# Sandbox: A Secured Testing Framework for Applications

**BY**

**ALLAN NACHAFIYADA WAMMANDA**

**ST/CS/HND/19/015**

**BEING A SEMINAR PAPER**

**PRESENTED TO THE DEPARTMENT OF COMPUTER SCIENCE**

**SCHOOL OF SCIENCE AND TECHNOLOGY**

**FEDERAL POLYTECHNIC, MUBI ADAMAWA STATE**

**DECEMBER, 2021**

## Abstract

*This seminar paper endeavors to review the concepts of sandboxing technique comprising the benefits of the technique, types and limitations. As malware has become more sophisticated, monitoring suspicious malware detection behavior has become difficult. Recent threats have incorporated advanced obfuscation techniques that can evade being detected by endpoints and network security products. Sandboxing protects the critical infrastructure of the organization from suspicious code as it operates in an isolated and separate environment. It also facilitates information technology organizations in testing malicious code in an isolated testing environment for understanding its working within the system and to detect similar malware attacks more quickly.*

### Introduction

A sandbox is a security platform for running unknown executables in a dedicated environment without the risk of affecting the production systems. Basically, sandboxes are virtualized environments that simulate live systems to ensure that the tested executable runs in way that is almost the same, if not identical, to the real environment. Sandbox systems allow the monitoring in an isolated environment of suspicious executable files while minimizing the risk of compromising live systems. Another important aspect of sandbox is that it minimizes human efforts in complex tasks like disassembling the executable to understand its purpose. This facilitates security administrator without extensive malware analysis training to perform a triage of suspicious files and only send confirmed malware for analysis (Adam, Collin and Charles, 2019).

Although malwares have been around since the early days of computers, the sophistication and innovation of malware has increased over the years. The latest ransomware has drawn attention to the dangers of malicious software, which can cause harm to private users as well as corporations, public services governments, and security institutions. To prevent this, malicious activity must be detected as early as possible, before it conducts its harmful acts which is a tedious task especially when dealing with new and unknown malware capable of virtually emulating entire end-user operating environments, a sandbox safely executes suspicious code so its output activity can be observed. Early security sandboxes could only scan executable files but advanced platforms are now able to scan Adobe Flash, JavaScript, and Microsoft Office files, among others. Cutting-edge sandboxing solutions today now provide tight integration into the rest of the security infrastructure (Patidar and Harshita, 2019).

“Sandboxes are increasingly important building materials for secure software systems. In recognition of their potential to improve the security posture of many systems at various points in the development lifecycle, researchers have spent the last several decades developing, improving, and evaluating sandboxing techniques” (Adam et al., 2019).

Sandboxing is a software management strategy that separates applications from critical system resources and other programs. It offers an additional protection layer that prevents system from the adverse impact of malware or harmful apps. To test new programming code, software developers use sandboxes in order to check possible malicious software, cybersecurity experts are using sandboxes. The sandboxes are used for safe execution of malicious code in order to avoid damage to the device, the network, or other devices connected. Runtime sandboxes is actually instancing of virtual machines. High profile applications like the Google Chrome browser, Internet Explorer Protect Mode, and Adobe Reader X employs sandboxing (Clemens, 2017).

Unlimited system resources access and user data can be possible for an application without sandboxing as shown in Fig:1 and the other way around, a sandboxed application can only access its own sandbox’s resources. A sandbox environment is a restricted space and memory area that includes only the resources needed by the software. If a program needs access to sources or files outside the sandbox, the system must explicitly give permission. For example, if a sandbox app is being installed in Operating system, a certain directory is created for the sandbox of the application. The app is provided with unrestricted write and read permission in that directory but no other files can be read or written on a computer storage device except where the system permits it (Bazargan and Zemerly, 2013).

Typically, sandboxing is extended to the most vulnerable components of the application. The sandboxed applications run with very limited and restricted privileges and uses DLL hooks and memory trampolines as required for certain Application programming interfaces (API). This means that the sandbox can examine and rewrite or block some system calls for malicious behavior (Blasing, Batyuk Schmidt, Camtepe & Albayrak, 2010).

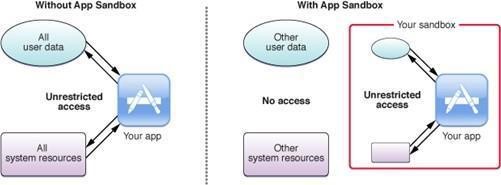


Fig 1: Resource access with and without sandbox (Blasing et al., 2010).

Google chrome browser is a perfect example. The software architecture of chrome is shown in fig 2. Due to Chrome's authors' focus on preventing exploits targeting the rendering engine and installing persistent host malware (the most common scenario for attacks) , they have chosen to just place only the rendering engine component in the sandbox and to allow other browser components to run unmanaged in the kernels browser (Ameiri and Salah, 2011).

Sandbox test detects malware proactively by running detonating code in an isolated and safe environment to monitor the behavior and output activities of the code. Sandbox testing whereas traditional security measures are reactive and based on the identification of signature, which works by searching trends in known malware instances. Since only previously defined threats are detected, sandboxes add another essential security layer. Furthermore, these defenses are only as good as the models which power those solutions even though the initial protection defense uses artificial intelligence or machine learning (signature detection is less), still the need of providing these solutions with advanced malware detection arises (Christiano, 2018).

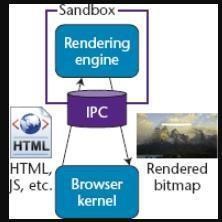


Figure 2: Google Chrome Security architecture using sandbox

### Sandbox Implementations

Many sandboxing solutions are available which can be more or less appropriate according to the needs of your organization. The implementation of sandbox includes three important scenarios:

1. Full Emulation System: The host machine’s physical hardware including CPU and memory is emulated in the sandbox providing deep visible content and effect on program behavior.
2. Operating Systems Emulation: The operating system but not the machine hardware is of end user is emulated in this implementation of sandbox.
3. Virtualization: suspicious programs are contained using virtual servers (Christiano, 2018).

### Cloud Sandbox and Appliance Sandbox

A major classification of sandboxes that exists today is cloud based and appliance-based sandboxes. The traditional method of sandboxing, the appliance sandboxing requires an organization to make an investment in the equipment needed to provide virtual server machines and their deployment in office premises. The operation and maintenance of these virtual servers was costly and occupied large physical space as well. Today's sandboxing appliance solutions are costly and thus usually installed only in the data Centre. When users leave the network, they are exposed to new threats, since they cannot be followed by sandbox protection. Appliances sandboxing or organization hardware sandboxing tests those applications, files, or downloads without any data leaving the organization network. In that case remote workers are exposed to network threats and the appliance sandbox goes blind when they are away from office or in transit. Appliance sandboxing, support limited investigation as malware has been known to hide in SSL traffic. If your software can not monitor all SSL traffic, threats can be broken through and exposed through hardware to the network (Christiano, 2018).

Since cloud sandboxing has remote working benefits, backups and recovery benefits and cost cut for in-house hardware, physical appliances are being utilized less. Cloud sandboxing means that in the sandbox, URLs, downloads or code are tested. Sandboxing on a virtual environment is entirely separated from the system or any other network devices. Potentially malicious files execution can still be unsafe for a business or personal appliance, being physically present and connected to the device. With cloud-based sandboxing, a costly instrument that requires maintenance, updates and ultimately depreciation is eliminated. A virtual server could be started in seconds once Companies wanting to deploy their cloud sandboxes decide a trusted provider and are ready with the requirements of virtual servers, memory and number of user slots they need (Greamo and Gosh, 2013).

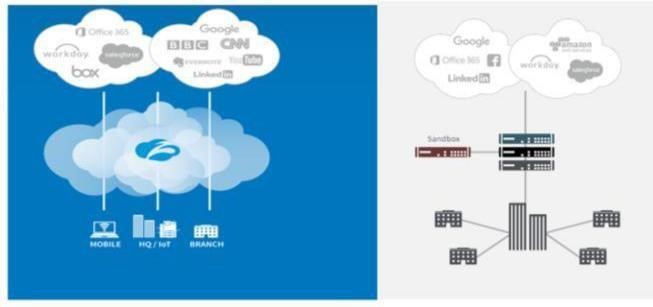


Figure 3: Cloud Sandbox and appliance sandbox

Perhaps the most important advantage of cloud sandboxing is that it can be covered remotely in comparison to appliance-based sandboxing. When a user leaves the network, they might be exposed to threats because company appliances cannot accompany them. Whatever the location, the cloud sandbox can protect the entire network (Greamo and Gosh, 2013).

**Sandbox Applications**

Some of the important Sandbox applications include: Browser plug-in content has always been dependent on the use of a sandbox, like Microsoft Silverlight, and Adobe Flash to view the content loaded from browser plugins.

However, it was notoriously difficult to sustain this form of content. While playing the Flash game on a web page was safer than downloading the game and running the game as a standard program, content publishers have mostly moved away from those plug-ins for publishing HTML5 content that contains sandbox features to enable your browser to deactivate any features that may cause security risks.

1. PDFs and other documents may include executable code, so Adobe Reader runs PDF files in a sandbox, which stops them from escaping the PDF viewer and interfering with the rest of the computer. Microsoft Office also has a sandbox mode that prevents system being tampered with unsafe macro structures.
2. Mobile apps are typically executed in sandboxes by mobile platforms. Applications on IOS, Android and Windows are prohibited in doing a variety of things standard desktop apps have permission for. For instance, mobile apps have to declare permissions for user's location access. The sandbox isolates applications and prevents the applications from tampering each o Web browsers: You can run a trustworthy web browser within a sandbox. If a web site exploits vulnerability in the browser, then the damage is limited to the sandbox and reduced.
3. Software protection: Certain tools are available that users can make use of to run software that are not trustworthy in sandboxes, so they cannot access their private data or damage their devices. The software generally cannot detect that it is limited to a virtual environment, because the sandbox seems to be a complete system for the software.
4. Security research: Security information professionals are using sandboxes to detect malicious code or for research purpose. For example, a security tool could visit sites to check files or install and run the software (Greamo and Gosh, 2013).

#### Sandboxing Benefits

Sandboxing solutions enables companies to set up, test, and launch software with the help of virtual environments it provides. This cutting-edge solution has become increasingly popular because they are accessible, flexible and can save a company a significant amount.

According to Jamalpur*,* Navya, Raja, and Rao (2018), virtual sandbox environments have several applications. It is widely used to streamline and optimize the development process, identifying bugs and fixing it, testing patches. It can also double up as a working directory depending on objective the company.

1. Sandboxing technology make use of virtual servers for testing software in an isolated environment. It enables Developers test certain features without having to worry about compatibility problems caused by other background programs.
2. Repeated usage nature of sandboxing environments is the perfect way to test IT solutions. It equips a company to analyze malicious code, untrusted software and other risks Without contaminating its own systems.
3. Sandboxes can also enable an external developer's mirrored production environment to develop an app using a sandbox web service. Sandbox solutions makes creating and deploying environments effortless even bigger at scale. It enables the users to test certain versions, incorporate new code lines and test them viz-a-viz control. This allows thirdparty developers to validate their codes ahead of taking it to the production environment.
4. A Sandbox Application Programming Interface enables development and testing of APIs. It imitates the features of the production environment so as to produce simulated responses for APIs representing the behavior of a real-time device.
5. Sandbox can be used to test software changes before they are launched, implying that there are lesser problems during and after testing because the testing environment is completely isolated from the actual production environment.
6. Sandboxing can also be used to exploit unreported vulnerabilities for quarantining zeroday risks. Although sandboxing cannot prevent zero-day threats, it provides an extra layer of security in isolating the threats from the rest of the network. When viruses and threats are quarantined, security researchers can use them analysis purpose and identify network vulnerabilities and patterns and help prevent attacks in the future.
7. Sandboxing also complements other security systems, including behavior management and virus systems. It provides additional protection from some malwares that cannot be easily detected by an antivirus software.
8. Cooperation is an important aspect of every company. The sandboxes are great to receive valuable feedback from different departments in the company because everyone with the right permissions can have access to them and can strengthen collaboration through all agencies
9. Provides advanced networking connectivity and support for nested virtualization. You gain access to complex topologies and advanced networking features without rearchitecting when you work with a reputable provider of sandbox technologies.
10. Integration of multiple builds of a project isn’t easy. Sandbox technology enable the checking of compatibility of different builds and validation of the overall solution being developed properly
11. Sales demos and virtual Proof of concepts usually include videos and different forms of multimedia presentations. Sandboxing enable the organization to engage potential customers and existing customers more interactively. Sandbox can enable testing a company’s software from anywhere in the world on the terms of the company.
12. Sandboxing also provides a Quality assured performance. Testing and performance optimization is a never-ending process in the field of software development. Sandbox can be used to test, optimize and enable the Quality Assurance teams to identify potential problems before they go unaddressed.
13. Provide significant cost savings for your company. It's an expensive process to purchase, maintain and manage your own development laboratories. You will only pay for services that you use with cloud-based sandboxing (Jamalpur *et al.,* 2018).

#### Sandbox Evasion Techniques

Malware developers are working effectively in circumventing the new and most sophisticated methods for threat detection. Some basic techniques for evading sandbox include the following

* Sandbox detection: Sandbox environments appears different slightly than a real environment of end user. Malwares either stall execution of harmful activities or terminate immediately if it detects a sandbox.
* Exploitation of Weaknesses and gaps in sandbox: As sophisticated as sandbox, Malware developers can also identify and exploit the weak points. One example is that the sandbox cannot process obscure file formats or large file sizes or the sandbox incurs a blind spot where malicious code can be deployed if the monitoring system of the sandbox is circumvented.
* Incorporating Context-Aware Triggers: Context- aware malware works by exploiting weaknesses of the automated sandbox technology. For example, logical bombs could postpone code detonation for a period of time or until a trigger occurs, such as system reboots or mouse and keyboard interactions, typically occurs only on the end-user system (Koyuncu and Bostanci, 2009).

#### Limitations

Though sandboxing provides users with additional safety, it can also restrict the application capabilities. For instance, A sandboxed application may not allow command line input because commands are executed at the system level, hence utilities like backup programs and keyboard shortcut managers may not operate correctly. Henceforth it is difficult to sandbox certain programs (Koyuncu and Bostanci, 2009).

#### Conclusion

Complex systems are always vulnerable, and the complexity of software only increases over time. How much ever secure coding practices and bug-resistant practices are adopted, attackers just have to pass through the defenses only once to succeed their effort of cracking sandbox. An Unsandboxed application has the full access permission of the resources and the user system running the app. If that app or any framework it is linked against contain security holes, an attacker can potentially exploit those holes to take control of that app, and in doing so, the attacker gains the ability to do anything that the user can do. Although Sandboxed app can’t prevent all attacks against the app, it minimizes the damage that can be brought about.

**RECOMMENDATIONS**

It is recommended that sandboxes serves as versatile tools from which every company and therefore should be employed in every phase of its development, ongoing testing at the project level, quality assurance and support.

It is also recommended that training be done more on the importance and effects of sandbox and its applications in order to utilized it fully.

# References

Adam, B., Collin, J. & Charles, R. (2019). *Google Chrome Team Google Inc. The Security Architecture of the Chromium Browser*. Retrieved 12th September, 2021 from <http://crypto.stanford.edu/websec/chromium/security-architecture.pdf>

Ameiri, F. & Salah, K. (2011). "Evaluation of popular application sandboxing,". *International Journal for Internet Technology and Secured Transactions, 4(3),* 358-362.

Bazargan, F. & Zemerly, J. (2013). State-of-the-Art of Virtualization, its Security Threats and Deployment Models. *International Journal for Information Security Research, (3)*10, 13-39.

Bläsing, L., Batyuk, A., Schmidt, S., Camtepe, H. & Albayrak, S. (2010). An Android Application Sandbox system for suspicious software detection, *2010 5th International Journal on Malicious and Unwanted Software,* 2(3), 55-62.

Clemens, K. (2017). Evasive Malware Tricks: How Malware Evades Detection by Sandboxes. *International Journal of Security and Malware Detection, 6(2)*, 34-64.

Cristiano, B., (2018). Detection of Intrusions and Malware, and Vulnerability Assessment. *International Journal of Computer and Security, 7(2),* 28–29.

Greamo, C. & Ghosh, A. (2013). Sandboxing and Virtualization: Modern Tools for Combating Malware. *International Journal of Electrical Electronics Engineering Security & Privacy, 9(2), 79-82*.

Jamalpur, Y., Navya, P., Raja, G. & Rao, K. (2018). "Dynamic Malware Analysis Using Cuckoo Sandbox," *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, 20(8)*, 1056-1060.

Koyuncu, B. & Bostanci, E. (2009). A Scenario Based Virtual Military Sandbox Implementation Using Web Services", *2009 International Conference on Advanced Computer Control*, 2(9), 767-771.

Patidar, C. & Harshita, K. (2019). Zero Day Attack Detection Using Machine Learning Techniques. *International Journal of Information Communication Technology and Security detection, 6(2),* 20-48.