BÁO CÁO TỔNG KẾT ĐỒ ÁN MÔN HỌC

**Môn học: Cơ chế hoạt động của mã độc**

**Tên chủ đề: Process Hollowing**

Mã nhóm: CK17 Mã đề tài: S10

**Lớp**: **NT230.P22.ANTT**

1. **THÔNG TIN THÀNH VIÊN NHÓM:**

(Sinh viên liệt kê tất cả các thành viên trong nhóm)

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Tôn Thất Bình | 21520639 | 21520639@gm.uit.edu.vn |
| 2 | Nguyễn Văn Hào | 20521293 | 20521293@gm.uit.edu.vn |
| 3 | Phạm Trần Hiếu | 21520882 | 21520882@gm.uit.edu.vn |

1. **TÓM TẮT NỘI DUNG THỰC HIỆN:[[1]](#footnote-1)**
   1. **Chủ đề nghiên cứu trong lĩnh vực Mã độc:** (chọn nội dung tương ứng bên dưới)

þ Dev Track

¨ Research Track

* 1. **Tên đề tài**

**Process Hollowing: Nhúng mã độc vào tiến trình hợp pháp**

* 1. **Liên kết lưu trữ mã nguồn của nhóm:**

Mã nguồn của đề tài đồ án được lưu tại: <https://github.com/itmehaone/NT230.P22.ANTT_Process-Hollowing>

(Lưu ý: GV phụ trách phải có quyền truy cập nội dung trong Link)

* 1. **Tên tài liệu tham khảo chính:**

|  |
| --- |
| **- Wikipedia, blogs, diễn đàn bảo mật.**  **- Thư viện pefile, MinHook, WinAPI, tài liệu MSDN. - Gihub:**   <https://github.com/m0n0ph1/Process-Hollowing>  <https://github.com/TsudaKageyu/minhook> |

* 1. **Tóm tắt nội dung chính:**

|  |
| --- |
| **Process Hollowing** là kỹ thuật cho phép mã độc ẩn mình bên trong một tiến trình hợp pháp bằng cách khởi tạo tiến trình ở trạng thái treo (suspended), xóa nội dung bộ nhớ image cũ và chèn mã độc vào, sau đó khởi chạy lại. Mục tiêu là vượt qua phát hiện của AV/EDR. Nhóm đã hiện thực quy trình process hollowing, xây dựng hệ thống phát hiện tĩnh (static) bằng cách phân tích PE file, và phát hiện động (dynamic) qua inject DLL và hook API ghi bộ nhớ, từ đó đối chiếu chữ ký độc hại. |

* 1. **Tóm tắt các kỹ thuật chính được mô tả sử dụng trong đề tài:**

|  |
| --- |
| **Kỹ thuật 1: Process Hollowing**  Cho phép chèn mã độc vào tiến trình hợp pháp bằng cách tạo tiến trình ở trạng thái suspended, giải phóng vùng nhớ gốc, ghi mã độc vào, chỉnh entry point và khởi chạy lại tiến trình.  **Kỹ thuật 2: Static Detection**  Phân tích PE file để trích xuất các API nguy hiểm thường xuất hiện trong process hollowing. Sử dụng thư viện pefile để phát hiện các chuỗi API nghi vấn và đánh dấu file đáng ngờ.  **Kỹ thuật 3: Dynamic Detection**  Inject DLL vào tiến trình mục tiêu, hook API WriteProcessMemory, giám sát dữ liệu được ghi và so sánh với chữ ký mã độc. Ghi log và chặn tiến trình nếu phát hiện hành vi đáng ngờ.  **Kỹ thuật 04: Bypass Windows Defender**  Né tránh phát hiện của AV/EDR bằng cách dùng dynamic API resolution (lấy hàm qua GetProcAddress), injection chỉ trong bộ nhớ, mã hóa payload bằng XOR, và tránh tạo file/process mới để giảm dấu vết. |

* 1. **Môi trường thực nghiệm của đề tài:**

|  |
| --- |
| **- Cấu hình máy tính:**  + Windows 10, RAM 8GB, CPU Intel i5  **- Công cụ hỗ trợ:**  + Python (tĩnh – thư viện pefile), C++ (hook API), MinHook, VS2019  **- Ngôn ngữ lập trình:** Python, C/C++  **- Đối tượng nghiên cứu:**  + Tệp PE mẫu có kỹ thuật process hollowing  **- Tiêu chí đánh giá:**  + Khả năng phát hiện (số vụ process hollowing được log)  + Số chữ ký đúng  + Khả năng tránh false positive |

* 1. **Kết quả đạt được: Công việc/tính năng/kỹ thuật mà nhóm thực hiện lập trình và triển khai cho demo:**

|  |
| --- |
| **- Hiện thực thành công kỹ thuật process hollowing**  **- Demo hoàn chỉnh hành vi và phát hiện.**  **- Xây dựng hệ thống phát hiện:**  + Tĩnh: Phân tích PE, phát hiện dựa vào API gọi  + Động: Inject DLL, hook API, log thao tác nghi ngờ  **- Ưu điểm:**  + Phát hiện nhanh, chính xác  + Có khả năng mở rộng với thêm chữ ký/mẫu  **- Hạn chế:**  + Hook có thể bị tránh  + Một số kỹ thuật shellcode nâng cao chưa xử lý hết |

* 1. **Các khó khăn, thách thức hiện tại khi thực hiện:**

|  |
| --- |
| **- Việc hook API nhạy cảm bị AV chặn hoặc phát hiện**  **- Khó tạo môi trường thực nghiệm an toàn với mã độc** |

1. **TỰ ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH SO VỚI KẾ HOẠCH THỰC HIỆN:**

|  |
| --- |
| **85%** |

1. **NHẬT KÝ PHÂN CÔNG NHIỆM VỤ:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Phân công nhiệm vụ** |
| 1 | + Thực hiện process hollwing truyền thống + Bypass window defender + Thực hiện phương pháp phát hiện | Tôn Thất Bình |
| 2 | + Thực hiện phương pháp phát hiện + Làm slide + Làm poster | Nguyễn Văn Hào |
| 3 | + Thực hiện process hollwing truyền thống + Làm slide + Làm poster | Phạm Trần Hiếu |

BÁO CÁO TỔNG KẾT CHI TIẾT

**Phần bên dưới của báo cáo này là tài liệu báo cáo tổng kết - chi tiết của nhóm thực hiện cho đề tài này.**

Qui định: Mô tả các bước thực hiện/ Phương pháp thực hiện/Nội dung tìm hiểu (Ảnh chụp màn hình, số liệu thống kê trong bảng biểu, có giải thích)

* 1. **Phương pháp thực hiện**

**1. Kiến trúc và thành phần hệ thống**

- Hệ thống được chia thành hai thành phần chính:

**+ Thành phần tấn công (Attack module):** Thực hiện kỹ thuật Process Hollowing bằng ngôn ngữ C/C++, sử dụng các hàm Windows API như CreateProcess, NtUnmapViewOfSection, WriteProcessMemory, VirtualAllocEx, SetThreadContext, ResumeThread, v.v.

**+ Thành phần phát hiện (Detection module):**

* Tĩnh (Static): Phân tích các PE file dựa trên chuỗi API nguy hiểm có trong file, được thực hiện bằng Python và thư viện pefile.
* Động (Dynamic): DLL được inject vào tiến trình mục tiêu để hook WriteProcessMemory, so sánh dữ liệu ghi với chữ ký mã độc đã biết.

**2. Quy trình thực hiện Process Hollowing**

**- Tạo tiến trình hợp pháp ở trạng thái treo (suspended)** bằng CreateProcessA(CREATE\_SUSPENDED).

**- Giải phóng vùng nhớ image gốc** bằng NtUnmapViewOfSection.

**- Cấp phát vùng nhớ mới** trong tiến trình đích bằng VirtualAllocEx.

**- Ghi header và các section** của file PE độc hại vào tiến trình đích bằng WriteProcessMemory.

**- Sửa context của thread chính** để trỏ tới entry point của mã độc (GetThreadContext, SetThreadContext).

**- Resume tiến trình**, mã độc được thực thi như một phần của tiến trình hợp pháp.

* 1. **Chi tiết cài đặt, hiện thực**

**- Ngôn ngữ sử dụng**: Python (detection tĩnh), C/C++ (hollowing & detection động)

**- Thư viện sử dụng:** pefile, MinHook, WinAPI

**- Môi trường cài đặt:**

+ OS: Windows 10

+ VS2019 (phát triển C++)

+ Python 3.x

+ RAM 8GB, Intel i5

**- Chuẩn bị dữ liệu:**

+ PE file độc hại được tạo trước bằng build mã C++ đã XOR payload

+ PE file hợp pháp: notepad.exe, calc.exe, ...

* 1. **Kết quả thực nghiệm**

**1. Tấn công process hollowing**

- Tiến trình notepad.exe được tạo ở trạng thái treo

- Mã độc được chèn vào notepad.exe mà không để lại dấu vết file trên ổ đĩa

- Mã độc thực thi thành công trong không gian tiến trình hợp pháp

**2. Phát hiện tĩnh**

- Công cụ monitor.py quét thư mục chứa file thực thi

- Tập các API nguy hiểm được xác định: CreateProcess, VirtualAllocEx, WriteProcessMemory, ...

- Hệ thống đánh dấu file đáng ngờ khi chứa đầy đủ chuỗi API

**3. Phát hiện động**

- DLL được inject vào tiến trình mục tiêu, hook WriteProcessMemory

- So sánh dữ liệu ghi với chữ ký shellcode lưu trong signature.txt

- Nếu trùng: log hành vi, hiện popup cảnh báo, hoặc kill tiến trình

**4. Demo**

- Mô phỏng thành công quá trình tấn công và phát hiện process hollowing

- Chứng minh có thể vượt qua Windows Defender bằng evasion techniques như:

**+ Dynamic API Resolution** (lấy hàm runtime bằng GetProcAddress)

**+ Memory-only injection** (không ghi ra ổ đĩa)

**+ XOR Payload** (mã hóa shellcode)

* 1. **Hướng phát triển**

**- Tăng khả năng phát hiện nâng cao:**

+ Kết hợp sandbox để phân tích hành vi tiến trình

+ Mở rộng chữ ký, nhận diện nhiều mẫu shellcode hơn

**- Cải thiện khả năng né tránh AV/EDR:**

+ Anti-hook, anti-debug, anti-VM

+ Sử dụng kỹ thuật DLL Proxy hoặc indirect syscalls

**- Triển khai thực tế:**

+ Kết hợp detection engine vào hệ thống giám sát tiến trình nội bộ doanh nghiệp

+ Cung cấp alert real-time đến admin nếu phát hiện hành vi đáng ngờ

Sinh viên báo cáo các nội dung mà nhóm đã thực hiện, có thể là 1 phần hoặc toàn bộ nội dung của bài báo. Nếu nội dung thực hiện có khác biệt với bài báo (như cấu hình, tập dữ liệu, kết quả,...), sinh viên cần chỉ rõ thêm khác biệt đó và nguyên nhân.

---

**Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này**

# **YÊU CẦU CHUNG**

* Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
* Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
* Sinh viên báo cáo kết quả thực hiện và nộp bài.

**Báo cáo:**

* File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
* Đặt tên theo định dạng: [Mã lớp]-Project\_Final\_NhomX\_Madetai. (trong đó X và Madetai là mã số thứ tự nhóm và Mã đề tài trong danh sách đăng ký nhóm đồ án).

Ví dụ: [NT521.N11.ANTT]-Project\_Final\_Nhom03\_CK01.

* Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
* Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

**Đánh giá**:

* Hoàn thành tốt yêu cầu được giao.
* Có nội dung mở rộng, ứng dụng.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

**HẾT**

1. Ghi nội dung tương ứng theo mô tả [↑](#footnote-ref-1)