Secure Your Save Files

Insecure Deserialization in Ren'Py Games - SWIntern Presentation

whoami?

Ian Eusebio

@iangg.

4th year - CIS

Competitions Officer and SWintern

Icebreaker

1.) Have you built a game with a narrative?
2.) Favorite premise or storyline in a video game?

Table of Contents

Learning Objectives

01 Explain how Ren'Py can be used in video game development

02 Describe serialization and deserialization within the context of Ren'Py

03 Understand how hackers can use insecure deserialization to manipulate a computer through a Ren'Py save file.

04 Demo the example video game save file trojan file

05 Identify ways to prevent insecure deserialization

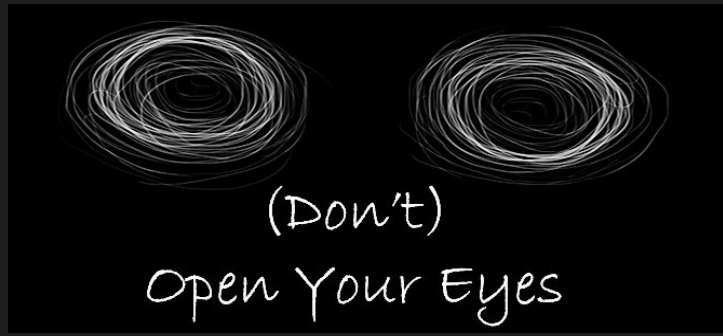
01 What is Ren'Py?

a Python-based video game engine for text-based choose your own adventure games

Why use Ren'Py?

- Free, open source, beginner-friendly
- "It's like coding a slideshow"
- Allows for branching narratives and multiple endings
- **Bonus**: SWIFT uses Ren'Py sometimes to make bonus CTF challenges

Game Examples:



Other Games

Black Closet, Magical Diary, Ren'Py Chess

How is this connected to cybersecurity?

Video games are also pieces of software that need to be secured. Save files are an attack vector for hackers.

Ren'Py being Python-based leaves it open to the same vulnerabilities as Python, including insecure deserialization.

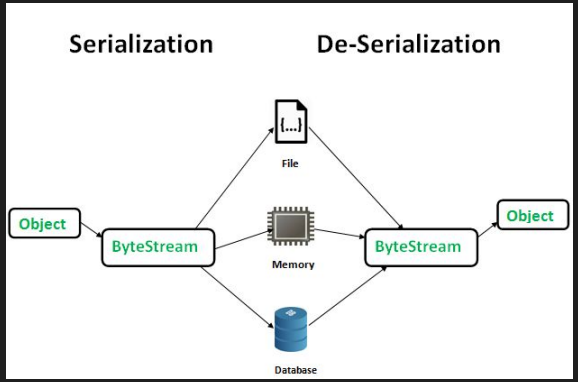
02
Insecure Deserialization

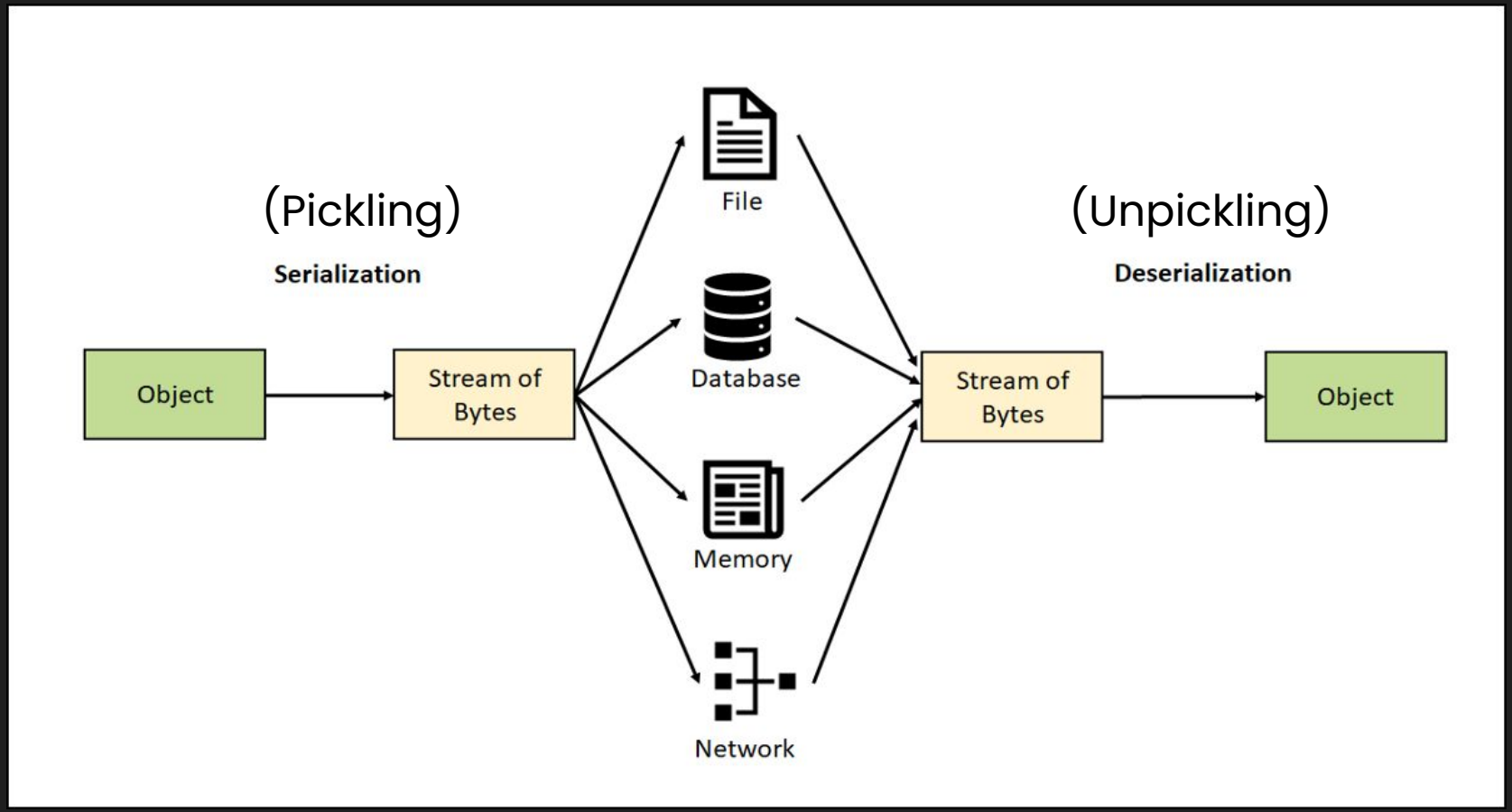
How Software/Malware is run
(Source: Dylan)

- Compiled Programs - generally dependent on hardware architecture and operating system (i.e. Windows)
  - Executables (.exe)
  - Installers/App Bundles (.msi, .msix)
  - Shared Libraries/Packages (.dll)
- Scripts - run through an interpreter
  - This virus is a script type because it uses Ren'Py/Python
  - Python: python (.py)

Definitions

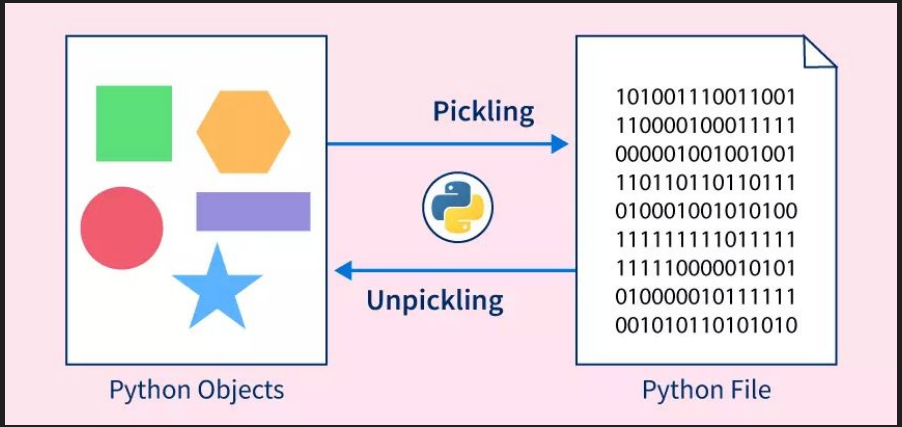
- **Serialization** - translating an object into a format that can be stored or transmitted (a byte stream)
  - **Byte stream** - an ordered sequence of bytes. They can represent virtually any kind of data, including video game save files.
- **Deserialization** - the reverse of serialization. The process of reconstructing an object from a byte stream.
- **Insecure Deserialization** - vulnerability that occurs when an attacker can run arbitrary code to a deserializer (OWASP Entry)
  - *This is also a bug class that can show up in cybersecurity competitions!*




(Pickling)
Serialization
(Unpickling)
Deserialization
Object
Stream of Bytes
File
Database
Memory
Network
Stream of Bytes
Object

Pickles - a Metaphor and Python Library

- **Python Pickle** - Python's process for serializing and deserializing objects.
  - Items are pickled to extend their shelf life for later consumption, similar to how data is pickled (serialized) to be retrieved (deserialized) later.
  - Ren'Py uses the Pickle library to handle its save files.
  - More dangerous than other forms of serialization because it can execute code



[Python Docs: Pickle]

03
Virus Analysis

Background and Demo of example save file trojan

Virus Background

- Created by a magician that became a security engineer, aka swoops on Github
- Magic tricks rely on deception, similar to malware

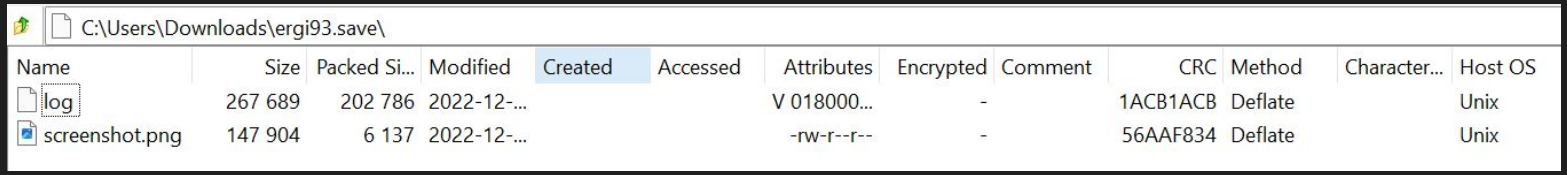
Origin of this Workshop

- Competed in the video game category last year for the BroncoHacks hackathon by making a dating sim

Demo Time!

- Backup Video:

The Save File


C:\Users\Downloads\ergi93.save\

Name	Size	Packed Si...	Modified	Created	Accessed	Attributes	Encrypted	Comment	CRC	Method	Character...	Host OS
log	267 689	202 786	2022-12-...			V 018000...	-		1ACB1ACB	Deflate		Unix
screenshot.png	147 904	6 137	2022-12-...			-rw-r--r--	-		56AAF834	Deflate		Unix

- Ren'Py save files are zip archives with a screenshot, metadata, and then game save data

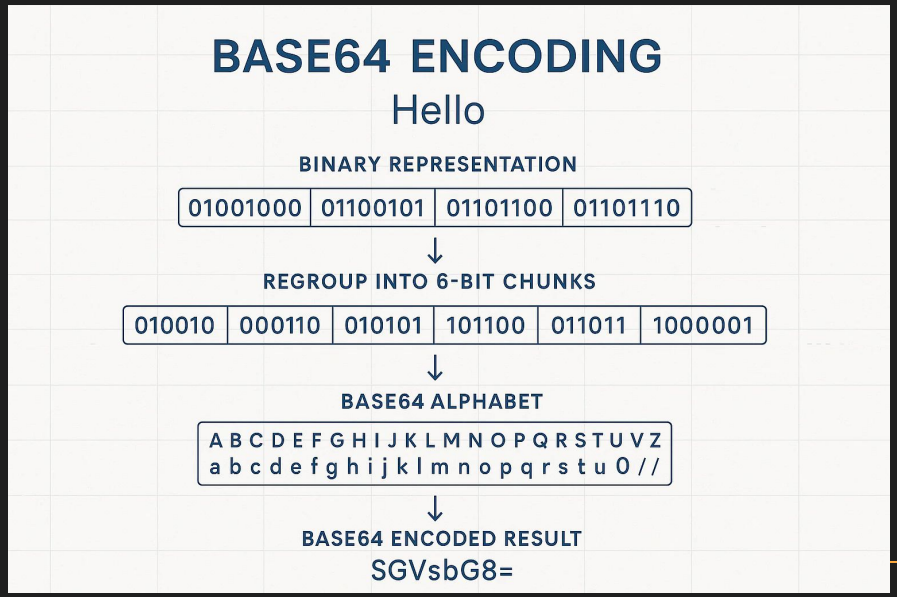
Obfuscated Payload (log)

```
1  ]2acpickle
2  loads
3  czlib
4  decompress
5  cbase64
6  b64decode
7  Tô SOH NUL NUL eJxVUbFKA0EQDcGArGDhV4hFUFEbReEKOcdSExSUZW9v7m4vm93zduOpWFj4CVaClV9
```

- The log file is the pickled save data
- Bottom line is obfuscated
- This is where the payload was hiding
  - It was compressed with zlib then encoded with base64

What is Base64?

- Encoding scheme that converts binary data into a printable format
- Commonly used in viruses to obfuscate malicious code



Deobfuscated Payload

```
label start:
    $ import sys, base64, zlib
    $ print(zlib.decompress(base64.b64decode('eJyLMPVXDFB1cAyINokJiIowNdEIiNM0d3bWNK9VcfV0dgzSDQ5x9HNxDHl
    $ if "emscripten" in sys.modules: renpy.run(OpenURL("https://www.youtube.com/watch?v=M5V_IXMewl4"))
    $ name_only("Get stickbugged")
    $ renpy.movie_cutscene("this_is_the_get_stickbug_video_filealsjdf0jwlkjalskjdfoiwjlkajsdf.webm")
    $ name_only("lol")
    jump start

label _after_load:
    jump start

label quit:
    $ name_only("No quit! Only stickbug")
    jump start
```

Payload - Interesting Lines

```
$ print(zlib.decompress(base64.b64decode('eJyLMPVXDFB1cAyINokJiIowNdEIiNM0d3bWNK9VcfV0dgzSDQ5x9HNxDHl
$ if "emscripten" in sys.modules: renpy.run(OpenURL("https://www.youtube.com/watch?v=M5V_IXMewl4"))
```

- $ print… - decodes an EICAR string so the test virus is detected by antivirus programs
  - European Institute for Computer Antivirus Research (EICAR)
- $ if “emscripten”… - so the test virus can work on the web browser version of Ren’Py

ProxMox Instructions

TO DO: Add instructions for ProxMox after exporting the Lubuntu virtual machine

Lubuntu Instructions

Lubuntu is the operating system of the virtual machine that we will be demoing the virus on.

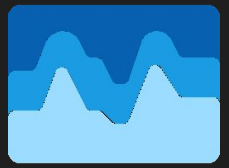
Username:Password -> test:test123

1. Open renpy-8.3.7-sdk folder
2. Run renpy.sh > Execute
3. Projects: Tutorial > Launch Project
4. Start
5. Press Save > Load
6. Load the top middle save > Yes > Yes again

The Virus

This is a harmless example virus made by a GitHub user for demonstration purposes. It traps the player in an endless loop of stick bug gifs and prevents them from exiting the game, forcing them to end the process manually. The creator claims it can work on almost any Ren'Py game if you insert it into the game's save file directory.

Exiting out of the virus



1. Open qterminal on the Desktop
2. Type the command “ps aux | grep renpy”
3. Type “kill -9 [Process ID]”

Killing a process through the command line is like the Linux equivalent of ending a task through Windows Task Manager.

*kill -9 (aka sigkill) forcefully terminates a process and should be used sparingly due to the risk of corrupting files. We use it here for educational purposes.

Video Example of Advanced Virus

Here is a video of a more advanced version of a video game save file trojan in Ren'Py:

https://youtu.be/ACTlg2M9tEs

In the video, the virus

- Deletes all the user's save files
- Manipulates files in the home directory
- Accesses the web cam

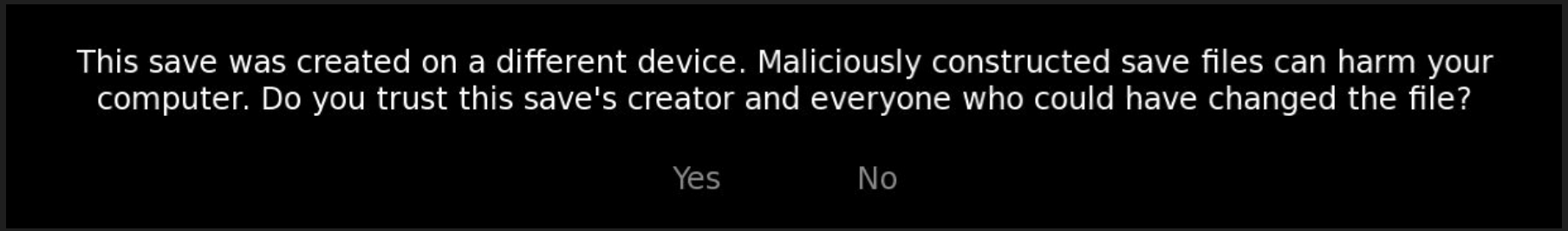
04

Prevention and Outro

Mitigation strategies, further learning, and closing thoughts.

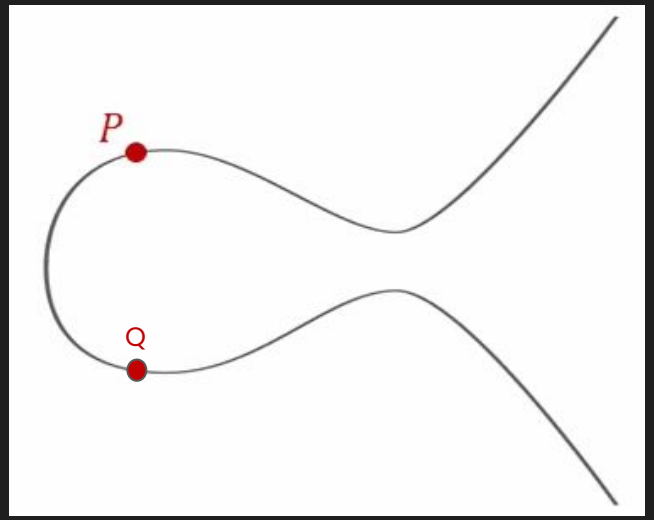
Ren’Py Security Measures

- “Ren'Py now uses tokens to warn users when a save file is moved between devices” (Changelog Link)
  - Save Token code - Thanks to Ren’Py being open source, we’re able to view its code on GitHub.
- The official documentation advises against sharing save files between players (Security - Ren’Py)
- Python’s documentation on its Pickle library similarly warns against unpickling data from untrusted sources (Pickle Docs)



Elliptic Curve Digital Signature Algorithm

- Ren'Py now uses ECDSA to verify save files are from trusted sources and not tampered with
- Q = nP, where Q = public key, n = private key.
  - Cannot use regular division to find n because Q and P are points on an elliptic curve
- Gif of elliptic curve scalar multiplication:



Snippet from savetoken.py in Ren'Py Source code

```
def create_token(filename):
    """
    Creates a token and writes it to `filename`, if possible.
    """

    try:
        os.makedirs(os.path.dirname(filename))
    except Exception:
        pass

    sk = ecdsa.SigningKey.generate(curve=ecdsa.NIST256p)
    vk = sk.verifying_key
    if vk is not None:
        line = encode_line("signing-key", sk.to_der(), vk.to_der())

        with open(filename, "w") as f:
            f.write(line)
```

Mitigation

While no option is foolproof, there are ways to mitigate risk when loading save files from unknown sources:

- Only downloading files from trusted sources
- Uploading the file to VirusTotal for scanning (does not catch every virus)
- Sandboxing or isolating the potentially malicious save files
  - Using a virtual machine like we did earlier
  - Load the file in Ren'Py's web browser version. Browsers tend to have better isolation due to their limited permissions

Prevalence

- Some small and niche Ren'Py communities share Ren'Py save files
- Some indie game devs have players upload their save files to identify game bugs - very dangerous!
- Unity, another video game engine, is also susceptible to insecure deserialization vulnerabilities due to many games' use of the obsolete BinaryFormatter class for their save file systems.

Further Learning

As of Ren'Py version 8.4.0, this example virus causes Ren'Py to crash and only works on previous versions, hence why we used version 8.3.7.

Possible Future Projects:

1. Try to create a version of the example virus that works on newer versions of Ren'Py.
2. Reverse engineer the advanced version of the virus from the video.

Sources

- GitHub: Video Game Save File Trojans
  - Write-up on Ren'Py save file vulnerability that this workshop was based off of.
  - Video: r2con 2024 - Cracking Pickles
- Ren'Py Source Code: Github
- Ren'Py Docs: Security
- Python Docs: Pickle Library
- OWASP: Insecure Deserialization
- Medium: Serialization and Deserialization in Python
- YouTube: Elliptic Curve Cryptography 101

Thanks

- https://linktr.ee/calpolyswift
- https://www.cppgamedev.com/

Please sign in if you haven't already!