Phát triển công cụ dịch ngược firmware trên thiết bị định tuyến

Trần Nghi Phú, Nguyễn Huy Trung
Ngô Quốc Dũng
Khoa Công nghệ và An ninh thông tin
Học viện An ninh nhân dân
Hà Nội, Việt Nam
Email: tnphvan@gmail.com

Nguyễn Ngọc Bình, Nguyễn Đại Thọ*
Trường Đại học Công nghệ, Đại học quốc gia
Hà Nội, Việt Nam
Email: binhnn, nguyendaitho@vnu.edu.vn
** UMI 209 UMMISCO IRD/UPMC*

***Tóm tắt nội dung*—Vạn vật kết nối (Internet of Things) hiện nay được sử dụng ngày càng rộng rãi trong cuộc sống, đi kèm với những tiện ích mới công nghệ này mang lại đó chính là vấn đề an ninh, an toàn thông tin. Kết quả các nghiên cứu gần đây đã chỉ ra rằng các lỗ hổng bảo mật và đặc biệt là mã độc xuất hiện rất nhiều trên hệ điều hành của các thiết bị mạng (firmware). Vì tính đặc thù cao của các thiết bị mạng cũng như của các firmware mà việc phân tích, rà quét các lỗ hổng bảo mật và mã độc gặp rất nhiều trở ngại. Hơn nữa, do chưa có được sự quan tâm đúng mức của các cá nhân, tổ chức lớn nên việc phát triển các công cụ cũng như phương pháp phân tích, phát hiện còn nhiều hạn chế. Hạn chế có thể kể đến là khả năng trích chọn firmware và dịch ngược chúng thành mã tường minh để từ đó có thể sử dụng các phương pháp phân tích lỗ hổng, mã độc khác nhau. Trong bài báo này, C500-Toolkit sẽ được giới thiệu với mục tiêu tăng hiệu xuất trích chọn và dịch ngược firmware của các thiết bị định tuyến. Gần 13.674 firmware từ 27 nhà phân phối được sử dụng để đánh giá và so sánh với các công cụ hiện có như fmk, binwalk và mô đun trích chọn,dịch ngược của firmadyne.**

***Keywords*-firmware; dịch ngược; mã độc; phân tích firmware; thiết bị định tuyến.**

I. GIỚI THIỆU

Sự gia tăng tích hợp, kết nối các thiết bị thông minh bằng công nghệ vạn vật kết nối (IoT - Internet of Things) đã mang lại các tiện ích phong phú cho người dùng như trong quản lý năng lượng thông minh, theo dõi sức khoẻ, thiết bị tự hành hay cơ sở hạ tầng thông minh... Để có được sự phát triển mạnh mẽ của IoT phải kể đến đóng góp lớn của cơ sở hạ tầng nhúng (embeded devices) cho phép tích hợp và biến mọi đồ vật trở nên "thông minh" khi có thể tương tác và kết nối với nhau cũng như với con người thông qua Internet.

Với ước tính 50 tỷ thiết bị IoT sẽ được kết nối vào mạng Internet đến năm 2020 [1] và có mặt ở mọi nơi mà đặc biệt là các thiết bị như CameraIP, VoIp, IpTV, thiết bị định tuyến... những thiết bị có thể sử dụng để giám sát, theo dõi người dùng mọi lúc, mọi nơi. Điều này đã làm dấy lên những lo ngại về việc lộ lọt thông tin của các tổ chức, cá nhân khi các thiết bị này bị tin tặc tấn công và chiếm quyền quản trị. Cho đến thời điểm hiện nay, những nghiên cứu, chính sách về bảo mật, an ninh an toàn thông tin chưa được quan tâm đúng mức so với sự phát triển nhanh và rộng của IoT. Một cách tổng quan, các thiết bị nhúng được thiết kế để thực hiện một công việc cụ thể và không có nhiều tương tác với người sử dụng. Đây cũng chính là một trong những lý do quan trọng khiến người sử dụng ít đặt các câu hỏi về an toàn, bảo mật hay mã độc đối với các thiết bị này. Theo IEEE Standard Glossary of Software Engineering Terminology, Std 610.12-1990 thì bộ xử lý trung tâm của các thiết bị IoT, hay còn gọi là firmware được định nghĩa như sau:

Định nghĩa 1. *Firmware là sự kết hợp của phần cứng và tập lệnh thực thi máy tính cùng với dữ liệu được lưu dưới dạng read-only trên máy đó.*

Trong khuôn khổ bài báo này, firmware được hiểu là một tập hợp các mã máy chạy trên bộ vi xử lý (chip) của phần cứng trên một thiết bị nhúng. Việc chỉnh sửa này giúp xác định rõ hơn trọng tâm nghiên cứu cũng như kết quả mong muốn đạt được. Một điểm cần lưu ý là firmware hoạt động ở mức thấp dưới tầng lõi hệ thống (Kernel) và tầng ứng dụng, do đó các phần mềm diệt virus như Kaspersky, Avast, Norton...không có khả năng rà quét và cách ly. Đây cũng chính là lý do mà phần mềm Lenovo Search Engine (LSE) được nhúng vào firmware của BIOS trước khi xuất xưởng và hoạt động mà không bị phát hiện. LSE cho phép

tin tặc (hacker) có thể khai thác để chiếm quyền điều khiển máy tính từ xa và người dùng dù có sử dụng các phần mềm diệt mã độc kể trên.

Một trong những điểm mấu chốt trong việc lộ lọt thông tin từ các thiết bị IoT đã được H.Grant [3] và đồng nghiệp nhấn mạnh trong nghiên cứu của mình nằm ở các thiết bị định tuyến và các thiết bị thu phát không dây và đây cũng chính là mục tiêu nghiên cứu của bài báo này. Đối với các thiết bị định tuyến, điểm cần lưu ý thứ hai đó là sự đa dạng hóa trong kiến trúc của các bộ vi xử lý (chip). Trái ngược với kiến trúc chip trên máy tính với đa số là i386 thì kiến trúc chíp sử dụng phổ biến trên các thiết bị mạng nói riêng và thiết bị nhúng nói chung là ARM (ARM7, ARM9, Cortex), Intel ATOM, MIPS, Motorola, Axis CRIS.

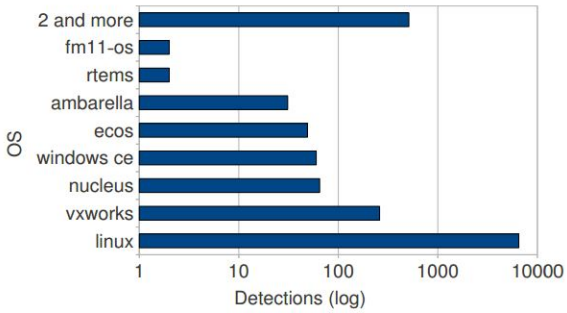
Khi tập trung vào phân tích mã độc các thiết bị mạng, đặc biệt là các thiết bị định tuyến, chúng ta có thể chia các phương pháp nghiên cứu làm hai nhóm chính đó là phân tích động và phân tích tĩnh. Phân tích tĩnh là phương pháp phân tích, rà quét mã độc trực tiếp trên mã nguồn mà không cần thực thi chúng. Một số nghiên cứu dựa trên phương pháp này đã tiến hành các phân tích diện rộng hoặc theo hướng chuyên biệt để tìm cổng hậu (backdoor), lỗ hổng bảo mật hay mã độc nhúng trên các thiết bị định tuyến như Angr [5] và FIE [6]. Phân tích tĩnh cho phép chi tiết hóa toàn bộ luồng điều khiển (Control-flow graph) và luồng dữ liệu (data-flow graph) cho từng tập tin hệ thống trong firmware để từ đó phát hiện mã độc bằng kỹ thuật phân tích đặc trưng (symbolic execution)[7].

Phân tích động là phương pháp phân tích hành vi các tập tin hệ thống của firmware trong quá trình thực thi. Toàn bộ hành vi của firmware sẽ được ghi lại và phân tích theo thời gian thực, nhất là đối với các hành vi được coi là bất thường. Dựa trên phương pháp này, Daming D. Chen [8] đã tiến hành phân tích 23.035 firmware từ 42 nhà cung cấp và phát hiện ra 887 firmware có lỗ hổng bảo mật, trong đó có 14 lỗ hổng bảo mật chưa được biết đến trước đây. Tại Việt Nam, tháng 6/2016, tập đoàn BKAV đã công bố kết quả khảo sát 21 triệu thiết bị trên Internet, trong đó có tới 5.6 triệu thiết bị trên thế giới bị nhiễm lỗ hổng PetHole [9], dẫn đến nguy cơ mất quyền điều khiển thiết bị.

Phân tích tĩnh hay phân tích động đều thực hiện dựa trên mã nguồn tường minh của các tập tin hệ thống. Tuy nhiên việc dịch ngược từ mã nhị phân một firmware thành mã nguồn tường minh không phải lúc nào cũng thực hiện được. Điều này ảnh hưởng do độ phức tạp của định dạng nén, kiến trúc chip và kỹ năng làm rối (Obfuscated code) của nhà sản xuất. Cho đến nay, tỉ lệ dịch ngược thành công của các công cụ là chưa cao và đây cũng chính là mục tiêu mà công cụ C500-Toolkit hướng tới. Trong bài báo này,13.674 firmware từ 27 nhà phân phối được sử dụng để đánh giá và so sánh hiệu năng dịch ngược với các công cụ hiện có như fmk [10], binwalk [11] và mô đun chuyên biệt của Firmadyne [8].

II. NHỮNG NGHIÊN CỨU LIÊN QUAN

Các nghiên cứu, khảo sát gần đây thống kê có gần 90% firmware thiết bị định tuyến hiện nay sử dụng nhân Linux 2.6 [12] [13] với nhiều loại tập tin hệ thống khác nhau như: Cramfs, Squashfs, Yaffs, Jffs2,RomFs....Mỗi kiểu tập tin hệ thống lại sử dụng các cách thức nén khác nhau như Squashfs dùng LZMA để tối ưu nén dữ liệu cùng tốc độ hay Cramfs dùng Zlib để nén... Các nền nảng hệ điều hành sử dụng phổ biến trong firmware được trình bày trong hình 1.



Hình 1: Nền tảng OS sử dụng phổ biến trong firmware [14]

Để giải nén và dịch ngược từ tập tin nhị phân thành code tường minh dùng trong việc phân tích, nhiều công cụ chuyên dụng đã được liên tục phát triển cập nhật như công cụ phân tích Binwalk [11] để xác định kiến trúc, thông tin tổng quan của một tập tin nhị phân. Trên cơ sở Binwalk, các công cụ dịch ngược cho các firmware được ra đời như Firmware Mod Kit [10] cho các nền tảng Linux, mô đun dịch ngược của Firmadyne [8]. Tuy nhiên, những công cụ này còn khá nhiều hạn chế, tồn tại.

- Firmware-mod-kit (fmk) [10] là một tập các câu lệnh dưới dạng kịch bản (script) và công cụ tích hợp nhằm mục đích dịch ngược và đóng gói các firmware có nền tảng là linux. Fmk chứa một tập các phiên bản khác nhau của bộ công cụ unsquashfs, sau khi dùng fmk để xác định điểm offset cũng như phiên bản nén của phần nhân chứa tập tin hệ thống, fmk sẽ thử lần lượt từng phiên bản khác nhau có trong bộ công cụ unsquashfs của mình để dịch ngược. Phương pháp này có một số tồn tại, đáng chú ý nhất là việc dịch ngược bằng một bộ công cụ unsquashfs nhất định chưa chắc chắn là phiên bản chuẩn dùng khi biên dịch squashfs. Việc này dễ xảy ra nếu nhà sản xuất thực hiện việc chỉnh sửa nhỏ trong phần header của tập tin.

- Binwalk [11] là một công cụ dùng để phân tích, dịch ngược và giải nén giữ liệu chứa trong ảnh của các firmware. Binwalk được sử dụng phổ biến nhất hiện nay và dễ dàng tương thích với các công cụ phân tích mã độc khác vì được viết chủ yếu bằng Python. Dựa trên binwalk, Costin và đồng nghiệp [15] đã phân tích trên tập lớn firmware với phương pháp phân tích tĩnh. Theo kết quả thử nghiệm, nhóm tác giả đã chỉ ra rằng Binwalk không thể nhận dạng được một số file nhị phân firmware dẫn đến tỷ lệ dương tính giả khá cao bởi không thể tìm thấy dấu hiệu (signal) hoặc dấu hiệu đã bị thay đổi khi sử dụng đối sánh trong tập tin magic [16]. Đặc biệt là những dòng IOS của Cisco khi dữ liệu nén kiểu gzip. Nói một cách khác, hạn chế của Binwalk không thể xác định được các loại firmware Integrated hay Partial updates.

- Mô đun dịch ngược của Firmadyne, Daming D.Chen [8] cùng đồng sự đã làm mịn các chức năng có trong binwalk. Tuy nhiên khả năng dịch ngược của Firmadyne cũng còn hạn chế khi tỉ lệ thành công chỉ chiếm gần 33%.

III. CẤU TRÚC FIRMWARE

Cấu trúc của firmware rất đa dạng, phụ thuộc vào chức năng và thiết kế của từng nhà sản xuất. Các firmware có thể được chia thành các kiểu như sau:

- Full-blown (full-OS/kernel + bootloader + libs + apps): đây thường là một Linux hoặc Windows firmware vì chúng chứa một hệ điều hành hoàn chỉnh nhưng tối giản. Các ứng dụng có thể chạy trong chế độ người dùng, kernel modules, drivers.
- Integrated (apps + OS-as-a-lib): đầy là một bản firmware không đầy đủ, các chức năng và hệ điều hành được xây dựng như một thư viện chứ không có đầy đủ các thành phần cần thiết như trong bản Full-blown.
- Partial updates (apps or libs or resources or support): Loại firmware này chỉ chứa các tập tin dùng trong việc cập nhật cho bản firmware cần nâng cấp.

Trong bài báo này, 2 loại firmware được tập trung phân tích đó là Full-blown và Intergrated firmware bởi vì khi chúng có chứa đầy đủ các tập tin hệ thống cần thiết cho việc phân tích động và tĩnh về sau.

Hệ điều hành hoàn chỉnh nhưng tối giản trong các firmware loại này thông thường chứa các tập tin hệ thống như sau [15]:

- Bootloader: là một đoạn mã được thực thi trước khi hệ điều hành bắt đầu chạy và nó cho phép nhà sản xuất thiết bị quyết định những tính năng nào người sử dụng được phép dùng hoặc bị hạn chế. Các loại Bootloader cho hệ thống nhúng:
  - U-boot (universal Boot loader) : bộ nạp khởi động cho PowerPC, ARM trên hệ thống nhúng Linux
  - RedBoot (RedHat eCos Derived): bộ nạp dành cho các máy di động sử dụng hệ thống nhúng
  - Rrload: bộ nạp khởi động cho ARM trên hệ thống nhúng Linux
  - FILO (Firs In Last Out) : bộ nạp khởi động tương thích x86
  - CRL/OHH : bộ nạp khởi động cho ARM dựa trên hệ thống nhúng Linux
  - PPCBoot: bộ nạp khởi động cho PowerPC dựa trên hệ thống nhúng Linux
  - Alios : hợp ngữ dựa trên bộ nạp Linux để khởi tạo các thành phần cứng cơ bản từ ROM hoặc RAM. Mục đích để loại bỏ thành phần Bios ra khỏi hệ thống nhúng.
- Kernel: được hiểu là hạt nhân là thành phần trung tâm của thiết bị định tuyến. Trách nhiệm của nó là giúp cho phần cứng và phần mềm của thiết bị có thể giao tiếp được với nhau.

Kernel là thành phần quan trọng nhất trong thiết bị, nếu kernel bị hỏng thì thiết bị đính tuyến đó sẽ dừng hoạt động và nó được lưu trữ trong bộ nhớ Flash của thiết bị.

- File–system images : chứa các tập tin hệ thống có chức năng tổ chức và kiểm soát quá trình hoạt động của thiết bị nhúng, các tính năng nổi bật của File system trong thiết bị nhúng đó là:
  - Tính năng Wear leveling nhằm phân phối đều số lần ghi/xóa dữ liệu trên mỗi block nhằm kéo dài tuổi thọ của bộ nhớ.
  - Hạn chế sự mất mát dữ liệu khi gặp sự cố về nguồn điện. Trong bài báo này sẽ trình bày thử nghiệm trên 5 loại file system được sử dụng phổ biến hiện nay đối với Flash là: Squashfs, Cramfs, YAFFS, JFFS2.
- Resources and support files: bao gồm các tài nguyên và các file hỗ trợ trong quá trình hoạt động của firmware, đối với mỗi loại firmware thi tài nguyên và các file hỗ trợ cũng sẽ khác nhau.
- Web–server/web–interface: đây chính là thành phần quan trong giúp cho người dùng có thể tương tác được với thiết bị một cách dễ dàng. Tất cả các thông tin và cấu hình thiết bị phần lớn được thao tác thông qua Web – server và Web – interface.

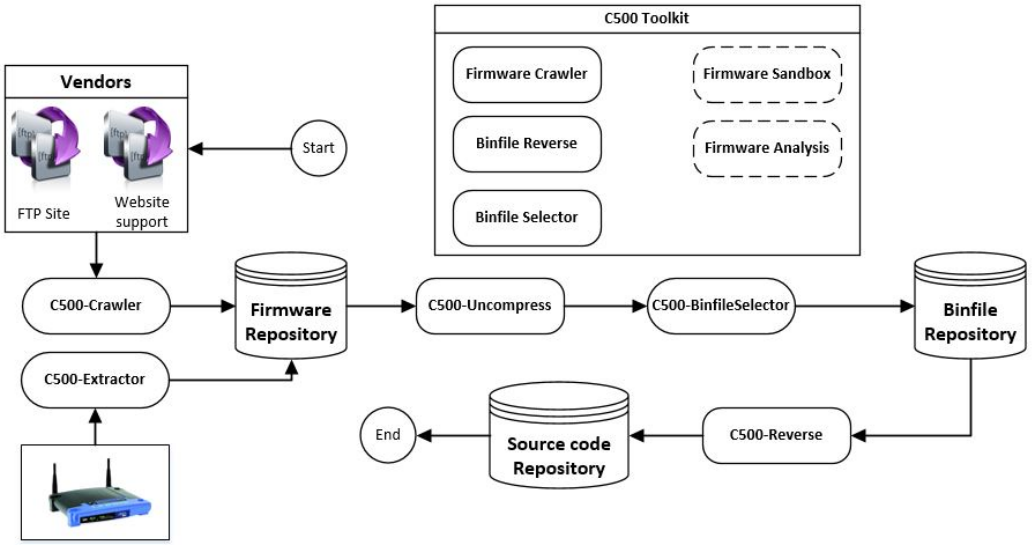
IV. GIẢI PHÁP ĐỀ XUẤT

Nhằm nâng cao khả năng dịch ngược, phân tích các firmware thành mã nguồn tường minh để thực hiện các kỹ thuật phân tích tĩnh hay động, bài báo này đề xuất công cụ C500-Toolkit. Những chức năng cơ bản của C500-Toolkit cũng như chu trình dịch ngược được giới thiệu trong hình 2. Các chức năng của C500-Toolkit được mô đun hoá tối đa giúp việc nâng cấp, bổ sung tính năng mới một cách dễ dàng.

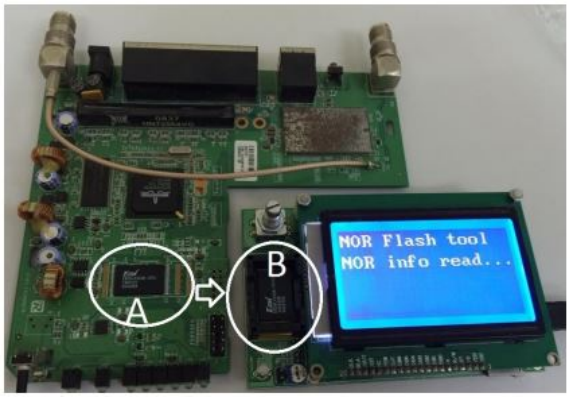
Chu trình dịch ngược sử dụng bộ công cụ C500-Toolkit gồm các bước sau:

- Bước 1: Thu thập firmware từ website các nhà phân phối thiết bị định tuyến. Phần này được thực hiện bởi mô đun C500-Crawler, 13.674 firmware từ 27 nhà phân phối đã được thu thập và lưu vào kho Firmware Repository. C500-Crawler sử dụng hai nguồn chính trong việc thu thập các firmware gồm:
  - các website nhà phân phối như: D-link, Belkin, Xerox, Tp-link, Asus...
  - các trang tin FTP chuyên dụng trong việc tìm kiếm và thu thập địa chỉ URL chứa firmware dựa vào từ khoá các nhà phân phối để tải về.

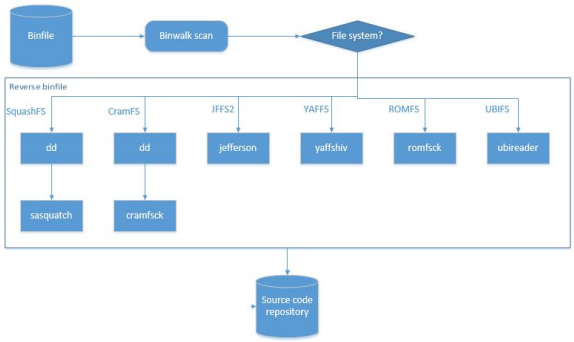
    Bên cạnh đó, bài báo còn có thể thu thập firmware trực tiếp từ thiết bị thông qua thiết bị ExC500 [18] mà hiện nay chưa nghiên cứu nào đề cập. Thiết bị này sao chép trực tiếp firmware từ bộ nhớ flash của thiết bị định tuyến như trong hình 3. Ở đây, chíp nhớ flash lưu firmware trên mạch của thiết bị định tuyến (vị trí A) sẽ được tách khỏi mạch và đưa vào khay (vị trí B) để tiến hành sao chép. Ngoài thiết bị Exc500, có thể trích xuất trực tiếp firmware từ flash, NVRAM, thông tin lưu trên RAM thông qua chức năng sao lưu firmware hoặc thông qua cổng JTAG bằng mạch FT2232H của FTDI.
- Bước 2: Thực hiện giải nén các bản ảnh firmware đã thu thập được thông qua mô đun C500-uncompress. Các firmware khi tải về thường được nén bằng các chuẩn như khác nhau. Modun C500-uncompress có khả năng giải nén các chuẩn zip, rar, tar, các định dạng khác sẽ được mô đun C500-uncompress duyệt đệ quy và đưa vào một thư mục vì có khả năng các tập tin này không phải định dạng nén.
- Bước 3: Trích chọn mã nguồn nhị phân bằng mô đun C500-Binfile Selector. Mỗi bản firmware đầy đủ sau khi giải nén thường chứa nhiều tập tin phụ như hướng dẫn sử dụng, lưu ý, v.v chứ không chỉ riêng mã nguồn cần tìm. Bên cạnh đó, các nhà phân phối cũng không sử dụng một định dạng đồng nhất mà một bản mã nhị phân firmware có thể được lưu dưới các định dạng .img, .bin, .iso, .tfp, .lod... hoặc thậm chí là không định dạng. Chính vì vậy, nếu chỉ dựa vào phần mở rộng của tập tin (extension) thì độ chính xác không cao vì ở mức thấp thì cách xác định định dạng như vậy không chính xác cao. Tuy nhiên, các tập tin mã nguồn nhị phân thường thuộc một kiểu tập tin hệ thống (filesystem) chính vì vậy để giải quyết vấn đề này, mô đun C500-Binfile Selector sử dụng Binwalk để duyệt đệ quy tất cả tập tin thu được ở Bước 2 có thuộc kiểu tập tin hệ thống nào không từ đó mô đun C500-Binfile Selector kiểm tra phần meta-data trong header



Hình 2: Kiến trúc khối của công cụ C500-Toolkits



Hình 3: Thiết bị ExC500 để sao chép firmware từ flash của thiết bị định tuyến



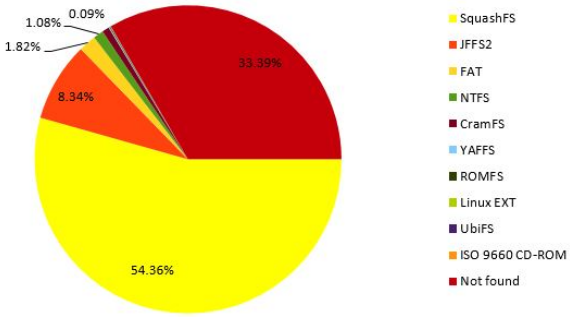
Hình 4: Thành phần dịch ngược trong mô đun C500-reverse

của các tập tin này bằng chức năng $file$ có trong Linux. Nếu tập tin nào trả về kiểu header là "application/octe-stream" thì có khả năng đó là tập tin nhị phân firmware cần tìm.

- Bước 4. Sử dụng mô đun C500-reverse để tiến hành dịch ngược những mã nguồn nhị phân đã trích chọn thành mã tường minh, chứa đầy đủ các tập tin cấu hình của thiết bị. Việc dịch ngược thành công các tập tin mã nguồn nhị phân phụ thuộc nhiều vào việc bộ dịch ngược có hiểu được các tập tin không, tức là xử lý được các kiểu tập tin hệ thống. Nhiều nghiên cứu hiện nay như Firmadyne hay FMK đều không đạt được tỷ lệ dịch ngược tốt vì chỉ tập trung vào 2 kiểu tập tin hệ thống thường dùng là SquashFS và CramFS. Chính vì thế, mô đun C500-reverse được tập hợp các công cụ cho phép dịch ngược các kiểu tập tin hệ thống phổ biến như: SquashFS, JFFS2, CramFS, YAFFS, ROMFS. Một bộ luật được xây dựng để xác định công cụ dịch ngược thích hợp dựa trên binwalk. Các thành phần để thực hiện bộ luật dịch ngược trong mô đun C500-reverse Hình 4.

V. KẾT QUẢ THỬ NGHIỆM

Trong bài báo này, bộ công cụ C500-Toolkit thử nghiệm với 13.674 bản ảnh firmware từ 27 nhà phân phối thiết bị định tuyến. C500-Toolkits được cài đặt trên một hệ thống gồm 03 máy tính cấu



Hình 5: Các kiểu tập tin hệ thống thường dùng trong thiết bị định tuyến

hình Intel Xeon CPU E5-2620 12 Cores 2.4GHz, 16GB RAM. Thời gian thực hiện đầy đủ chu trình thu thập, giải nén và dịch ngược cho 13.674 firmware là sấp xỉ 90 ngày nếu chạy trên 01 máy. Do đó, chương trình được lập trình hướng song song nhằm phân tách 13.674 firmware này trên 03 máy, nhờ đó mà thời gian thực hiện được giảm xuống còn 40 ngày.

Dựa trên phân tích 13.674 firmware, nhóm tác giả đã thống kê được các định dạng tập tin hệ thống thông dụng trong thiết bị định tuyến như Hình 5.

Qua Hình 5, kiểu tập tin hệ thống được dùng nhiều nhất trong thiết bị định tuyến là SquashFS (54,36%), tiếp đó là CramFS (33,39%). Bên cạnh đó, một số nhà sản xuất thiết bị định tuyến hiện nay đã sử dụng kiểu tập tin hệ thống JFFS2 (8,34%). Chính vì vậy, khi tiến hành thử nghiệm bằng công cụ FMK (Firmware-mod-kit) trên 13.674 firmware, tỷ lệ dương tính giả (false positive) ở mức 18%. Điều này là do kiểu tập tin hệ thống JFFS2, ROMFS,... chưa được FMK hỗ trợ. Đối với Firmadyne, mặc dù sử dụng API của Binwalk trong việc dịch ngược mã nguồn nhị phân firmware nhưng chỉ hỗ trợ một số phương thức nén chuẩn như gzip, tar, lzma,... vì thế trường hợp nhà sản xuất thiết bị định tuyến tự xây dựng các phương pháp nén riêng cho firmware thì Binwalk sẽ không xử lý mà bỏ qua.

Trong 13.674 bản ảnh firmware có chứa một số lượng không nhỏ các bản cập nhật (partial update firmare) nên mô đun C500-binfile selector tự loại bỏ và không thực hiện việc trích bọn mã nguồn nhị phân. 10.034 tập tin mã nguồn nhị phân đã được trích chọn và trong đó C500-reverse đã dịch ngược thành công 6.623 tập tin nhị phân thành mã nguồn tường minh. Chi tiết tỉ lệ dịch ngược thành công trên tổng số firmware tải về được trình bày trong bảng I.

Bảng I: Kết quả dịch ngược các firmware của từng hãng đã thu thập

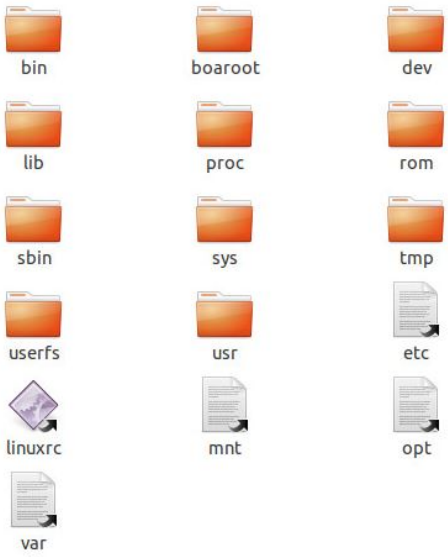
Index	Vendor	Download	Extracted
1	Belkin	138	42
2	Buffalo	134	8
3	Centurylink	10	7
4	Dlink	543	117
5	Foscam	62	11
6	Linksys	49	29
7	Microtrain	1	0
8	Mikrotick	27	20
9	Netgear	125	1
10	Openwireless	2	1
11	Pfense	2	0
12	Polycom	951	58
13	Qnap	177	2
14	Seiki	19	0
15	Supermicro	91	6
16	Tenda	7	1
17	Tenvis	22	12
18	Ti	10	0
19	Tp-link	740	360
20	Ubiquiti	113	57
21	Verizon	10	0
22	Xerox	2	0
23	Zyxel	269	51
24	Openwrt	6531	5228
25	Synology	2309	2
26	Trendnet	288	101
27	Asus	1042	509
	Total	13.674	6.623

Với việc phân tách các chức năng của C500-Toolkit thành các mô đun riêng lẻ và độc lập nên việc cập nhập, chỉnh sửa hay thêm, bớt các chức năng sẽ trở nên dễ dàng. Kết quả dịch ngược của C500-Toolki cũng đạt kết quả tốt hơn Fmk, Firmadyne hay Binwalk. Điều này được biểu diễn thông qua bảng II.

Bảng II: Kết quả thử nghiệm dịch ngược 13.674 bản ảnh firmware của 27 hãng phân phối thiết bị định tuyến

Firmware Mod Kit	18 %
Firmadyne	37 %
C500-Toolkits	48 %

Sau khi dịch ngược thành công, hệ thống có các tập tin như hình 6. Tập tin tường minh này tiếp sau sẽ được sử dụng trong việc phân tích mã độc hoặc lỗ hổng bảo mật tuỳ theo mục tiêu yêu cầu.



Hình 6: Các tập tin hệ thống sau khi dịch ngược

VI. KẾT LUẬN

Trong bài báo này, chúng tôi đã giới thiệu công cụ C500-Toolkit cho phép thực hiện 2 chức năng chính bao gồm trích chọn file nhị phân firmware và dịch ngược firmware. Từ đó tạo tiền đề cho việc áp dụng phân tích tĩnh, phân tích động vào phát hiện mã độc, lỗ hổng bảo mật trên fimware. Bài báo mới dừng lại ở các firmware dựa trên Linux bởi đây là nền tảng sử dụng phổ biến nhất hiện nay. Tuy nhiên, hướng nghiên cứu tiếp theo của nhóm tác giả là mở rộng tiếp việc dịch ngược đối với các nền tảng khác như VxWorks, QNX,.... Bên cạnh đó, các mô đun chuyên dụng cho phân tích tĩnh và phân tích động dựa trên tập tin tường minh cũng sẽ được nghiên cứu và phát triển trong thời gian tới.

Kết quả được thử nghiệm trên 13.647 firmware từ 27 nhà phân phối đưa ra được những kết quả ban đầu khả quan. Thời gian tới, nhóm nghiên cứu sẽ tiếp tục thu thập thêm firmware mới và mở rộng phân tích fimware trên thiết bị nhúng nói chung chứ không chỉ tập trung vào các thiết bị định tuyến.

CẢM TẠ

Nhóm tác giả xin chân thành cảm ơn sự góp ý chuyên môn của PGS.TS Nguyễn Linh Giang - Đại học Bách khoa Hà Nội.

TÀI LIỆU

[1] Sebastian Muniz, *Killing the myth of Cisco IOS rootkits*, DIK, 2008. In EUSecWest

[2] IEEE Standard Glossary of Software Engineering Terminology. IEEE Std 610.12-1990, pages 1–84,1990.

[3] H.Grant, O.Arias, D.Buentello, and Y.Jin, *Smart nest thermostat: A smart spy in your home*, Black Hat USA, 2014.

[4] C. Heffner, *Reverse Engineering a D-Link Backdoor*, October 2013

[5] C.Kruegel and Y.Shoshitaishvili, *Using Static Binary Analysis To Find Vulnerabilities And Backdoors In Firmware*, Black Hat USA, 2015.

[6] Yan Shoshitaishvili, Ruoyu Wang, Christophe Hauser, Christopher Kruegel, Giovanni Vigna, *Firmalice - Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware*, NDSS Symposium, 2015

[7] D.Davidson, B.Moench, S.Jha,and T.Ristenpart, FIE on Firmware, *Finding vulnerabilities in embedded systems using symbolic execution* in Proceedings of the 22nd USENIX Security Symposium. USENIX, 2013, pp. 463–478. [Online]. Available: https://www.usenix.org/conference/ usenixsecurity13/technicalsessions/paper/davidson

[8] Daming D.Chen*, Manuel Egele, Maverick Woo and David Brumley, *Towards Automated Dynamic Analysis for Linux-based Embedded Firmware*, Carnegie Mellon University, 2015.

[9] *http://pethole.net/*

[10] Firmware Mod Kit. Available: *https://github.com/mirror/firmware-mod-kit*.

[11] Binwalk [Online]. Available *http://binwalk.org*

[12] Nikolai Hampton, Patryk Szewczyk, *A survey and method for analysing SoHo router firmware currency*, Australian Information Security Management, 2015

[13] Shodan [Online]. Available: *http://shodan.io*

[14] A.Costin, J.Zaddach, A.Francillon and D. Balazarotti, *A large-scale analysis of the security of embedded firmwares*, in Proceedings of the 23rd USENIX Security Symposium, 2014, pp.95-110 [Online].Available: https://www.usenix.org/conference/usenixsecurity14/ techincal-sessions/presentation/costin.

[15] Jonas Zaddach, Andrei Costin, *Embedded Devices Security and Firmware Reverse Engineering*, EURECOM,Sophia-Antipolis, Biot, France, 2013.

[16] *https://pypi.python.org/pypi/filemagic/*

[17] Hui Suo, Jiafu Wan, Caifeng Zou, Jianqi Liu, *Security in the Internet of Things: A Review LaTeX*, Guangzhou, China, 2012.

[18] Trần Nghi Phú, Ngô Quốc Dũng, Nguyễn Huy Trung, Nguyễn Ngọc Bình, *Mô hình phát hiện mã độc trong phần mềm nhúng trên thiết bị định tuyến*, Hội thảo quốc gia lần thứ XIX: Một số vấn đề chọn lọc của CNTT&TT - Hà Nội, 2016.

[19] FTP indexing engines firmware Crawler. Available *http://www.mmnt.ru/*
http://filemare.com/
http://www.filesearching.com/

[20] Xuan Nam Nguyen, Dai Tho Nguyen, Long Hai Vu, *POCAD: a Novel Payload-based One-Class Classifier for Anomaly Detection, 2016 3rd National Foundation for Science and Technology Development (NAFOSTED) Conference on Information and Computer Science (NICS)*, Da Nang, 2016