(Work in Progress): Clustering-Based Characterization of Database Server Ransom Scams

Kevin van Liebergen*†, Gibran Gomez*†, Srdjan Matic*, and Juan Caballero*
*IMDEA Software Institute, †Universidad Politécnica de Madrid

***Abstract*—We perform the first study of *database server ransom scams*, a class of attacks where attackers scan for database servers, log in by leveraging the lack of authentication or by using guessed credentials, drop the database contents, and demand a ransom to return the deleted data. To enable our study, we leverage 5,792 unique ransom notes collected by an Internet scanning engine from 27,750 compromised ElasticSearch and MySQL database servers over a period of two years. We propose a novel automated three-step clustering approach. First, it leverages similarity of the ransom notes text to identify servers infected by the same campaign. Then, it identifies campaigns run by the same threat group by merging note similarity clusters that reuse IOCs (i.e., Bitcoin payment addresses, email addresses, Tor onion addresses). Finally, it merges IOC reuse clusters whose notes contain Bitcoin addresses co-spent in Bitcoin transactions. This process groups the 27,750 database server infections into 94 clusters, identifying a dominant threat group that is responsible for 49% of the infections.**

I. Introduction

Database servers are a key asset of digital services as they store the data required for the services to operate. Given the large amount of data they store and the criticality of that data, database servers are a valuable target for attackers. Compromising a database server allows attackers to block the server owners from accessing their data, and to demand a ransom to restore access to the data.

Compared to ransomware attacks that encrypt or block access to data stored in desktop computers or mobile devices [1], [2], [3], [4], [5], attackers targeting database servers do not need to employ social engineering techniques to convince victims to install their malware. Instead, they can identify targets by scanning the IPv4 address space for Internet-connected vulnerable database servers [6] or leverage the information obtained from Internet scanning engines [7], [8], [9]. Furthermore, popular database software (e.g., MySQL, ElasticSearch) can run on top of different platforms (e.g., Linux, Windows, MacOS, FreeBSD, UNIX) enabling attackers to target database servers regardless of the underlying operating system the server uses.

In this work, we perform the first study of *database server ransom scams*. In this class of attacks, the attacker identifies target database servers by scanning the IPv4 address space or by using Internet scanning engines. Once a target is located, the attacker tries to log in to the database server by leveraging the lack of authentication, using default credentials, or guessing weak credentials. If the attacker manages to log into the database server, it examines the databases the compromised account has access to, drops the content of those databases, and leaves a ransom note (e.g., by creating a new database table with a catchy name) that tells the victim how to get the data back. The note provides ransom payment details (e.g., a Bitcoin address and the ransom amount) or a contact method to request the payment details (e.g., email address, Tor hidden service address).

We consider such attacks scams, as opposed to ransomware, because the data is deleted rather than encrypted and because it is highly unlikely that the victim can get the deleted data back, even if the ransom is paid. For the attacker to be able to return the data, it needs to have exfiltrated the data before its deletion. Whether such exfiltration is possible depends on the privileges of the compromised account. We have searched for reports of database server ransom scams in abuse databases [10] and using search engines. So far, we have not found any report stating that the data was recovered after paying the ransom. In contrast, the recent DeadBolt server ransomware, which encrypts data stored in network-attached storage (NAS), releases the decryption key after the ransom is paid [11].

Database server ransom scams are not new, e.g., a campaign in January 2017 infected 28K MongoDB servers that lacked authentication [12]. However, database server ransom scams have not yet been systematically analyzed, arguably due to the difficulty of obtaining sizable data about them. To enable our analysis of database server ransom scams we leverage publicly available data independently collected by the LeakIX Internet scanning engine [9]. LeakIX has vulnerability plugins that scan for popular database servers that lack authentication. From those vulnerable servers, it collects the ransom notes left by the attackers. From LeakIX data, we extract 5,792 unique ransom notes collected from 27,750 compromised ElasticSearch and MySQL servers over two years (May 2021 - May 2023).

One challenge when analyzing scams is how to identify which victims may have been affected by the same scam campaign and which campaigns may be run by the same group of attackers. To address this challenge, we propose a novel three-step automated clustering approach. The first step groups ransom notes by the similarity of their text. A note similarity (NS) cluster captures a scam campaign whose ransom notes follow the same template, allowing for small changes in the note text such as different payment addresses or minor text structure modifications. The second step extracts indicators of compromise (IOCs) from the ransom notes (i.e., Bitcoin addresses, email addresses, Tor onion addresses) and merges

ISBN:978-84-09-62140-8

NS clusters that share IOCs into IOC reuse (IR) clusters. This step allows to identify campaigns that are run by the same group, but use different ransom note templates. The third step augments the Bitcoin addresses extracted from the ransom notes with information extracted from the Bitcoin blockchain to generate the co-spending or multi-input (MI) clusters [13], [14], [15], [16]. If two Bitcoin addresses in the same MI cluster have been extracted from notes in different IR clusters, those IR clusters are merged, as they are operated by the same group. The produced clusters capture threat groups managing one or multiple scam campaigns.

Using the collected dataset and the produced clusters, we answer the following research questions.

(RQ1) How prevalent are database server ransom scams? We found 27,750 infected database servers (identified by their IP address). This number is a lower bound given that LeakIX only collects ransom notes from unauthenticated MySQL and ElasticSearch database servers. Our prevalence estimate does not include database servers using other database software (e.g., Oracle, MongoDB), as well as MySQL and ElasticSearch databases with some, but weak, authentication, which may have been compromised through credential guessing.

(RQ2) How many groups are responsible for the database server infections? Our clustering groups the 27,750 infections into 94 group clusters, and identifies one dominant group responsible for 49% of all infections. This group runs multiple campaigns over time and targets different geographical locations (captured by the use of ransom notes written in English and Chinese).

While this research is work-in-progress, we can already report some contributions:

- We perform the first analysis of database server ransom scams, where the attackers target database servers with weak or no authentication, drop the database contents, and demand a ransom to recover the data.
- We develop a novel three-step automated clustering approach to group database server infections into campaigns launched by the same threat group.
- We apply our clustering to 27,750 infected database servers, producing 94 group clusters. We identify a dominant group responsible for 49% of the infections.

II. Database Server Ransom Scams

Attackers running database server ransom scams typically perform four actions: (1) identify target database servers; (2) break into the database servers; (3) drop the database contents; and (4) place a ransom note. To identify target database servers, attackers can scan the IPv4 address space on default database ports (e.g., 3306/tcp for MySQL, 9200/tcp for ElasticSearch), or leverage Internet scanning engines such as Shodan [8], Censys [7], and LeakIX [9]. Once a target server has been identified, the attacker tries to access it by exploiting a vulnerability or compromising an account by guessing account credentials. While some recent server ransomware families have leveraged zero-day vulnerabilities to compromise servers [17], [18], public reports on database server ransom scams mention the attacker guessing credentials to log into the server [19]. The guessed credentials can correspond to database accounts or server accounts (e.g., accounts with SSH access).

Next, the attacker executes the payload, i.e., exfiltrate data (optional) and drop the database contents. Exploiting a vulnerability or compromising an SSH account with enough privileges may allow users to run their own code on the server. On the other hand, compromising a database account may not provide the attackers with the ability to run code, but allows them to perform other actions that violate data integrity such as dropping the database tables, adding a new user to the database, or modifying the stored data. Most often, the attacker simply executes database commands and there is no malware installed on the server [20]. The lack of malware makes attributing the scam harder and motivates our automated clustering approach.

Finally, the attackers place a ransom note to tell the database server administrator that the data was purposefully deleted and how to pay a ransom to get the data back. Alternatively, they provide a contact method (e.g., an email address) for the victim to obtain the ransom payment details. If the attackers only compromised a database account, they may be limited to placing the ransom note in a database table, e.g., creating a new table with a name such as “README: how to recover your data”.

Such attacks are often called ransomware because the attacker demands a ransom for the victim to recover its data. However, it is more accurate to consider them scams, as there is no evidence that the attackers exfiltrated the data, and they are able to return it upon payment. Even if the data could be exfiltrated, the attackers may decide that it is not worth the effort as the exfiltration may take a long time, it requires significant data storage resources from the attacker, and makes it easy to identify the attacker’s server receiving the data. Furthermore, if the victim pays, returning the data to the compromised server would be too risky as that server would likely be monitored at that point. Placing it on a Tor hidden service is a better option, but if the data is large the attacker still needs to invest in storage.

III. Datasets

We obtain our data from the LeakIX [9] Internet scanning engine. LeakIX has plugins that check for specific server vulnerabilities, and provides a search API to obtain the events produced by each plugin. Among others, LeakIX provides plugins for identifying unauthenticated databases (e.g., ElasticSearch, MySQL). These plugins offer an *infected* field that identifies events where LeakIX identified an unauthenticated database that contains a ransom note. To identify ransom notes, LeakIX applies regular expressions to the names of files, collections, and tables in the database. The regular expressions look for keywords in the names of ransom notes (e.g., readme, hacked, infected). This approach may miss infections that do not use the expected keywords to name the ransom note.

Plugin	Events	IP	Port	CC	ASN	Notes	Addr.
Elastic	84,746	20,104	37	107	1,229	3,557	67
MySQL	38,158	7,691	7	102	922	2,235	209
All	122,904	27,750	41	124	1,745	5,792	272

TABLE I: LeakIX dataset summary.

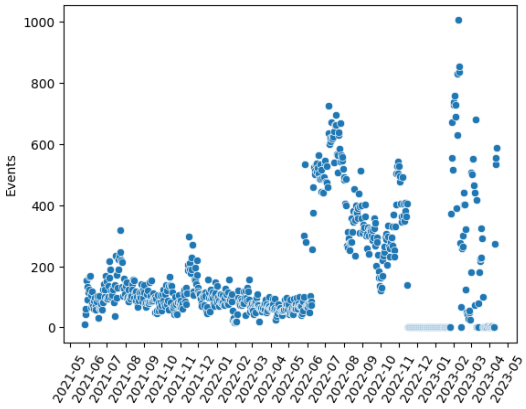

Fig. 1: Daily number of infection events.

We build a dataset by querying LeakIX for events with the infected flag set for the *ElasticSearchOpen* and *MysqlOpen* plugins, which collect the ransom note text. We query LeakIX once a month for 6 months between November 2022 and April 2023. We obtain 204,257 initial events, of which only 60.2% contain a ransom note, 52.9% of the ElasticSearch infected events, and 86.5% of the MySQL infected events. We discard events without a ransom note. The regular expressions used by LeakIX for identifying ransom notes can occasionally introduce false positives, e.g., detect as a ransom note a database table with configuration information. To address this issue, we filter 0.64% of the events whose notes do not contain at least one keyword associated with ransom (e.g., hacked, backed, BTC, bitcoin).

Table I summarizes the filtered dataset, which contains 122,904 events with valid ransom notes obtained from 27,750 infected IP addresses running ElasticSearch or MySQL services on 41 ports. The IP addresses are hosted in 124 countries and belong to 1,745 ASNs. We use the number of IP addresses to approximate the number of infected database servers. This may overestimate the infected servers as the same server could appear multiple times in the dataset under different IP addresses. An infected server stays marked as such in LeakIX until it is cleaned (i.e., LeakIX regular expressions no longer detect an infection). When we queried the LeakIX for ElasticSearch or MySQL infected servers, we notice infection events that date back to 25th of May 2021, over a year earlier than our data collection start date. In Figure 1 we report the number of daily infection events.

ElasticSearch and MySQL software can run on a variety of platforms. LeakIX was able to identify the OS in 88.9% of the ElasticSearch events and 98.5% of the MySQL events. Of the ElasticSearch events, 87.4% correspond to Linux servers, 1.5% to Windows, and the remaining 0.03% include Mac OS, FreeBSD, OpenBSD, Solaris, and UNIX. Of the MySQL events, 78.9% are for Linux servers, 18.9% for Windows servers, and 0.7% are for other OSes.

Clustering type	All	Non-singl.	Singl.	Max.	Med.	Mean
Note similarity	408	94 (94.6%)	314	3,507	5	58.3
+IOC reuse	99	62 (99.4%)	37	4,046	10	92.8
+MI clust. (Groups)	94	59 (99.4%)	35	4,073	10	97.8

TABLE II: Clusters obtained after each clustering step. Bottom row corresponds to the final clustering. Percentage is over number of notes.

The dataset contains 5,792 unique ransom notes (identified by SHA256 hash), and the same ransom note appears on average in 35.3 events, from one or multiple IP addresses. Occasionally, the same IP shows multiple ransom notes, likely because it was re-infected. For MySQL, the ransom notes are in plain text, while for ElasticSearch they are in JSON format with the attackers placing the ransom note in a field that entices to read its content (e.g., *Readme*, *readThis*). We find notes in two languages: English (89.0%) and Chinese (10.7%). The remaining 0.3% notes are too short to identify the language.

IV. Clustering

Server infections in the dataset may come from a variety of campaigns run by the same or different threat groups. We apply a novel clustering approach to group server infections belonging to campaigns run by the same group. Our clustering comprises three steps. The first step clusters unique ransom notes by computing their text similarity. We call the resulting clusters *note similarity (NS) clusters*. This step captures ransom notes that belong to the same campaign and thus use a common template in the campaign's ransom notes. Since attackers may run multiple campaigns, this step may produce multiple NS clusters (e.g., one per campaign) that are operated by the same attackers. The second step merges NS clusters that share IOCs, namely Bitcoin addresses, email addresses, and onion addresses. We call the resulting merged clusters *IOC reuse (IR) clusters*. This step identifies different campaigns run by the same group that reuse indicators. The third step analyzes the Bitcoin blockchain to further merge IR clusters if they contain addresses that belong to the same multi-input (MI) cluster [13], [14], [15], [16]. We call the resulting clusters *group clusters*. This step identifies campaigns run by the same group that do not reuse contact indicators across campaigns, but aggregate the profits from different campaigns before cashing out. For each group cluster, we obtain the associated LeakIX infection events where the note appears to produce clusters of server infections.

Our clustering groups the 5,792 unique ransom notes into 94 group clusters, each characterizing infection campaigns run by the same group. Table II summarizes the clusters obtained after each step and Table IV the top 5 clusters by number of notes at each clustering step. Next, we detail the three clustering steps using these tables.

	Clusters			
Threshold	**Total**	**Non-singleton**	**Singleton**	**Precision**
5	553	106	447	100.00%
6	408	94	314	100.00%
7	321	82	239	99.98%
8	252	77	175	99.98%
9	216	71	145	98.84%
10	184	65	119	81.27%

TABLE III: Note similarity threshold evaluation. The best threshold is 6 because it provides perfect precision while minimizing the number of clusters.

A. *Note Similarity*

We first cluster the 5,792 ransom notes by the similarity of their text into 408 NS clusters. The goal of this step is to capture ransom notes that have very similar text such as those having the same text template with different indicators (e.g., different Bitcoin ransom payment addresses) or different names of databases dropped (specific to the victim server), as well as those with minor differences in their text. To capture text similarity we use SimHash [21], a fuzzy hash that produces similar digests for similar inputs. SimHash is designed to detect near-duplicate data more efficiently than classical pairwise similarity metrics, like the Jaccard index, which has quadratic complexity. It is Google's preferred algorithm for detecting near-duplicate web pages when crawling the Web [22]. We compute the SimHash of each note and add the digests to an index where digests with similarity above a threshold are placed in the same bucket, i.e., NS cluster.

We evaluate different threshold values. Since we have no ground truth, for each threshold value, we manually examine the 10 largest clusters, which contain 70%–85% of all notes depending on the threshold. We examine each of the entries in the top 10 clusters classifying them as false positives (FPs) if they are not similar to the rest of the cluster, and true positives (TPs) otherwise. TPs and FPs are aggregated across the 10 clusters and we compute $Precision = TP/(TP + FP)$. Table III summarizes the threshold selection evaluation. As the threshold increases, the precision decreases and the number of clusters reduces. We cannot compute the recall as we lack a ground truth to identify the false negatives. But, the fewer clusters with perfect precision the higher the recall will be.

We select 6 as threshold because it produces the smallest number of clusters (408) with perfect precision. The first row in Table II captures the results of the note similarity clustering. Of the 408 clusters, 94 contain multiple notes and 314 are singletons. Excluding the singleton clusters, the median cluster size is 5 notes with a mean of 58 notes, and the largest cluster has 3,507 notes. Manual examination shows that NS clusters are very tight containing small variations of the same template introduced by replacing indicators (e.g., Bitcoin addresses, emails) in the text or by performing small modifications to the text. All NS clusters contain notes written in a single language.

Table IV details the 5 largest clusters for each clustering step. For each cluster, it shows an identifier and a label; the number of notes; the number of Bitcoin addresses, email addresses, onion addresses, and ransom amounts in the notes; the number of events; the number of IP addresses and DB plugins in the events (i.e., MySQL, ElasticSearch, or both); and the number of NS clusters merged into the cluster.

For each cluster, we compute the longest common subsequence (LCS) of the notes in the cluster. From the produced sequence of substrings, we remove all the substrings with less than 4 characters and add a separator ("[...]") between the substrings. This produces a pattern that captures the common template for the notes in the cluster. For example, for cluster NS4 in Table IV the LCS string is: "All your data was backed up from your server. You need to email us at rambler+[...]@onionmail.org to recover your data. If you dont contact us we will reach the General Data Protection Regulation, GDPR,". This template captures that all notes in the cluster start with the same prefix and contain contact emails with the same structure starting with "rambler+" and using the onionmail.org mail service. The cluster label column in Table IV, shows the initial part of the cluster's LCS string. Notes in clusters NS1 and NS5 start with the same prefix "All your data is backed up. You must pay", suggesting that they belong to the same campaign, even if their notes had enough differences to be placed in different NS clusters. Next, we address such cases, as well as identify different campaigns of the same group, by merging clusters that share IOCs.

B. *IOC Reuse*

A threat group may be responsible for multiple campaigns. To identify campaigns run by the same threat group, we merge clusters that share IOCs. For example, if notes in different NS clusters request the victims to pay to the same Bitcoin address, those campaigns should be run by the same group, as the profits are being aggregated into the same Bitcoin address.

IOC extraction. To extract IOCs from the ransom notes we use the *iocsearcher* IOC extraction tool [23]. *iocsearcher* uses regular expressions to extract a large variety of IOCs including addresses for a dozen different blockchains, email addresses, and networking endpoints such as domains, URLs, IP addresses, and onion addresses for Tor hidden services. Compared to other IOC extraction tools, *iocsearcher* validates the checksum embedded in blockchain addresses and Tor onion addresses (version 3) avoiding FPs due to hashes that are not valid Bitcoin or onion addresses.

The most common IOCs are email addresses (3,846) where victims can contact the attacker, URLs (958), and Bitcoin addresses (272) for the victim to pay the ransom. There are also 3 Tor onion addresses, two version 3 addresses and one in the deprecated version 2 format. Onion addresses are used in notes that ask the victim to visit a Tor hidden service and provide an identifier in the ransom note to obtain the payment information. There is also one Monero payment address. Addresses for other popular blockchain (e.g., Ethereum, BitcoinCash, Cardano) are not present. Other extracted IOCs are domain names (64) and universal unique identifiers (32).

The IOC reuse step focuses only on three IOC types: blockchain addresses, email addresses, and onion addresses.

Clustering	ID	Cluster Label	Notes	Addr.	Email	Onion	Val.	Events	IP	DB	NSC
Note similarity	NS1	All your data is a backed up. You must pay	3,507	38	3,481	0	15	27,803	7,798	2	1
	NS2	To recover your lost databases and avoid leaking it	950	0	0	2	0	3,885	780	1	1
	NS3	To recover your lost Database send	312	47	75	0	24	1,759	354	1	1
	NS4	All your data was backed up from your server.	52	0	52	0	0	207	54	1	1
	NS5	All your data is a backed up. You must pay	45	24	0	0	8	19,324	3,917	1	1
IOC reuse	IR1	(different ransom notes)	4,046	139	3,602	0	47	56,291	13,560	2	100
	IR2	To recover your lost databases and avoid leaking it	973	0	0	2	0	3,960	795	1	24
	IR3	(different ransom notes)	66	34	24	0	5	360	72	1	16
	IR4	All your data was backed up from your server.	52	0	52	0	0	207	54	1	1
	IR5	To recover your databases, visit http://godransm3nnlo...	51	0	0	1	0	239	49	1	11
Bitcoin analysis (MI)	BC1	(different ransom notes)	4,073	142	3,606	0	47	56,421	13,586	2	109
	BC2	To recover your lost databases and avoid leaking it	973	0	0	2	0	3,960	795	1	24
	BC3	(different ransom notes)	66	34	24	0	5	360	72	1	16
	BC4	All your data was backed up from your server.	52	0	52	0	0	207	54	1	1
	BC5	(different ransom notes)	51	0	0	1	0	239	49	1	11

TABLE IV: Top 5 clusters at each clustering step. Addr. corresponds to Bitcoin addresses, Val. to unique ransom amounts, DB to targeted databases (MySql, ElasticSearch), and NSC to the number of NS clusters merged into that cluster.

The other IOC types are not used because domains are often benign, the URLs contain simply a domain with a scheme (i.e., there is no resource), and each UUID only appears in one note and thus cannot be used for merging.

Merging clusters by IOC reuse. This step first generates the set of IOCs (i.e., Bitcoin, email, and onion addresses) extracted from all the ransom notes in a NS cluster. Then, it merges NS clusters that share IOCs. The merging process starts with an empty set of IR clusters, and iterates on the list of NS clusters. For each NS cluster, it checks if its IOC set has a non-empty intersection (i.e., shares at least one IOC) with the IOC set of any of the IR clusters. If the NS cluster shares IOCs only with one IR cluster, the NS cluster is merged into that IR cluster. If the NS cluster shares IOCs with multiple IR clusters, those IR clusters as well as the NS cluster are merged together. If the NS cluster is not similar to any IR cluster, a new IR cluster is created for it. After the iteration on NS clusters completes, the IR clusters are output.

IOC reuse clusters. Table II shows that the 408 NS clusters are merged into 99 IR clusters. Of the 99 IR clusters, 62 contain more than one note, accounting for 99.4% of the 5,792 notes. The remaining 37 are singleton clusters with only one note. Overall, IOC reuse reduces the number of clusters by 75%. The reduction is even more significant for singleton clusters that decrease from 314 to 37 (12%). On average, an IR cluster contains 50% more ransom notes than an NS cluster. The median size of an IR cluster is 10 notes (mean of 92.8), compared to a median of 5 (mean of 58.3) for NS clusters.

The group of rows in the middle of Table IV shows the five largest IR clusters. The rightmost (NSC) column in the table captures the number of NS clusters merged into that IR cluster. For example, the IR1 cluster corresponds to 100 merged NS clusters including three of the top 5 NS clusters (NS1, NS3, NS5). However, not all NS clusters are enlarged through IOC reuse, for example, IR4 is the same as NS4. This cluster contains 52 email addresses for victims to contact the attackers for payment instructions, and it does not contain any Bitcoin or onion addresses. None of the 52 email addresses is present in any other NS cluster, and thus the cluster remains unchanged. When multiple NS clusters are merged into an IR cluster (e.g., IR1, IR3), the IR cluster may not have anymore a common template for the ransom notes, as each NS cluster uses a different template. In this case, we replace the cluster label in Table IV with the string "(different ransom notes)".

The IOC reuse step reduces the number of clusters by 75%. Still, it is possible that some NS clusters from the same owners did not reuse IOCs and have not been merged. To further identify campaigns from the same owners that do not reuse IOCs, we use the information from the Bitcoin blockchain.

C. Bitcoin Multi-Input Clustering

We leverage the public Bitcoin transaction data to further merge IR clusters. For this, we use the open-source WatchYourBack [24] platform to analyze the Bitcoin blockchain up to block height 810,200, corresponding to October 1st, 2023. First, we examine which payment addresses extracted from the ransom notes have received funds. Of the 272 Bitcoin addresses, 137 (50.4%) have received at least one deposit. We call these 137 Bitcoin addresses the *seeds*.

Next, we expand the set of seeds using multi-input (MI) clustering [13], [25], [15], [14]. MI clustering considers that addresses that co-spend together in the same transaction are under the control of the same owner since signing the transaction requires access to the private keys of the input addresses. We use previously proposed heuristics to exclude CoinJoin transactions that do not satisfy this property [26], [27]. Clusters are created transitively; if a transaction has addresses A and B as inputs, and another transaction has B and C as inputs, then A, B, and C are all considered to belong to the same owner and thus belong to the same MI cluster (MIC). If a seed corresponds to an online wallet in a service like an exchange, then the MIC of that seed may incorrectly include online wallets from other unrelated clients of the service [11]. To avoid this problem, we identify online wallets using the tag database in WatchYourBack. This step identifies two seeds belonging to MICs from exchanges, one in Poloniex and the other in Coinbase. We also ran the WatchYourBack machine learning exchange classifier on the seeds, identifying an additional 7 seeds on 6 untagged MI clusters belonging to exchanges. Other addresses in the MICs of those 9 seeds may belong to unrelated clients of the exchanges. Thus, for

Clustering	All	Non-singl.	Singl.	Max.	Med.	Mean
Note similarity	408	94 (94.6%)	314	3,507	5	58.3
IOC reuse	281	141 (97.6%)	140	3,451	6	40.1
MI clustering	1,336	162 (79.7%)	1,174	2,977	5	28.5

TABLE V: Ablation study results for clustering the 5,792 notes using each clustering step separately. The note similarity row is the same as the first row in Table II.

以下数据库已被删除: example_nodejs, 我们有完整的备份, 要恢复它, 您必须向我们的比特币地址\
1J8jK64528P9CKJm8Sk5oY6eea2Qm5UHYK 支付 0.005 比特币 (BTC), 如果您需要证明, 请通过以下电子邮件与我们联系, xiao77@tutanota.com, 任何与付款无关的邮件都将被忽略!\
1J8jK64528P9CKJm8Sk5oY6eea2Qm5UHYK xiao77@tutanota.com

Fig. 2: Chinese note with same Bitcoin address observed in English notes.

those 9 seeds, we do not use their MICs and instead place each seed by itself in its own MIC. Overall, the 137 seeds belong to 122 MICs. Of those, 39 MICs contain at least two addresses, and 83 are singletons (i.e., they only include one seed). The 39 non-singleton clusters contain 57 additional non-seed addresses that were not part of the LeakIX dataset, and were discovered using the MI clustering.

Finally, we use the MICs to merge IR clusters that contain addresses in the same MIC. The merging procedure is analogous as the one described in Section IV-B, but it requires a non-empty intersection of MIC identifiers (instead of IOC sharing) as a signal to merge clusters. As shown in the bottom row of Table II, MI clustering of Bitcoin addresses further reduces the number of clusters from 99 to the final 94 clusters. The reduction is significantly smaller than the one obtained with IOC reuse, which may be due to attackers avoiding MI clustering on purpose, as recently pointed out [11]. Still, this step eliminates 5 clusters and increases the average (non-singleton) cluster size from 92.8 to 97.8 notes.

It is worth noting that the 57 additional non-seed addresses identified are not added to the final clustering, as they do not come from any ransom note in our dataset. Instead, they are kept as a separate MIC feature. Given a final cluster, we can output what additional Bitcoin addresses from the same owners the MIC discovered.

D. Ablation Study

We perform an ablation study to quantify how much our three-step clustering improves compared to using each clustering step in isolation. Table V summarizes the results of the ablation study on the clustering results. Using only NS clustering we obtain the same results in the first row of Table II with 408 clusters (94 with more than one ransom note). Clustering notes that share IOCs produce 281 clusters (141 with multiple notes). To apply the MI clustering, we first need to perform a lightweight IOC reuse clustering where we group notes that contain the same Bitcoin address. Notes with no Bitcoin addresses are placed in a singleton cluster. Next, we calculate the MI clustering on the Bitcoin addresses and merge clusters containing addresses in the same MI cluster. This approach produces 1,336 clusters (162 with more than one ransom note). The clustering step in isolation groups the most is the one targeting the IOC reuse. In contrast, IOC reuse of Bitcoin addresses combined with MI clustering groups the least, suggesting that the reuse of email and onion addresses is an important factor. To conclude, our three-step clustering produces 94 clusters, less than half the minimum of 281 clusters when using any individual clustering step.

V. Cluster Analysis

In this section we analyze more in detail selected clusters.

BC1. This is the largest cluster containing 4,073 notes (70.3% of all notes) observed in 13,586 victim servers (49% of all IP addresses). We observe activity associated to this cluster for nearly two years, from May 2021 to the end of our analysis. The long lifetime, together with the fact that the cluster includes nearly half of the infected servers, indicates that the group behind this cluster is a major player in the database server ransom scam ecosystem. The cluster contains 109 NS clusters indicating that it runs a variety of campaigns with different text in the ransom notes. Thus, there is not a single template for the notes, The ransom notes are written either in English (96.5%) or Chinese (3.5%) indicating that they are targeting users in different geographical locations. We manually examine the NS clusters confirming the link between English and Chinese notes. For example, BTC address *1J8jK64528P9CKJm8Sk5oY6eea2Qm5UHYK* appears in English notes like "All your data is backed up. You must pay 0.15 BTC to 1J8jK64528P9CKJm8Sk5oY6eea2Qm5UHYK 48 hours to recover it." or "To recover your lost Database and avoid leaking it: Send us 0.015 Bitcoin (BTC) to our Bitcoin address 1J8jK64528P9CKJm8Sk5oY6eea2Qm5UHYK", but also in Chinese notes such as the one in Figure 2.

The 4,073 notes in the cluster contain 142 payment addresses (51.8% of all Bitcoin addresses), with 65 of them having received deposits. The ransom amounts are low, ranging from 0.0015 BTC up to 0.26 BTC. Such low amounts can be an incentive for victims to pay, even when they have doubts about whether their data can be recovered.

BC2. The second largest cluster is responsible for 795 infected servers and contains 973 notes. All the notes share the same LCS: "To recover your lost databases and avoid leaking it: visit http://[...] and enter your unique token [...] and pay the required amount of Bitcoin to get it back. Databases that we have:[...]. Your databases are downloaded and backed up on our servers. If we dont receive your payment in the next 9 Days, we will sell your database to the highest bidder or use them otherwise.[...]". None of the notes contain a Bitcoin or email address. Instead, the attackers request the victim to visit a Tor hidden service, and enter a token in the note to obtain the amount to pay and the ransom address. Adding this level of indirection prevents Internet scanning engines to observe the ransom addresses, which in turn hinders the ability to analyze them using the Bitcoin public transaction data. The cluster contains two onion addresses; initially, the attackers used a version 2 address hn4wg4o6s5nc7763.onion, that moved later

to a newer version 3 address o42xfh5kao7mrtesnok5jgdsfagjsgzxlxdlpkpd2x6lpckhzk225yad.onion

We use the note LCS template to search for public reports for the campaign captured by the cluster. We find reports indicating that the cluster corresponds to a database server ransom scam, rather than a real ransomware [19], [28]. The attackers rely on brute force to exploit weak credentials on MySQL servers. After gaining access, they drop the databases and leave the ransom note, but they do not copy the data out of the server. Thus, even if the victim pays the ransom, the attackers have no way of returning the deleted data. While the report mentions only MySQL servers being targeted and English ransom notes, our clustering reveals that the same attackers also run other campaigns targeting ElasticSearch and using notes with text in Chinese.

VI. Related Work

Due to its privacy properties and poorly regulated legal status in some countries, the Bitcoin ecosystem has attracted cybercriminal operations such as ransomware [1], [2], [3], [4], [5], thefts [16], scams [29], [30], [16], [31], [32], [33], human trafficking [34], cryptojacking [35] hidden marketplaces [36], [37], and money laundering [38]. In contrast, we investigate database server ransom scams, which target database servers without authentication, drop the data, and demand a ransom to recover the deleted data.

Clustering. Many works automatically group similar malicious instances such as scam websites [39], [40], [33], scam emails [29], and malware samples [41], [42], [43], [44], [45], [46], [47], [48]. Our work differs because it specifically targets the clustering of ransom notes. Most related is the work by Paquet-Clouston et al. [29] that clusters emails where the last 50 words are similar, identifies sextortion emails through keyword search, and then manually merges related email clusters. In contrast, our clustering computes the similarity of the whole ransom note and automatically merges related clusters that reuse IOCs.

Ransomware. Due to its impact, ransomware targeting desktops and mobile devices has attracted much research [1], [2], [3], [4], [5], [49], [50], [51]. Instead, database server ransom scams target servers. Recently, Gomez et al. [11] investigated the DeadBolt ransomware that encrypts data in network-attached storage (NAS) servers. In contrast, database server ransom scams do not encrypt data, but delete it without exfiltrating it. Thus, the victim has no means to recover the data even after paying the ransom.

Scams. A variety of scams have been analyzed by prior work including technical support scams [52], [53], sextortion [29], romance scams [54], survey scams [55], pet scams [56], and tax scams [57]. To the best of our knowledge, our work first investigates database server ransom scams.

VII. Discussion

Evasion. We observe two main evasion techniques. First, some campaigns do not provide the ransom payment details in their notes, instead providing Tor onion addresses or contact emails to obtain them. Such indirection complicates the collection of Bitcoin addresses. However, it may be possible to obtain the Bitcoin addresses through scam-baiting techniques [52], [58] such as automating the submission of identifiers to Tor hidden services and the sending of initial contact emails. Second, attackers try to hamper IOC extraction by removing spaces and punctuation before or after the IOCs. Such evasion does not work for blockchain and onion addresses if the extraction verifies the embedded checksum. However, for other IOCs such as emails and URLs, spurious text may be added at the beginning or end. Obfuscating the IOCs to extraction tools also obfuscates them for real victims, which may in turn prevent them from paying even if they want to. As attackers become aware of our approach they may incorporate other evasions such as minimizing the reuse of IOCs across campaigns. However, avoiding IOC reuse does not come for free as it may require automated approaches to handle many IOCs.

Ethical considerations. We do not collect data from users or infected servers ourselves. Instead, we leverage publicly available data independently collected by the LeakIX scan engine. In particular, LeakIX collects data from database servers that have not set up authentication. From those servers, they collect the name of database tables and the table content for tables whose name matches their regular expressions for ransom notes names. Since those regular expressions can occasionally have false positives, we apply an additional filter to eliminate data that does not look like a valid ransom note. LeakIX does not modify any data in the server.

VIII. Conclusions

We present the first study of *database server ransom scams*, a class of attacks targeting database servers with no authentication or guessable credentials. We leverage the LeakIX Internet scanning engine to collect 5,792 unique ransom notes from 27,750 infected ElasticSearch and MySQL database servers. We propose a novel approach to cluster infections into campaigns run by the same threat group. Our clustering leverages the similarity of the ransom note text to group notes into campaigns. Then, it merges note similarity clusters that reuse IOCs to identify campaigns run by the same group. Finally, it combines the IOC reuse clusters using the MI clustering extracted from the Bitcoin blockchain. Our approach clusters the 27,750 infections into 94 groups, identifying a dominant group responsible for 49% of the infections.

Acknowledgments

We thank the reviewers for their time and valuable comments. We are especially grateful to LeakIX administrators for providing us with access to the data. This work was partially funded by the Spanish Government MCIN/AEI/10.13039/501100011033/ through grants TED2021-132464B-I00 (PRODIGY), PID2022-142290OB-I00 (ESPADA), PRE2019-088472, and PREP2022-000165. The above grants are co-funded by European Union ESF, EIE, FEDER, and NextGeneration funds.

REFERENCES

[1] M. Spagnuolo et al., "BitIodine: Extracting Intelligence from the Bitcoin Network," in *Financial Cryptography and Data Security*, 2014.

[2] K. Liao et al., "Behind Closed Doors: Measurement and Analysis of CryptoLocker Ransoms in Bitcoin," in *APWG Symposium on Electronic Crime Research*, 2016.

[3] M. Conti et al., "On the economic significance of ransomware campaigns: A bitcoin transactions perspective," *Computers & Security*, vol. 79, pp. 162–189, 2018.

[4] D. .Y. Huang et al., "Tracking Ransomware End-to-end," in *IEEE Symposium on Security and Privacy*, 2018.

[5] M. Paquet-Clouston et al., "Ransomware Payments in the Bitcoin Ecosystem," *Journal of Cybersecurity*, vol. 5, no. 1, 2019.

[6] D. et al., "ZMap: Fast Internet-Wide Scanning and its Security Applications," in *USENIX Security Symposium*, 2013.

[7] "Censys," 2024, https://censys.io/.

[8] "Shodan," 2024, https://www.shodan.io/.

[9] "LeakIX," 2022, https://leakix.net/.

[10] "Bitcoin abuse," 2023, https://www.bitcoinabuse.com.

[11] G. Gomez et al., "Cybercrime bitcoin revenue estimations: Quantifying the impact of methodology and coverage," in *ACM SIGSAC Conference on Computer and Communications Security*, 2023.

[12] L. Abrams, "MongoDB Apocalypse: Professional Ransomware Group Gets Involved, Infections Reach 28K Servers," January 2017, https://www.bleepingcomputer.com/news/security/mongodb-apocalypse-professional-ransomware-group-gets-involved-infections-reach-28k-servers/.

[13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[14] E. Androulaki et al., "Evaluating User Privacy in Bitcoin," in *Financial Cryptography and Data Security*, 2013.

[15] D. Ron et al., "Quantitative Analysis of the Full Bitcoin Transaction Graph," in *Financial Cryptography and Data Security*, 2013.

[16] S. Meiklejohn et al., "A Fistful of Bitcoins: Characterizing Payments among Men with No Names," in *Internet Measurement Conference*, 2013.

[17] Microsoft, "Windows Common Log File System Driver Elevation of Privilege Vulnerability," April 2023, https://msrc.microsoft.com/update-guide/vulnerability/CVE-2023-28252.

[18] B. Larin, "Windows CLFS and five exploits used by ransomware operators (Exploit #5 - CVE-2023-28252)," December 2023, https://securelist.com/windows-clfs-exploits-ransomware-cve-2023-28252/111601/.

[19] L. O'Donnell, "Please read me Ransomware Attacks 85K MySQL Servers," December 2020, https://threatpost.com/please_read_me-ransomware-mysql-servers/162136.

[20] J. Ullrich, "Quick analysis of a recent mysql exploit," 2016, https://isc.sans.edu/diary/Quick+Analysis+of+a+Recent+MySQL+Exploit/20781.

[21] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Annual ACM Symposium on Theory of Computing*, 2002.

[22] G. S. Manku, A. Jain, and A. D. Sarma, "Detecting near-duplicates for web crawling," in *International Conference on World Wide Web*, 2007.

[23] "iocsearcher," 2023, https://github.com/malicialab/iocsearcher.

[24] G. Gomez et al., "Watch your back: Identifying cybercrime financial relationships in bitcoin through back-and-forth exploration," in *ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[25] F. Reid et al., "An analysis of anonymity in the bitcoin system," in *2011 IEEE PASSAT/SocialCom*, 2011.

[26] H. Kalodner, M. Möser, K. Lee, S. Goldfeder, M. Plattner, A. Chator, and A. Narayanan, "BlockSci: Design and Applications of a Blockchain Analysis Platform," in *USENIX Security Symposium*, 2020.

[27] S. Goldfeder, H. A. Kalodner, D. Reisman, and A. Narayanan, "When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies," *PoPETs*, vol. 2018, pp. 179–199, 2018.

[28] G. Corfield, "'Malwareless' ransomware campaing operators pwned 83k victims' MYSQL servers, 250k databases up for sale," Dec. 2020, https://www.theregister.com/2020/12/10/mysql_malwareless_ransomware/.

[29] M. Paquet-Clouston et al., "Spams Meet Cryptocurrencies: Sextortion in the Bitcoin Ecosystem," in *ACM Conference on Advances in Financial Technologies*, 2019.

[30] M. Bartoletti et al., "Data Mining for Detecting Bitcoin Ponzi Schemes," in *Crypto Valley Conference on Blockchain Technology*, June 2018.

[31] P. Xia et al., "Characterizing Cryptocurrency Exchange Scams," *Computers & Security*, vol. 98, 2020.

[32] M. Bartoletti et al., "Cryptocurrency Scams: Analysis and Perspectives," *IEEE Access*, vol. 9, pp. 148 353–148 373, 2021.

[33] X. Li et al., "Double and nothing: Understanding and detecting cryptocurrency giveaway scams," in *Network and Distributed Systems Security Symposium*, 2023.

[34] R. S. Portnoff et al., "Backpage and Bitcoin: Uncovering Human Traffickers," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[35] E. Tekiner et al., "SoK: Cryptojacking Malware," in *IEEE European Symposium on Security and Privacy*, 2021.

[36] N. Christin, "Traveling the Silk Road: A measurement analysis of a large anonymous online marketplace," in *The World Wide Web Conference*, 2013.

[37] S. Lee et al., "Cybercriminal Minds: An Investigative Study of Cryptocurrency Abuses in the Dark Web," in *Network and Distributed Systems Security Symposium*, 2019.

[38] M. Möser et al., "An Inquiry into Money Laundering Tools in the Bitcoin Ecosystem," in *APWG eCrime Researchers Summit*, 2013.

[39] J. Drew et al., "Automatic identification of replicated criminal websites using combined clustering," in *IEEE Security and Privacy Workshops*, 2014.

[40] R. Phillips et al., "Tracing cryptocurrency scams: Clustering replicated advance-fee and phishing websites," in *IEEE International Conference on Blockchain and Cryptocurrency*, 2020.

[41] U. Bayer et al., "Scalable, Behavior-Based Malware Clustering," in *Network and Distributed System Security*, 2009.

[42] R. Perdisci et al., "Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces," in *USENIX Symposium on Networked Systems Design and Implementation*, 2010.

[43] J. Jang et al., "BitShred: Feature Hashing Malware for Scalable Triage and Semantic Analysis," in *ACM Conference on Computer and Communications Security*, 2011.

[44] X. Hu et al., "MutantX-S: Scalable Malware Clustering Based on Static Features," in *USENIX Annual Technical Conference*, 2013.

[45] M. Z. Rafique et al., "FIRMA: Malware Clustering and Network Signature Generation with Mixed Network Behaviors," in *International Symposium on Research in Attacks, Intrusions and Defenses*, 2013.

[46] Y. Li et al., "Experimental study of fuzzy hashing in malware clustering analysis," in *Workshop on Cyber Security Experimentation and Test*, 2015.

[47] M. Sebastian et al., "AVClass: A Tool for Massive Malware Labeling," in *Research in Attacks, Intrusions, and Defenses*, 2016.

[48] H. Faridi et al., "Performance evaluation of features and clustering algorithms for malware," in *IEEE International Conference on Data Mining Workshops*, 2018.

[49] H. Oz et al., "A survey on ransomware: Evolution, taxonomy, and defense solutions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 11s, pp. 1–37, 2022.

[50] N. Andronio et al., "Heldroid: Dissecting and Detecting Mobile Ransomware," in *International Symposium on Research in Attacks, Intrusions, and Defenses*, 2015.

[51] A. Kharraz et al., "UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware," in *USENIX Security Symposium)*, 2016.

[52] N. Miramirkhani et al., "Dial one for scam: Analyzing and detecting technical support scams," in *Network and Distributed System Security Symposium*, 2016.

[53] B. Srinivasan et al., "Exposing search and advertisement abuse tactics and infrastructure of technical support scammers," in *World Wide Web Conference*, 2018.

[54] G. Suarez-Tangil et al., "Automatically dismantling online dating fraud," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1128–1137, 2019.

[55] A. Kharraz et al., "Surveylance: Automatically detecting online survey scams," in *IEEE Symposium on Security and Privacy*, 2018.

[56] B. Price et al., "Resource networks of pet scam websites," in *APWG Symposium on Electronic Crime Research*, 2020.

[57] M. Bidgoli et al., ""hello. this is the irs calling.": A case study on scams, extortion, impersonation, and phone spoofing," in *APWG Symposium on Electronic Crime Research*, 2017.

[58] M. Edwards, C. Peersman, and A. Rashid, "Scamming the scammers: towards automatic detection of persuasion in advance fee frauds," in *International Conference on World Wide Web Companion*, 2017.