

Safely and Flexibly Sandboxing Python Code in a C++ Runtime

Thema:

Safely and Flexibly Sandboxing Python Code in a C++ Runtime

Art:

BA, MA

BetreuerIn:

Jürgen Hahn

ErstgutachterIn:

Raphael Wimmer

Status:

Entwurf

Stichworte:

Sandboxing, Python3, C++, Security

angelegt:

2019-10-07

Hintergrund

Das Sandboxing von Applikationen ist eine Methode in der Software Entwicklung, um die Kontexte, in denen Code ausgeführt werden kann, einzuschränken. Dementsprechend wird die Sicherheit einer Applikation vor externen Angreifern, Schadsoftware aber auch gegenüber unerwünschten Zugriffen auf Systemressourcen oder von anderen Applikationen gesteigert. Eine Art des Sandboxing ist die Verwendung von Skriptsprachen in größeren Software-Systemen, wie Betriebssysteme oder Game Engines, die traditionell in C/C++ geschrieben sind. In diesem Benutzungskontext wird mithilfe der Skriptsprache das „Was?“ modelliert, das passieren soll, während das einbindende Software-System das „Wie?“ größtmöglich generisch regelt. Ein Beispiel aus dem Videospielekontext ist die Implementierung eines Projektils. Der Anwendungsentwickler „scriptet“ ein Projektil und definiert das Aussehen, die Größe, die Geschwindigkeit, den Schaden bei einem Treffer (das „Was“), usw., während die Engine die rechenaufwendigen Aspekte übernimmt, wie die Physiksimulation (z.B. Schwerkraft), Kollisionserkennung (Hit-Boxes), etc. (das „Wie“). Skriptsprachen werden für solche Anwendungsszenarien bevorzugt eingesetzt, da sie schnelle Iteration erlauben, leichter zu verwenden sind als C/C++ und auch Entwicklungen dritter zulassen (Modding

Zielsetzung der Arbeit

In dieser Arbeit soll anhand des Threat-Modelling-Prinzips für eine gegebene C++-Runtime, die Scripting mit Python 3 unterstützt, erarbeitet werden, innerhalb welcher Szenarien es für Benutzer möglich ist sich aus der Runtime „rauszuhacken“. Hierbei geht es primär um unbeabsichtigte „Attacken“, wie Bugs und undefiniertes Verhalten in den Scripts abzufangen, als konkret sich gegen böswillige Angriffe zu verteidigen. Im nächsten Schritt soll ausgehend vom State of the Art von Sandboxing anhand der evaluierten Szenarios eine Integration der Sandbox für Python 3 Scripting in die C++-Runtime vorgenommen werden. In einem weiteren Schritt soll evaluiert werden, ob das Sandboxing hinreichend gegen das Threat-Model schützt.

Konkrete Aufgaben

- Aufbereitung von Literatur zum Thema (1 Woche)
- Erstellen des Threat-Models (1 Wochen)
- Integration des Thread Models in C++-Runtime (2 Wochen)
- Testing des Thread Models und Bugfixing (1 Woche)
- Evaluation der Sanbox (1 Woche)
- Schriftliche Ausarbeitung (2 Wochen)

Erwartete Vorkenntnisse

- gute Programmierkenntnisse in Python 3 (von Vorteil: sehr gute Kenntnisse)
- rudimentäre Programmierkenntnisse in C/C++ (von Vorteil: gute Kenntnisse)
- technisches Interesse und sorgfältige Arbeitsweise werden vorausgesetzt
- von Vorteil: Erfahrung mit Security-Aspekten und Sandboxing

Weiterführende Quellen

[1] <https://docs.unrealengine.com/en-US/Engine/Blueprints/index.html> [2] Sewell, B. (2015). Blueprints Visual Scripting for Unreal Engine. Packt Publishing Ltd. [3] <https://de.wikipedia.org/wiki/Skriptsprache> [4] Prevelakis, V., & Spinellis, D. (2001, June). Sandboxing Applications. In USENIX Annual Technical Conference, FREENIX Track (pp. 119-126). [5] Ansel, J., Marchenko, P., Erlingsson, Ú., Taylor, E., Chen, B., Schuff, D. L., ... & Yee, B. (2011, June). Language-independent sandboxing of just-in-time compilation and self-modifying code. In ACM SIGPLAN Notices (Vol. 46, No. 6, pp. 355-366). ACM.

From:
<https://wiki.mi.ur.de/> - MI Wiki

Permanent link:
https://wiki.mi.ur.de/arbeiten/safely_and_flexibly_sandboxing_python_code_in_a_c_runtime?rev=1570451334

Last update: **07.10.2019 12:28**

