July 2020

Sandboxing Files Downloaded Via A Web Browser

Santhosh Kumar Edukulla

Sandboxing Files Downloaded Via A Web Browser

ABSTRACT

When browsing the web, users often download various types of files that are saved directly to the user's device. Such files can potentially contain malware or trigger the download of malicious software when opened, thus making users vulnerable to cyberattacks. Typical web browsers scan downloaded files for malware. However, such scans cannot detect malware in certain circumstances, e.g., new malware for which the signature is not known, large file size of the download, etc. This disclosure describes a sandbox environment for isolating files downloaded during web browsing and scanning such files for malware prior to being transferred to the device filesystem.

KEYWORDS

- Malware scanner
- Web browser
- Browser sandbox
- Sandbox environment
- Virtual machine
- Deep content inspection
- Shell extension
- Filesystem

BACKGROUND

When browsing the web, users often download various types of files such as documents, spreadsheets, images, videos, archives, etc. The downloaded files are saved directly to the local filesystem of the user's device. Such files can potentially contain malware or trigger the

download of malicious software when they are opened, thus making users vulnerable to cyberattacks. Oftentimes, such attacks occur in the background without the user's knowledge. Traditionally, web browsers and email programs scan downloaded files to check for malware so that the user can be protected from the harmful consequences of opening them. However, the scans cannot detect malware in a file if the malware signature is not available (in global databases of known malware), if the downloaded file is bigger than the size limits of files that can be scanned, etc.

DESCRIPTION

This disclosure describes the use of a sandbox environment for isolating files downloaded during web browsing to protect users from potentially harmful consequences of opening unscanned malicious files. The sandbox environment can be implemented within a cloud platform and/or be implemented on the user device.

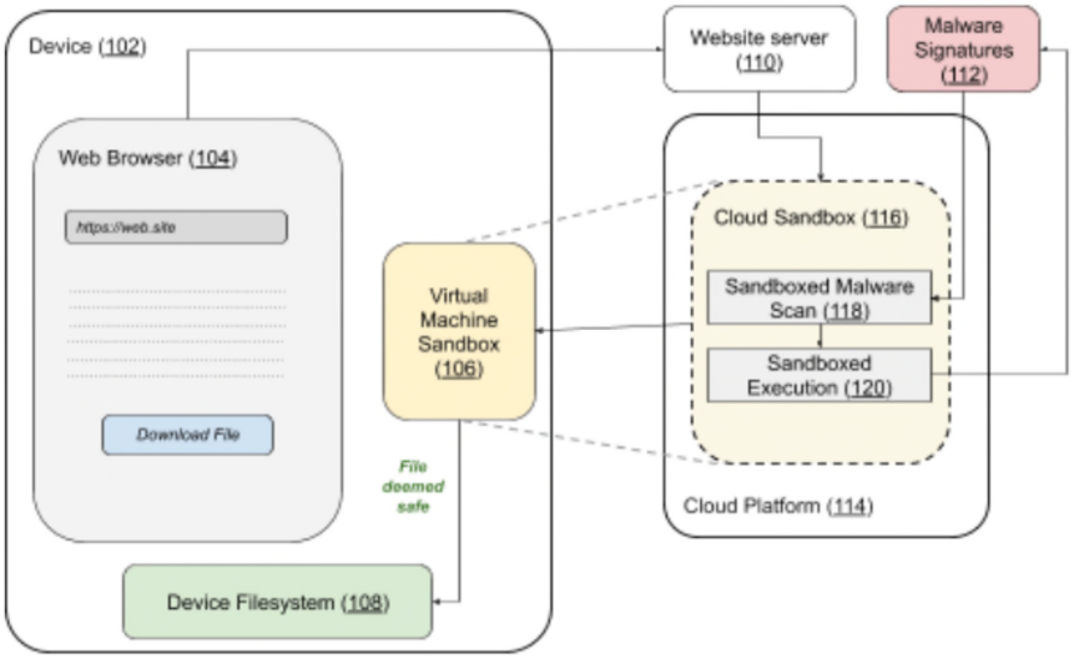

Fig. 1: Sandboxed inspection of downloaded file to check for malware

Fig. 1 shows an example operational implementation of the techniques described in this disclosure. A user browsing a website using a web browser (104) on a device (102) encounters a button to download a file. The file from the website server (110) is downloaded to a sandbox (116) within a cloud platform (114) that is external to the device. Alternatively, or in addition, the sandbox can be implemented within a virtual machine local to the user device (106).

The downloaded file is scanned for malware (118) by matching against a global database of known signatures (112). If no existing signatures match, the file is executed (120) within the sandbox to determine whether it is malicious. If the sandboxed execution indicates that the file is malicious, the global database of malware signatures is updated accordingly with user permission. Otherwise, the file is deemed safe and permitted to move out of the sandbox to the local filesystem of the device (108).

When a user initiates a file download, the file is not directly downloaded to the local filesystem of the device. Instead, the file is first saved to the sandbox that provides a virtual file system accessible via the cloud or implemented within the virtual machine on the user device, e.g., using local control groups (cgroups) and namespaces isolated from the main filesystem of the device.

The sandbox provides capabilities to scan files for malware by comparing against a database of signatures and hashes of known malware. If there is a match, the file is considered malware. When the file does not match known malware, it can be examined further via deep content inspection to check for the presence of executable malicious code using techniques such as: heap spray, port accesses, command and control communication, etc. Further, any new libraries required by the downloaded file are compared with a list of trusted libraries provided in the sandbox. If it is determined upon deep content inspection that the file is likely to include

malicious content despite not matching a known malware signature, the file is marked as malware, and if the user permits, its signature is automatically added to the global database of signature for future reference.

The malware scanning process described above occurs on-the-fly as files are downloaded and saved to the sandbox. Alternatively, an on-access scan can be initiated when the user attempts to open the downloaded file. If the sandboxed malware analysis does not flag the file as malicious, it is deemed safe and transferred out of the sandbox to the local filesystem of the user device. The sandbox includes a shell extension to access the file and read its contents, if possible. The shell serves as the interface to access the file.

The sandbox environment can be implemented locally on the device and/or externally on a cloud platform. In case both local and cloud sandboxes are available, such sandboxes may operate in sync or independently. The global database of malware signatures can be included within the sandbox and/or hosted externally, e.g., as a third party service.

The techniques described in this disclosure can be implemented within any web browser. Further, the techniques can be extended to support inclusion within any software that involves user download of external files. The techniques enable scanning downloaded files for malware regardless of their size, thus making it possible to detect malware embedded in large sized files. In addition, the techniques can detect malware even if the malware signature is not present in the database of known malware signatures, thus protecting against previously unknown malware.

Implementation of the techniques eliminates the need for users to be dependent on separate third party malware scanning software and enhances the security of web browsing by protecting users against harmful consequences resulting from opening malicious downloaded files. Further, with user permission, the techniques can contribute to the detection of new

malware via the intelligence gathered in a distributed manner via sandboxes of the users that permit use of such data.

The described techniques can be extended by appropriate user interface (UI) mechanisms to mark known malware status of a file within the web browser. For instance, download links for files known to be malware can be tagged with an alert, e.g., a red check mark, while those for files known to be safe can be accompanied with a green check mark.

Further to the descriptions above, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs or features described herein may enable collection of user information (e.g., information about a user's file downloads, a user's preferences), and if the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user, or a user's geographic location may be generalized where location information is obtained (such as to a city, ZIP code, or state level), so that a particular location of a user cannot be determined. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user.

CONCLUSION

This disclosure describes a sandbox environment for isolating files downloaded during web browsing and scanning such files for malware prior to being transferred to the device filesystem. The scanning is performed by matching with known malware signatures and deep content inspection. The sandbox environment can be provided within a cloud platform and/or be implemented on the user device. The techniques described in this disclosure can be implemented

within any web browser. Implementation of the techniques eliminates the need for users to be dependent on separate third party malware scanning software and enhances the security of web browsing by protecting users against harmful consequences resulting from opening malicious downloaded files.

REFERENCES

1. Barth, Adam, Collin Jackson, Charles Reis, and TGC Team. "The security architecture of the chromium browser." In *Technical report*. Stanford University, 2008.