Analysis Report for: main.pyc

Decoded using latin-1...

The provided code is a chaotic mix of seemingly unrelated sections: a large block of uninterpretable binary data, a PowerShell script, and fragments of Python code. The binary data is likely not C code and is essentially gibberish in this context. The PowerShell and Python sections appear to be separate, independent programs. A proper analysis requires separating and analyzing these parts individually.

Overall Functionality

The code appears to consist of three distinct parts:

1. **Binary Data:** A large section of binary data that is uninterpretable without further context or knowledge of its origin. It's not valid C code.
2. **PowerShell Script:** This script retrieves information from an Active Directory, specifically:
* Inactive users (last logon more than 6 months ago).
* Inactive computers (last logon more than 3 months ago).
* Disabled users (disabled for more than 3 months).
The script then exports this information into CSV files.
3. **Python Code Fragments:** These fragments seem to be part of a program that generates an HTML table from CSV data. It also includes sections related to sending an email using SMTP, potentially to report the Active Directory findings.

PowerShell Script Analysis

* **Function Summaries:** The PowerShell script doesn't define explicit functions but uses cmdlets (Get-ADUser, Get-ADComputer, Export-Csv, etc.). These are built-in functions for managing Active Directory.
* **Control Flow:** The script follows a straightforward sequential flow:
1. Sets thresholds for inactive users and computers.
2. Queries Active Directory for users and computers based on the defined criteria, excluding service accounts.
3. Selects specific properties and formats the output.
4. Exports the results into CSV files.
* **Data Structures:** The main data structures used are objects returned by `Get-ADUser` and `Get-ADComputer` cmdlets. These are essentially collections of user/computer attributes.
* **Potential Issues:** The script could be improved by adding more robust error handling and logging.

Python Code Fragments Analysis

* **Function Summaries:** The Python code consists of fragments without clear function definitions, but seems to include functions for:
* Reading CSV files (`csv.reader`)
* Creating HTML tables from CSV data (`crea_tabella_html_from_file`)
* Sending emails using SMTP (`smtplib`).
* **Control Flow:** The control flow is difficult to ascertain due to incomplete code, but based on the fragments, it likely involves:
1. Reading CSV data.
2. Formatting it into HTML table structure.
3. Constructing an email message.
4. Sending the email using SMTP.
* **Data Structures:** The Python code utilizes lists and potentially dictionaries for storing CSV data and email message components.
* **Potential Issues:** The code lacks exception handling for potential errors (e.g., file not found, SMTP connection errors), making it unstable.

Data Structures (Overall)

The significant data structures used across the code are:

* **Active Directory Objects:** PowerShell uses Active Directory user and computer objects, representing their attributes.
* **CSV Data:** Both PowerShell and Python interact with CSV data, using it as an intermediate format for data exchange.
* **Lists/Dictionaries (Python):** Python likely uses these for storing data read from CSV files and for assembling the email message.
* **HTML Table Structure:** The Python code generates an HTML table as an output data structure.

Malware Family Suggestion

Based solely on the provided code snippets, it's impossible to definitively label it as belonging to a specific malware family. However, the PowerShell script's function—gathering sensitive information from Active Directory and potentially sending it via email—shows characteristics of reconnaissance and exfiltration tools used by various malware families (e.g., information stealers, spyware). The inclusion of what might be obfuscated code further elevates concern. A comprehensive malware analysis would be needed, which involves examining the binary data, performing dynamic analysis and checking for any malicious behavior beyond the straightforward functionality seen in the PowerShell and Python snippets. The potential to send information via email suggests a goal beyond simple reconnaissance. The combination of these three elements warrants a deeper look into the origin and behaviour of the overall file. It is highly suspicious.