Undergraduate final project workshop in Cyber

Security Protection against ransomware attacks that hide in models capable of running hostile code or that hide in metadata

Raz Elbaz, Ran Dubin and Amit Dvir
Ariel Cyber Innovation Center,
Department of Computer Science,
Ariel University Ariel, Israel
rand, amitdv@ariel.ac.il

February 2023

1 pickle file

What is a pickle file type? The pickle module implements binary protocols for serializing and de-serializing a Python object structure. "Pickling" is the process whereby a Python object hierarchy is converted into a byte stream, and "unpickling" is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

When entering the Python website we can see that under the description of the file type the following warning appears: The pickle module is not secure. Only unpickle data you trust. It is possible to construct malicious pickle data which will execute arbitrary code during unpickling. Never unpickle data that could have come from an untrusted source, or that could have been tampered with. Consider signing data with hmac if you need to ensure that it has not been tampered with. Safer serialization formats such as json may be more appropriate if you are processing untrusted data. See Comparison with json.

A pickle file is a binary file format used in Python programming language to serialize and deserialize Python objects. Pickling is the process of converting a Python object hierarchy into a byte stream, while unpickling is the inverse process of converting the byte stream back into a Python object hierarchy.

The main advantage of using pickle files is that they can be used to save and load complex Python objects, including custom classes, lists, dictionaries, and other data structures, while preserving their state and relationships. This can be useful for saving the state of a program, storing data for later use, or sharing data between different Python programs.

Pickle files have some limitations, such as being specific to Python and not being interoperable with other programming languages. Additionally, they can potentially introduce security risks if they are not handled properly, as they can be used to execute arbitrary code when loading malicious pickled objects. Overall, pickle files are a powerful and flexible tool for working with Python objects, but care should be taken to ensure that they are used safely and securely.

Why does using pickle in the ML model contain security risks? Using pickle to serialize and deserialize machine learning models can introduce security risks because pickle is a powerful and flexible format that can execute arbitrary code. This means that if an attacker can craft a malicious pickle file and convince a user to open it, they could potentially execute arbitrary code on the user's machine. For example, an attacker could craft a pickle file that, when unpickled, causes the system to delete all files in the current directory. Or, attacker could craft a pickle file that, when unpickled, causes the system to run a shell command and exfiltrate data from the machine.

2 attacks

There are no specific cyber attacks that target pickle files specifically. However, pickle files are vulnerable to various types of attacks that can compromise the security and integrity of the data they contain, such as:

1. Deserialization attacks: An attacker may manipulate the contents of a pickle file to execute arbitrary code on the system when the file is deserialized. This can lead to remote code execution and other types of system compromise.
2. Tampering attacks: An attacker may modify the contents of a pickle file to change the data it contains, or to inject malicious code into the file. This can be used to steal sensitive information or to disrupt the operation of the system.
3. Denial of Service (DoS) attacks: An attacker may flood a system with pickle files to consume system resources and cause the system to become unresponsive. This can be used to disrupt the normal operation of the system and prevent legitimate users from accessing it.

It's important to always be careful when working with pickle files, and to ensure that they are only opened from trusted sources and processed using secure methods.

3 Metadata

Pickle files store not only the data but also the metadata that describes the structure of the data. Attackers can carry out attacks on the metadata of pickle files in various ways:

1. Malformed metadata: Attackers can modify the metadata of a pickle file to include invalid or malicious values. For example, an attacker could change the length or type of a data structure in the metadata, causing the pickle file to fail to load properly or to execute malicious code when loaded.
2. Metadata injection: Attackers can inject their own metadata into a pickle file, which can be used to modify the behavior of the program that loads the file. For example, an attacker could inject a fake object into the metadata that appears to be a trusted object but actually executes malicious code when loaded.
3. Metadata tampering: Attackers can modify the metadata of a pickle file to change the behavior of the program that loads the file. For example, an attacker could modify the metadata of a pickle file to disable security checks or to bypass authentication.

To protect against attacks on the metadata of pickle files, it's important to use secure coding practices when creating and processing these files. This includes validating input data, using encryption and hashing techniques to protect sensitive data, and limiting access to pickle files to trusted users and processes. Additionally, it's important to keep software and libraries up-to-date to ensure that any known vulnerabilities are patched.

4 RELATED WORK

[3] This article discusses Python's Pickle module which provides a well-known capability to invoke arbitrary Python functions and, by extension, enable remote code execution; However, there is no public guide to exploiting Pickle and published exploits are only simple examples. The article describes the Pickle environment, obstacles a shellcoder faces, and provides guidelines for writing Pickle shellcode.

[2] This article accurately describes DoS attacks and their mitigations.

[1] This blog describes and demonstrates how to run ransomware automatically from a machine learning model.

5 Tasks

1. Learn about a pickle file: what it is, how to attack it. Execute an attack - I implemented a malicious pickle creation, I implemented sender and receiver programs to implement an attack that activates a code that runs automatically on the receiver and performs an attack.

2. Develop a code that statically reads the pickle file and scans the active parts in it.

3. Removing malware from a pickle file

4. The task is to modify the pickle package by removing parts that could be used for malicious attacks, add a new capability called cdr.py, and then test that this change works by running parse and cdr, and verifying that parse returns ”clean” after cdr is called. I need to remove the parts that can cause an attack because a pickle has parts that make it explosive. I want to take their package. Add a new capability called cdr.py. and then in the tests to perform: analysis cdr analysis when the latter should show me ”clean”

   Here are the steps I will take:

   I will identify the parts of the pickle package that may be dangerous and should be removed. I'll refer to the Python documentation on pickle for more information.

   After identifying the parts that need to be removed, a create a different version of the pickle package that does not include these parts. I will modify the existing pickle package or create a new package that is a different version of the original.

   I will add a new capability called cdr.py. I will create this file in the same directory as the modified pickle package.

   In cdr.py, I will write code that performs the desired functionality.

   I will write tests that call the analysis and cdr, I will make sure that the analysis returns ”clean” after reading cdr.

   I will run the tests and make sure they pass.

   If the tests fail, I will debug the code and fix problems.

   After the tests pass, I can use the custom pickle package and cdr.py in your project.

   Changing a Python core package like a pickle can be dangerous and may have unintended consequences. It's important to thoroughly test your changes and make sure they don't break any existing code.

References

[1] Tom Bonner Eoin Wickens, Marta Janus. Weaponizing machine learning models with ransomware.

[2] Kousei Tanaka Taiichi Saito. Python deserialization denial of services attacks and their mitigations.

[3] Marco Slaviero. Sour pickles.