Security Assessment of Dangerzone's Web and Client Applications and Review of Application Architecture Security on Behalf of OTF

OPEN TECHNOLOGY FUND

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Include Security (IncludeSec)

IncludeSec brings together some of the best information security talent from around the world. The team is composed of security experts in every aspect of consumer and enterprise technology, from low-level hardware and operating systems to the latest cutting-edge web and mobile applications. More information about the company can be found at www.IncludeSecurity.com.

Assessment Objectives

The objective of this assessment was to identify and confirm potential security vulnerabilities within targets in-scope of the SOW. The team assigned a qualitative risk ranking to each finding. Recommendations were provided for remediation steps which Dangerzone could implement to secure its applications and systems.

Scope and Methodology

Include Security conducted a comprehensive security assessment of the Dangerzone Web Application, Client Application, and Application Architecture on behalf of the Open Technology Fund. This 12-day effort, carried out from December 4th, 2023, to December 19th, 2023, employed a Standard Grey Box assessment methodology. The assessment involved a thorough review of all components as outlined in the original Statement of Work (SOW), ensuring a methodical and structured evaluation in alignment with the predefined objectives and scope.

Findings Overview

IncludeSec identified a total of 10 findings. There were 3 deemed to be “Low-Risk,” which pose some tangible security risk, and 7 “Informational” level findings that do not immediately pose a security risk.

IncludeSec encourages the Dangerzone development team to redefine the stated risk categorizations internally in a manner that incorporates internal knowledge regarding business model, customer risk, and mitigation environmental factors.

Next Steps

IncludeSec advises the Dangerzone development team to remediate as many findings as possible in a prioritized manner and make systemic changes to the Software Development Life Cycle (SDLC) to prevent further vulnerabilities from being introduced into future release cycles. This report can be used as a basis for any SDLC changes. IncludeSec welcomes the opportunity to assist the Dangerzone development team in improving their SDLC in future engagements by providing security assessments of additional products. For inquiries or assistance scheduling remediation tests, please contact us at remediation@includesecurity.com.

RISK CATEGORIZATIONS

At the conclusion of the assessment, Include Security categorized findings into five levels of perceived security risk: Critical, High, Medium, Low, or Informational. **The risk categorizations below are guidelines that IncludeSec understands reflect best practices in the security industry and may differ from a client's internal perceived risk. Additionally, all risk is viewed as "location agnostic" as if the system in question was deployed on the Internet. It is common and encouraged that all clients recategorize findings based on their internal business risk tolerances. Any discrepancies between assigned risk and internal perceived risk are addressed during the course of remediation testing.**

Critical-Risk findings are those that pose an immediate and serious threat to the company's infrastructure and customers. This includes loss of system, access, or application control, compromise of administrative accounts or restriction of system functions, or the exposure of confidential information. These threats should take priority during remediation efforts.

High-Risk findings are those that could pose serious threats including loss of system, access, or application control, compromise of administrative accounts or restriction of system functions, or the exposure of confidential information.

Medium-Risk findings are those that could potentially be used with other techniques to compromise accounts, data, or performance.

Low-Risk findings pose limited exposure to compromise or loss of data, and are typically attributed to configuration, and outdated patches or policies.

Informational findings pose little to no security exposure to compromise or loss of data which cover defense-in-depth and best-practice changes which we recommend are made to the application. Any informational findings for which the assessment team perceived a direct security risk, were also reported in the spirit of full disclosure but were considered to be out of scope of the engagement.

The findings represented in this report are listed by a risk rated short name (e.g., C1, H2, M3, L4, and I5) and finding title. Each finding may include if applicable: Title, Description, Impact, Reproduction (evidence necessary to reproduce findings), Recommended Remediation, and References.

INTRODUCTION

Dangerzone is a solution designed to enhance digital security and privacy through a suite of applications, including web and client applications, alongside a complex application architecture. It aims to safeguard confidential information and ensure secure communication within its operational environment.

The assessment team undertook a security assessment over a specified period, focusing on Dangerzone's multifaceted components. This evaluation aimed to examine the security measures implemented within the Dangerzone system, including the web application, client application, and the overarching application architecture. The assessment was structured to encompass a wide range of testing methodologies and security checks to ensure a thorough examination of Dangerzone's defense mechanisms.

Assessment Overview

The assessment was conducted with the following objectives:

A. **Information & Documentation Gathering and Configuration**: Initial stages involved close collaboration with the Dangerzone team to gather essential documentation and configure the testing environment to mirror real-world operations closely.

B. **Contextual Familiarization with Dangerzone**: The team reviewed Dangerzone's development history, documentation, and technical advisories to fully understand its operational context.

C. **Comprehensive Testing and Review**: This included manual code review, dynamic testing of the installer, software scans, and an in-depth security review of the application architecture. Special attention was given to Docker container/sandbox configurations and Qubes OS integration.

D. **Focused Analysis on Security Concerns**: The team concentrated on identifying potential security vulnerabilities, ranging from local and remote attack surfaces to protocol attack vectors, emphasizing the system's resilience against various cybersecurity threats.

E. **Security Architecture Design Improvements**: Throughout the assessment, strategies for enhancing the security posture of the Dangerzone system were explored, with a focus on identifying and rectifying potential vulnerabilities.

F. **Web Application Grey Box Testing**: A detailed network security assessment and web application security testing were conducted to identify and confirm security vulnerabilities, ensuring the robustness of Dangerzone's web presence against common and advanced security threats.

The assessment period spanned from December 4th, 2023, to December 19th, 2023, employing a range of dynamic testing tools and methodologies. This enabled the team to perform penetration testing effectively, scrutinize the software for its response to malicious inputs, and review the source code for potential vulnerabilities.

The following components were reviewed in this assessment:

- Dangerzone Web Application
- Client Application (Desktop)
- Application Architecture
- Docker Container/Sandbox Configuration
- Qubes OS Integration

LOW-RISK FINDINGS

L1: [MacOS] Opportunities for MacOS Client Entitlements Hardening

Description:

Upon examining the installer packages across different platforms, the assessment team found that the MacOS installer was missing several security best-practice hardening features. Specifically, the installer exhibited certain configuration settings that could compromise its security:

- The configuration **com.apple.security.app-sandbox** was missing.
- The configuration **com.apple.security.cs.allow-unsigned-executable-memory** was set.

Impact:

The **com.apple.security.app-sandbox** and the **com.apple.security.cs.allow-unsigned-executable-memory** entitlements in macOS are important for system security, if exploited, the following issues could occur:

- **com.apple.security.app-sandbox**: An attacker could potentially bypass these restrictions, gaining unauthorized access to system resources or user data. This could lead to a range of security issues, including data theft, privacy breaches, and the execution of malicious code with elevated privileges.
- **com.apple.security.cs.allow-unsigned-executable-memory**: An attacker could leverage this lack of entitlement for the execution of arbitrary, unsigned code, potentially leading to system compromise.

As outlined in Apple's official documentation, the **com.apple.security.app-sandbox** entitlement is important for safeguarding system resources and user data. It restricts an app's access to only those resources explicitly permitted through entitlements. Additionally, Apple mandates that any app distributed through the official Mac App Store must include this entitlement. Consequently, the MacOS version of **Dangerzone** lacked the protective measures provided by the **com.apple.security.app-sandbox** entitlement.

The setting of the **com.apple.security.cs.allow-unsigned-executable-memory** entitlement in an application allows the app to create memory segments that are both writable and executable without the constraints normally enforced by the **MAP_JIT** flag. This could potentially expose the application to various additional security vulnerabilities.

Reproduction:

The current entitlements of the latest available **Dangerzone** installers for MacOS can be obtained by running the following command on a MacOS host:

```
$ codesign -d --entitlements :- /Applications/Dangerzone.app/
Executable=/Applications/Dangerzone.app/Contents/MacOS/dangerzone
Warning: Specifying ':' in the path is deprecated and will not work in a future release
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"https://www.apple.com/DTDs/PropertyList-1.0.dtd"><plist version="1.0"><dict><key>com.apple.security.cs.allow-
unsigned-executable-memory</key><true/><key>com.apple.security.files.user-selected.read-
write</key><true/><key>com.apple.security.hypervisor</key><true/><key>com.apple.security.inherit</key><true/><key>c
om.apple.security.network.client</key><true/><key>com.apple.security.network.server</key><true/></dict></plist>
```

At the time of the assessment, the output from this command showed that the application was missing the recommended security-related entitlement. Additionally, it showed that the application had the **com.apple.security.cs.allow-unsigned-executable-memory** entitlement enabled.

Recommended Remediation:

The assessment team recommends removing the **com.apple.security.cs.allow-unsigned-executable-memory** entitlement and setting the **com.apple.security.app-sandbox** entitlement for defense-in-depth measures of the **Dangerzone** MacOS build.

References:

Allow Unsigned Executable Memory Entitlement
App Sandbox

L2: [MacOS] [Windows] [Linux] [QubesOS] LibreOffice Security Hardening Options

Description:

The assessment team's evaluation of LibreOffice's configuration, particularly in the context of converting documents to pixels, revealed that **Dangerzone's** setup at the time of the assessment could be improved by optimizing LibreOffice's security-related settings. In particular, the assessment team identified features and services of LibreOffice prone to security vulnerabilities which could be disabled.

Impact:

LibreOffice includes many functionalities, and some of these functionalities are infrequently utilized or have historically presented security challenges. These functionalities, for example, include Macros or further Dynamic Content. It is advisable to have the capability to disable these specific features.

Macros, which are essentially scripts or command sequences, are integrated into documents, especially those handled by office suites like LibreOffice. From a security perspective, macros pose a significant risk due to their potential to execute harmful code. Upon opening a document containing a macro, it can automatically initiate, running any code it contains.

By providing a malicious file containing macros or dynamic content, which is then processed by **Dangerzone**, an attacker could potentially execute arbitrary or malicious code inside the isolated container.

Reproduction:

The **Dangerzone** development team was previously unaware of these configuration options, and they confirmed that these settings were not enabled at the time of assessment.

Recommended Remediation:

The assessment team recommends further enhancing the present configuration of LibreOffice and to consider disabling:

- OLE objects (**DisableActiveContent** configuration option)
- Macros (**DisableMacrosExecution** configuration option)
- LibreLogo scripts (at the time of testing, there was no available configuration option to disable this feature; however, a pull reference is available for further details, as mentioned in the references section)
- DDE commands (at the time of testing, there was no available configuration option to disable this feature; however, a pull reference is available for further details, as mentioned in the references section)

The official LibreOffice documentation offers an extensive overview of the available configuration settings, enabling users to selectively deactivate the mentioned features.

References:

Bug 158375 - Ability to Disable Active Content in LibreOffice
Libreoffice Configuration Options

L3: [Web] Deprecated TLS Ciphers Supported

Description:

The assessment team analyzed the SSL/TLS configuration of the static website **dangerzone.rocks**. At the time of assessment, the application's HTTPS service used deprecated encryption algorithms or protocols. The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are the encryption components of the HTTPS protocol suite. They provide transport-layer security to ensure data confidentiality, authenticity, and integrity.

Impact:

If deprecated algorithms are enabled, a suitably located attacker could mount a cipher downgrade attack, forcing both client and service to use a less complex cipher than they would normally choose. This could allow the attacker to obtain confidential information as it travels between clients and the service.

As part of this evaluation, it was observed that the deprecated TLSv1.0 version was supported, thus rendering it susceptible to attack strategies such as BEAST. Moreover, the analyzed server also supported TLSv1.1, which is considered deprecated according to IETF.

Reproduction:

The following output from the tool **testssl.sh** shows that the server supported deprecated protocols and ciphers:

```
testssl.sh https://dangerzone.rocks
```

Recommended Remediation:

The assessment team recommends reconfiguring the HTTPS service to ensure that only cryptographically strong protocols (e.g., TLS1.2 and later) and ciphers are supported.

References:

Testing for SSL-TLS (OWASP-CM-001) – OWASP
testssl.sh
IETF Deprecating TLS 1.0 and TLS 1.1
Guide to TLS Standards Compliance
PCI Compliance – Disable SSLv2 and Weak Ciphers
Mozilla: SSL Configuration Generator

INFORMATIONAL FINDINGS

I1: [MacOS] [Windows] [Linux] [QubesOS] Non-Essential Binaries Included in Container Images

Description:

During the assessment for possible container escape avenues within the containers used for converting documents to sanitized ones, the assessment team observed that the container image included non-essential executables, such as **wget** and **nc**. These utilities are not known to be needed for the **Dangerzone** business-use case.

A container escape, in the context of technologies like Docker, refers to a security incident where an attacker manages to break out of an isolated container environment and gain unauthorized access to the host system or other containers. This breach effectively compromises the isolation principle that containers are supposed to provide, allowing the attacker to potentially access security-relevant data, manipulate the host system, or launch further attacks.

Impact:

Although the included binaries posed no immediate threat to the application, given the container's robust hardening and effective prevention of all outgoing network connections, the assessment team does not believe these binaries need to be included in the container. This is due to the potential evolution of the **Dangerzone** software complex in the future, where these binaries could be exploited by attackers as part of a more extensive attack chain. Therefore, their presence is viewed as an avoidable risk.

Reproduction:

The presence of tools like **wget** and **nc** can be confirmed by establishing a BASH connection to the container responsible for the conversion of the file.

Recommended Remediation:

While this vulnerability was not identified as a security risk during the testing phase, the assessment team recommends the removal of all non-essential binaries from the container images as a precautionary measure for an enhanced defense-in-depth security posture.

References:

wget
nc

I2: [MacOS] [Windows] [Linux] [QubesOS] Missing Password Protection Feature

Description:

During the evaluation of the **Dangerzone** application, the assessment team observed that the application did not offer a feature that would enable users to encrypt the final, sanitized PDF file. This functionality could be a beneficial addition for users seeking enhanced security for the converted documents.

The **Dangerzone** application is a security tool that sanitizes PDFs, office documents, or images by first converting them into a PDF, then into raw pixel data, and finally back into a sanitized PDF, all within isolated sandboxes to prevent malicious code execution. This process ensures that the final PDF is free from any embedded malware or malicious scripts.

Impact:

The absence of an encryption feature in the **Dangerzone** application could lead to the temporary storage of the converted, sanitized document (PDF) on the host system. In certain situations, retaining any form of persistent data or artifacts of the processed document on the host computer might not be preferable.

Reproduction:

This missing feature is noticeable when operating the **Dangerzone** utility through both its graphical user interface (GUI) and command-line interface (CLI). This can be observed by performing the following steps:

1. Start the **Dangerzone** application.
2. Select a file to be converted.
3. Click on the **Convert to Safe Documents** button.
4. Observe the lack of any feature allowing the user to password protect the resulting file.

Recommended Remediation:

The assessment team recommends exploring the implementation of a password protection feature that would encrypt the resulting document prior to saving it on the host system.

References:

CWE-311: Missing Encryption of Sensitive Data

I3: [MacOS] [Windows] [Linux] Missing Software Status Check For Docker and Docker Desktop

Description:

During the examination for potential container escape vulnerabilities, the assessment team observed that the **Dangerzone** application did not include a mechanism to ensure that the Docker or Docker Desktop applications installed on the system were current and running the latest software versions.

A container escape, in the context of technologies like Docker, refers to a security incident where an attacker manages to break out of an isolated container environment and gain unauthorized access to the host system or other containers. This breach effectively compromises the isolation principle that containers are supposed to provide, allowing the attacker to potentially access security-relevant data, manipulate the host system, or launch further attacks.

Impact:

Vulnerabilities in Docker and Docker Desktop on the host system could potentially affect users of **Dangerzone**. For example, if an attacker were to gain code execution within a container tasked with converting documents to pixels and then to PDF, any security gaps resulting from an outdated Docker installation could be exploited. In the worst-case scenario, these vulnerabilities might be used to execute a guest-to-host escape, compromising the host system.

Note: This vulnerability has been classified as **Informational**. This designation reflects the understanding that addressing this concern would serve as an enhancement, contributing to a more robust defense-in-depth strategy, rather than indicating a direct security vulnerability.

Reproduction:

The lack of such a check becomes evident when using the Dangerzone utility either over the GUI or the CLI. Upon starting **Dangerzone**, no such check was observed to take place which would inform the user in case an outdated Docker/Docker Desktop version is installed on the host.

Recommended Remediation:

The assessment team recommends implementing a feature in **Dangerzone** that checks the host system for the latest version of Docker/Docker Desktop upon startup. If the feature detects an outdated Docker installation, **Dangerzone** could then provide a warning to the user, recommending an update. This precaution aims to mitigate the risk of attackers exploiting vulnerabilities within the container to escape to the host system.

References:

Docker Security Cheat Sheet

I4: [CLI] dangerzone-cli Disclosed File Names to Shell History

Description:

While evaluating **Dangerzone** for disclosure vulnerabilities, the assessment team noted that employing **dangerzone-cli**, like any other command-line interface tool invoked via a shell, inadvertently resulted in the arguments supplied to the utility being recorded in the **.bash_history** file. This observation highlighted a common security concern regarding command-line operations where input arguments could be unintentionally exposed. The **dangerzone-cli** provides an alternative way to use the **Dangerzone** utility in addition to a graphical user interface via the command line.

Impact:

An attacker able to obtain access to a device that utilizes the **Dangerzone CLI** interface could identify the file names for files that were processed by the **Dangerzone** application. File names alone can often reveal the nature of documents that have undergone sanitization, which could lead to an attacker discovering additional vulnerabilities.

Reproduction:

Command-line interface (CLI) arguments can be recorded in the **.bash_history** file by utilizing the **dangerzone-cli** interface and then inspecting the command history, which is typically located at **{$HOME}/.bash_history** on Linux systems for users operating with BASH as their shell. The following provides a snippet showing how the output is stored in the BASH history:

```
$ history
[...]
 1330  dangerzone employe_max_mustermann_termination_agreement.pdf
[...]
```

Recommended Remediation:

This vulnerability can be mitigated by careful management of security-relevant information in command-line environments. The assessment team recommends either explicitly mentioning this behavior in the application's documentation and allowing users to acknowledge the inherent risk or decreasing the risk by implementing a mechanism that clears the **.bash_history** file each time the **dangerzone-cli** command is executed.

References:

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

I5: [MacOS] [Windows] [Linux] [QubesOS] Limited User Feedback for File Conversion Process

Description:

In the dynamic evaluation of the **Dangerzone** utility, the assessment team noted that user feedback during the file conversion process was mainly limited to a visual progress bar. The absence of detailed feedback, such as time estimates for the completion of a file conversion, could be a drawback to users. This is particularly relevant for users handling large numbers of files or exceptionally sizable individual files, especially when operating under time constraints.

The **Dangerzone** application is a security tool that sanitizes PDFs, office documents, or images by first converting them into a PDF, then into raw pixel data, and finally back into a sanitized PDF, all within isolated sandboxes to prevent malicious code execution. This process ensures that the final PDF is free from any embedded malware or malicious scripts.

Impact:

Incorporating a feature that provides an estimated time for the completion of file conversions would greatly enhance the user experience. Such an enhancement would allow users to make informed decisions about whether they can sanitize their files within the required timeframe, improving both the utility and efficiency of the **Dangerzone** tool.

Reproduction:

This absence of time estimation becomes apparent during the file conversion process using the **Dangerzone** utility. This can be observed by performing the following steps:

1. Start the **Dangerzone** application.
2. Select a file to be converted.
3. Click on the **Convert to Safe Documents** button to begin the file conversion process.
4. Observe the lack of a time estimation of the conversion process.

Recommended Remediation:

To further enhance the user experience, the assessment team recommends adding a time estimation feature alongside the existing progress bar.

References:

Dangerzone

I6: [MacOS] [Windows] [Linux] [QubesOS] Possible Attack Vectors via OCR Engine

Description:

As part of the **Dangerzone** security evaluation, the assessment team explored potential attack vectors, including the optical character recognition (OCR) translation engine used in the system.

OCR is used to convert visually represented text, e.g., handwritten text, into machine-encoded text. OCR is used by the **Dangerzone** application to perform this conversion and can be enabled by the user during the conversion process.

Impact:

A malicious actor could craft a document that appears as broken pixels but is designed to exploit the mechanics of the OCR engine. Such a document might not only produce inaccurate results but also could potentially yield malicious outcomes, such as data exfiltration or RCE in a worst case scenario. Additional in-

depth analysis would be required to determine if these outcomes would be actual vulnerabilities in this application. This scenario hinges on the OCR engine misinterpreting the visual data, leading to the generation of unintended or harmful outputs.

Although the likelihood of an attacker exploiting this vulnerability is low and executing such an attack would require considerable expertise, the potential impact of this vulnerability is severe, potentially enabling an attacker to produce a document that is perceived as benign but is, in fact, malicious. This deceptive tactic could lead to the generation and opening of a seemingly safe document on the host computer, causing harm. The severity of this scenario lies in the ability of the attacker to exploit the system's trust in the safety of the processed document, potentially compromising the integrity and security of the host where the document is accessed.

Reproduction:

During this security review, it was not possible for the assessment team to develop an actual proof-of-concept (PoC) exploitation of this vulnerability due to the time-limited nature of the assessment. Nevertheless, it was mutually agreed upon with the **Dangerzone** team to include this finding in the report. This inclusion is intended to ensure that the **Dangerzone** team investigates and further strengthens the **Dangerzone** application against such potential threats.

Recommended Remediation:

This aspect of the OCR engine as a potential attack vector warrants further investigation to ensure the robustness and safety of **Dangerzone** against sophisticated attack methods.

References:

Evaluating the Robustness of OCR Systems
Attacking Optical Character Recognition (OCR) Systems with Adversarial Watermarks

I7: [MacOS] [Windows] [Linux] [QubesOS] Out-of-Date Libraries in Use

Description:

The **Dangerzone** application was found to use outdated libraries which are affected by publicly known vulnerabilities.

Impact:

The assessment team found several libraries used by the application to be out of date. These components have publicly known vulnerabilities, and an attacker who discovers out-of-date software within the application could use them to focus exploit attempts. Note that these vulnerabilities require very specific conditions to be exploitable; the extent to which the out-of-date components can be exploited depends largely on how these libraries are used within the application.

Notably, the **ghostscript** library was impacted by **CVE-2023-43115**, which represents a **Critical** vulnerability with a CVSS score of 9.8.

As detailed in the official advisory, the mentioned **ghostscript** library was employed in an isolated environment where documents are processed. Owing to the stringent and effective isolation from the host, the assessment team believes that the security vulnerability in the **ghostscript** library does not pose an immediate risk to **Dangerzone** users. However, if coupled with a more severe vulnerability, often referred to as a container escape, a malicious document could potentially compromise **Dangerzone's** security. It is important to note that IncludeSec did not identify any vulnerabilities in the isolation layer that would permit

such a container escape at the time of assessment. Additionally, this vulnerability was addressed by the **Dangerzone** team mid-project, as described within the security advisory (https://github.com/freedomofpress/dangerzone/blob/v0.5.1/docs/advisories/2023-12-07.md). The assessment team has adjusted the risk for this finding to **Informational** due to these mitigating factors.

Reproduction:

The following software packages were identified as out-of-date and potentially vulnerable, utilizing the **grype** tool, as shown below:

```
$ grype dangerzone.rocks/dangerzone | head
[...]
 ✓ Vulnerability DB                [updated]
 ✓ Loaded image
dangerzone.rocks/dangerzone:latest
 ✓ Parsed image
sha256:9c6efa81b6bc335560d4c94c706c71c1eaa7af0bb4a138e0cccf48426b80cda8
 ✓ Cataloged packages              [305 packages]
 ✓ Scanned for vulnerabilities     [79 vulnerability matches]
   ├── by severity: 1 critical, 6 high, 50 medium, 14 low, 0 negligible (8 unknown)
   └── by status:   32 fixed, 47 not-fixed, 0 ignored
NAME                          INSTALLED    FIXED-IN     TYPE  VULNERABILITY   SEVERITY
avahi-libs                    0.8-r13                   apk   CVE-2023-38473  Medium
avahi-libs                    0.8-r13                   apk   CVE-2023-38472  Medium
avahi-libs                    0.8-r13                   apk   CVE-2023-38471  Medium
avahi-libs                    0.8-r13                   apk   CVE-2023-38470  Medium
avahi-libs                    0.8-r13                   apk   CVE-2023-38469  Medium
busybox                       1.36.1-r4                 apk   CVE-2023-42366  Medium
busybox                       1.36.1-r4                 apk   CVE-2023-42365  Medium
busybox                       1.36.1-r4                 apk   CVE-2023-42364  Medium
busybox                       1.36.1-r4                 apk   CVE-2023-42363  Medium
busybox-binsh                 1.36.1-r4                 apk   CVE-2023-42366  Medium
busybox-binsh                 1.36.1-r4                 apk   CVE-2023-42365  Medium
busybox-binsh                 1.36.1-r4                 apk   CVE-2023-42364  Medium
busybox-binsh                 1.36.1-r4                 apk   CVE-2023-42363  Medium
cups-libs                     2.4.7-r0                  apk   CVE-2018-6553   High
ghostscript                   10.01.2-r0   10.02.0-r0   apk   CVE-2023-43115  Critical
ghostscript                   10.01.2-r0                apk   CVE-2023-36664  High
ghostscript                   10.01.2-r0                apk   CVE-2023-38560  Medium
ghostscript                   10.01.2-r0                apk   CVE-2023-38559  Medium
gnupg-dirmngr                 2.4.3-r0                  apk   CVE-2022-3219   Low
gnupg-gpgconf                 2.4.3-r0                  apk   CVE-2022-3219   Low
gnupg-keyboxd                 2.4.3-r0                  apk   CVE-2022-3219   Low
gpg                           2.4.3-r0                  apk   CVE-2022-3219   Low
gpg-agent                     2.4.3-r0                  apk   CVE-2022-3219   Low
gpgsm                         2.4.3-r0                  apk   CVE-2022-3219   Low
graphicsmagick                1.3.40-r1                 apk   CVE-2007-0770   High
gst-plugins-bad               1.22.5-r0    1.22.7-r0    apk   CVE-2023-44446  Unknown
gst-plugins-bad               1.22.5-r0    1.22.7-r0    apk   CVE-2023-44429  Unknown
gst-plugins-bad               1.22.5-r0    1.22.7-r0    apk   CVE-2023-40476  Unknown
gst-plugins-bad               1.22.5-r0    1.22.7-r0    apk   CVE-2023-40475  Unknown
gst-plugins-bad               1.22.5-r0    1.22.7-r0    apk   CVE-2023-40474  Unknown
libcrypto3                    3.1.3-r0     3.1.4-r0     apk   CVE-2023-5363   High
libcrypto3                    3.1.3-r0     3.1.4-r1     apk   CVE-2023-5678   Medium
libpq                         15.4-r0      15.5-r0      apk   CVE-2023-5870   Unknown
libpq                         15.4-r0      15.5-r0      apk   CVE-2023-5869   Unknown
libpq                         15.4-r0      15.5-r0      apk   CVE-2023-5868   Unknown
libraw1394                    2.1.2-r4                  apk   CVE-2023-1729   Medium
libraw1394                    2.1.2-r4                  apk   CVE-2020-22628  Medium
libreoffice                   7.5.5.2-r0                apk   CVE-2012-5639   Medium
libreoffice-base              7.5.5.2-r0                apk   CVE-2012-5639   Medium
libreoffice-calc              7.5.5.2-r0                apk   CVE-2012-5639   Medium
libreoffice-common            7.5.5.2-r0                apk   CVE-2012-5639   Medium
libreoffice-connector-postgres 7.5.5.2-r0               apk   CVE-2012-5639   Medium
libreoffice-draw              7.5.5.2-r0                apk   CVE-2012-5639   Medium
```

libreoffice-impress	7.5.5.2-r0		apk	CVE-2012-5639	Medium
libreoffice-lang-en_us	7.5.5.2-r0		apk	CVE-2012-5639	Medium
libreoffice-math	7.5.5.2-r0		apk	CVE-2012-5639	Medium
libreoffice-writer	7.5.5.2-r0		apk	CVE-2012-5639	Medium
libreofficekit	7.5.5.2-r0		apk	CVE-2012-5639	Medium
libssl3	3.1.3-r0	3.1.4-r0	apk	CVE-2023-5363	High
libssl3	3.1.3-r0	3.1.4-r1	apk	CVE-2023-5678	Medium
libx11	1.8.4-r4	1.8.7-r0	apk	CVE-2023-43787	High
libx11	1.8.4-r4	1.8.7-r0	apk	CVE-2023-43786	Medium
libx11	1.8.4-r4	1.8.7-r0	apk	CVE-2023-43785	Medium
openjdk8	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22081	Medium
openjdk8	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22067	Medium
openjdk8	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22049	Low
openjdk8	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22045	Low
openjdk8-jre	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22081	Medium
openjdk8-jre	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22067	Medium
openjdk8-jre	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22049	Low
openjdk8-jre	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22045	Low
openjdk8-jre-base	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22081	Medium
openjdk8-jre-base	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22067	Medium
openjdk8-jre-base	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22049	Low
openjdk8-jre-base	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22045	Low
openjdk8-jre-lib	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22081	Medium
openjdk8-jre-lib	8.372.07-r0	8.392.08-r0	apk	CVE-2023-22067	Medium
openjdk8-jre-lib	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22049	Low
openjdk8-jre-lib	8.372.07-r0	8.382.05-r0	apk	CVE-2023-22045	Low
openjpeg	2.5.0-r3		apk	CVE-2015-1239	Medium
pixman	0.42.2-r1		apk	CVE-2023-37769	Medium
ssl_client	1.36.1-r4		apk	CVE-2023-42366	Medium
ssl_client	1.36.1-r4		apk	CVE-2023-42365	Medium
ssl_client	1.36.1-r4		apk	CVE-2023-42364	Medium
ssl_client	1.36.1-r4		apk	CVE-2023-42363	Medium
tiff	4.5.1-r0		apk	CVE-2023-6277	Medium
tiff	4.5.1-r0		apk	CVE-2023-41175	Medium
tiff	4.5.1-r0		apk	CVE-2023-40745	Medium
tiff	4.5.1-r0		apk	CVE-2015-7313	Medium

Recommended Remediation:

The assessment team recommends updating all out-of-date components to their most recent releases. If this is not possible, the assessment team recommends updating all dependencies to at least the earliest version that addresses all publicly known vulnerabilities.

References:

CVE-2023-43115

APPENDICES

Statement of Coverage

The assessment team performed a Standard Grey Box security assessment of the **Dangerzone** client application, as well as a basic cursory review of the **dangerzone.rocks** web application (in the production environment). The **Dangerzone** application was subjected to a limited engagement which focused on critical aspects of application design and deployment, including its sandboxing mechanism and adherence to security best practices.

In general, the assessment team was able to test the most critical parts of the **Dangerzone** application for the MacOS, Windows, Linux, and Qubes OS platforms.

The assessment team implemented a comprehensive approach in evaluating the **Dangerzone** application, employing dynamic testing methods across all platforms and conducting in-depth code audits within the application's open-source GitHub repository. This dual-strategy involved actively interacting with the application under various conditions and configurations to observe its behavior in real-time, while also meticulously examining the source code hosted on GitHub.

Installation Process

The installation process for the various platforms was examined. Marginal improvements have been identified with regards to the MacOS build, where certain security-related entitlements could be hardened. Information about this is included in the findings of this report.

Software Dependencies

The assessment team evaluated software dependencies as part of the assessment. The assessment revealed that the **Dangerzone** team has established procedures to swiftly deploy updates whenever a vulnerability classified as Critical is discovered in any of the third-party libraries in use. Additionally, a finding (**Vulnerable Software Dependencies**) was created to highlight software dependency vulnerabilities.

Docker Container/Sandbox

Given the high severity and critical importance assigned to this aspect of the **Dangerzone** application by the client, this area received special focus during the assessment. The assessment team endeavored to breach the container(s) where conversions occur. For that purpose, a local debugging environment was established, allowing the assessment team to establish a shell into the containers. It was determined that the isolation layer across all platforms was robust and effectively protected against attacks from entities that might achieve code execution within the container.

Application Architecture Security Review

The assessment team thoroughly reviewed the overall architecture of the **Dangerzone** application. The strategic choice to process potentially dangerous files within fully isolated containers, which are shielded from the network and host access, is commendable from a security perspective. The **Dangerzone** team informed IncludeSec that they are proactively addressing the risk of data disclosure, such as in scenarios where a device running **Dangerzone** is seized or acquired by an attacker. The **Dangerzone** team stated they are evaluating the elimination of persistent files and are considering implementing solutions that exclusively utilize RAM in the future. This approach would ensure that no persistent traces are left on the disk, further enhancing the security and privacy aspects of the application.

Qubes OS Integration Security Testing

An in-depth review of the Qubes OS was not achieved. The required ordering, delivery, and setup of the Qubes OS laptop by the assessment team delayed the start of this area of assessment, and thus the duration of assessment time for this area was shortened to conclude within the allotted overall assessment time. Additionally, the assessment team dedicated time to install Qubes OS software and to become familiar with Qubes OS prior to conducting the assessment after the laptop arrived. Despite this shortened assessment window, the assessment team believes this area was adequately assessed and found that Qubes OS did not expose security-relevant vulnerabilities at the time of assessment.

dangerzone.rocks Review

The static website at https://dangerzone.rocks underwent a basic review and scan, focusing on security best practices. This included an SSL/TLS scan, which revealed that the site supported outdated and deprecated TLS versions. Additionally, an analysis of the NGINX configuration and Dockerfile for this static website showed them to be well-configured, with no vulnerabilities detected.

Further/Future Testing

During the evaluation of the **Dangerzone** application, certain components, e.g., the OCR step, were not tested as extensively as would be desirable to evaluate for all security vulnerabilities. The assessment team identified a potential vulnerability in this area, which has been included in the findings. However, the assessment team did not deeply explore the exploitability of this vulnerability, as it would have required significant research into areas not directly related to **Dangerzone**.

Specifically, **Dangerzone** was found to utilize a third-party library for OCR, and an in-depth analysis of this utilization of a third-party library would have extended beyond the primary scope of the assessment. In general, regarding external third-party components, the assessment was focused mainly on basic security best practices in terms of how **Dangerzone** configured and used these external third-party components and whether the application relied on outdated or vulnerable versions. It is important to note that no comprehensive checks were conducted on the third-party libraries themselves.