Walkthrough

!!! Spoilers Ahead !!!

This walkthrough contains **spoilers and detailed instructions**.

If you want detailed instructions, please proceed to the next page.

Analysis of the Dropper: `procmon64` Step 1

1. **Right-click** the dropper and examine its **properties/description**.
2. Rename the dropper as `w.exe`
3. Start `procmon64` and add a filter where the **Process Name equals** `w.exe`.
4. **Execute** the dropper
5. Filter the `procmon64` log to identify **where the files are dropped and how they are executed**.

Analysis of the Dropped Files

Analyze the `msedgewebview4.exe` file

1. **Open in** `pe-bear`:
   - Examine the **imports**, **strings**, and **sections**. Does anything look suspicious?
2. **Open in** `IDAFree`:
   - **Understand the general layout**:
     - Find the function that **reads and loads the file**.
     - Find the function that **loops over the encrypted file content** and determine **where the encryption key is stored**.
   - **Understand how the encryption is implemented**:
     - Analyze the **XOR loop** (which suggests a **Pseudo-Random Number Generator (PRNG)**).
     - Find the **encryption key** (which serves as the **seed for the PRNG**).

Analyze the `SearchHost.bin` file

1. **Decrypt the bytecode** using `decrypt_bytecode.py` with the **correct key (seed)**.
2. **Check if it matches a well-known architecture** using `find_shellcode_arch.py`:
   - **What heuristic** does `find_shellcode_arch.py` implement?
3. **Open the shellcode in** `ghidra` (specifying the **correct architecture**):
   - Analyze the **API resolution function**:
     - Use `hash_x65599_exports.py` to **identify which APIs are imported** (e.g., `kernelbase!VirtualProtect`, `kernel32!CreateThread`).
     - Determine **how the API is used**.
   - Analyze the **main loop function**.

Toward the Second Stage: `procmon64` Step 2

1. **Continue monitoring with `procmon64`**:
   - Identify the **APIs that are invoked** by analyzing the **stack trace**.
   - Look for `GetComputerName` and `GetUserNameW` in the stack trace.
2. **Create a user**:

   ```
   net user <username> <pass> /add
   ```

3. **Add the user to the local administrators group** (optional):

   ```
   net localgroup administrators <username> /add
   ```

4. **copy the dropped files** into `c:\windows\temp` and **execute the dropper under the new user's credentials**:

   ```
   cd c:\windows\temp
   Start-Process powershell.exe -ArgumentList "-Command & { Start-Process msedgewebview4.e
   ```

5. **Attach to the process with `windbg`** (**activate all user processes view** in windbg - requires elevation):
   - Place a **breakpoint** on `kernelbase!VirtualProtect`.
   - Wait for the breakpoint to **hit**. Look at the stack trace with `kb`. **Identify the parameters**; we expect `r8` **to equal** `0x40` **(Read-Write-Execute - RWX)**.
   - **Execute** `VirtualProtect` and open `SystemInformer`. Find the process and look for the **RWX allocated memory regions**. **Dump the content** and **trim the dump** properly using a hex editor.
   - Greetings! You've found **Stage 2**!