have not initially been in GitHub and they have received stars only after the project moved to GitHub. So, a popular tool might have a low GitHub star count as it only recently moved to GitHub. To study this, we also collected the project creation dates from GitHub.

2.7 Tool Cross-References

None of the presented metrics give values for all tools. So, we constructed a new metric that would give a popularity value for most tools. Many Internet pages refer to multiple infosec tools. Some list tools which are used by the author, others provide a set of tools for a particular infosec task, some may compare tools, some intent to provide a comprehensive list of tools, etc. To tap this information, we took advantage of pages we downloaded for the tool name search to see how many cross-references there are between the tools.

Tool's *cross-reference popularity* (P_c) is calculated by adding up the number of downloaded pages concerning the tool t,

$$P_c = \log \sum_p \begin{cases} 1, & t \subseteq R_p \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

where p stands for the downloaded pages and R_p is the set of tools referenced in the page. We take *log* of the value as the distribution visually looks exponential When the reference count was less than three, we treat the value as undefined. The resulting metric, shown in Figure 1, is not normally distributed.

2.8 Aggregate Popularity Metric

When we examined the different popularity metrics, we noticed a strong correlation between them, but also that they do not completely agree on the popularity of the tools. This may be as the sources are indicating a different kind of popularity. We do not know the motives of people creating web pages, tweeting, or marking tools as their favorites. We may not have found all the relevant pages, and not all references come with URLs, but a tool may be mentioned using the name only.

We need to extract the common popularity, if any, from the source metrics. Principal component analysis (PCA) is a way to reduce dimensions from data in multiple dimensions [31]. PCA constructs new uncorrelated variables, where the first variable explains as much variation as possible in a single axis. The next variable explains the second largest chunk of variation, and so on. As the source variables in our case are popularity metrics, we expect the first axis to represent the underlying overall popularity. The amount of variation explained by the first axis tells how much the sources agree on the popularity.

We decided to exclude some source metrics from the aggregate popularity:

- URL popularity results were not reliable
- The tweet popularity calculation contains subjectively selected category names and manual filtering of the non-relevant tweets
- SecTools metric is only defined for 50 tools, with many high profile tools missing, so we do not think it is representative enough to be used.

We are left with three source popularities: name popularity P_n, Github popularity F_g, and cross-reference popularity P_c. For a tool to be included, we require at least two of the three values to be defined. For PCA we need to *impute* the missing value for the tools only having two source metric values. As the source metrics correlate strongly, we use linear regression to estimate the missing value using the two source metrics with value. Provided that the first principal component from PCA captures the majority of the variance, we can extract loadings L_n, L_g, and L_c for the source popularity metrics and calculate the aggregate popularity P for tools

$$P = L_n P'_n + L_g P'_g + L_c P'_c. \tag{6}$$

We mark the source metrics with ', as we need to normalize them to zero mean and unit variance.

2.9 Dividing Tools into Categories

We want to inspect tool popularity by tool categories. There exist several different categorizations for tools, e.g. by Ellison [27], Hogue [30], Kali Linux [13], *SecTools.Org* [20]. There is no consensus on how the tools should be categorized. We need a complete and relevant categorization, which is simple to assign for a large set of tools. With all this in mind, we defined our tool categories systematically along two dimensions: tool scope and tool activity type. Tool scopes are network, storage, host, passwords, and other. Active tool interacts with a live system, while passive only monitors the system or inspects artifacts.

A network tool actively sending and receiving data is categorized as a *packet injection* tool. A tool only listening, or receiving, data is categorized as a *packet capture* tool. When a tool scans a network, we categorize it as a *network scan* tool without repeating injection and capture categories. A tool that analyzes captured traffic is a *traffic analysis* tool. An important sub-category is *web security* tools that interact with Web servers or browsers. If the tool only interacts over HTTP, we do not add other categories.

Storage scoped tools work with file systems or with file-like data in databases, e-mails, etc. Tools working with file-like objects are categorized as *file analysis* tools and the ones working directly with file systems are categorized as *disk forensics* tools. We consider all these tools passive, as they work with data at rest.

Host scoped tools work with operating system or host machine. Tools which work with a live system are *host interactive* tools, while tools working with memory are *memory forensics* tool. Tools in the *malware execution* category are used to analyze and/or sandbox a running malware (or other kinds of software). *Disassembly* tools are intended to statically analyze or reverse-engineer software.

Many tools are intended to crack passwords or other authentication tokens. We divide these tools into *password on-line* and *password off-line* tools depending on activity type. Finally, the tools which do not conveniently fit any of the categories above, are categorized as *other*. Table 1 summarizes the presented categories.

Table 1. Categorization of security analysis tools

	Network	**Storage**	**Host**	**Passwords**
Passive	Traffic analysis Packet capture	Disk forensics File analysis	Memory forensics Disassembly	Password off-line
Active	Network scan Packet injection Web security	*(always passive)*	Host interactive Malware execution	Password on-line

3 Results

The results were collected on November 2020. Our corpus included 423 different tools. Table 2 summarizes the source metrics by number of tools, type of value distribution (after taking the logarithms, when applied), and the top-3 tools by the metric. Figure 1 shows the value distribution histograms for the source metrics.

We correlated name popularity with the other metrics to see how well they agree or disagree with the popular tools, the results are plotted in Figure 2.

There is a significant correlation between the metrics. However, individual tool places are scattered, so the metrics do not agree on the ranking of each tool. Other metrics, except cross-reference, do not give ranking for tools that are low in the name popularity. The cross-reference metric provides popularity values for almost all tools which have name popularity. However, the correlation is somewhat weaker than by the other metrics. The values of cross-reference popularity concentrate around the median value, which means that small changes in those cross-reference values would mean large swings in the ranking of a tool.

Table 2. Source popularity metrics

Metric		**Tools**	**Distribution**	**Top-3**
Name	P_n	237	Normal	wireshark, nikto, nmap
Tweet	P_t	239	Unspecified	metasploit, wireshark, nmap
GitHub	P_g	232	Weak normal	jadx, ghidra, metasploit
SecTools	P_s	50	Linear	wireshark, metasploit, aircrack-ng
Cross-ref.	P_c	388	Unspecified	nmap, wireshark, metasploit

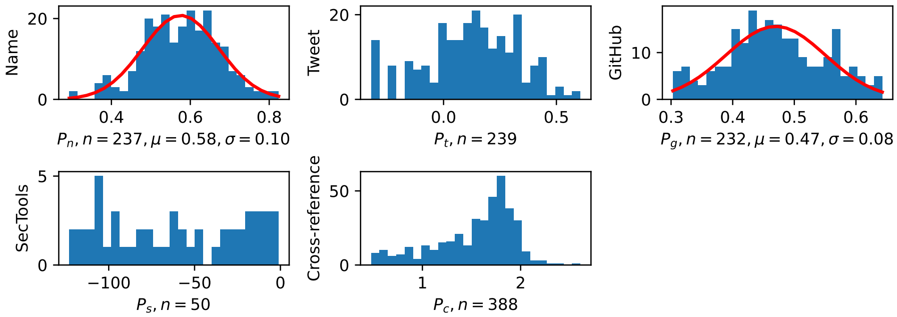

Fig. 1. Source metric value distribution histograms. Sample size is n. Mean (μ), standard deviation (σ), and ideal normal distribution graph is given for the metrics which follow normal distribution

3.1 Most Popular Tools

The aggregate popularity was created using principal component analysis of source metrics name popularity P_n, GitHub popularity P_g, and cross-reference popularity P_c. To calculate the most popular tools, we first imputed the missing source metric values for the tools which only had two values, using linear regression and the two values we know. Then we performed the analysis with

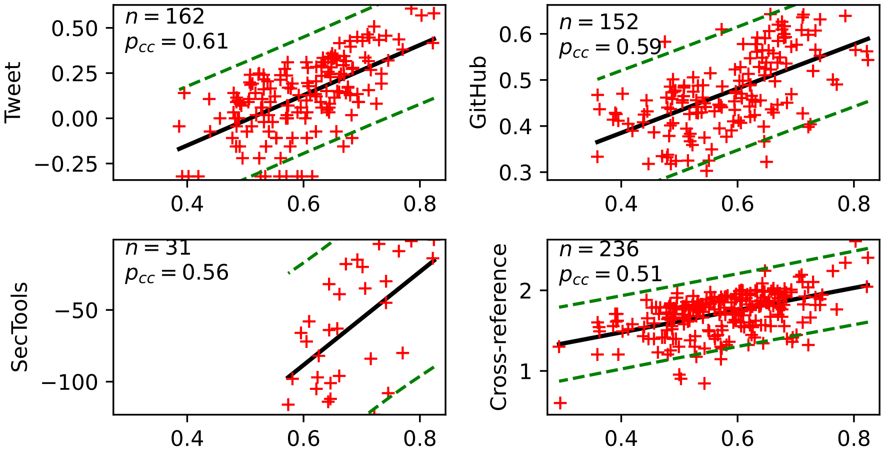

Fig. 2. Correlation of name popularity and other source metrics, n = sample size and p_{cc} = Pearson correlation coefficient. Dotted lines mark the 80% correlation band.

values normalized to zero mean and unit variance. The aggregate popularity was calculated for 310 tools. The first component explains 70% of the variance, and we use it as the aggregate popularity metric. We do not use the other two components, explaining 20% and 9% of the variance. Using the loadings from the first component, we can calculate the aggregate popularity for tools like this:

$$P = 0.6273P'_n + 0.5578P'_g + 0.5435P'_c \tag{7}$$

Where P'_n, P'_g, and P'_c are normalized (zero mean and variance of one) popularity values of name, GitHub, and cross-reference. The loadings have the same sign and their values are close to each other, so the aggregate popularity is quite close to the average of the source popularities. Table 4 shows the resulting 100 most popular tools. The three most popular tools are Metasploit penetration testing framework [15], Nmap network scanner [16], and Wireshark traffic capture and analysis tool [24]

3.2 Reliability of the Results

Next we consider how reliable the average popularity metric is. For the Google search, it is uncertain how well the value matches the real page count as we lack visibility to the internals of the search system. The GitHub stars count is exact, but some tools have been longer than others in GitHub. In our analysis, there was only a small correlation between project age and the start count. The calculated cross-reference is not accurate if we have not found all relevant pages, or some references do not use the URLs we have collected.

We simulated the impact of errors by making changes to source metric values, recalculating the aggregate popularity, and noting the changes in the tool ranking due to the simulated errors. Table 3 shows the variation at ranks 3, 20, 50, and 100 for GitHub popularity error range $+100\%... -25\%$ and name popularity and cross-reference popularity error ranges $+100\%... -25\%$ and $+200\%... -50\%$. For example, when a tool around ranking 50 have name cross-correlation popularity error in range $+100\%... -25\%$, then the tool aggregate popularity ranking would vary by a maximum of ±30 positions.

Table 3. Ranking variation caused by source metric errors, by ranking

Rank	$P_g\ {}^{+100}_{-25}\%$	$P_n\ {}^{+100}_{-25}\%$	$P_n\ {}^{+200}_{-50}\%$	$P_c\ {}^{+100}_{-25}\%$	$P_c\ {}^{+200}_{-50}\%$
3	±1	±1	±1	±2	±2
20	±9	±6	±9	±13	±30
50	±20	±16	±21	±30	±37
100	±27	±22	±32	±39	±66

We should also consider the impact of imputing source metric values for those tools which only had two source values available. Within the set of 310 tools with aggregate popularity, the nick popularity value was imputed for 74 tools, the github popularity for 84 tools, and 152 tools had all three source metrics. Cross reference popularity was available for all included tools.

3.3 Most Popular Tools per Category

We then divided the top 100 tools into the categories we had defined. Table 5 then presents the tools in each of the categories. The superscript after each tool is the popularity rank of the tool. There is substantial overlap in the categories, as many tools are in several different categories. If we calculate the tool counts per scopes from Table 1 we get that 51 tools are in the network categories, 27 tools in host categories, 15 tools in storage categories, 13 tools in passwords categories, and four in the other category.

3.4 Availability of the result data

Tool data, including tool URLs, and raw popularity data is available in the Internet at https://gitlab.com/CinCan/infosec-tools.

4 Discussion

In this study, we examined the popularity of open-source infosec tools. We did not find earlier studies on the subject. Objective data on tool popularity is required e.g. to use representative tools in research, to examine the properties of the popular tools, and to find alternative tools for a task.

Table 4. Top 100 tools

Tool	Name	GitHub	Cross	Tool	Name	GitHub	Cross
metasploit	4	3	3	arachni	53	53	81
nmap	3	32	1	etherape	60	-	26
wireshark	1	52	2	dsniff	62	-	30
nikto	2	38	14	sleuthkit	56	70	31
mitmproxy	19	4	9	hping	25	111	33
mimikatz	20	11	6	ubertooth	42	84	41
sqlmap	8	5	64	capstone	-	34	151
hashcat	15	15	15	linenum	-	45	129
ghidra	17	2	95	social-engineer-toolkit	90	23	180
radare2	31	8	17	sslyze	98	59	39
tcpdump	6	79	5	rkhunter	51	-	85
apktool	10	12	99	dirbuster	57	-	74
powersploit	55	20	4	patator	69	58	115
ophcrack	21	-	7	rainbowcrack	80	-	37
socat	7	-	43	extundelete	67	-	68
smali	11	30	55	dnsrecon	91	82	22
masscan	39	7	61	pdfcrack	87	-	42
bloodhound	-	29	10	steghide	66	-	88
wpscan	26	28	19	sqlninja	97	-	25
dex2jar	30	17	45	oletools	100	81	28
aircrack-ng	12	64	8	bytecode-viewer	123	9	230
dnspy	34	6	140	volatility	-	43	212
john	-	40	12	dcfldd	88	-	60
ntop	9	46	103	bdf	-	56	166
jadx	48	1	191	skipfish	43	158	16
lynis	33	19	102	ngrep	46	150	27
p0f	16	-	50	kismet	-	124	13
nishang	63	33	11	weevely	47	63	235
cyberchef	49	13	91	tcpick	102	-	40
ilspy	35	10	190	wafw00f	54	61	228
impacket	73	24	23	ncrack	74	131	24
whatweb	41	54	29	fping	40	136	78
ddrescue	29	-	35	iodine	-	47	217
beef	-	27	72	netsniff-ng	94	109	21
zaproxy	70	18	79	dc3dd	93	-	73
crackmapexec	71	41	20	invoke-obfuscation	139	67	34
wifite	28	55	56	safecopy	99	-	71
clamav	5	87	138	winhex	23	-	270
cuckoo	-	37	62	sslscan	79	88	148
binwalk	38	21	174	rarcrack	114	-	46
theharvester	50	35	76	mtr	-	75	156
wfuzz	64	49	32	httpry	116	-	49
jd-gui	32	14	234	slowhttptest	104	101	63
chkrootkit	37	-	36	tcpreplay	68	128	110
responder	-	50	47	inetsim	118	-	52
ratproxy	45	-	18	exiftool	24	-	281
gobuster	44	44	108	angryip	105	62	204
w3af	36	48	125	oledump	111	-	84
androguard	82	51	44	testdisk	18	162	219
scapy	22	25	261	ssdeep	13	188	196

We identified potential sources to extract tool popularity data: Google name search, Twitter search, GitHub stars, SecTools.org ranking, and tool cross-references. We noted a clear correlation between the different metrics. There is an underlying overall popularity, which is reflected on all of the metrics. However, the consensus does not go to the individual tool level, there is a lot of variation on the rank of a tool between the metrics. We used principal component analysis to

Table 5. Top 100 tools by categories, tool$^{\text{ranks}}$ by superscript

Network scan	metasploit^{1} nmap^{2} sqlmap^{7} powersploit^{13} masscan^{17}	bloodhound^{18} whatweb^{32} crackmapexec^{36} wifite^{37} theharvester^{41}	gobuster^{47} arachni^{51} hping^{55} dnsrecon^{66} skipfish^{75}	kismet^{77} fping^{82} sslscan^{89} mtr^{91} angryip^{97}
Traffic analysis	wireshark^{3} tcpdump^{11}	ntop^{24} p0f^{27}	scapy^{50} dsniff^{53}	netsniff-ng^{84}
Packet capture	wireshark^{3} tcpdump^{11} aircrack-ng^{21} ntop^{24}	p0f^{27} scapy^{50} etherape^{52} dsniff^{53}	hping^{55} ubertooth^{56} ngrep^{76} kismet^{77}	tcpick^{79} netsniff-ng^{84} httpry^{92}
Packet injection	metasploit^{1} mitmproxy^{5} sqlmap^{7} socat^{15}	aircrack-ng^{21} impacket^{31} crackmapexec^{36} wfuzz^{42}	responder^{45} scapy^{50} hping^{55} ubertooth^{56}	iodine^{83} netsniff-ng^{84} slowhttptest^{93} tcpreplay^{94}
Web security	nikto^{4} mitmproxy^{5} wpscan^{19} whatweb^{32}	beef^{34} zaproxy^{35} wfuzz^{42} ratproxy^{46}	w3af^{48} arachni^{51} sslyze^{60} dirbuster^{62}	sqlninja^{69} skipfish^{75} wafw00f^{80}
Disk forensics	ddrescue^{33} sleuthkit^{54}	extundelete^{65} dcfldd^{73}	dc3dd^{85} safecopy^{87}	testdisk^{99}
File analysis	radare2^{10} clamav^{38}	pdfcrack^{67} oletools^{70}	winhex^{88} exiftool^{96}	oledump^{98} ssdeep^{100}
Host interactive	metasploit^{1} mimikatz^{6} powersploit^{13}	bloodhound^{18} lynis^{26} nishang^{28}	chkrootkit^{44} linenum^{58} rkhunter^{61}	bdf^{74} weevely^{78}
Memory forensics	volatility^{72}			
Malware execution	radare2^{10}	cuckoo^{39}	inetsim^{95}	
Disassembly	ghidra^{9} radare2^{10} apktool^{12} smali^{16}	dex2jar^{20} dnspy^{22} jadx^{25} ilspy^{30}	binwalk^{40} jd-gui^{43} androguard^{49} capstone^{57}	bytecode-viewer^{71}
Password on-line	sqlmap^{7} ophcrack^{14}	aircrack-ng^{21} crackmapexec^{36}	wifite^{37} dsniff^{53}	patator^{63} ncrack^{81}
Password off-line	hashcat^{8} john^{23}	rainbowcrack^{64} pdfcrack^{67}	rarcrack^{90}	
Other	cyberchef^{29}	social-engineer-toolkit^{59}	steghide^{68}	invoke-obfuscation^{86}

extract the aggregate popularity from three source metrics: Google name search, GitHub stars, and tool cross-references. A tool had to have two out from the three source metrics to be included to the aggregate metric. We imputed values for missing GitHub star counts and Google name search counts. For these tools, the resulting aggregate metric is only based on two source metrics. Especially when we lack GitHub star count, we are quite dependent on Google search results as even the cross-reference count is calculated from the pages originally returned by Google.

The common popularity component explained 70% of the variance in the metrics. We then ranked the top 100 tools by the aggregate metric. The three top ranking tools are Metasploit, Nmap, and Wireshark. Generally, the top 100 tools contains roughly the tools we expected to be high on the list and many tools which we were not familiar with.

We estimated the reliability of the aggregate metric and concluded that errors in the source metrics in the range of -50%...+100% change the ranking of the tools around ranking 50 by a maximum by ±30 positions, less in higher rankings and if source errors are smaller. So, one should not take a tool rank as an exact number, but rather as an estimate of the popularity of the tool. We may have caused errors by missing a popular tool or failing to collect an relevant URL. We tried to avoid missing tools by collecting the corpus from many sources. Further, we have reviewed the URL lists by multiple people and also checked popular URLs in the downloaded pages.

To present the popularity by tool categories, we divided the tools by their scope and type of activity. Roughly half of the tools, 51 tools, are in the network scope and all top 5 tools are network tools. All network categories, network scan, traffic analysis, packet capture, packet injection, and web security, are popular. The large number of versatile network tools highlight the interest to the network security. Some of the packet and traffic tools may also be used for non-security development and administration.

In the storage scope, there are 15 tools equally split between disk and file tools. The scope is less popular than the network tools. There are 27 tools in the host scope, the disassembly tools being the most popular. Host interactive contains tools for all major operating systems, Windows being the most popular. Memory forensics and malware execution are a lot less popular categories, which may indicate that these tasks are less practised. In the password scope there are 13 tools, several tools in both on-line and off-line categories. There are just four tools in the other tools category.

Several popular tools are tool suites or frameworks, e.g. Metasploit and Radare 2, and they are in many categories. We assign just one popularity metric for a tool, so a tool popular in many categories may be in reality popular in only some of them. It was also quite challenging to categorize so many tools and sometimes it was not easy to determine the correct categories to the tools by reading the tool documentation only.

4.1 Future Work

This study provides information on which open-source tools, and tool categories, are popular in the infosec. The data can be used to direct future research to the relevant tools and tool categories.

One could study the properties of the popular tools and tool categories, e.g. the used programming language, tool age, number of active developers, etc. Further, it could be useful to map tools into infosec tasks to get insight into tasks and the tools used in them. The tool categorization could be refined to better capture the types of tools available. It would also be critical to detect if there are popular tools that have some risks associated with them, like lack of maintenance or use of outdated technology.

The reliability of the tool rankings could be improved by adding more sources of data, e.g. by scraping more web pages or scraping academic publications. It would be important to lessen the uncertainty by the use of the Google search engine. This is highlighted by the fact that after this study we have observed changes in the behavior of the Google Custom Search engine. It appears to return fewer pages compared to the results collected in November 2020. This does not seem to impact the relative rankings of the tools, but clearly, it would be highly beneficial to have more sources for the data.

Acknowlegements This work is done in the CinCan project funded by CEF programme (2016-FI-IA-0095) and in the SECREDAS project funded by Horizon 2020 programme (grant agreement nr. 783119) and by Business Finland.

References

1. ArchStrike linux. https://archstrike.org/, accessed: 2021-04-14
2. BlackArch linux. https://blackarch.org/, accessed: 2021-04-14
3. CAINE linux. https://www.caine-live.net/, accessed: 2021-04-14
4. CinCan project. https://cincan.io/, accessed: 2021-04-14
5. Debian linux. https://www.debian.org/, accessed: 2021-04-14
6. Fedora Security Lab. https://labs.fedoraproject.org/en/security/, accessed: 2021-04-14
7. FLARE VM image. https://github.com/fireeye/flare-vm, accessed: 2021-04-14
8. GetOldTweets3. https://pypi.org/project/GetOldTweets3/, accessed: 2021-04-14
9. GitHub home page. https://github.com, accessed: 2021-04-14
10. GitLab home page. https://gitlab.com/, accessed: 2021-04-14
11. Google Code Archive. https://code.google.com/archive/, accessed: 2021-04-14
12. Google Custom Search. https://developers.google.com/custom-search, accessed: 2021-04-14
13. Kali linux. https://www.kali.org/, accessed: 2021-04-14
14. Merriam-Webster. https://www.merriam-webster.com/, accessed: 2021-04-14
15. Metasploit. https://www.metasploit.com/, accessed: 2021-04-14
16. Nmap. https://nmap.org/, accessed: 2021-04-14
17. Pentoo linux. https://www.pentoo.ch/, accessed: 2021-04-14
18. Programmable search engine help: Custom Search vs Google.com. https://support.google.com/programmable-search/answer/70392, accessed: 2021-04-14
19. REMnux linux. https://remnux.org/, accessed: 2021-04-14
20. SecTools.Org. https://sectools.org/, accessed: 2021-04-14
21. SIFT workstation linux. https://digital-forensics.sans.org/community/downloads, accessed: 2021-04-14
22. Sourceforge home page. https://sourceforge.net/, accessed: 2021-04-14
23. Ubuntu linux. https://ubuntu.com/, accessed: 2021-04-14
24. WireShark. https://www.wireshark.org/, accessed: 2021-04-14
25. Bagrow, J., Rozenfeld, H., Bollt, E., ben Avraham, D.: How famous is a scientist? – famous to those who know us. EPL (Europhysics Letters) **67** (05 2004)
26. Cilibrasi, R.L., Vitanyi, P.M.B.: The google similarity distance. IEEE Transactions on Knowledge and Data Engineering **19**(3), 370–383 (2007)

27. Ellison, D., Ikuesan, R.A., Venter, H.S.: Ontology for reactive techniques in digital forensics. In: 2019 IEEE Conference on Application, Information and Network Security (AINS). pp. 83–88 (2019)
28. Harichandran, V.S., Breitinger, F., Baggili, I., Marrington, A.: A cyber forensics needs analysis survey: Revisiting the domain's needs a decade later. Computers & Security **57**, 1 – 13 (2016)
29. Hibshi, H., Vidas, T., Cranor, L.: Usability of forensics tools: A user study. In: 2011 Sixth International Conference on IT Security Incident Management and IT Forensics. pp. 81–91 (2011)
30. Hoque, N., Bhuyan, M.H., Baishya, R., Bhattacharyya, D., Kalita, J.: Network attacks: Taxonomy, tools and systems. Journal of Network and Computer Applications **40**, 307 – 324 (2014)
31. Jolliffe, I., Cadima, J.: Principal component analysis: A review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **374**, 20150202 (04 2016)
32. Karie, N.M., Venter, H.S.: Measuring semantic similarity between digital forensics terminologies using web search engines. In: 2012 Information Security for South Africa. pp. 1–9 (2012)
33. Mandal, N., Jadhav, S.: A survey on network security tools for open source. In: 2016 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC). pp. 1–6 (2016)
34. Weiss, D.: Measuring success of open source projects using web search engines. In: Scotto, M., Succi, G. (eds.) Proceedings of The First International Conference on Open Source Systems (OSS 2005), Genova, Italy. pp. 93–99 (2005)