Towards Automated Dynamic Analysis for Linux-based Embedded Firmware (https://www.ndss-symposium.org/wp-content/uploads/2017/09/towards-automated-dynamic-analysis-linux-based-embedded-firmware.pdf) 摘要

摘要

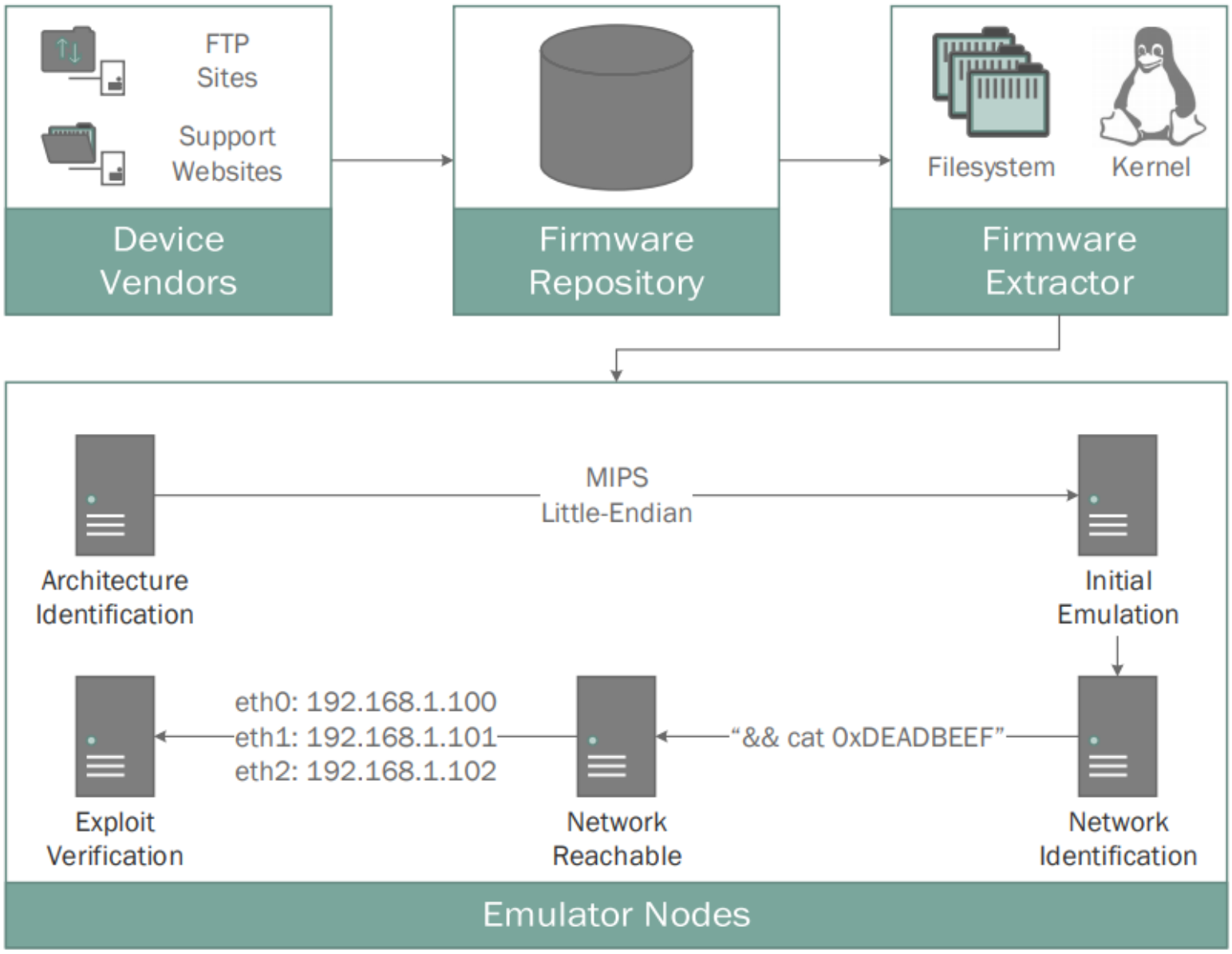
- 背景：日常生活中嵌入式設備在網路方面佔有重要的地位，因為它們可能是連線的唯一管道；然而這些設備的安全性可能有問題。
- 主題：在這篇論文中，使用**FIRMADYNE**來模擬並分析這些設備，說明並解決遇到的問題。
- 成果：模擬42家供應商的23035個映像檔，在其中9486個檔案上追蹤到指定的74個漏洞、887個有漏洞的映像檔影響89個產品、找到14個新的漏洞。

介紹

- IoT設備越來越多，但是卻很少、甚至沒有更新有關漏洞的東西。
- 這些低安全性的設備卻可能影響到整個網路，為了提高電腦設備的安全性，該論文決定挑戰準確判別在嵌入式韌體中的漏洞。
- 以往的分析方法：
  - 使用實體設備進行動態分析，硬體成本太高、且會有接口不符的問題。
  - 靜態分析可能會有結果誤報或是複合多個程式語言難以分析的問題。
- 為了克服以上問題而使用的方法是用軟體模擬整個系統進行動態分析：
  - 好處：不須真的擁有硬體、不受到開發語言的限制、分析出可能的漏洞之後會提供成功利用該漏洞的行為。
  - 挑戰：在一些特別的硬體設備(NVRAM)可能會有問題需要解決、動態生成文件的處理。
- 開發了**FIRMADYNE**，修改了Linux kernel使其對Linux-based的設備有更好的支援能力；使用**QEMU** (https://zh.wikipedia.org/wiki/QEMU)模擬硬體。
- FIRMADYNE包含爬蟲可以自動尋找供應商網頁的映像檔並輸入至系統中。
- 完全模擬系統必須要正確地設定與目標互動的網路介面(interfaces)，為此系統會先將目標模擬在一個獨立的環境中，監控所有互動後分析出正確的設定，再重新配置整個環境。
- 發現最有效的攻擊影響五個不同供應商的韌體，多達資料集裡面10%的映像檔；其中受影響的除了公開的程式碼也包含未公開的，代表code share在上游製造商中很普遍。
- https://github.com/firmadyne/firmadyne (https://github.com/firmadyne/firmadyne)

總覽

元件



- FIRMADYNE分為四個元件，也分別是四個步驟。
  - 爬蟲韌體：
    - 最大的獨立元件。
    - 可以爬支援的硬體供應商網頁。
    - 分辨韌體映像檔和其他二進制檔。
    - 提供建立日期、發行版本或是MIB (https://en.wikipedia.org/wiki/Management_information_base) (Management information base)的SNMP (https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol)(Simple Network Management Protocol)等等對於動態分析或漏洞開發有用的資訊。
    - 對於不好爬的動態網頁，犧牲掉metadata的資料而改爬供應商的FTP站。
  - 提取韌體filesystem：
    - FIRMADYNE使用**binwalk**的自訂提取功能，撈出映像檔中的kernel和root filesystem。
  - 初始化模擬：
    - 當成功提取filesystem之後，FIRMADYNE會確認硬體架構(上圖為例就是*MIPS Little-Endian*)。然後會在QEMU中預建一個與目標架構相同的Linux kernel。
    - 支援little-endian ARM, little-endian MIPS, and big-endian MIPS。
    - 初始化推測出系統和網路的設定(像上圖的eth0, 1, 2)。
  - 動態分析：
    - 當上面的步驟成功設定好環境之後，FIRMADYNE就可以進行動態分析。
    - 高擴展性以應對新的漏洞。
    - 分別彙整出資料集中每個檢查結果(上圖是測試到command injection的例子)。
    - 三個通道來檢測漏洞，並從kernel監控漏洞是否觸發成功。

動機

- 說明為何要使用完整系統模擬來分析，其中的優點、缺點和挑戰。

Application-Level

- 最直接的方法是直接用支援的應用程式執行，例如許多設備含有Web網頁內容，可以複製並使用Apache執行。
- 缺點：不符合通用性的目標，許多function是自定義加入到web server中，上游的開源程式並不支援。
- 最後方法：只檢測應用層中的特定數據漏洞。

Process-Level

- 另一種分析方法是使用原始的系統文件檔模擬每個process的行為。
  - 可使用QEMU完成，直接啟用原始的Web server來模擬出Web介面。
- 儘管該方法比前一個方法更加準確，但依然存在許多缺點：
  - 調取特定硬體設備時可能出現錯誤(NVRAM)
  - 記憶體實體位址不存在時可能出現錯誤
  - 模擬時可能產生內存的備份檔，因為這些檔案是動態產生，當symbolic links到這些檔案時，靜態分析會以為是毀損。

System-Level

- 系統層級的模擬可以符合硬體設備的介面接口，與真實設備相同的方式動態生成數據，並分析系統啟動時的所有process(包含HTTP、FTP和Telnet等)，因此FIRMADYNE就是以完整系統模擬為基礎設計。
- 將原始的kernel替換為修改過專門為模擬用的kernel，否則原來的kernel會因為被編譯為只支援特定硬體的平台而崩潰。
  - 該方法成功執行他們數據庫中96.6%的初始模擬。
- 然而這種方法並不支援核心模組(kernel modules)，但他們認為不需要支援也沒關係：
  - 他們的資料集裡超過99%的out-of-tree kernel modules在系統中沒有使用。
  - 較新的kernel可以在in-tree找到等同out-of-tree實現的功能。

概念

- 簡述動態分析Linux-based映像檔架構和背後的概念。

架構

- 系統包含一個韌體數據庫伺服器(repository)和一個資料庫(dataset)，前者會儲存並對應每個韌體的二進制檔，後者則是會追蹤每個韌體的相關訊息，如提取狀態、結構、商標或是相關文件等。
- 整個虛擬化作業流程為：
  - 提取系統文件和kernel。
  - 如提取成功，數據庫會記錄暫存。
  - 初始化設定、紀錄網路互動行為。
  - 模擬並分析。

資料蒐集

- 為了蒐集韌體映像檔數據集，開發了Web爬蟲。
- 針對42家供應商的頁面去解析，能區分更新的映像檔或是其他檔案(如驅動程式等)。
- 回復映像檔的Metadata，例如供應商、產品名稱、發布日、版本或是更新日誌等。
- 某些不好自動化的網頁，採用FTP或是手動收集。

提取

- 使用binwalk的API開發提取程式，從映像檔恢復系統檔和kernel，並壓縮成tar儲存於數據庫中。

模擬

- 在提取出系統檔之後，FIRMADYNE會分析映像檔所需要的系統配置。
- 藉由系統檔裡二進制文件的標頭得到目標架構和位元組順序(endianness)。
  - 使用QEMU對應架構。
  - ARM little-endian、MIPS littleendian和MIPS big-endian已經預先編譯好kernel，因為這三種佔了數據集中的90.8%。
- 接下來會對網路系統互動並記錄，為網路接口分配IP。
- 最後就進入實際的模擬階段，使用上述的網路配置與模擬的韌體互動。

自動化分析

- 在動態分析的架構中實作了三個自動分析，檢測到14個以前未知的漏洞。

實作

- 本節討論前面提到的各元件的實作。

資料蒐集

- 爬蟲使用Scrapy框架開發，數據集中42家供應商都是不同的爬蟲。
- 產品包含IP攝影機、路由器、智慧電視等等。
- 使用XPath selectors分別去解需每個供應商的頁面。
- 爬取不同地理位置的網頁(不同國家)。
- 供應商網頁如果使用大量的動態產生的內容，就去爬FTP站。
- 只下載副檔名較相關的文件。
- 若下載有所限制，就手動去下載。
- 刪除重複的映像檔，若有多個產品共享同一個映像檔，則可透過Metadata去找到對應檔案。

提取

- binwalk內建的提取器無法使用，因此只好自行利用API開發一個。
- 目標是從映像檔中拿出filesystem和kernel，並且最小化時間與空間。
  - 建立黑名單，排除掉PE32、ELF、PDF或Microsoft Office等等的非目標檔案。
  - 檢查其餘檔案的存檔格式、標頭檔案、kernel版本字串，排序出處理優先度。
  - 根據檔案系統階層標準，檢查是否存在4個根目錄來決定是否已經完成提取。
- 對於JFFS2和SquashFS這兩種filesystem，使用jefferson和sasquatch這兩個第三方工具完成。
  - 由於這些提取工具太久沒有更新，反倒filesystem的壓縮算法一直改良，導致時常提取失敗；為此團隊提交不少版本改良，大多都已經合併到最新版本中。
- 儘管以上的開發與改良提高了提取的成功率，還是有很多情況會提取失敗：
  - 供應商只提供一個產品的部分映像檔。
  - 一個映像檔中有多個filesystem。
  - 映像檔有加密。
  - 非Linux或是根本無法辨識的系統。
-

Type	# Images
Linux	9,379
Unidentified (UNIX-like)	2,187
VxWorks	857
Unknown	10,612
Total	23,035

  - 按系統分類的映像檔數目。

模擬

NVRAM

- 提取出來的映像檔中，至少有52.6%使用libnvram.so這個library存取NVRAM並儲存相關配置。
- 自行開發一個library可以攔截對NVRAM的所有互動，並重新實現對外部設備的API，而不用去模擬整個外部設備。
- 遇到的第一個挑戰是，為了將自行開發用於監控的lib加入到環境中，可能導致編譯時出錯，因為不同的映像檔編譯的toolchain(編譯器、連結器、函式庫)不同，有些無法修改。
  - 解決辦法是使用lazy link，使function使用到的時候才去link，然後使用-nostdlib去加載自己的lib。
- 另一個挑戰是NVRAM需要預設值，否則即便可以使用API去溝通也不知道要傳遞的值為何。
  - 試過返回NULL、0或是其中一組預設值但都失敗。
  - 後來發現許多映像檔將一組NVRAM的初始值放置在幾個常見的目錄之下(/etc/nvram.default、/etc/nvram.conf或/var/etc/nvram.default)。

    - 有些則是在libaray中會出現router_defaults或是Nvrams等等的變數，可以去檢查他們是否被初始化。
- NVRAM的模擬依然不適用於所有的映像檔，可能是許多原因導致：
    - 使用未模擬到的函數。
    - 函數的參數或返回值和預期的不同。
    - 實現在Linux上的MTD(Memory Technology Device)上。
- NVRAM的模擬失敗是整體導致模擬失敗的主要原因之一，尷尬的是目前修復的主要方法是手動修復。

Kernel

- 前面提過，kernel不使用預設的，而替換成針對ARM和MIPS結構預編譯過的kernel。
- 使用kprobe監控20個system call，可以攔截並更改執行環境，幫助後續正確配置網路設定。這也可以檢查是否出現預先定義的異常值(如0xDEADBEEF、0x41414141)。
- 某些映像檔會預期存在filesystem(如/dev、/proc)，因此需要使用rdinit腳本，預先初始化這些filesystem；同樣的原因，使用nandsim這個kernel module模擬/dev/mtdX，這個部分在嵌入式系統時常被使用。
- 因為NVRAM的模擬是揮發性的，因此禁止重啟系統，除非重新呼叫init這個process。
- 支援新的硬體架構並不是自動化。QEMU不支援VirtIO的技術或是無法模擬PCI線路，且kernel也必須支援QEMU選中的硬體平台，因此需要手動修復並修改各種問題。

系統設定

- 因為主要對實現網路功能的設備感興趣，因此需要對某些特定的設備做更動。
- 理想狀況下，網路設備透過DHCP協定就可以完成設定，但會遇到一些問題：
    - 路由器、交換器等設備需要提供DHCP服務，而非接受服務。
    - 路由器可能有四個網路接口，而IP攝影機只有一個，造成接口數不同的情況。
- 對於每個設備，最初都會有60秒的學習階段，設定並收集網路相關的設定。
    - 追蹤分配給接口的IP位址。
    - 追蹤作為聚合的IEEE 802.1d bridge。
    - 檢查IEEE 802.1Q VLAN是否隔離無線網路和有線網路。
- 然後將訊息回傳到模擬系統中，使QEMU更加準確的配置設定。
    - 在host實體化出一個網路設備TAP，並將其網路的接口與韌體的接上。
    - 分配一個IP給TAP，和韌體在同一個子網中。
    - 透過ICMP協定並使用Nmap掃描網路連接。

QEMU

- 除了NVRAM，還希望模擬其他外部設備(如watchdog timer或快閃記憶體)，但許多製造商不是在kernelspace實作出這些功能，而是在userspace完成，導致模擬難度上升。
- 修改了QEMU的原始碼，使其支援了138個受快閃設備影響的映像檔。

自動化分析

- 在系統中實作了三個基本自動化動態分析，且每個在系統中都被註冊為callback function，當進入網路推斷階段的時候，每個callback都會被觸發，有助於檢測漏洞。

1. 可存取的網頁

- 為了檢測各種資訊洩漏、buffer overflow、command injection等漏洞，從LAN尋找可公開訪問的網頁。
- 使用python撰寫，遍歷疑似是server的路徑(例如/www/)，且非靜態資源(png、css等)，嘗試直接用web介面開啟。
- 忽略非2xx的回應；重新導向的類型不會被忽略，但會被標記為低可信度，其餘網頁則為正常可信度。

2. SNMP訊息

- 使用snmpwalk編寫了一個分析程式去儲存未經身分驗證的SNMP訊息。
- MIB文件是否存在敏感資訊，需要手動檢查。

3. 漏洞

- 先檢測資料集是否包含Metasploit中已知的60個漏洞。每個漏洞是按照順序執行，如果成功會提供payload和exploit log。
- 對於自行發現的新漏洞，預定義了有毒參數(如0xDEADBEEF)，接著檢測kernel log是否包含該值。

額外功能

- 開發了許多助於模擬框架、漏洞利用的功能。
  - 程式碼執行的動態追蹤，可導入到IDA Pro等工具中。
  - 禁用了context_switch()的inlining功能，可以追蹤user process的執行。
  - 在/dev/ttyS1啟動了一個console，該console是由QEMU轉發到Host上，有助於在執行時修改映像檔。

評估

- 本節包含：
  - 分析各階段對模擬結果的影響。
  - 展示如何利用系統辨識漏洞。
  - 透過添加已知漏洞，評估漏洞對資料集的影響。
- 需要注意的是，映像檔的發布不統一會影響解釋的結果，即便蒐集了MetaData，依然可能有問題。
  - 相同型號之下，不同區域的產品映像檔不同。
  - 映像檔和產品並非一對一關係，難以建立關係。
  - 結果：難以辨識那些映像檔已棄用、或是哪些是最新版本。

統計

- 上述資料的分析統計結果。

架構

- 透過檢查busybox的標頭或是/sbin/目錄下的檔案來識別架構種類。

Architecture (Endian)	# Image(s)
TILE (LE)	1
ARC (LE)	10
Motorola 68k (BE)	10
x86 (LE)	31
MIPS 64-bit (BE)	50
PPC (BE)	84
ARM (BE)	102
x86-64 (LE)	147
Unknown	439
ARM (LE)	843
MIPS (BE)	3,137
MIPS (LE)	4,632
Total	9,486

- MIPS和ARM(little-endian)佔了整體90.8%，因此以這兩種架構為目標。

作業系統

- 透過提取出來的filesystem和kernel的簽章去分類。

Type	# Images
Linux	9,379
Unidentified (UNIX-like)	2,187
VxWorks	857
Unknown	10,612
Total	23,035

- UNIX-like指確定為UNIX-based，但提取kernel時出現錯誤的種類。
  - 提取遇到路徑問題、不支援的壓縮法、韌體缺乏kernel。
- VxWorks數量較少，因此對這些設備的支援性較低。

Kernel Modules

- 根據文件名稱進行分類。

Category	# Modules
NetUSB	853
Unclassified	1,384
Cryptography	12,603
USB	30,683
Filesystems	43,271
Miscellaneous	55,344
Peripheral Drivers	64,085
Networking	296,592
Total	504,815

- 58.8%實現網路功能、12.7%實現外部設備溝通。
- 其中有不到0.2%為KCodes NetUSB，用於USB Over IP的技術，已知擁有buffer overflow的漏洞。

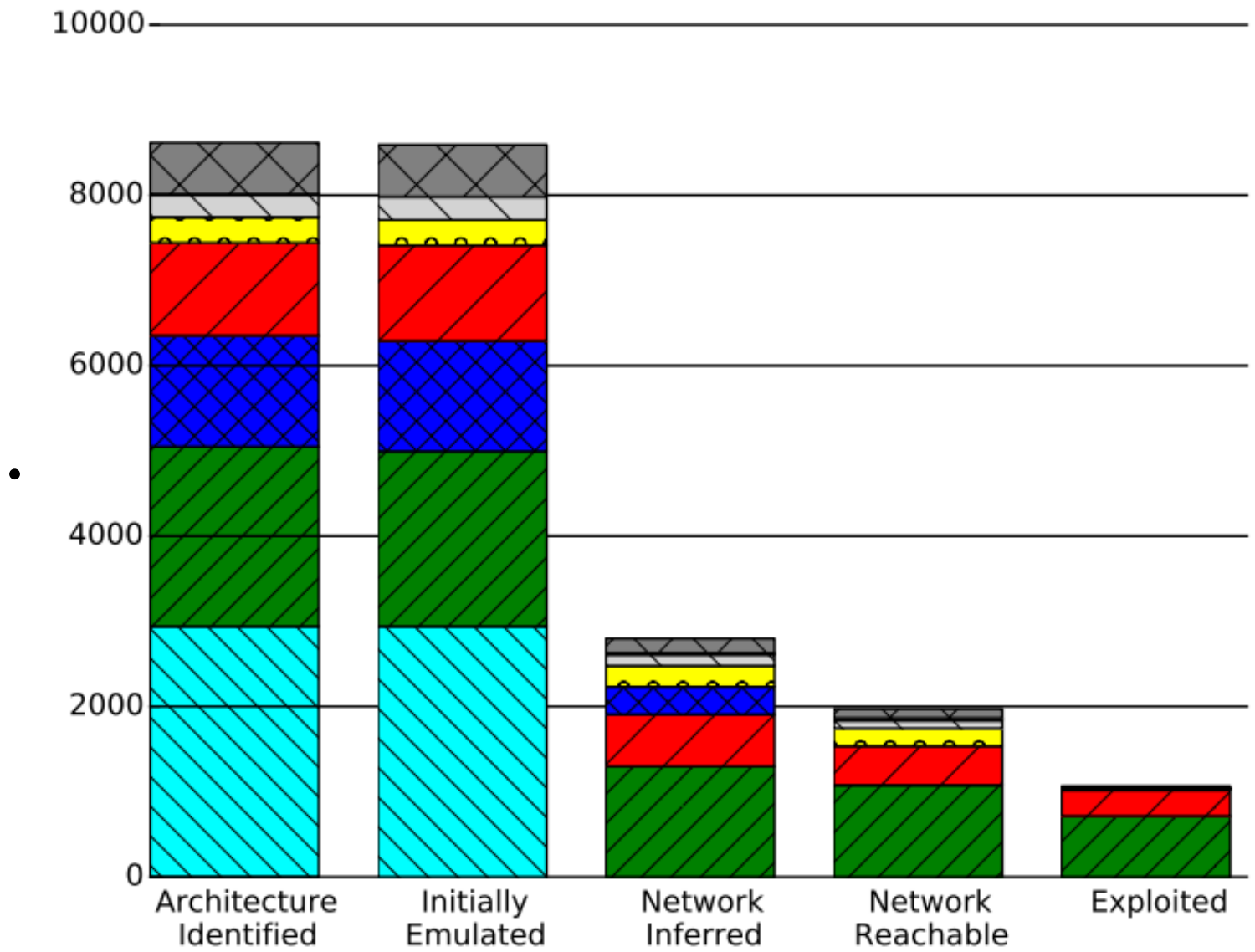
網路服務

- 針對有網路功能的設備，利用nmap工具搜尋TCP 1~1024 ports上的服務。

# Images	TCP Port/Service	# Vendor(s)
928	80/http	9
708	23/telnet	7
536	53/domain	6
250	3333/dec-notes	1
188	443/https	7
187	5000/upnp	2
136	1900/upnp	1
162	49152/unknown	4
63	2602/ripd	2
57	5555/freeciv	3

- 47.3%的設備可能有Web-based的頁面(HTTP或HTTPS)。
- 超過1024的port，大多是為了路由器常見的服務去檢查的。

模擬程度



- 全部共8617個映像檔，初始模擬成功96.6%；失敗原因包含：
  - /init/中缺少檔案，或是init的檔案名稱有所不同導致無法辨認。
  - 前面提過的提取失敗問題。
- 進入學習階段後，只有32.3%成功推斷出網路設置；失敗原因包含：
  - 大多是前面提過的NVRAM問題。
  - 網路接口設定問題。
- 2797個完成網路設置的設備中，使用ping只有70.8%有反應；失敗原因包含：
  - 防火牆規則擋掉了。
  - 前一步的網路設置其實是誤報。
- 最後，在可以透過網路溝通的設備中，有45%至少受到一種漏洞的攻擊。

結果

- 漏洞編號0~100代表來自Metasploit的已知漏洞，而大於等於200的則是以前未知的漏洞(除了202，它是非Metasploit的已知漏洞)。

# Exploits	# Images	# Vendor(s)	# Products
5	2	1	1
4	8	1	3
3	30	2	10
2	86	5	14
1	761	9	77
0	1,910	22	263
Total	2,797	42	322

- 有多個漏洞的映像檔只有4.5%。
- 大多數有多個漏洞的都是D-Link和Netgear生產的路由器，但可能只是漏洞和映像檔的資料集偏差。

注入攻擊

- 在自動化分析網頁的結果，在PHP伺服端發現了六個注入漏洞。
  - 可使系統參數遭到修改，例如MAC或是映像檔的地區。
  - 或是將MAC、流水號、或是版本號寫入到快閃記憶體中。
- 主要影響Netgear的路由器和基地台。

Buffer Overflow

- 一樣在自動化分析網頁發現了該漏洞。
- 發生在用戶身分驗證時，會去驗證uid的cookie的值；只要惡意修改cookie成一個過長的值就能使server崩潰。
- 主要影響D-Link的路由器。

資訊洩漏

- 透過自動化分析網頁，發現七個會洩漏資訊的漏洞。
- 其中一個洩漏了路由器的診斷訊息，包含WPS PIN和密碼。
- 其餘漏洞則在SNMP中洩漏了使用的憑證。

Sercomm Configuration Dump

- 該漏洞來自Metasploit，漏洞報告為CVE-2014-0659 (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0659)。
- 攻擊scfgmgr服務，進而從NVRAM取得Shell。
- 影響了資料集裡，14.3%可用網路溝通的映像檔。

MiniUPnPd Denial of Service

- 該漏洞來自Metasploit，漏洞報告為CVE-2013-0229 (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-0229)。
- 利用MiniUPnP的SSDP這個協定，觸發DoS攻擊。
- 影響了資料集裡，8.5%可用網路溝通的映像檔。
- 在2009-10-30的MiniUPnP1.4應該就修復了這個漏洞，但Metasploit統計到2013-1-29為止，仍有69%的設備使用1.0或更早的版本。
    - 這個結果指出開發商共享元件造成漏洞的普遍性。

OpenSSL ChangeCipherSpec

- 該漏洞來自Metasploit，漏洞報告為CVE-2014-0224 (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0224)。
- 利用OpenSSL 0.98za、1.0.0m、1.0.1h以前的所有版本中，錯誤的狀態機實作，導致中間人攻擊。
- 影響了資料集裡，8.5%可用網路溝通的映像檔，包含89.9%有HTTPS的映像檔；同時影響8.4%的所有產品，是最多的。

討論與限制

- FIRMADYNE可改善的地方：
    - 許多故障的地方需要手動修復。
    - 建立新的分析方法也需要手動，但完成之後可以對全部資料庫中的映像檔自動分析。
    - 分析結果可能因為支援的供應商而有偏差。
    - 因為使用預編譯的kernel，如果供應商提供的kernel或kernel modules有漏洞則無法檢測。
    - 無法確認網路是在WAN還是LAN運作，所以不知道漏洞是在Internet觸發還是只能在本地觸發。

相關工作

- 相關攻擊、漏洞：
    - Heffner等人對嵌入式映像檔做提取，得到了超過2000個hardcoded的SSL私鑰。
    - Costin等人用靜態分析的方式分析了32000個映像檔，發現了38個新漏洞，包含hardcoded back-door、XSS等。
    - Cui和Stolfo等人利用Nmap工具，發現540000個嵌入式設備在四個月中只有不到3%的憑證有修改，顯示用戶意識不足。
    - Heninger等人利用ZMap工具，發現2.45%的TLS憑證可能被暴力攻擊攻破，1.03%的DSA私鑰會被破解。
    - Cui等人發現攻擊者可以利用遠端更新的功能植入惡意程式。

    - Weinmann和Bonkoski等人發現某些遠端功能有漏洞，可使攻擊者控制系統或執行任意code。
    - Masketiecz和Nohl等人發現可以利用插入USB硬體控制主機系統。
- 抵禦攻擊、偵測漏洞的技術：
    - Davidson等人開發KLEE(一種symbolic executor)來偵測漏洞。
    - Li等人將QEMU模擬器放到BIOS中去驗證SoC。
    - Zaddach等人開發一種框架，可使模擬器的I/O轉到真實設備上，進而動態分析。
- 但以上方法在分析成本和時間上無法擴充，因此開發FIRMADYNE。

結論和未來發展

- 希望透過FIRMADYNE降低發現漏洞的門檻，以增加製造商對產品安全性的重視度。
- 希望可以支援更多系統(VxWorks)。
- 希望可以利用統計分析的技術，改進提取的部分。

tags: **paper**