

TCVN

TIÊU CHUẨN QUỐC GIA

DỰ THẢO

TCVN XXXX:2026
ISO/IEC TR 5891:2024

Xuất bản lần 1

**BẢO MẬT THÔNG TIN, AN NINH MẠNG VÀ BẢO VỆ
QUYỀN RIÊNG TƯ — CÔNG NGHỆ GIÁM SÁT PHẦN
CỨNG ĐỂ ĐÁNH GIÁ BẢO MẬT PHẦN CỨNG**

*Information security, cybersecurity and privacy protection — Hardware
monitoring technology for hardware security assessment*

HÀ NỘI – 2026

Mục lục

Lời nói đầu.....	4
Lời giới thiệu.....	5
1 Phạm vi áp dụng.....	7
2 Tài liệu viện dẫn.....	7
3 Thuật ngữ và định nghĩa.....	7
4 Thuật ngữ viết tắt.....	8
5 Mối quan hệ với các tiêu chuẩn hiện hành.....	8
5.1 Các tiêu chuẩn về đánh giá bảo mật.....	8
5.2 Mối quan hệ với ISO/IEC 15408-3.....	8
5.3 Mối quan hệ với ISO/IEC TS 30104.....	8
6 Bối cảnh.....	8
6.1 Độ phức tạp và bảo mật.....	8
6.2 Thách thức trong việc xác định các kỹ thuật đánh giá bảo mật phần cứng.....	9
7 Các công nghệ giám sát phần cứng.....	9
7.1 Tổng quan.....	9
7.2 Nghiên cứu trong lĩnh vực học thuật.....	10
7.3 Các trường hợp trong công nghiệp.....	10
7.4 Mục đích.....	11
7.5 Loại phương tiện mang (carrier type).....	15
7.6 Thực thể đích.....	22
7.7 Các mẫu mục tiêu (objective patterns).....	24
7.8 Phương pháp triển khai.....	25
8 Sử dụng các công nghệ giám sát để đánh giá bảo mật phần cứng.....	27
8.1 Các cách tiếp cận đánh giá bảo mật tiên tiến hiện có.....	27
8.2 Giám sát phần cứng có thể hỗ trợ như thế nào.....	27
8.3 Thách thức.....	28
9 Chứng nhận cho phần cứng giám sát.....	31
Thư mục tài liệu tham khảo.....	33

TCVN XXXX:2026

Lời nói đầu

TCVN XXXX:2026 hoàn toàn tương đương với ISO/IEC TR 5891:2024.

TCVN XXXX:2026 do Ban Cơ yếu Chính phủ biên soạn, Bộ Quốc phòng đề nghị, Ủy ban Tiêu chuẩn Đo lường Chất lượng Quốc gia thẩm định, Bộ Khoa học và Công nghệ công bố.

Lời giới thiệu

Các thành phần phần cứng và hệ sinh thái tính toán ngày càng trở nên phức tạp. Do đó, việc đánh giá bảo mật của phần cứng ngày càng khó khăn. Ngay cả trong giai đoạn thiết kế, rất khó xác định các phần bất thường có thể gây ra khiếm khuyết trong số hàng triệu dòng mã nguồn hoặc hàng tỷ bóng bán dẫn, cũng như các kết nối vật lý giữa chúng. Các lĩnh vực công nghệ khác sử dụng giám sát để hỗ trợ việc đánh giá nhằm giảm bớt những khó khăn như vậy. Trong các công nghệ đó, những hoạt động tại thời gian chạy, chẳng hạn như thay đổi trạng thái bên trong hoặc bên ngoài, có thể được giám sát để nhận diện các sai lệch so với các mẫu hành vi bình thường, và bằng cách này, việc đánh giá có thể tập trung vào một tập nhỏ các mẫu mà theo đó đối tượng được giám sát thường vận hành. Phương pháp này hiện trở thành một lựa chọn có thể áp dụng để hỗ trợ đánh giá bảo mật phần cứng. Trong những trường hợp như vậy, đối tượng của việc đánh giá bảo mật hoặc được coi là "bảo mật dựa trên hành vi phần cứng tại thời gian chạy", hoặc được đưa vào như một cách tiếp cận chủ động đối với bảo mật.

Nhiều tiêu chuẩn về thẩm định và đánh giá, chẳng hạn như ISO/IEC TS 30104, ISO/IEC 19790 và ISO/IEC 17825, tập trung vào bảo mật vật lý (xâm lấn/không xâm lấn) tại ranh giới phần cứng. Tuy nhiên, các tiêu chuẩn này không tập trung vào dữ liệu giám sát, dù là ngoại tuyến hay theo thời gian thực.

Bảo mật thông tin, an ninh mạng và bảo vệ quyền riêng tư — Công nghệ giám sát phần cứng để đánh giá bảo mật phần cứng

Information security, cybersecurity and privacy protection — Hardware monitoring technology for hardware security assessment

1 Phạm vi áp dụng

Tiêu chuẩn này khảo sát và tổng hợp các phương pháp giám sát phần cứng hiện có, bao gồm các công trình nghiên cứu và các ứng dụng công nghiệp. Các công nghệ giám sát được khảo sát được phân loại theo lĩnh vực áp dụng, loại phương tiện mạng, thực thể đích, mẫu mục tiêu và phương pháp triển khai. Ngoài ra, tiêu chuẩn này tổng hợp những cách thức khả dĩ để sử dụng các công nghệ giám sát cho việc đánh giá bảo mật phần cứng cùng với một số cách tiếp cận đánh giá bảo mật tiên tiến nhất hiện có.

Phần cứng được đề cập trong tiêu chuẩn này chỉ bao gồm phần cứng xử lý lõi, chẳng hạn như bộ xử lý trung tâm (CPU), bộ vi điều khiển (MCU) và hệ thống trên chip (SoC), trong hệ thống von Neumann và không bao gồm các thiết bị chỉ có một đầu vào hoặc một đầu ra như bộ nhớ hoặc màn hình hiển thị.

Công nghệ giám sát phần cứng được thảo luận trong tiêu chuẩn này có các lưu ý và giới hạn sau:

- đối tượng được giám sát thuộc giai đoạn hậu silicon, không thuộc giai đoạn thiết kế (ví dụ: thiết kế RTL hoặc netlist);
- việc giám sát chỉ được áp dụng cho hệ thống đang chạy.

2 Tài liệu viện dẫn

Các tài liệu viện dẫn sau đây là cần thiết để áp dụng tiêu chuẩn này. Đối với các tài liệu viện dẫn ghi năm công bố thì áp dụng phiên bản được nêu. Đối với các tài liệu viện dẫn không ghi năm công bố thì áp dụng phiên bản mới nhất, bao gồm cả các sửa đổi, bổ sung (nếu có).

ISO/IEC 15408-1, *Information security, cybersecurity and privacy protection — Evaluation criteria for IT security — Part 1: Introduction and general model* (Bảo mật thông tin, an ninh mạng và bảo vệ quyền riêng tư — Tiêu chí đánh giá bảo mật công nghệ thông tin — Phần 1: Giới thiệu và mô hình tổng quát)

ISO/IEC/TS 30104:2015, *Information Technology — Security Techniques — Physical Security Attacks, Mitigation Techniques and Security Requirements* (Công nghệ thông tin — Các kỹ thuật bảo mật — Tấn công bảo mật vật lý, kỹ thuật giảm thiểu và yêu cầu bảo mật)

3 Thuật ngữ và định nghĩa

Trong tiêu chuẩn này, áp dụng các thuật ngữ và định nghĩa nêu trong ISO/IEC 15408-1, ISO/IEC TS 30104 và các thuật ngữ, định nghĩa sau đây.

3.1

giám sát phần cứng (hardware monitoring)

Thành phần phần cứng hoặc phần mềm cho phép một cá nhân giám sát các thiết bị được kết nối với máy tính.

3.2

bảo mật dựa trên hành vi phần cứng tại thời gian chạy (runtime hardware-behaviour-based security)

Chức năng của phần cứng nhằm bảo vệ các thiết bị vật lý đang hoạt động khỏi tác hại do các chuyển đổi trạng thái bất thường hoặc không mong muốn gây ra.

CHÚ THÍCH 1 cho thuật ngữ: Các chuyển đổi như vậy bắt nguồn từ lỗi hỏng, các phần không được khai báo hoặc logic độc hại.

4 Thuật ngữ viết tắt

CPU	central processing unit	Bộ xử lý trung tâm
DRAM	dynamic random-access memory	Bộ nhớ truy cập ngẫu nhiên động
EDA	electronic design automation	Tự động hóa thiết kế điện tử
FSM	finite state machine	Máy trạng thái hữu hạn
I/O	input/output	Đầu vào/đầu ra
MCU	microcontroller unit	Bộ vi điều khiển
NIC	network interface controller	Bộ điều khiển giao diện mạng
RAS	reliability availability and serviceability	Độ tin cậy, tính sẵn sàng và khả năng bảo trì
RTL	register transfer level	Mức truyền thanh ghi
SoC	system on a chip	Hệ thống trên chip
JTAG	Joint Test Action Group	Nhóm hành động kiểm thử chung
IP	intellectual property	Sở hữu trí tuệ
CISC	complex instruction set computer	Máy tính có tập lệnh phức tạp
QoS	Quality of Service	Chất lượng dịch vụ
ISA	instruction set architecture	Kiến trúc tập lệnh
ECC	error checking and correction	Kiểm tra và sửa lỗi
ROM	read-only memory	Bộ nhớ chỉ đọc
EEPROM	electrically erasable programmable ROM	ROM lập trình được, xóa được bằng điện
VMM	virtual machine manager	Trình quản lý máy ảo
FPGA	field programmable gate array	Mảng cổng lập trình được trên trường

5 Mối quan hệ với các tiêu chuẩn hiện hành

5.1 Các tiêu chuẩn về đánh giá bảo mật

Các phương pháp đánh giá bảo mật và tiêu chuẩn kỹ thuật hiện hành gặp nhiều thách thức trong việc xử lý tính không thể kiểm soát của phần cứng. ISO/IEC TS 30104, ISO/IEC 19790 và ISO/IEC 17825 tập trung vào bảo mật vật lý (xâm lấn/không xâm lấn) tại ranh giới phần cứng, nhưng không vượt ra ngoài ranh giới đó. ISO/IEC TR 20004 bổ sung cho việc phân tích lỗ hổng của ISO/IEC 15408-3 từ góc độ phần mềm. Trong số các tiêu chuẩn này, chưa có tiêu chuẩn nào liên quan đến phần cứng.

5.2 Mối quan hệ với ISO/IEC 15408-3

ISO/IEC 15408-3 định nghĩa việc đánh giá lỗ hổng. Tiêu chuẩn này nhằm khảo sát các công nghệ hỗ trợ việc đánh giá lỗ hổng phần cứng có thể được thực hiện tại thời gian chạy.

5.3 Mối quan hệ với ISO/IEC TS 30104

Tiêu chuẩn này nhằm bổ sung cho 7.2 và 7.3 của ISO/IEC TS 30104:2015.

6 Bối cảnh

6.1 Độ phức tạp và bảo mật

Các mạch hiện đại rất phức tạp và độ phức tạp này đang gia tăng nhanh chóng trong môi trường tính toán hiện đại, càng bị gấp lên nhiều lần bởi áp lực về thời gian đưa sản phẩm ra thị trường. Do vậy, các hãng thiết kế thường xuyên sử dụng IP bên ngoài, và phần lớn các doanh nghiệp thiết kế vi mạch hoạt động theo mô hình không có nhà máy chế tạo (fabless).

Độ phức tạp của các hệ thống hiện đại làm gia tăng bề mặt tấn công. Do ngành công nghiệp bán dẫn đã chuyển sang mô hình kinh doanh theo chiều ngang đối với chuỗi cung ứng vi mạch tích hợp, phần cứng độc hại (Trojan phần cứng) có thể bị cấy vào trong các giai đoạn hoặc thành phần không tin cậy, ví dụ:

các lỗi IP thương mại, công cụ EDA, dịch vụ chế tạo và lắp ráp. Những sửa đổi độc hại như vậy đối với mạch gốc được kẻ tấn công đưa vào nhằm khai thác phần cứng hoặc lợi dụng các cơ chế phần cứng để tạo cửa hậu trong thiết kế.

6.2 Thách thức trong việc xác định các kỹ thuật đánh giá bảo mật phần cứng

Rất khó giải quyết tất cả các rủi ro bảo mật nêu trên do độ phức tạp của các quy trình và thành phần, việc thuê ngoài khâu thiết kế và chế tạo, cũng như mức độ tinh vi ngày càng tăng của các cuộc tấn công tiềm ẩn.

Đối với một sản phẩm phần cứng cho trước, đặc biệt là một hệ thống phức tạp như SoC hiện đại, cũng rất khó nhận diện những sửa đổi độc hại nhỏ so với thiết kế gốc (ví dụ: ở netlist mức cổng logic). Ngay cả khi có mã nguồn tin cậy, một sản phẩm phần cứng vẫn không thể được bảo đảm là không chứa Trojan trong giai đoạn hậu silicon. Các phép kiểm thử truyền thống (ví dụ: kiểm thử độ bao phủ chức năng, kiểm thử lỗi và kiểm thử trường hợp ngẫu nhiên) ít có khả năng hỗ trợ việc phát hiện này, bởi vì Trojan phần cứng thường được thiết kế để kích hoạt theo các điều kiện xác định trước, chẳng hạn như các chuỗi lệnh cụ thể, các tổ hợp tín hiệu bên ngoài nhất định, một bộ định thời hoặc một ngưỡng nhiệt độ. Kẻ tấn công có thể kích hoạt Trojan và tiến hành tấn công, sau đó vô hiệu hóa Trojan trong khi phần cứng vẫn đang hoạt động. Nói cách khác, các phương pháp truyền thống như kiểm toán mã nguồn hoặc phát hiện lỗi chế tạo là không hiệu quả đối với việc đánh giá bảo mật phần cứng.

Trojan phần cứng là phần được chèn thêm hoặc sửa đổi một cách độc hại trong phần cứng. Một Trojan phần cứng bao gồm một mạch kích hoạt và một mạch tải độc hại. Mạch kích hoạt kích hoạt mạch tải độc hại trong một điều kiện cụ thể, và mạch tải độc hại thực hiện hành vi độc hại của Trojan phần cứng. Trojan phần cứng có thể làm rò rỉ thông tin bí mật trong phần cứng, hoặc vượt qua hay vô hiệu hóa các chức năng bảo mật của phần cứng.

Việc phát hiện Trojan phần cứng có thể áp dụng trong nhiều giai đoạn của quá trình sản xuất và phân phối phần cứng. Ví dụ, việc phát hiện dựa trên netlist có thể được áp dụng trong giai đoạn thiết kế. Các cách tiếp cận dựa trên kênh kẻ và kiểm thử logic có thể được áp dụng trong giai đoạn chế tạo hoặc giai đoạn sử dụng. Dựa trên thực tế là Trojan phần cứng làm thay đổi hành vi của mạch, cách tiếp cận kênh kẻ phát hiện hành vi bất thường từ thông tin kênh kẻ. Cách tiếp cận dựa trên kiểm thử logic tạo ra các mẫu kiểm thử để phát hiện Trojan phần cứng thông qua đầu ra. Một số cách tiếp cận tiên tiến sử dụng các công nghệ học máy để phát hiện Trojan phần cứng từ netlist hoặc thông tin kênh kẻ của phần cứng.

Các khiếm khuyết phần cứng, chẳng hạn như Meltdown, Spectre và một loạt khiếm khuyết mới được công bố, [99-101] là hệ quả của việc theo đuổi hiệu năng, chẳng hạn tính song song, trong quá trình phát triển vi kiến trúc. Thứ nhất, chúng khó sửa chữa, và chi phí sửa chữa có thể lớn hơn lợi ích thu được từ việc tối ưu hóa phần cứng. Thứ hai, một số khiếm khuyết này tồn tại khoảng 10 năm đến 15 năm trước khi bị phát hiện. Việc phát hiện truyền thống trong giai đoạn tiền silicon khó phát huy tác dụng vì các khiếm khuyết không phải là những sửa đổi độc hại. Có ý kiến cho rằng một số kỹ thuật xác minh bảo mật tiên tiến có khả năng tình cờ tìm ra các khiếm khuyết như vậy từ sớm trong vòng đời thiết kế. [102] Tuy nhiên, thay vì được sử dụng trong một phương pháp đánh giá, các kỹ thuật này có nhiều khả năng hỗ trợ khâu thiết kế và có thể không sẵn có cho các bên thứ ba. Đối với các sản phẩm phần cứng chứa IP thương mại, việc áp dụng đánh giá bảo mật ở mức vi kiến trúc là rất khó khăn do yêu cầu bảo vệ bí mật thương mại.

7 Các công nghệ giám sát phần cứng

7.1 Tổng quan

Công nghệ giám sát phần cứng khởi đầu trong giai đoạn bùng nổ máy tính vào thập niên 1980. Ban đầu, công nghệ này được dùng để hỗ trợ gỡ lỗi và sau đó phát triển thành các ứng dụng trong nhiều lĩnh vực khác nhau. Tuy nhiên, cùng với sự phát triển nhanh chóng của phần cứng máy tính và mạng, đặc biệt là sự xuất hiện của các bộ xử lý đa lõi và các hệ thống phức tạp, bao gồm cả các hệ thống bộ xử lý đa lõi, công nghệ giám sát phần cứng cũng đã trải qua những thay đổi lớn. Trong khi môi trường phần mềm và phần cứng phát triển nhanh đã dẫn đến các chức năng ứng dụng phức tạp, môi trường này cũng phải đối mặt với những thách thức bảo mật ngày càng phức tạp. Trong các hệ thống dựa trên điện toán đám mây, tính ổn định và bảo mật tại thời gian chạy là rất quan trọng, vì các hệ thống này cung cấp dịch vụ

trực tuyến liên tục. Các đặc tính thời gian chạy riêng có của công nghệ giám sát phần cứng mang lại cho công nghệ này lợi thế vốn có trong việc ứng phó với các kịch bản này.

Giám sát phần cứng đã đạt được những tiến bộ đáng kể trong lĩnh vực học thuật và các ứng dụng công nghiệp. Công nghệ này được sử dụng rộng rãi trong các lĩnh vực và cho các mục đích sau:

- bảo mật;
- gỡ lỗi và kiểm thử;
- phân tích, đánh giá và tối ưu hóa hiệu năng;
- khả năng chịu lỗi và độ tin cậy của hệ thống;
- đo lường tham số vật lý và cảnh báo sớm.

Mặc dù trọng tâm của tiêu chuẩn này là đánh giá bảo mật phần cứng, các kỹ thuật giám sát dùng cho những mục đích khác cũng có ý nghĩa đối với việc xây dựng các mô hình đánh giá. Ví dụ, công nghệ dùng cho chạy thử và phân tích độ tin cậy tương tự công nghệ dùng cho việc phát lại trong quy trình đánh giá an toàn; công nghệ dùng cho phân tích hiệu năng có sự tương đồng nhất định với công nghệ phát hiện phần cứng chứa Trojan tại thời gian chạy. Do đó, trong tiêu chuẩn này, các từ khóa như "gỡ lỗi (debug)" và "hiệu năng (performance)" được sử dụng rộng rãi khi tra cứu tài liệu.

7.2 Nghiên cứu trong lĩnh vực học thuật

Về phía học thuật, các nghiên cứu khác nhau đã xem xét tổng quan các công nghệ giám sát từ những góc độ khác nhau.

Ian Cassar và các cộng sự^[1] chia các kỹ thuật chèn đo (instrumentation) giám sát tại thời gian chạy thành hai loại: ngoại tuyến và trực tuyến. Việc phân đoạn chi tiết loại trực tuyến trải từ các cách tiếp cận chèn đo giám sát đồng bộ hoàn toàn (CS) liên kết chặt, đến các cách tiếp cận giám sát không đồng bộ hoàn toàn (CA) liên kết lỏng.

Heidar Pirzadeh và các cộng sự^[2] chia các công nghệ giám sát thành công nghệ giám sát bằng phần mềm và bằng phần cứng, đồng thời chia nhỏ các công nghệ giám sát bằng phần mềm thành giám sát hỗ trợ, chèn đo thủ công, chèn đo trực tuyến, trình biên dịch có chức năng chèn đo, chèn đo bằng trình thông dịch và chèn đo bằng hệ điều hành. Các bộ giám sát khác nhau được so sánh đối chiếu với nhau về các mặt bảo mật, chi phí, tính linh hoạt, tính xâm lấn, hiệu năng và độ bao quát.

Lihua Gao và các cộng sự^[3] cùng Frank Cornelis và các cộng sự^[4] đã phân chia chi tiết công nghệ giám sát phần mềm tại thời gian chạy theo các cách riêng. Tài liệu tham khảo [3] phân loại và thảo luận các cấp độ khác nhau của mục tiêu giám sát (chức năng, mô-đun, kiến trúc và hệ thống con), trong khi tài liệu tham khảo [4] tập trung vào các sự kiện không tắt định và đề cập đến nhu cầu của các công nghệ khác nhau đối với tài nguyên bên ngoài (thời gian, thứ tự, ngôn ngữ, v.v.) để làm tiêu chí phân chia và so sánh.

Georgios Kornaros và các cộng sự^[5] tập trung vào công nghệ giám sát trên chip trong hệ thống SoC đa lõi và thảo luận chi tiết công nghệ giám sát từ góc độ chức năng và phương pháp luận.

7.3 Các trường hợp trong công nghiệp

Về mặt ứng dụng công nghiệp, các nhà sản xuất chip chủ đạo và các nhà cung cấp dịch vụ công nghệ thông tin cũng sử dụng công nghệ giám sát phần cứng trong sản phẩm của họ.

Từ năm 2013, Intel® đã đưa vào các bộ xử lý thương mại một công nghệ có thể hỗ trợ đắc lực cho việc gỡ lỗi, vì công nghệ này cung cấp vết hoạt động chính xác và chi tiết, đồng thời có khả năng kích hoạt và lọc để giúp tách riêng những vết quan trọng.

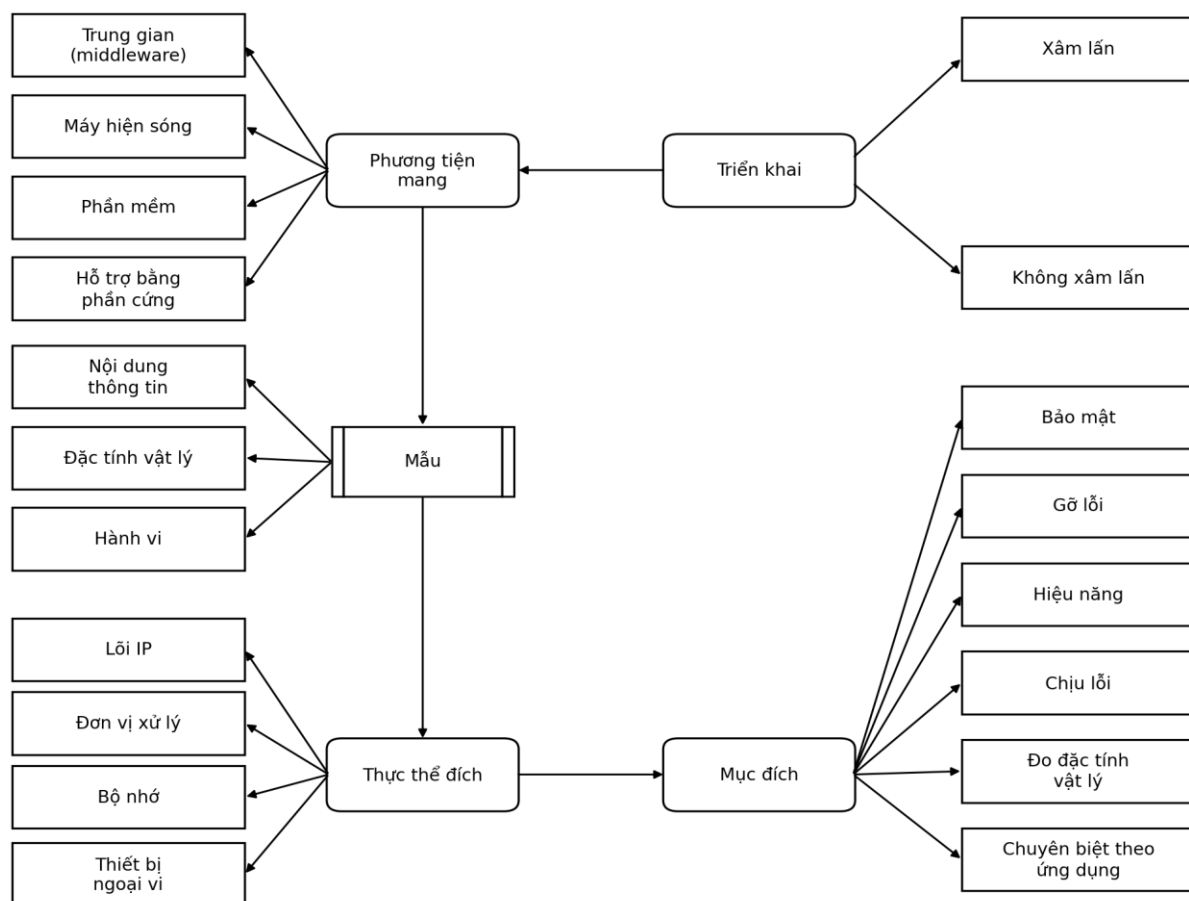
AMD® đã cung cấp một công nghệ tương tự để giám sát và điều khiển các bộ xử lý. Công cụ chuyên dụng này được thiết kế riêng cho một dòng thiết bị độc quyền, qua đó cho thấy các nhà thiết kế phần cứng và nhà thiết kế công cụ tiện ích cùng phối hợp để mang lại dịch vụ tốt nhất cho sản phẩm của mình.

ARM® đã thiết kế một bộ tiện ích, bao gồm nhiều macrocell theo vết khác nhau, các phép đo hệ thống và phần mềm cho bộ xử lý ARM, cùng một bộ khối IP hoàn chỉnh, để gỡ lỗi và theo vết các SoC đa lõi phức tạp nhất.

CHÚ THÍCH: Các tên thương mại Intel®, AMD® và ARM® được nêu vì lý do lợi ích công cộng hoặc an toàn công cộng. Thông tin này được đưa ra nhằm tạo thuận lợi cho người sử dụng tiêu chuẩn này và không phải là sự xác nhận của ISO hay IEC.

Ngoài ra còn có các chương trình giám sát phần cứng được dùng để đọc các cảm biến chính về tình trạng hoạt động của hệ thống máy tính cá nhân: điện áp, nhiệt độ, công suất, dòng điện, tốc độ quạt, mức sử dụng và tốc độ xung nhịp trong thời gian chạy.

Bảo mật phần cứng là một khái niệm phức tạp. Các loại phần cứng rất phức tạp. [Hình 1](#) mô tả toàn diện các công nghệ giám sát phần cứng từ năm góc độ: thực thể đích, mục đích, phương tiện mang, mẫu mực tiêu và triển khai.



Hình 1 — Phân loại các công nghệ giám sát phần cứng

7.4 Mục đích

7.4.1 Bảo mật

Việc ứng dụng công nghệ giám sát phần cứng cho mục đích bảo mật chủ yếu nhằm so sánh xem hành vi đang chạy có khớp với kỳ vọng hay không. Sau đó, việc kiểm soát bảo mật được thực hiện dựa trên kết quả so khớp. Kiểm soát bảo mật có thể chấm dứt hoạt động của hệ thống, thực thi lại mô-đun đích, hoặc chỉ ghi lại các sự kiện và trạng thái hiện tại để phục vụ phân tích ngoại tuyến hoặc kiểm toán bảo mật.

Tùy theo các mục tiêu giám sát khác nhau, các phương pháp giám sát khác nhau được áp dụng. Xét theo phạm vi, các phương pháp này có thể được chia thành hai nhóm: nhóm giám sát một thành phần phần cứng then chốt cụ thể trong hệ thống và nhóm giúp bảo đảm hoạt động an toàn của toàn bộ hệ thống. Việc bảo mật phần cứng trọng yếu có thể được chia thành hai hướng: giám sát việc cấy ghép Trojan độc hại và bảo vệ các cơ chế vận hành phần cứng trước lỗi hỏng bảo mật.

Hỗ trợ giám sát ở mức kiến trúc cho mục đích bảo mật thường tập trung vào việc đạt được khả năng mã hóa và khả năng chống can thiệp trái phép. Một số công nghệ tích hợp sẵn trên chip, chẳng hạn như toàn vẹn luồng điều khiển (CFI),^[6] có thể hỗ trợ người dùng xác minh tính toàn vẹn với hiệu năng cao.

Các công nghệ này tập trung vào khả năng chống tấn công của chính phần cứng lõi. Một số công nghệ^[7] có thể giúp phòng vệ trước phần mềm độc hại chiếm quyền điều khiển luồng, ví dụ: theo dõi rẽ nhánh gián tiếp và ngăn xếp bóng (shadow stack). Tính toàn vẹn và tính kịp thời của chính ngăn xếp bóng CFI cũng là một trọng tâm cần chú ý.^[8]

Để phát hiện Trojan, một số cách tiếp cận dựa vào chip chuẩn (golden chip) thông qua cảm biến tích hợp sẵn^[9] nhằm giám sát các đặc tính vật lý của một số mạch then chốt. Nếu phát hiện điện áp hoặc dòng điện bất thường, hiện tượng đó được nhận diện là hành vi của Trojan. Ngược lại, một số cách tiếp cận không cần chip chuẩn. Dựa vào tri thức sẵn có, các cách tiếp cận này trích xuất các đặc trưng điển hình của Trojan phần cứng và thực hiện nhận dạng mẫu bằng cách sử dụng RTL hoặc các mô tả phần cứng khác như netlist.^{[10][11]} Một số cách tiếp cận tiên tiến sử dụng các công nghệ học máy để phát hiện Trojan phần cứng từ netlist hoặc thông tin kênh kề của phần cứng.^[12-15]

Việc giám sát toàn hệ thống tập trung vào dữ liệu và logic sự kiện ở tầng trên, đồng thời cung cấp giám sát bảo mật theo cách dễ hiểu hơn đối với người dùng. Cách tiếp cận giám sát này thuận lợi hơn trong việc hướng dẫn đồng đạo người dùng không chuyên áp dụng các biện pháp bảo mật hiệu quả. Sentry^[16] là một hệ thống thu thập dữ liệu qua bus nội bộ kết nối với hệ thống để xác định hành vi tấn công, đồng thời đóng vai trò bộ lọc dữ liệu giữa hệ thống bên trong và hệ thống bên ngoài, mang lại khả năng kiểm soát bảo mật hiệu quả.

Các công nghệ giám sát phần cứng cũng có thể tận dụng các thuật toán phát hiện bất thường theo thời gian thực. Những công cụ tinh vi này có khả năng theo dõi các sai lệch so với hành vi kỳ vọng bằng cách sử dụng các mô hình thống kê, học máy hoặc trí tuệ nhân tạo để đưa ra cảnh báo tức thời về các mối đe dọa bảo mật tiềm ẩn.

Các điểm sau đây cũng cần được xem xét đối với bảo mật của hoạt động giám sát phần cứng:

- a) Vai trò của giám sát phần cứng trong kỷ nguyên tính toán lượng tử cũng cần được chú ý. Khi máy tính lượng tử đặt ra mối đe dọa đối với các phương pháp mã hóa hiện tại, giám sát phần cứng có thể đóng vai trò then chốt trong việc triển khai và bảo đảm tính hiệu quả của các phương pháp mật mã hậu lượng tử.
- b) Gốc tin cậy phần cứng (HROT) là một yếu tố thiết yếu của bảo mật phần cứng. HROT đóng vai trò điểm khởi đầu của sự tin cậy, bảo đảm tính toàn vẹn và tính bí mật của các giai đoạn tiếp theo trong quá trình vận hành hệ thống.
- c) Việc giám sát các bộ đếm hiệu năng phần cứng (HPC) cũng có thể cung cấp thông tin có giá trị về hành vi của hệ thống và các vi phạm bảo mật tiềm ẩn. Việc phân tích các bộ đếm này có thể phát hiện các mẫu bất thường cho thấy dấu hiệu của phần mềm độc hại hoặc Trojan phần cứng.
- d) Việc áp dụng các thông lệ tốt nhất về giám sát phần cứng sẽ bảo đảm đo lường và so sánh nhất quán mức độ bảo mật phần cứng giữa các hệ thống khác nhau, thúc đẩy các thực hành bảo mật rộng rãi và hiệu quả hơn trong toàn ngành.

7.4.2 Gỡ lỗi

Gỡ lỗi phần cứng, với khả năng giám sát hành vi của phần cứng tại thời gian chạy, là một phương pháp giám sát việc thực hiện phần cứng. Việc giám sát và gỡ lỗi hệ thống tính toán, đặc biệt là với bộ xử lý đơn, đã trở thành một lĩnh vực phát triển trưởng thành về công cụ và công nghệ.^{[17][18][19]} Giao diện JTAG (Joint Test Action Group) là một cách tiếp cận tiêu chuẩn phổ biến được dùng để xác minh thiết kế và kiểm thử các chức năng của vi mạch tích hợp. Một số nhà cung cấp bộ xử lý nổi tiếng đã phát triển các chức năng theo vết riêng của mình. JTAG sử dụng công nghệ quét biên (boundary-scan), cho phép kỹ sư thực hiện gỡ lỗi và chẩn đoán sâu rộng trên hệ thống thông qua một số lượng nhỏ chân kiểm thử chuyên dụng. Các tín hiệu được quét nối tiếp vào và ra khỏi các ô I/O của thiết bị nhằm điều khiển đầu vào và kiểm thử đầu ra của thiết bị trong các điều kiện khác nhau.

Trong một số bộ xử lý Intel, có thể đặt điểm dừng (breakpoint) tại các lệnh rẽ nhánh, ngắt và ngoại lệ, đồng thời thực hiện gỡ lỗi và phân tích từng bước từ nhánh này sang nhánh kế tiếp. Các cải tiến của những bộ xử lý này cho phép tạo ra các cơ chế bản ghi rẽ nhánh gần nhất (LBR) và lưu trữ vết rẽ nhánh (BTS). Các bộ xử lý dựa trên vi kiến trúc Intel Core™ còn hỗ trợ lấy mẫu chính xác dựa trên sự kiện (PEBS), cơ chế này lưu một tập thông tin trạng thái bổ sung phục vụ việc giám sát sự kiện chính xác.

CHÚ THÍCH: Core là nhãn hiệu của sản phẩm do Intel cung cấp. Thông tin này được đưa ra nhằm tạo thuận lợi cho người sử dụng tiêu chuẩn này và không phải là sự xác nhận của ISO hay IEC đối với sản phẩm được nêu tên. Có thể sử dụng các sản phẩm tương đương nếu chứng minh được rằng chúng cho kết quả tương tự.

Có một số cách tiếp cận phần cứng cho việc gỡ lỗi. Nếu hành vi sai được điều tra trong quá trình phát triển thiết bị, thì việc tìm kiếm lỗi trong silicon còn được gọi là xác nhận và gỡ lỗi hậu silicon, hay gọn hơn là xác nhận hậu silicon. Các kỹ thuật thiết kế phục vụ gỡ lỗi (DFD) được dùng để định vị các lỗi chức năng thông qua thăm dò logic, chuỗi quét và thu thập vết theo thời gian thực, và thường được gọi là các phương pháp phân tích logic nhúng. Nhiều cách tiếp cận khác nhau đề cập đến khung hỗ trợ gỡ lỗi và chuẩn hóa giao diện cho SoC^{[20][24]} cũng như các cấu trúc gỡ lỗi phục vụ chèn đo, chẳng hạn như bộ kiểm tra gỡ lỗi trên chip dựa trên khẳng định (assertion), bộ kích hoạt và bộ đếm sự kiện.^{[22][19]} Các kỹ thuật xác nhận hậu silicon điển hình được áp dụng để ghi lại giá trị các tín hiệu bên trong của mạch vào một bộ đệm vết trên chip và nạp các giá trị thu được vào một khung mô phỏng nhằm tái hiện hành vi sai.^[23] Do khối lượng dữ liệu gỡ lỗi rất lớn, việc ghi lại thông tin ở mức cao hơn, chẳng hạn như dấu vết lệnh đối với bộ xử lý, sẽ hiệu quả hơn.^[24]

Mặc dù năng lực của các trình mô phỏng tiếp tục được cải thiện một cách tuyến tính, việc không thể mô phỏng đầy đủ mọi hoán vị khả dĩ trong giai đoạn tiền silicon đã dẫn đến các giải pháp thay thế, chẳng hạn như giả lập (emulation) và tạo mẫu dựa trên FPGA trước khi hoàn tất thiết kế. Chèn đo trên chip là cải tiến then chốt cho phép quan sát và xác minh hiệu quả hơn trong bối cảnh chức năng được tích hợp liên tục, cùng với tốc độ xung nhịp bên trong ngày càng nhanh và I/O tốc độ cao, phức tạp. Hơn nữa, gỡ lỗi tại thời gian chạy cũng ngày càng quan trọng đối với các hệ thống đa lỗi, đặc biệt để phát hiện tình trạng tranh chấp (race condition) hoặc khóa chết (deadlock), đòi hỏi các cải tiến về kiến trúc và sự hỗ trợ của công cụ.

7.4.3 Điều chỉnh hiệu năng

Giám sát hiệu năng là một trong những mục tiêu then chốt của thiết kế hệ thống tính toán. Các đối tượng giám sát hiệu năng bao gồm thời gian thực thi, trùng/trượt bộ nhớ đệm, mức chiếm dụng CPU và bộ nhớ, cùng hiệu năng đọc và ghi của các thiết bị lưu trữ then chốt. Tài liệu tham khảo ^[25] chia giám sát hiệu năng thành bốn loại: phần mềm, phần cứng, lai và bộ đếm hiệu năng tích hợp sẵn trên chip. Kết quả so sánh đã chứng minh rằng bộ đếm trên chip là hiệu quả nhất. Hiện nay, tất cả các bộ xử lý chủ đạo đều cung cấp bộ đếm hiệu năng. Tuy nhiên, song song với việc phần cứng được cập nhật rất nhanh, độ phức tạp của bản thân hệ thống cũng tăng theo cấp số nhân. Các bộ đếm hiệu năng không thể được dùng để ghi lại tất cả các sự kiện. Do đó, việc lọc dữ liệu và sự kiện cốt lõi trở thành then chốt. Tài liệu tham khảo ^[26] lần đầu tiên đề xuất một kiến trúc mảng bộ đếm lai mới dựa trên các mảng SRAM và các bộ đếm rời rạc, có thể theo dõi đồng thời nhiều sự kiện.

Tài liệu tham khảo ^[27] chia hồ sơ hiệu năng thành hai loại: hồ sơ dựa trên thời gian và hồ sơ dựa trên sự kiện. Hồ sơ dựa trên thời gian dựa vào việc ngắt quá trình thực thi của ứng dụng theo các khoảng thời gian đều đặn. Để hỗ trợ lấy mẫu dựa trên sự kiện (EBS), phần cứng giám sát hiệu năng thường sinh ra một ngắt giám sát hiệu năng khi một bộ đếm sự kiện hiệu năng bị tràn. Alan Mink và các cộng sự^[28] đề xuất một hệ thống đo hiệu năng lai thu thập các mẫu dữ liệu và sự kiện bằng cách nhúng phần cứng trên bo mạch chủ; hệ thống này có thể được dùng để giám sát hiệu năng đa bộ xử lý.

Đối với nền tảng SoC đa lỗi, tài liệu tham khảo ^[29] trình bày một đơn vị giám sát hiệu năng (PMU) cho bus trên chip của SoC. PMU có thể đo các chỉ số hiệu năng chính, chẳng hạn như độ trễ bus đối với các yêu cầu từ thành phần dẫn dắt (leader) cụ thể và lưu lượng bộ nhớ trong những khoảng thời gian xác định. PMU cũng có thể đo mức độ cạnh tranh truy cập bus giữa các thành phần dẫn dắt và thành phần theo sau (follower) trong SoC. Bên cạnh các nghiên cứu về giám sát hiệu năng cho phần cứng then chốt trong hệ thống, còn có các nghiên cứu về giám sát hiệu năng của toàn hệ thống. Tài liệu tham khảo ^[30] chia việc giám sát hệ thống thành ba bước: giám sát hệ thống, mô hình hóa hệ thống và cải thiện hệ thống. Các tác giả đã xây dựng một công cụ tên là Charmonto để giám sát các đặc tính phần mềm và phần cứng, và sử dụng công cụ này cho việc giám sát dài hạn. Charmonto có thể chạy liên tục trong khi lấy mẫu nhiều loại chỉ số phần cứng khác nhau thông qua các bộ đếm giám sát hiệu năng (PMC)^[31] với những tần suất nhất định.

7.4.4 Khả năng chịu lỗi và QoS

Việc bảo đảm độ tin cậy của hệ thống liên quan đến nhiều tầng trừu tượng, từ mạch và vi kiến trúc đến thuật toán và tầng ứng dụng. Vì lý do này, nhiều phương pháp giám sát khác nhau đã được phát triển

trên các tầng này nhằm cung cấp khả năng tự chẩn đoán, tự thích ứng và tự phục hồi tại thời gian chạy. Phần lớn các cơ chế điều chỉnh độ tin cậy bất đối xứng, theo quan điểm thận trọng rằng một số thành phần gần như không có lỗi, và do đó, các cơ chế này được kỳ vọng bảo vệ các thành phần riêng lẻ của hệ thống như bus, mạng trên chip, bộ nhớ và đường dữ liệu trước các lỗi nhất thời hoặc vĩnh viễn. Ngoài ra, các đơn vị giám sát và chẩn đoán có yêu cầu hiệu năng không cao, nên chúng có thể hoạt động ở điện áp và tần số thấp, đồng thời thường sử dụng dự phòng tích hợp ở mức cao, khiến chúng gần như miễn nhiễm với sự cố.^[32] Tuy nhiên, các sơ đồ giám sát tập trung tiềm ẩn rủi ro về độ tin cậy do điểm lỗi đơn của chúng.

Các bộ giám sát trực tuyến nhằm trích xuất các chỉ số để xác định chính xác các mô hình lão hóa, thực hiện phân tích độ tin cậy hoàn chỉnh^[33] hoặc cung cấp các đặc tính độ tin cậy của các thành phần. Các kiến trúc tương lai chứa những thành phần thể hiện các đặc tính tin cậy khác nhau, chẳng hạn như các lỗi xử lý với độ chính xác số học khác nhau và các bộ nhớ với mức bảo đảm độ tin cậy khác nhau. Các ứng dụng sẽ được phép đánh đổi độ tin cậy phần cứng cao lấy hiệu năng cao thông qua việc chuyển đổi chế độ trong các hệ thống đa lỗi chế độ hỗn hợp với dự phòng mô-đun kép có thể cấu hình lại.^[34] Hơn nữa, bằng cách sử dụng các chú giải về độ tin cậy, trình biên dịch có thể tạo ra một ảnh xạ bảo đảm việc gán mã và các đối tượng dữ liệu trọng yếu về độ tin cậy cho những thành phần đáng tin cậy nhất của hệ thống, hoặc hệ điều hành có thể thực hiện di trú luồng chủ động theo định hướng độ tin cậy và tạo luồng bóng (ví dụ: các luồng bóng (shadow threads) có thể được gán cho các lỗi đáng tin cậy có ứng suất nhiệt thấp).

Nhiều hệ thống nhúng và hệ thống phân tán hiện đại (bao gồm cả hệ thống thời gian thực và phi thời gian thực) sử dụng các bộ giám sát mức sử dụng, bộ điều biến tốc độ, hoặc các bộ điều khiển dự báo theo mô hình hay bộ điều khiển khối lượng công việc với các kỹ thuật điều khiển phản hồi nhằm bảo đảm chất lượng dịch vụ.^{[35][36]} Để ứng phó với biến động khối lượng công việc trong các môi trường không thể dự đoán, việc phân bổ tài nguyên động có thể được dùng để đạt mức sử dụng bộ xử lý cao trong khi vẫn đáp ứng các ràng buộc thời gian thực, để áp đặt các giới hạn mức sử dụng phù hợp bảo đảm khả năng lập lịch, hoặc để xử lý động lực học hệ thống do cân bằng tải gây ra đối với các cụm máy chủ quy mô lớn. Từ một góc độ khác, các bộ giám sát giúp tránh tình trạng bão hòa bộ xử lý, vốn có thể gây sự cố hệ thống hoặc suy giảm dịch vụ nghiêm trọng. Tuy nhiên, các cách tiếp cận quản lý dựa trên điều khiển phản hồi này đòi hỏi một mô hình hệ thống chính xác, điều có thể khó đạt được đối với các hệ thống nhúng phân tán hoặc đa lỗi trong thực tế.

Việc giám sát để duy trì QoS có thể là yếu tố then chốt đối với những lĩnh vực ứng dụng nhất định. Chẳng hạn, các máy chủ cung cấp dịch vụ truyền phát video thích ứng khai thác tính thích ứng vốn có của các ứng dụng video để thực hiện những điều chỉnh có kiểm soát và hài hòa đối với chất lượng cảm nhận của luồng video MPEG hiển thị, nhằm ứng phó với các dao động về QoS do hạ tầng bên dưới cung cấp. Hiệu quả được nâng cao khi dịch vụ giám sát không độc lập với nền tảng. Trong thiết kế hệ thống nhúng, nơi chi phí và tài nguyên của nền tảng bị giới hạn, việc giám sát và kiểm soát chất lượng với sự trợ giúp của hệ thống và phối hợp cùng các ứng dụng là điều tất yếu.^[37]

Một thành phần giám sát riêng lẻ có thể cung cấp các chỉ số liên quan đến QoS cho các tầng trừu tượng cao hơn của hệ thống. Thông tin này được các tầng trên sử dụng để phân tích và phản ứng với các biến động QoS, ví dụ: chuyển sang khối chức năng dự thừa hoặc thứ cấp, hoặc chạy ở chế độ dự phòng. Đối với các ứng dụng trọng yếu về an toàn khác, các mô hình riêng lẻ có thể ra quyết định dựa trên ngưỡng dung sai QoS của chúng.^[41]

7.4.5 Đo đặc tính vật lý

Khi mật độ công suất tăng lên cùng với kích thước hình học không ngừng thu nhỏ, những biến thiên nhiệt độ lớn có thể xảy ra trên chip và cần được ngăn ngừa hoặc ít nhất là giảm thiểu. Các cơ chế giám sát đóng vai trò nền tảng trong việc xây dựng các kiến trúc chip thích ứng, hướng tới đạt hiệu năng lý thuyết cao nhất trong một cấu hình điều chỉnh điện áp và tần số động (DVFS) an toàn về nhiệt cho các khối lượng công việc cụ thể hoặc thay đổi.

Các bộ vi xử lý hiện đại luôn có cảm biến nhiệt độ số tích hợp sẵn cho phép giám sát nhiệt độ theo thời gian thực. Các cảm biến này nằm trong từng lõi xử lý riêng lẻ, gần vị trí nóng nhất. Dữ liệu thu thập từ các cảm biến này rất chính xác vì không phụ thuộc vào mạch ngoài đặt trên bo mạch chủ để báo nhiệt độ. Ví dụ, các bộ vi xử lý Intel gần đây chứa ít nhất một cảm biến nhiệt độ cho mỗi lõi, và thường có

thêm nhiều cảm biến khác. Đã có báo cáo ghi nhận hơn 16 giá trị nhiệt độ trên toàn bộ die (tám bán dẫn).

Cũng có nhiều công trình nghiên cứu trong lĩnh vực này, do việc khám phá không gian thiết kế cho quản lý công suất và nhiệt độ tại thời gian chạy liên quan đến một số lượng tham số rất lớn. Có một số cách tiếp cận cho bộ xử lý đơn và một số cách tiếp cận cho kỹ nguyên đa lõi. [38][39][40][41] Mục tiêu chính là chọn được chiến lược giám sát tốt nhất về mặt hiệu quả, phù hợp với từng loại chip, hệ thống và môi trường cụ thể.

7.4.6 Giám sát chuyên biệt theo ứng dụng

Giám sát chuyên biệt theo ứng dụng bao gồm các cơ chế được phát triển để đáp ứng bất kỳ mục tiêu nào đã thảo luận ở trên nhưng được tùy biến sâu hơn nhằm xét đến hành vi của một ứng dụng cụ thể. Chẳng hạn, loại này bao gồm các kỹ thuật giám sát để khám phá trực tuyến các giá trị tham số hiệu năng/năng lượng tốt nhất, có thể được tùy biến theo đặc tính sử dụng bộ nhớ của ứng dụng.

Các bộ giám sát và bộ quản lý tại thời gian chạy đã đề xuất các bộ xử lý mở rộng được với tập lệnh tùy biến, sử dụng lệnh đặc biệt phù hợp nhất cho một bộ mã hóa video đa phương tiện. [44] Ý tưởng chính của cách tiếp cận giám sát trực tuyến này là ước lượng chất lượng sử dụng của các lệnh đặc biệt được cấu hình lại trong đường dữ liệu của bộ xử lý. Phương pháp này sử dụng các bộ đếm để phản ánh số lần thực thi của các lệnh tùy biến này trong mỗi lần lặp của một vòng lặp tính toán chuyên sâu.

Hơn nữa, trong phát triển hệ thống nhúng, các bộ xử lý có tập lệnh chuyên biệt theo ứng dụng (ASIP) đã trở nên phổ biến vì chúng kết hợp tính linh hoạt của phần mềm với hiệu quả năng lượng và hiệu năng tính toán có thể mở rộng của các giải pháp thực hiện bằng phần cứng chuyên dụng. Bằng cách điều chỉnh phương pháp phát triển ASIP, việc giám sát có thể nâng cao khả năng quan sát và khả năng thích ứng của hệ thống. Chẳng hạn, tài liệu tham khảo [45] mô tả một phương pháp cho các thủ tục giám sát nhằm kiểm tra các thao tác không bảo mật thông qua việc bổ sung các vi lệnh phụ vào bên trong những lệnh máy để bị tổn thương. ASIP có thể được tổng hợp và nhắm tới các ứng dụng khác nhau, và các ứng dụng này có thể sử dụng những phương pháp giám sát bảo mật khác nhau (giám sát bus dữ liệu của bộ nhớ lệnh, giám sát đường dữ liệu, giám sát ngăn xếp địa chỉ trả về và giám sát lệnh rẽ nhánh).

Đối với các ứng dụng trọng yếu về hiệu năng, các mục tiêu tùy biến cho ứng dụng hoặc các mô-đun con chuyên biệt theo ứng dụng có thể tận dụng một hoặc nhiều chỉ số giám sát hiện có. Chẳng hạn, QoS có thể là một chỉ số về độ bền vững vốn có và hiệu năng của hệ thống. Sự sụt giảm QoS có thể là tác động trực tiếp của một vi phạm bảo mật. Các mục tiêu tổng hợp như vậy có thể được coi là chuyên biệt theo ứng dụng và không mang tính khái quát.

7.5 Loại phương tiện mạng (carrier type)

7.5.1 Trung gian (middleware)

7.5.1.1 Trong các ô mạch (cell)

Trong số các chiến lược phát hiện lỗi nhất thời do bức xạ hoặc nguồn quang gây ra, các cảm biến dòng điện khối tích hợp sẵn (BBICS) đưa ra một giải pháp phù hợp với các luồng thiết kế hệ thống dựa trên các ô mạch chuẩn CMOS của các thư viện thương mại. BBICS đạt hiệu quả phát hiện cao tương đương các sơ đồ chịu lỗi tốn kém (ví dụ: nhân đôi kèm so sánh), trong khi chi phí phụ trội về diện tích và công suất lại thấp tương đương các kỹ thuật giảm thiểu kém hiệu quả hơn, chẳng hạn các cách tiếp cận dự phòng theo thời gian. [46][47][48][49] Không ảnh hưởng đến tần số hoạt động của hệ thống, BBICS xử lý được các lỗi nhất thời có thời lượng ngắn và dài cũng như nhiều lỗi đồng thời. Hơn nữa, do BBICS giám sát sát sao các vùng nơi lỗi phát sinh, việc phát hiện sớm là khả thi ngay sau khi lỗi xảy ra, nhờ đó ngăn ngừa việc phát sinh và lan truyền lỗi sang các chu kỳ xung nhịp hoặc khối hệ thống khác. [50]

7.5.1.2 Các thành phần phần cứng gắn kèm bên trong

Tài liệu tham khảo [51] mô tả MAMon, một hệ thống giám sát có thể giám sát cả mức logic và mức hệ thống trong các SoC đơn/đa bộ xử lý. Một đơn vị thăm dò phần cứng nhỏ được tích hợp vào thiết kế SoC và kết nối qua liên kết cổng song song với một môi trường công cụ giám sát trên máy tính chủ (host).

Tài liệu tham khảo [52] mô tả RG-Secure, giải pháp kết hợp chiến lược thiết kế tin cậy đối với sở hữu trí tuệ của bên thứ ba với công nghệ phân tích đặc trưng netlist của chuỗi quét, và được trang bị một thuật toán tăng cường gradient phân tán, hạng nhẹ có tên lightGBM.

Để cho phép phát lại tất định toàn hệ thống các phiên thực thi đa bộ xử lý, Flight Data Recorder (FDR) [53] bổ sung một lượng nhỏ phần cứng vào một hệ thống đa bộ xử lý nhất quán tuần tự dựa trên thư mục.

7.5.1.3 Phần cứng lọc đặt ngoài hệ thống

Trái ngược với phần cứng được nhúng trong hệ thống, một bộ giám sát có thể được đặt bên ngoài hệ thống và được dùng để lọc và giám sát hoạt động giao tiếp của hệ thống với môi trường bên ngoài, cũng như kiểm tra hành vi tương tác có tuân thủ quy định hay không. Ưu điểm của bộ giám sát đặt bên ngoài là không đòi hỏi phải cải tạo quy mô lớn hệ thống gốc. Cách bố trí này thuận tiện cho việc đánh giá và giám sát hành vi mức cao của hệ thống cũng như báo cáo các vấn đề bảo mật theo cách người dùng dễ hiểu hơn.

Các phương pháp điển hình bao gồm TrustGuard. [54] Các bảo đảm về bảo mật và quyền riêng tư trong TrustGuard được xây dựng trên một phần tử phần cứng đơn giản, có thể cắm rời, gọi là Sentry. Với mô hình Sentry cải tiến, [16] TrustGuard hiện chỉ cho phép xuất dữ liệu ra từ quá trình thực thi đúng đắn của phần mềm đã được ký. Sentry trong TrustGuard kiểm tra:

- a) tính đúng đắn của việc thực thi lệnh so với ISA (bao gồm việc xác định lệnh kế tiếp theo thứ tự chương trình); và
- b) mỗi lệnh có thuộc một chương trình đã được ký hay không.

Chính sách quy định hành động cần thực hiện khi phát hiện một phiên thực thi sai. TrustGuard hỗ trợ nhiều chính sách khác nhau đối với thực thi sai, bao gồm dừng thực thi, tiếp tục thực thi trong khi chặn đầu ra sai, cảnh báo người dùng và báo cáo lệnh sai. Bản chứng minh khái niệm này hỗ trợ một hệ thống với bộ xử lý đơn lõi chạy phần mềm, bao gồm cả hệ điều hành, được ký bởi một cơ quan tin cậy. Ngoài ra, TrustGuard yêu cầu mọi hoạt động I/O phải xuất phát từ các lệnh I/O tường minh.

7.5.2 Phần mềm

Số lượng chủng loại phần cứng ngày càng tăng, và cấu trúc phần cứng ngày càng phức tạp. Một bộ giám sát phần cứng có thể được kết hợp với một bộ giám sát phần mềm để đạt kết quả đánh giá tốt hơn. Do đó, điều nhỏ này tóm tắt ngắn gọn công nghệ giám sát bằng phần mềm.

Công nghệ giám sát bằng phần mềm được phân chia theo mức và có thể được xếp vào các loại sau:

- a) Tầng hệ điều hành, cung cấp các API giám sát hoặc trình điều khiển hệ thống.

Do các hạn chế về quyền của hệ điều hành, các ứng dụng tầng trên không thể truy cập trực tiếp các tài nguyên nhạy cảm. Chương trình giám sát gọi API giám sát đặc thù của hệ điều hành, chẳng hạn như phương pháp của Instant-Replay, [55] để thu thập thông tin vận hành liên quan.

- b) Tầng ứng dụng, bổ sung các chức năng giám sát hoặc nhúng mã giám sát.

Mã được thêm vào một hàm được thực thi bình thường, hoặc các tiến trình hay luồng song song được khởi động để giám sát các nội dung cần quan tâm. Ví dụ, có thể bổ sung các biến quan sát hoặc việc ghi lại thông tin đường đi, chẳng hạn các lệnh nhảy. Các bộ giám sát phần mềm dùng ở tầng ứng dụng bao gồm JaRec, [56] một môi trường ghi/phát lại độc lập nền tảng cho các ứng dụng Java™ đa luồng, được thực hiện dựa trên giao diện lập hồ sơ máy ảo Java (JVMPI).

CHÚ THÍCH: Java™ là nhãn hiệu của sản phẩm do Oracle cung cấp. Thông tin này được đưa ra nhằm tạo thuận lợi cho người sử dụng tiêu chuẩn này và không phải là sự xác nhận của ISO hay IEC đối với sản phẩm được nêu tên.

- c) Tầng kiến trúc, xây dựng một kiến trúc hệ thống có năng lực giám sát.

Việc giám sát được xét đến ngay trong quá trình xây dựng khung hệ thống. Quy trình gọi và các quy định sử dụng định hướng cho việc thực hiện các yêu cầu giám sát về sau. Các bộ giám sát phần mềm dùng ở tầng kiến trúc bao gồm ReVirt. [57]

Các tài liệu liên quan được tổng hợp trong [Bảng 1](#).

Bảng 1 — Phân loại các bộ giám sát phần mềm

Phương pháp	Mức	Không xâm lấn	Bộ xử lý đơn lỗi hay đa lỗi	Mô tả
Halsall-Hui ^[58]	Kiến trúc	Không	Đa lỗi	Bộ giám sát này được thiết kế để thu thập dữ liệu từ từng nút xử lý của một hệ thống nhúng thời gian thực dựa trên kiến trúc phân tán.
ReVirt ^[57]	Kiến trúc	Có	Đa lỗi	ReVirt có thể phát lại toàn bộ quá trình thực thi của một máy ảo theo từng lệnh, ngay cả khi quá trình thực thi đó phụ thuộc vào các sự kiện không tắt định như ngắt và đầu vào của người dùng. Quản trị viên có thể sử dụng kiểu phát lại này để trả lời những câu hỏi chi tiết tùy ý về những gì đã xảy ra trước, trong và sau một cuộc tấn công.
BugNet ^[59]	Kiến trúc	Có	Đa lỗi	Kiến trúc BugNet tập trung vào việc theo vết liên tục quá trình thực thi chương trình. BugNet chú trọng việc phát lại một cách tắt định các lệnh được thực thi trong mã người dùng và các thư viện dùng chung.
Instant Replay ^[55]	Hệ điều hành	Có	Đa lỗi	Instant Replay là một kỹ thuật phát lại các truy cập bộ nhớ dùng chung bằng cách tiếp cận dựa trên thứ tự. Kỹ thuật này giới hạn chương trình chỉ được truy cập các đối tượng bộ nhớ dùng chung thông qua các nguyên thủy của giao thức CREW (nhiều tiến trình đọc đồng thời - một tiến trình ghi độc quyền) được định nghĩa rõ ràng, mà Instant Replay sử dụng cho việc phát lại quá trình thực thi.
Interrupt Replay ^[60]	Hệ điều hành	Không	Đa lỗi	Audenaert và Levrouw đã mô tả một phương pháp phát lại các chương trình song song trên các hệ thống đa bộ xử lý bộ nhớ dùng chung dựa trên bus có sử dụng ngắt. Bộ giám sát này được dùng như một phần mở rộng của Instant Replay.
HMON ^[61]	Hệ điều hành	Có	Đa lỗi	HMON được phát triển để giám sát hiệu năng của Kiến trúc lục giác cho các hệ thống thời gian thực (HARTS), một hệ thống thời gian thực phân tán.
IGOR ^[62]	Ứng dụng	Không	Đơn lỗi	Dựa trên các kỹ thuật điểm kiểm tra (checkpoint), IGOR là một hệ thống phát lại chương trình. Với một điểm kiểm tra cho trước, IGOR tái dựng trạng thái của chương trình tại điểm đó và cho phép chương trình tiếp tục thực thi. IGOR không ghi lại I/O bên ngoài, nên phiên thực thi được phát lại có thể khác với phiên thực thi gốc nếu môi trường đã thay đổi. Ngoài ra, IGOR không xử lý tính không tắt định do các chương trình đa luồng gây ra.
JaRec ^[56]	Ứng dụng	Không	Đa lỗi	JaRec hoạt động hoàn toàn ở mức bytecode Java. Mỗi lớp được JVM (máy ảo Java) nạp vào sẽ được chuyển qua JVMPI (giao diện lập hồ sơ JVM) đến một bộ chèn đo, sau đó lớp đã được chèn đo sẽ được nạp.

jRapture ^[63]	Ứng dụng	Có	Đơn lỗi	jRapture là một công cụ được thiết kế để ghi lại các tương tác giữa một ứng dụng Java và hệ thống bên dưới, tức là các tương tác với giao diện người dùng, tệp, bàn phím. Trong giai đoạn phát lại, jRapture cung cấp cho các luồng đang thực thi đúng chuỗi đầu vào mà chúng đã nhận được trong giai đoạn ghi.
Recap ^[64]	Ứng dụng	Có	Đa lỗi	Recap là một công cụ tạo cảm giác như đang thực thi ngược cho các chương trình song song. Recap sử dụng kết hợp điểm kiểm tra và phát lại để mang lại cho người dùng cảm giác về việc thực thi ngược.
DejaVu ^[65]	Kiến trúc	Không	Đơn lỗi	DejaVu là một hạ tầng phát lại thực thi được thiết kế tại IBM® nhằm xử lý tính không tắt định do các luồng và những cấu trúc tương tranh liên quan khác gây ra trong các ứng dụng Java.
Nhãn hiệu này được nêu vì lý do lợi ích công cộng hoặc an toàn công cộng. Thông tin này được đưa ra nhằm tạo thuận lợi cho người sử dụng tiêu chuẩn này và không phải là sự xác nhận của ISO hay IEC.				

7.5.3 Các bộ giám sát có hỗ trợ phản cứng

Các bộ giám sát có hỗ trợ phản cứng là những công nghệ hoặc hệ thống phục vụ gỡ lỗi, điều chỉnh hiệu năng, chịu lỗi, hoặc đo lường công suất, năng lượng và nhiệt độ. Chúng không được thiết kế với mục đích đánh giá bảo mật. [Bảng 2](#) liệt kê các tài liệu chính liên quan đến các bộ giám sát có hỗ trợ phản cứng.

Bảng 2 — Phân loại các bộ giám sát có hỗ trợ phản cứng

Phương pháp	Mục đích	Phương tiện mang	Thực thể đích	Các mẫu mục tiêu	Phương pháp triển khai	Mô tả
FDR ^[53]	Gỡ lỗi	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Tương tự hộp đen ghi dữ liệu chuyến bay của máy bay, FDR liên tục ghi lại quá trình thực thi để chờ một sự kiện kích hoạt, có thể là một sự cố nghiêm trọng hoặc một vi phạm khẳng định (assertion) của phần mềm.
SafetyNet ^[66]	Gỡ lỗi	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	SafetyNet định kỳ tạo điểm kiểm tra trạng thái hệ thống để cho phép hệ thống khôi phục trạng thái về một điểm kiểm tra nhất quán trước đó. Nếu phát hiện lỗi, SafetyNet khôi phục trạng thái về điểm khôi phục, là điểm kiểm tra cũ gần nhất đã được xác nhận là không có lỗi.

RnR-Safe ^[67]	Bảo mật	Trung gian	Đơn lỗi	Các hành vi	Không xâm lấn	RnR-Safe hỗ trợ các tính năng bảo mật phần cứng nhằm giảm tải các phép kiểm tra xâm nhập và/hoặc loại bỏ sự thiếu chính xác của kiểm tra. RnR-Safe giảm chi phí phần cứng bảo mật bằng cách cho phép phần cứng kém chính xác hơn khi phát hiện tấn công, do đó có thể báo cáo dương tính giả.
PASM ^[68]	Gỡ lỗi	Trung gian	Đa lỗi	Thông tin và các hành vi	Xâm lấn	PASM là một bộ giám sát phần cứng lập trình được, cung cấp một bộ công cụ linh hoạt để người dùng chỉ định các sự kiện cho nhiều loại ứng dụng giám sát khác nhau. Người dùng có thể kèm theo ứng dụng một phân đoạn giám sát định nghĩa các sự kiện cần quan tâm, các hành động cần thực hiện khi phát hiện các sự kiện này, và sự ràng buộc giữa sự kiện với hành động.
ART ^[69]	Gỡ lỗi	Trung gian	Đa lỗi	Các hành vi	Xâm lấn	Cách tiếp cận này tập trung vào việc trực quan hóa hành vi định thời của các tiến trình hệ thống. Bộ giám sát này hỗ trợ các thuật toán đơn điệu theo tốc độ (rate monotonic) và máy chủ trì hoãn được (deferrable server), và tác vụ giám sát được thực hiện như một phần của hệ thống đích.
ZM4 ^[70]	Gỡ lỗi và hiệu năng	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Trong cách tiếp cận này, một hệ thống phần cứng gọi là ZM4 và một phần mềm xử lý vết sự kiện gọi là SIMPLE, hoạt động độc lập với bộ giám sát, đã được phát triển. Kết nối giữa phần cứng và hệ thống được

						giám sát thuộc loại mạng cục bộ.
Trams ^[28]	Gỡ lỗi	Trung gian	Đa lỗi	Các hành vi	Xâm lấn	Kiến trúc của hệ thống này gồm phần mềm kích hoạt sự kiện có chức năng chen các lệnh ghi vào mã và một hệ thống con phần cứng dùng để lấy mẫu thời gian và danh tính của CPU.
Dựa trên SoC ^[74]	Gỡ lỗi, hiệu năng, chịu lỗi và bảo mật	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Bộ giám sát dựa trên hệ thống trên chip này sử dụng một phương pháp lai để xác minh tại thời gian chạy các hệ thống nhúng. Bộ giám sát gồm một bộ nhận diện sự kiện, một công cụ xác minh và đầu ra của bộ giám sát. Bộ nhận diện sự kiện quyết định dữ liệu thu thập được có phù hợp với định nghĩa sự kiện hay không. Sau khi qua bước này, dữ liệu sự kiện được gửi đến phân đoạn xác minh, nơi chúng được so sánh với các ràng buộc yêu cầu.
MAMon ^[51]	Gỡ lỗi	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	MAMon là một hệ thống giám sát cho các hệ điều hành thời gian thực được tăng tốc bằng phần cứng.
Roadnoc ^[2]	Khả năng quan sát, mức sử dụng bộ đệm của các bộ định tuyến NoC	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Roadnoc là một thành phần giám sát dựa trên bộ đếm được gắn vào bộ định tuyến.
ADAM ^[73]	Tối ưu hóa: ảnh xạ tác vụ trên NoC	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	ADAM thực hiện ánh xạ ứng dụng tại thời gian chạy trên NoC theo cách phân tán bằng cách tiếp cận dựa trên tác tử.

Vicis ^[74]	Độ tin cậy cho NoC	Trung gian	Đơn lỗi	Các hành vi	Xâm lấn	Cơ chế tự kiểm thử tích hợp tại mỗi bộ định tuyến chẩn đoán số lượng và vị trí các lỗi cứng, và phương pháp này bao gồm ECC, một bus vòng tránh thanh chéo (crossbar bypass bus) và hoán đổi cổng.
NUDA ^[75]	Gỡ lỗi (phát hiện tranh chấp)	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	NUDA là một khung gỡ lỗi không xâm lấn cho các hệ thống nhiều lõi, hoạt động song song với kết nối dữ liệu gốc, nhờ đó cho phép các phương pháp gỡ lỗi "không xâm lấn".
HARD ^[76]	Gỡ lỗi (phát hiện tranh chấp)	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Thuật toán lockset được sử dụng trong phần cứng nhằm khai thác năng lực phát hiện tranh chấp của thuật toán này bằng cách sử dụng các bộ lọc Bloom.
Non-INTER ^[77]	Gỡ lỗi và hiệu năng	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Dựa trên một kiến trúc giám sát không can thiệp, non-INTER giám sát quá trình thực thi của các hệ thống thời gian thực phân tán mà không can thiệp vào quá trình thực thi của chúng, bằng cách sử dụng phần cứng bổ sung để thu thập thông tin trạng thái và hỗ trợ việc giám sát.
TrustGuard ^[54]	Bảo mật	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	TrustGuard làm cho Sentry tương thích với nhiều loại thiết bị mà không đòi hỏi sửa đổi phần cứng của hệ thống chủ, đồng thời cho phép nhà phát triển tùy chọn sử dụng mã tin cậy để xác minh việc thực thi của mã không tin cậy, nhờ đó giảm kích thước của cơ sở mã tin cậy.
Jintide® ^[78]	Bảo mật	Trung gian	Đa lỗi	Các hành vi	Không xâm lấn	Jintide® giám sát các lỗi Xeon tại thời gian chạy bằng một chip bộ xử lý

						tính toán cấu hình lại được (RCP). Jintide® nắm bắt các đặc trưng cụ thể bằng cách ghi lại và phân tích các hành vi ở mức vi kiến trúc vốn trong suốt đối với phần mềm.
Các cách tiếp cận dựa trên kênh kè ^{[14][15]}	Bảo mật	Phần cứng và phần mềm (tùy chọn)	Đơn lõi/đa lõi	Các hành vi	Không xâm lấn	Các cách tiếp cận này giám sát hành vi phần cứng bằng cách sử dụng thông tin kênh kè để phát hiện Trojan phần cứng.
Jintide® là nhãn hiệu của sản phẩm do Montage cung cấp. Thông tin này được đưa ra nhằm tạo thuận lợi cho người sử dụng tiêu chuẩn này và không phải là sự xác nhận của ISO hay IEC đối với sản phẩm được nêu tên. Có thể sử dụng các sản phẩm tương đương nếu chứng minh được rằng chúng cho kết quả tương tự.						

7.5.4 So sánh giải pháp phần mềm và giải pháp có hỗ trợ phần cứng

Việc phát hiện hành vi phần cứng bằng phần mềm có nhiều hạn chế:

- Phần mềm thu thập không thể ghi lại mọi truy cập bộ nhớ của CPU và chỉ có thể sử dụng độ chi tiết ở mức trang, do đó gây ra chi phí phụ trội (overhead) lớn về hiệu năng và không gian bộ nhớ, làm hạn chế đáng kể phạm vi sử dụng.
- Phần mềm thu thập cũng không thể ghi lại thao tác DMA của thiết bị truyền dẫn trong suốt (transparent transmission device) và chỉ có thể thực hiện việc ghi sự kiện chính xác trên thiết bị tương tự (analogue device).
- Ở phía phát lại, việc sử dụng kiến trúc khác nhau gây ra dương tính giả, còn việc sử dụng cùng kiến trúc lại dẫn đến vấn đề mức độ tin tưởng (confidence) không đủ.

Phần mềm có thể được kết hợp với các mô-đun phần cứng để đạt mức độ tin tưởng cao hơn cùng hiệu năng cao hơn. Cụ thể là, phần mềm thu thập trạng thái CPU, phần cứng ghi lại các truy cập bộ nhớ và I/O; trên cơ sở đó, một khung giám sát phần cứng được thực hiện thông qua các lõi phần cứng khác chạy trình mô phỏng lệnh.

7.6 Thực thể đích

7.6.1 Lỗi IP

7.6.1.1 Khái quát

Bảo mật của các lỗi IP xuất phát từ hai khía cạnh:

- Để duy trì uy tín thương mại, các nhà cung cấp IP đưa ra các giải pháp chống làm giả IP (IP anti-counterfeiting) nhằm xác minh tính hợp pháp của chúng.
- Người sử dụng lỗi IP bảo đảm rằng các IP của bên thứ ba được nhúng vào sẽ không gây ra rủi ro về an toàn.

7.6.1.2 Chống sao chép trái phép IP (IP anti-piracy)

Các giải pháp chống làm giả IP có thể được chia đại thể thành bốn chiến lược: làm rối (obfuscation), đánh dấu chìm (watermarking), lấy dấu vân tay (fingerprinting) và đo đếm (metering). Việc chống làm giả IP chỉ có thể áp dụng cho các IP có nguồn gốc từ những nhà cung cấp hợp pháp, và giải pháp này dựa trên uy tín kinh doanh cùng các phương pháp xác minh của chính nhà cung cấp. Do đó, trong trường hợp yêu cầu bảo mật cao, người sử dụng có thể bổ sung một bộ giám sát phần cứng để xác minh sự tuân thủ của IP bên thứ ba.

7.6.1.3 Giám sát sự tuân thủ của IP

Các IP của bên thứ ba thường được tích hợp vào SoC với vai trò chức năng bảo mật lõi, chẳng hạn như một khối mã hóa (encryption engine). Bảo mật của chính chúng quyết định trực tiếp mức độ bảo mật của toàn hệ thống. Cũng tồn tại các rủi ro tiềm ẩn là những chức năng độc hại, ví dụ Trojan phần cứng, có thể bị chèn vào các lõi IP. Đặc biệt, do các ràng buộc về chi phí và thời gian, các lõi IP không có chuỗi xác thực hoàn chỉnh được sử dụng trong nhiều tình huống. Vì vậy, việc bảo đảm các lõi IP hoạt động an toàn, không có cửa hậu bảo mật, cũng như các công nghệ giám sát phần cứng bảo đảm không có chức năng độc hại là rất quan trọng.

7.6.1.4 Bảo đảm tính toàn vẹn phần cứng

Trong phạm vi vận hành phần cứng hậu silicon, các phương pháp như chứng thực tại thời gian chạy trở nên thiết yếu để khẳng định rằng các thành phần phần cứng không bị thay đổi trái phép trong chu kỳ thực thi của chúng. Cách tiếp cận này là một phần của các chiến lược giám sát phần cứng nhằm bảo đảm tính toàn vẹn của IP bên thứ ba, bổ sung hiệu quả cho các giải pháp chống làm giả do nhà cung cấp IP đưa ra. Cách xác minh như vậy trở nên đặc biệt quan trọng trong các ứng dụng bảo mật cao, nơi bất kỳ sự can thiệp trái phép nào cũng có thể dẫn đến sự cố nghiêm trọng cho toàn hệ thống.

7.6.1.5 Xác minh và xác nhận lõi IP

Các cơ chế xác minh và xác nhận lõi IP đóng vai trò trung tâm trong việc xác lập tính đúng đắn về chức năng, hiệu năng và các đặc tính bảo mật của IP bên thứ ba. Các cơ chế này có thể bao gồm các phương pháp xác minh hình thức, mô phỏng và giả lập phần cứng. Tương tự các giải pháp chống làm giả, những quy trình này phụ thuộc nhiều vào các phương pháp xác minh do nhà cung cấp hợp pháp đưa ra. Chúng bảo đảm sự an toàn trong vận hành của các lõi IP, ngăn ngừa mọi cửa hậu bảo mật tiềm ẩn có thể tồn tại.

7.6.1.6 Giám sát lõi IP tại thời gian chạy

Giám sát phần cứng theo thời gian thực mang lại sự bảo đảm liên tục cho hoạt động bảo mật của các lõi IP. Các công nghệ có khả năng thu nhận và phân tích dữ liệu theo thời gian thực đặc biệt có giá trị trong bối cảnh này. Chúng có thể quan sát các biến động về mức tiêu thụ điện năng, phân tích định thời hoặc các chỉ số hiệu năng để nhận diện bất kỳ hoạt động bất thường nào có thể là dấu hiệu của một vi phạm bảo mật. Do các lõi IP thường đảm nhiệm các chức năng bảo mật lõi trong SoC, chẳng hạn như khối mã hóa, việc giám sát tại thời gian chạy có thể mang ý nghĩa then chốt trong việc duy trì bảo mật của toàn hệ thống, đặc biệt khi chuỗi xác thực của các lõi IP không hoàn chỉnh.

7.6.2 Các đơn vị xử lý

Các đơn vị xử lý có thể được chia thành bộ xử lý đơn, đa bộ xử lý và mạng trên chip (NoC) theo số lượng lõi. Việc giám sát bộ xử lý đơn chủ yếu xuất hiện trong các tài liệu thời kỳ đầu.^[67] Với sự phát triển nhanh chóng của công nghệ chế tạo chip và sự phát triển mạnh mẽ của điện toán đám mây, Internet vạn vật và các lĩnh vực khác, nhiều nghiên cứu tập trung chủ yếu vào đa bộ xử lý^{[18][29]} và NoC^[79] nhằm giải quyết các vấn đề gỡ lỗi, chịu lỗi và bảo mật trong các kịch bản đa lõi.

7.6.3 Bộ nhớ

Bộ nhớ gồm hai loại: khả biến và không khả biến. Đại diện của bộ nhớ khả biến là RAM. Do tính chất chỉ đọc của ROM, bộ nhớ không khả biến chỉ được thảo luận ở phạm vi các thành phần của SoC (Flash/EEPROM, v.v.). Bộ nhớ không khả biến nằm ngoài hệ thống (chẳng hạn ổ đĩa cứng hoặc USB) được thảo luận ở phạm vi thiết bị ngoại vi.

Bộ nhớ truy cập ngẫu nhiên (RAM) là bộ nhớ truy cập ngẫu nhiên khả biến với tốc độ truy cập cao và thời gian đọc, ghi bằng nhau, không phụ thuộc vào địa chỉ. RAM có thể được chia thành RAM tĩnh (SRAM) và RAM động (DRAM). SRAM lưu dữ liệu tạm thời sinh ra trong quá trình hoạt động của máy tính đơn chip. SRAM có tốc độ rất cao. SRAM hiện là thiết bị lưu trữ có tốc độ đọc, ghi nhanh nhất. SRAM thường được dùng làm bộ nhớ đệm tốc độ cao, chẳng hạn bộ đệm cấp một và cấp hai của CPU; DRAM lưu giữ dữ liệu trong thời gian ngắn. DRAM chậm hơn, rẻ hơn và cung cấp dung lượng bộ nhớ máy tính lớn hơn SRAM.

Việc giám sát SRAM tập trung vào hai khu vực chính: bộ nhớ đệm của CPU và SRAM dùng trong SoC. RG-Secure thu thập thông tin bộ nhớ đệm bằng cách giám sát dữ liệu của bộ điều khiển bus, nhờ đó có thể xử lý dữ liệu động của SRAM trong các kịch bản đa lỗi. FDR^[53] nhận diện tranh chấp bộ nhớ bằng cách nhúng một trường đặc biệt (VIC/CIC) vào bộ nhớ đệm.

Các bộ giám sát DRAM được chia thành hai hình thức thực hiện: phần mềm và phần cứng. SMM^[80] là một giải pháp thực hiện bằng phần mềm, cung cấp thông tin chi tiết và giao diện đồ họa về các chương trình đang chạy bên trong máy tính. Khi được thực hiện bằng phần cứng, SMM có thể mang lại hiệu năng và độ an toàn tốt hơn. Jintide^[78] sử dụng phần cứng chuyên dụng, cụ thể là MTR, để thu thập dữ liệu DRAM một cách hiệu quả. Jintide có thể chỉ thu thập phần bộ nhớ thay đổi giữa các điểm kiểm tra theo nhu cầu, nhờ đó giảm đáng kể kích thước nhật ký.

7.6.4 Thiết bị ngoại vi

Thiết bị ngoại vi được chia thành hai loại: đầu vào và đầu ra. Các thiết bị này xử lý đầu vào/đầu ra của dữ liệu hoặc tín hiệu, bao gồm các thiết bị lưu trữ không khả biến (chẳng hạn vật mang dữ liệu) và các thiết bị đầu vào/đầu ra cho các sự kiện không tất định (chẳng hạn đầu vào bàn phím của người dùng và các tín hiệu ngắt/bất thường).

Việc giám sát thiết bị ngoại vi được thực hiện trực tiếp hoặc gián tiếp dựa trên giao diện I/O (I2C/SPI/PCIe, v.v.) do ranh giới hệ thống cung cấp. Các phương pháp thu thập trực tiếp bao gồm thu thập dữ liệu bus ngoại vi. Các phương pháp gián tiếp bao gồm sử dụng chức năng giám sát do công nghệ hypervisor cung cấp để thu thập thông tin ngắt và dữ liệu.

FDR^[53] ghi nhật ký DMA bằng cách mô hình hóa các giao diện DMA như những bộ xử lý giả và áp dụng các kỹ thuật ghi nhật ký tranh chấp bộ nhớ. Jintide^[78] sử dụng phần cứng chuyên dụng ITR để thu thập thông tin I/O.

Hypervisor còn được gọi là VMM lưu trú (hosted VMM), vì hypervisor chạy như một ứng dụng trong một hệ điều hành thông thường được gọi là hệ điều hành chủ. Hệ điều hành chủ không nhận biết VMM và coi VMM như mọi tiến trình khác. Hệ điều hành chủ thường thực hiện I/O thay mặt cho hệ điều hành khách. Hệ điều hành khách phát ra yêu cầu I/O, yêu cầu này tạo ra một bẫy (trap) để hệ điều hành chủ xử lý. Hệ điều hành chủ sau đó gửi yêu cầu I/O đến trình điều khiển thiết bị thực hiện I/O. Yêu cầu I/O hoàn tất lại được chuyển ngược về hệ điều hành khách thông qua hệ điều hành chủ.

7.7 Các mẫu mục tiêu (objective patterns)

7.7.1 Nội dung thông tin

Giám sát dựa trên nội dung thông tin tập trung vào luồng thông tin giữa các thành phần bên trong hệ thống hoặc giữa hệ thống với bên ngoài, bao gồm định thời liên quan đến truyền thông và các dữ liệu giao thức khác, cùng các nhật ký kiểm toán liên quan đến logic nghiệp vụ tầng trên. Loại bộ giám sát này chú trọng vào bản thân thông tin và phản ánh tình trạng vận hành của hệ thống thông qua việc giám sát nội dung.

Thông tin liên quan đến giao thức truyền thông có thể được thu thập bằng máy phân tích logic hoặc máy hiện sóng, hoặc trực tiếp sử dụng thành phần trung gian bằng cách chặn bắt trên bus. Đối với thông tin liên quan đến nghiệp vụ, có thể dùng phần mềm để ghi lại thông tin nhật ký.

7.7.2 Đặc tính vật lý

Việc giám sát các đặc tính vật lý của phần cứng bao gồm thu thập theo thời gian thực các dữ liệu vật lý của phần cứng, chẳng hạn dòng điện, điện áp, mức tiêu thụ điện năng, phát xạ điện từ, thời gian tính toán và nhiệt, thông qua cảm biến, đầu dò hoặc phần cứng trung gian. Các mục đích chính của giám sát đặc trưng vật lý bao gồm kiểm soát theo ngưỡng, tối ưu hóa hiệu năng, chịu lỗi, QoS và giám sát bảo mật (để phát hiện Trojan phần cứng, v.v.).

Từ góc độ chuỗi cung ứng, việc giám sát các đặc tính vật lý đóng vai trò quan trọng trong mọi mắt xích. Ở giai đoạn thiết kế nội bộ, giám sát đặc tính vật lý có thể giúp xác định tính hợp pháp của IP bên thứ ba. Ở giai đoạn kiểm thử, giám sát đặc tính vật lý có thể giúp xác minh rằng các mô-đun then chốt có khả năng kháng nhất định trước phân tích kênh kề. Ở giai đoạn hậu silicon, giám sát đặc tính vật lý có thể bảo đảm môi trường vận hành ổn định và hiệu năng vượt trội.

Các đặc trưng vật lý có thể được rút ra từ các giá trị chuẩn (golden values) hoặc được điều chỉnh động bằng các thuật toán học máy và trí tuệ nhân tạo.

7.7.3 Các hành vi

Các bộ giám sát dựa trên hành vi nhấn mạnh sự tuân thủ logic ở mức cao. Ví dụ, Trojan phần cứng có thể được phát hiện thông qua việc phân tích hành vi của phần cứng. Khác với các bộ giám sát dựa trên nội dung thông tin và dựa trên đặc tính vật lý, các bộ giám sát dựa trên hành vi có một đặc trưng quan trọng về thời gian chạy.

Hành vi phần cứng bị ràng buộc bởi môi trường hệ thống nơi thực thể tồn tại. Việc giám sát hành vi thường bao gồm phân tích tổng thể toàn bộ hệ thống và không giới hạn ở các thành phần bảo mật quan trọng.

7.8 Phương pháp triển khai

7.8.1 Khái quát

Việc triển khai giám sát cần xem xét:

- a) tính xâm lấn;
- b) việc triển khai là ngoại tuyến hay trực tuyến;
- c) việc triển khai là đồng bộ hay không đồng bộ; và
- d) có một hay nhiều bộ giám sát.

7.8.2 Tính xâm lấn

Tính xâm lấn được phản ánh qua tác động về thời gian và không gian đối với hệ thống đích đang chạy. Trong một số tình huống phức tạp, các bộ giám sát được thiết kế kém thậm chí có thể gây ra lỗi vận hành và rủi ro bảo mật. Xét về chi phí, tính xâm lấn thấp là một trong những mục tiêu chính của thiết kế bộ giám sát. Ưu điểm của bộ giám sát dựa trên phần mềm là mang lại tính linh hoạt tốt. Khi có đủ tài nguyên, một bộ giám sát chạy song song với hệ thống đích có thể thể hiện tính không xâm lấn tốt. Tuy nhiên, đối với các hệ thống nhúng, giải pháp thuần phần mềm thường gây ra chi phí phụ trội đáng kể. Ưu điểm của bộ giám sát dựa trên phần cứng là, nhờ hiệu năng tốt của phần cứng giám sát song song, chúng có thể mang lại tính xâm lấn thấp hơn. Tuy nhiên, so với các giải pháp phần mềm, giải pháp phần cứng đòi hỏi đầu tư nhiều hơn về kinh phí phát triển và chu kỳ phát triển, đồng thời tính linh hoạt cũng bị hạn chế. Các giải pháp có hỗ trợ phần cứng cấu hình được dựa trên FPGA đã dung hòa ở mức độ nhất định các ưu điểm về tính linh hoạt và chi phí phụ trội thấp. Việc cân bằng giữa chi phí, tính linh hoạt và mức xâm lấn thấp có thể là một vấn đề trong giai đoạn triển khai bộ giám sát.

7.8.3 Ngoại tuyến hay trực tuyến

Các phương pháp ngoại tuyến có thể đạt tính xâm lấn thấp hơn các phương pháp trực tuyến vì thời gian xử lý có thể được trì hoãn. Tuy nhiên, khác với các giải pháp trực tuyến, các giải pháp ngoại tuyến phải lưu trữ các bản ghi lấy mẫu hoặc theo vết, do đó phát sinh thêm chi phí phụ trội về không gian. Các phương pháp triển khai trực tuyến phù hợp hơn với các hệ thống thời gian thực. Khi giám sát các hành vi bất thường, phương pháp này có thể cung cấp khả năng kiểm soát bảo mật kịp thời. Tuy nhiên, cách tiếp cận trực tuyến hoặc làm tăng chi phí hoặc gây ra chi phí phụ trội lớn hơn.

7.8.4 Đồng bộ hay không đồng bộ

Để dung hòa ưu điểm và nhược điểm của các phương pháp ngoại tuyến và trực tuyến, có thể sử dụng bộ giám sát không đồng bộ nhằm đạt được sự cân bằng nhất định giữa độ trễ thời gian và tính kịp thời của việc phát hiện. So với giải pháp đồng bộ, giải pháp không đồng bộ gây ra một độ trễ nhất định, nhưng với điều kiện độ trễ nằm trong phạm vi chấp nhận được của hệ thống, giải pháp giám sát không đồng bộ là giải pháp tốt hơn với tính xâm lấn thấp hơn.

7.8.5 Một hay nhiều bộ giám sát

Cho đến nay, chiến lược xử lý tập trung với một bộ giám sát duy nhất^[80] vẫn phổ biến hơn, nhưng với độ phức tạp ngày càng tăng của các hệ thống SoC và NoC, các giải pháp đa bộ giám sát phân tán đã dần xuất hiện. Tài liệu tham khảo [81] chia các giải pháp đa bộ giám sát thành hai loại: bộ giám sát điều phối tập trung (orchestrated) và bộ giám sát phối hợp phân tán (choreographed). Điều phối tập trung dựa vào một thực thể điều phối duy nhất để thu thập, sắp xếp và phân tích các sự kiện, trong khi cách tiếp cận phối hợp phân tán phân bổ các nhiệm vụ này cho nhiều bộ giám sát.^{[82][83]} Trong khi điều phối tập trung thường dễ tổng hợp hơn và do đó dễ đạt tính đúng đắn hơn, phối hợp phân tán lại phù hợp hơn với các đặc tính của tính toán phân tán, nhờ đó dẫn đến lưu lượng mạng thấp hơn và mức độ chịu lỗi cao hơn.^[84]

7.8.6 Khả năng mở rộng

Khả năng mở rộng là một yếu tố quan trọng cần cân nhắc đối với các hệ thống giám sát phần cứng khi độ phức tạp và yêu cầu giám sát của các hệ thống hiện đại không ngừng tăng lên. Một giải pháp giám sát thiếu khả năng mở rộng có thể trở nên không đáp ứng được số lượng ngày càng tăng của các thành phần, thiết bị hoặc điểm dữ liệu cần giám sát. Để duy trì khả năng mở rộng, việc thiết kế các hệ thống giám sát phần cứng theo kiến trúc mô-đun là thiết yếu, vì kiến trúc này sẽ tạo thuận lợi cho việc mở rộng và thích ứng dễ dàng. Chẳng hạn, tài liệu tham khảo [42] trình bày một khung giám sát phần cứng có khả năng mở rộng, tận dụng việc thu thập và xử lý dữ liệu phân tán để đáp ứng các yêu cầu giám sát trên các hệ thống phân tán quy mô lớn.

7.8.7 Khả năng phục hồi và dự phòng

Thiết kế của các hệ thống giám sát phần cứng ưu tiên khả năng phục hồi và dự phòng nhằm bảo đảm giám sát liên tục, ngay cả khi gặp sự cố hoặc gián đoạn. Thời gian ngừng hoạt động hoặc trục trặc của hệ thống giám sát có thể dẫn đến mất dữ liệu quan trọng, gây ra những hậu quả nghiêm trọng. Một cách tiếp cận khả thi cho tình huống này là thực hiện các cơ chế lưu trữ dữ liệu dự phòng và chuyển đổi dự phòng. Như minh họa trong tài liệu tham khảo [43], phương pháp này sử dụng kiến trúc giám sát dự phòng để duy trì việc thu thập và phân tích dữ liệu không bị gián đoạn.

7.8.8 Tính tương thích

Việc tích hợp liền mạch các giải pháp giám sát thường phụ thuộc vào tính tương thích của chúng với các hệ thống phần cứng và phần mềm hiện có. Về lý tưởng, thiết kế của các hệ thống này tạo thuận lợi cho việc tương tác với nhiều loại cấu hình phần cứng và môi trường phần mềm khác nhau, qua đó tránh mọi xung đột hoặc gián đoạn. Một ví dụ có thể thấy trong tài liệu tham khảo [95], trong đó trình bày chi tiết một kiến trúc hệ thống trên chip (SoC) tích hợp nhiều bộ xử lý với các khối bộ nhớ khác. Thiết kế như vậy nâng cao khả năng thích ứng của hệ thống với nhiều cấu hình phần cứng khác nhau, thúc đẩy việc triển khai thuận lợi trong nhiều loại hệ thống.

7.8.9 Tác động đến hiệu năng

Về lý tưởng, ảnh hưởng của việc giám sát phần cứng đến hiệu năng của hệ thống được quan sát được giữ ở mức thấp nhất có thể nhằm tránh can thiệp vào hoạt động bình thường của hệ thống. Cách thiết kế tốt nhất cho các hệ thống giám sát là bảo đảm độ chính xác cao để tạo gánh nặng ít nhất có thể lên tài nguyên của hệ thống đích. Chẳng hạn, tài liệu tham khảo [112] nghiên cứu các phương pháp giám sát phần cứng hạng nhẹ với tác động gần như không đáng kể đến hiệu năng của CPU, nhờ đó cho phép giám sát liên tục mà không làm giảm khả năng đáp ứng của hệ thống.

7.8.10 Các quy định và yêu cầu về xử lý dữ liệu hợp pháp và có đạo đức

Việc các hệ thống giám sát phần cứng tuân thủ các quy định pháp luật và hướng dẫn đạo đức cho phép xử lý dữ liệu một cách hợp pháp và có đạo đức. Khi quyền riêng tư và bảo mật dữ liệu ngày càng trở thành mối quan tâm nổi bật, việc ưu tiên bảo vệ dữ liệu thu thập được và sử dụng dữ liệu một cách có trách nhiệm trở nên đặc biệt quan trọng. Chẳng hạn, tài liệu tham khảo [113] nhấn mạnh giá trị của việc tuân thủ các quy định bảo vệ dữ liệu và việc thực hiện các kỹ thuật bảo vệ quyền riêng tư trong các hệ thống giám sát phần cứng, giúp duy trì niềm tin của người dùng.

8 Sử dụng các công nghệ giám sát để đánh giá bảo mật phần cứng

8.1 Các cách tiếp cận đánh giá bảo mật tiên tiến hiện có

Các giai đoạn của các cách tiếp cận đánh giá bảo mật phần cứng hiện có có thể được phân thành giai đoạn tiền silicon và giai đoạn hậu silicon. Giai đoạn tiền silicon được dùng để định hướng việc đáp ứng các yêu cầu an toàn cụ thể trong khâu thiết kế, còn giai đoạn hậu silicon được dùng chủ yếu để đánh giá các rủi ro an toàn của sản phẩm.

Các phương pháp dùng trong giai đoạn tiền silicon có thể được chia thành phương pháp từ trên xuống và phương pháp từ dưới lên nhằm định hướng khâu thiết kế phần cứng đáp ứng các yêu cầu an toàn. R. Huang và các cộng sự^[85] đề xuất một sơ đồ đánh giá bảo mật phần cứng cung cấp cách thức có hệ thống để đo lường và phân loại các mối quan ngại bảo mật của các tính năng phần cứng. Sơ đồ đề xuất tập trung vào việc đo lường bảo mật trong giai đoạn thiết kế ban đầu bằng cách tận dụng kinh nghiệm của các chuyên gia bảo mật và kiến thức thiết kế của những người phụ trách tính năng. Sơ đồ đánh giá sử dụng định dạng bảng hỏi hai cấp để chấm điểm và phân loại các phép đo bảo mật.

B. Sherman và các cộng sự^[86] đề xuất cách tiếp cận thiết kế từ dưới lên, chuyển từ cách tiếp cận lấy sản phẩm làm trung tâm sang cách tiếp cận dựa trên IP, qua đó cải thiện hiệu quả việc tái sử dụng các thành phần chung của nhiều sản phẩm. Kết quả của cách tiếp cận này là công cụ Đánh giá rủi ro bảo mật (SRA), dùng để hướng dẫn một nhóm công tác IP tiến hành các đánh giá bảo mật.^[86] Công cụ SRA chia quy trình đánh giá bảo mật thành hai lớp: bộ lọc mức cao và đánh giá mức thấp. Mục đích của bộ lọc mức cao là nhận diện nhanh các IP, và nếu không có mối quan ngại bảo mật đáng kể, các IP đó được loại khỏi diện xem xét vì được coi là có rủi ro tối thiểu. Đánh giá mức thấp bao gồm một tập câu hỏi được tổ chức thành các nhóm chủ đề trọng tâm, trong đó mỗi câu hỏi được gán một mức rủi ro và một điểm số có trọng số.

Hasegawa và các cộng sự^{[12][13]} đề xuất một phương pháp phát hiện Trojan phần cứng dựa trên học máy cho các netlist mức cổng logic bằng cách sử dụng mạng nơ-ron nhiều tầng. Các tác giả đã trích xuất 11 đặc trưng cho mỗi net trong một netlist. Sau đó, các tác giả phân loại các net trong một netlist chưa biết thành tập các net Trojan và tập các net bình thường bằng mạng nơ-ron nhiều tầng.

Ở giai đoạn hậu silicon, việc đánh giá bảo mật được thực hiện trên sản phẩm hoàn chỉnh. CC EAL (Tiêu chí chung, Mức bảo đảm đánh giá) dựa vào bộ đặc tả CC để kiểm định mức độ của sản phẩm cụ thể và thiết lập cơ chế thừa nhận lẫn nhau như một sự bảo đảm an toàn cho việc lưu hành sản phẩm. Bộ đặc tả CC định nghĩa khung bảo mật thông tin tổng thể với hai cấp độ: chức năng bảo mật và bảo đảm bảo mật.

Các ngành công nghiệp khác nhau có những yêu cầu đánh giá bảo mật phần cứng khác nhau tùy theo đặc thù của mình. Ví dụ, Hiệp hội Kỹ sư Ô tô quốc tế (SAE International) đang tích cực tham gia xây dựng các bộ tiêu chuẩn, chẳng hạn AS6171A.^[104] SAE đã đưa ra một bộ phương pháp kiểm định để phát hiện linh kiện giả mạo bằng cách kiểm định các thuộc tính vật lý và thuộc tính điện của các linh kiện khác nhau.^[87] SAE sử dụng các công cụ hình ảnh và công cụ đo lường vật lý để đánh giá bảo mật phần cứng.

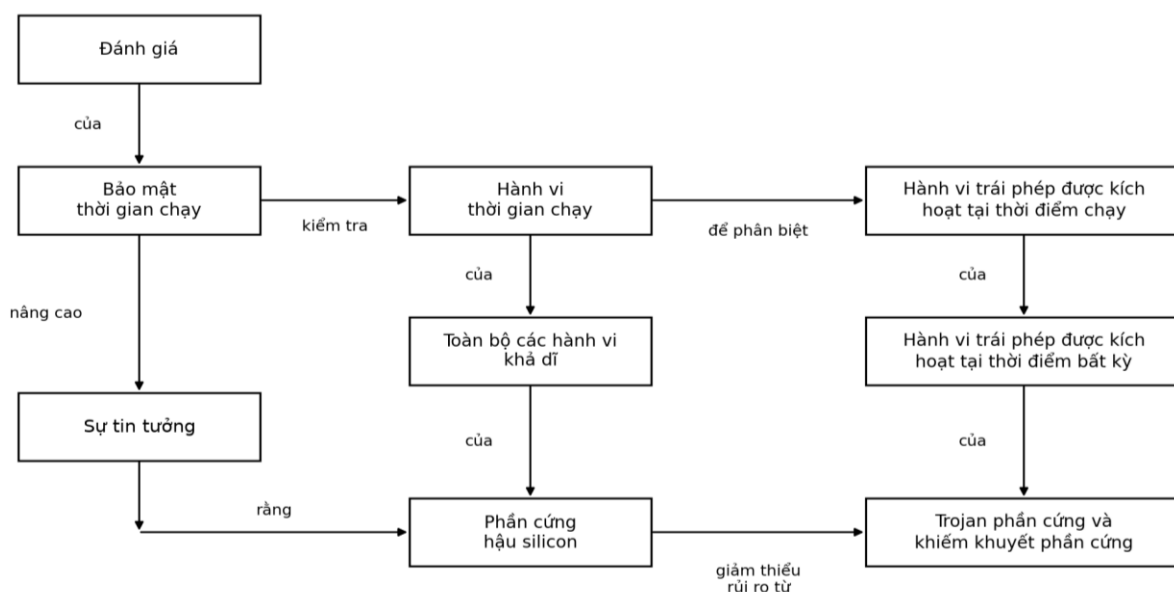
Do tính nhạy cảm của một số lĩnh vực (hàng không, quân sự, dân sinh, v.v.), các quốc gia khác nhau đã đưa ra những yêu cầu đánh giá và chứng nhận phần cứng nội địa, chẳng hạn NIST FIPS 140-2 (Hoa Kỳ).^[105] Trong khi đó, ISO/IEC 19790 quy định rằng các sản phẩm đã được chứng nhận có thể được sử dụng trong các cơ quan cụ thể. Để cấp các chứng nhận này, các phòng thử nghiệm được ủy quyền tiến hành kiểm định và xác minh trên các mẫu nộp để kiểm tra, đối chiếu với từng yêu cầu an toàn cụ thể, rồi lập báo cáo đánh giá an toàn trình cơ quan có thẩm quyền xem xét.

8.2 Giám sát phần cứng có thể hỗ trợ như thế nào

Khi phân tích tính đầy đủ của các biện pháp đối phó cho các vấn đề bảo mật như Trojan phần cứng hoặc khiếm khuyết phần cứng, mục tiêu chủ yếu của một phương pháp luận đánh giá bảo mật là đánh giá bảo mật dựa trên hành vi phần cứng tại thời gian chạy. Việc đánh giá được thực hiện trong các khoảng thời gian rời rạc gọi là các khoảng giám sát.

Với mục đích này, khái niệm bảo mật dựa trên hành vi phần cứng tại thời gian chạy được định nghĩa là một cách tiếp cận chủ động đối với bảo mật cho phần cứng trong giai đoạn hậu silicon, trong đó hoạt động thời gian chạy có liên quan của phần cứng được giám sát sao cho các sai lệch so với các mẫu hành vi bình thường có thể được nhận diện và xử lý nhanh chóng.

Theo [Hình 2](#), việc đánh giá bảo mật phần cứng tại thời gian chạy kiểm tra các hành vi thời gian chạy để phân biệt các hoạt động trái phép hoặc không mong muốn trên phần cứng hậu silicon đang chạy. Vì các hành vi thời gian chạy là một tập con của tất cả các hành vi khả dĩ, và các hành vi được kích hoạt tại thời gian chạy là một tập con của các hành vi được kích hoạt tại thời điểm bất kỳ, việc đánh giá bảo mật thời gian chạy trên phần cứng hậu silicon đích có tiềm năng nâng cao sự tin tưởng rằng các rủi ro từ các mối đe dọa, chẳng hạn Trojan phần cứng hoặc khiếm khuyết phần cứng, đã được giảm thiểu.



Hình 2 — Bảo mật dựa trên hành vi phần cứng tại thời gian chạy — các khái niệm và mối quan hệ

Một đặc trưng rõ ràng của Trojan phần cứng là các thao tác (hành vi) của phần cứng chủ có thể dẫn đến trạng thái không mong muốn trái với các đặc tả gốc khi Trojan được kích hoạt. Ví dụ, thao tác có thể tạo ra các thanh ghi vi phạm sở tay hướng dẫn lệnh hoặc giao thức để sinh ra một cuộc tấn công dẫn đến lỗi nghiêm trọng, hoặc một bí mật nội bộ có thể bị đưa trái phép ra ngoài tới các chân đệm (pad) qua một đường vòng. Tuy nhiên, các điều kiện kích hoạt và hành vi tương ứng của phần lớn khiếm khuyết liên quan đến lỗi chức năng và lỗi kênh kẻ có thể được tra cứu để có tri thức tiên nghiệm. Ví dụ, nếu phát hiện một chương trình thường xuyên đo độ trễ của một địa chỉ bộ nhớ mà lẽ ra chỉ được CPU truy cập khi thực thi suy đoán, có thể kết luận rằng đây là một phần của cuộc tấn công kênh kẻ qua bộ nhớ đệm CPU.

Hiện nay, các công nghệ giám sát phần cứng đã được phát triển cho nhiều mục đích. Các công nghệ này được sử dụng rộng rãi để thu thập thông tin động thể hiện những thay đổi tại thời gian chạy về thuộc tính, nội dung, trạng thái bên trong hoặc biểu hiện bên ngoài của đối tượng đích. Bằng việc phân tích sâu hơn các thông tin đó, có thể trích xuất các hoạt động có ý nghĩa của đối tượng đích tại thời gian chạy, và các hoạt động này có thể được nhận diện là các hành vi.

8.3 Thách thức

Cũng tồn tại những thách thức khi áp dụng các công nghệ giám sát. Ví dụ về các thách thức đó được liệt kê dưới đây.

— Giám sát tại thời gian chạy có thể gây thêm chi phí phụ trội cho hệ thống gốc, và bảo mật của chính việc giám sát cũng quan trọng.

— Một mô-đun bên trong hoặc một tác nhân bên ngoài thực hiện việc giám sát, với vai trò một phần chức năng bổ sung của đối tượng đích, nhiều khả năng sẽ gây thêm chi phí phụ trội, do đó có thể làm giảm hiệu năng tổng thể.

— Chứng nhận cho phần cứng giám sát là yêu cầu thiết yếu để vận hành việc giám sát một cách tin cậy. Phần cứng giám sát có thể phối hợp với các thiết bị bên ngoài để đạt hiệu năng tốt hơn. Các mối đe dọa tiềm ẩn đối với phần cứng giám sát tồn tại trong cả hai trường hợp. Ví dụ, dữ liệu

giám sát có thể bị lộ ra bên ngoài. Do đó, việc chứng nhận dựa trên các yêu cầu bảo mật nhằm ứng phó các mối đe dọa này có thể được xem xét khi áp dụng phần cứng giám sát.

— Phần lớn các công nghệ giám sát phần cứng hiện có mang tính tùy biến theo tình huống (ad hoc). Việc tối ưu hóa bổ sung hoặc thiết kế lại phần cứng đích có thể là cần thiết cho các nhiệm vụ đánh giá tại thời gian chạy.

— Trong các trường hợp hiện có, các công nghệ giám sát trạng thái phần cứng nhiều khả năng đóng vai trò hỗ trợ thiết kế hơn là hỗ trợ đánh giá. Các công nghệ này được thiết kế để phục vụ các mục đích khác nhau (xem 6.2) với nhiều tính năng đa dạng, một số trong đó có thể không thuận tiện cho việc đánh giá. Một thách thức phổ biến là một số công nghệ này chỉ có thể được thực hiện bằng cách tiếp cận xâm lấn đối với mạch, trong khi tính tái sử dụng, tính cấu hình được và tính không xâm lấn lại là những đặc tính được mong muốn hơn cho việc đánh giá.

— Các thành phần phần cứng khác nhau thể hiện các thuộc tính, tính năng và hành vi rất khác nhau.

— Các tổ chức bên thứ ba hoặc người dùng có thể phải đối mặt với nhiều loại thành phần phần cứng khác nhau. Hiện nay, các tổ chức bên thứ ba hoặc người dùng muốn giám sát các thành phần phần cứng để đánh giá chúng phải thực hiện theo từng trường hợp cụ thể. Do đó, cần có một phương pháp luận tổng quát về đánh giá bảo mật tại thời gian chạy.

— Dương tính giả và âm tính giả ảnh hưởng đến hiệu quả giám sát phần cứng, gây lãng phí tài nguyên hoặc khiến hệ thống dễ bị tổn thương trước các mối đe dọa không được phát hiện.

— Dương tính giả và âm tính giả là những mối quan ngại nghiêm trọng trong các hệ thống giám sát phần cứng. Dương tính giả là các trường hợp hệ thống giám sát nhận diện nhầm một hoạt động bình thường là mối đe dọa, dẫn đến các cảnh báo không cần thiết và tiêu tốn tài nguyên. Ngược lại, âm tính giả xảy ra khi hệ thống không phát hiện được các mối đe dọa thực sự, khiến hệ thống có nguy cơ bị tổn hại. Cả hai tình huống đều có thể gây tác động bất lợi đến hiệu quả tổng thể của hệ thống giám sát.

— Các vấn đề này thường phát sinh do độ phức tạp của các thuật toán dùng để phát hiện mối đe dọa và tính bất định vốn có khi giám sát các hệ thống phần cứng quy mô lớn. Ví dụ, trong các hệ thống phát hiện xâm nhập (IDS), một lần khớp chữ ký sai hoặc phát hiện bất thường sai có thể kích hoạt dương tính giả. Tương tự, phần mềm độc hại tinh vi có thể né tránh các cơ chế phát hiện, dẫn đến âm tính giả.

— Việc giảm thiểu dương tính giả và âm tính giả đòi hỏi một sự cân bằng tinh tế giữa việc giảm các cảnh báo không chính xác và việc duy trì độ bao phủ vững chắc đối với các mối đe dọa. Bằng cách tích hợp các kỹ thuật học máy tiên tiến và huấn luyện liên tục với dữ liệu thực tế, độ chính xác của việc phát hiện mối đe dọa có thể được cải thiện đáng kể. Việc áp dụng các phương pháp tổ hợp (ensemble), kết hợp nhiều thuật toán phát hiện, hỗ trợ giảm thiểu dương tính giả và âm tính giả.

— Việc phân bổ tài nguyên kém hiệu quả cho năng lực xử lý và bộ nhớ trong các hệ thống giám sát phần cứng có thể ảnh hưởng đến hiệu năng tổng thể của hệ thống.

— Phân bổ tài nguyên là một khía cạnh quan trọng của các hệ thống giám sát phần cứng, đặc biệt trong các môi trường hạn chế về tài nguyên. Việc phân bổ năng lực xử lý và bộ nhớ kém hiệu quả có thể dẫn đến hiệu năng hệ thống dưới mức tối ưu, ảnh hưởng đến chức năng tổng thể và khả năng đáp ứng của hệ thống được giám sát.

— Trong các hệ thống nhúng hạn chế tài nguyên hoặc các thiết bị Internet vạn vật (IoT), việc vận hành hệ thống giám sát phần cứng có thể gặp nhiều khó khăn do tài nguyên hạn chế. Ví dụ, một hệ thống nhúng có thể khó đáp ứng các yêu cầu tính toán cao của việc giám sát và phân tích theo thời gian thực do năng lực xử lý bị hạn chế.

— Việc giải quyết thách thức này có thể bao gồm việc thực hiện các thuật toán hạng nhẹ và hiệu quả trong các hệ thống giám sát phần cứng. Một cách thực hiện là tận dụng các đơn vị xử lý tăng tốc bằng phần cứng như FPGA hoặc ASIC. Kỹ thuật này có thể giảm tải tính toán cho CPU chính. Ngoài ra, việc chú trọng tối ưu hóa lưu trữ và truyền dữ liệu có tiềm năng tiết kiệm tài nguyên bộ nhớ.

— Sự khan hiếm chuyên môn kỹ thuật cản trở việc thiết kế, triển khai và bảo trì hiệu quả các hệ thống giám sát phần cứng.

— Độ phức tạp của các hệ thống giám sát phần cứng đòi hỏi chuyên môn kỹ thuật chuyên sâu. Tuy nhiên, sự khan hiếm các chuyên gia lành nghề với kiến thức sâu về phần cứng, an ninh mạng và kiến trúc hệ thống có thể cản trở việc thực hiện và duy trì thành công các hệ thống như vậy.

— Các hệ thống giám sát phần cứng đòi hỏi hiểu biết sâu về nhiều lĩnh vực, bao gồm thiết kế phần cứng, hệ điều hành, mạng, mật mã và an ninh mạng. Chẳng hạn, việc thực hiện giám sát phần cứng bảo mật đòi hỏi kiến thức về mô hình hóa mối đe dọa, đánh giá rủi ro và các giao thức bảo mật.

— Để khắc phục sự khan hiếm chuyên môn kỹ thuật, các tổ chức có thể đầu tư vào các chương trình đào tạo và giáo dục cho nhân sự của mình. Ngoài ra, việc hợp tác với các cơ sở học thuật và các trung tâm đào tạo an ninh mạng có thể giúp lấp đầy khoảng trống kỹ năng và tạo nguồn nhân lực có trình độ.

— Việc quản lý vòng đời, đặc biệt trong các hệ thống nhúng, gây ra những phức tạp và chi phí khi cập nhật hoặc thay thế các thành phần.

— Các hệ thống giám sát phần cứng, đặc biệt trong các hệ thống nhúng, đối mặt với những thách thức riêng trong việc quản lý vòng đời các thành phần của chúng. Quá trình cập nhật hoặc thay thế các thành phần phần cứng có thể phức tạp, tốn kém và mất thời gian.

— Các hệ thống nhúng thường có tính mô-đun hạn chế, và các thành phần phần cứng có thể được tích hợp chặt chẽ với kiến trúc tổng thể của hệ thống. Ví dụ, việc thay thế một thành phần trọng yếu trong một hệ thống nhúng có thể đòi hỏi thiết kế lại và xác nhận lại đáng kể toàn bộ hệ thống.

— Để giải quyết các thách thức này, việc lập kế hoạch cẩn thận và có tầm nhìn xa ngay từ giai đoạn thiết kế ban đầu là rất quan trọng. Việc áp dụng các kiến trúc phần cứng mô-đun và bảo đảm tính tương thích của các thành phần với các bản cập nhật tương lai có thể giúp việc quản lý vòng đời dễ dàng hơn. Ngoài ra, các công nghệ cập nhật phần mềm từ xa và cập nhật qua mạng không dây (OTA) có thể tạo thuận lợi cho việc cập nhật các thành phần trong các hệ thống nhúng một cách hiệu quả và tiết kiệm chi phí hơn.

Hơn nữa, theo tài liệu tham khảo [88], các mối đe dọa bảo mật phần cứng được chia thành năm loại: Trojan phần cứng, tấn công kênh kề, sao chép trái phép IP, tấn công xâm nhập và dịch ngược. Vì bảo mật của chính bộ giám sát phần cứng cũng rất quan trọng, bộ giám sát đã được đưa vào hàng cuối cùng của [Bảng 3](#). Các mối đe dọa mới cũng có thể phát sinh từ bối cảnh mối đe dọa không ngừng biến đổi, do đó các kỹ thuật giám sát phần cứng phù hợp và chuyên biệt là hữu ích để phát hiện các mối đe dọa như vậy.

Bảng 3 — Giá trị và hạn chế của công nghệ giám sát phần cứng phục vụ đánh giá

Mối đe dọa [88]	Biện pháp phát hiện hoặc đối phó	Giá trị và hạn chế của công nghệ giám sát phần cứng phục vụ đánh giá
Trojan phần cứng	Xác minh chức năng[89]; kích hoạt Trojan phần cứng[90]; phân tích kênh kề[91]; thiết kế bảo mật[92]	Kiểm tra bằng cấu trúc dư thừa (redundant checks) và xác minh chức năng dựa trên phần cứng tin cậy. Nhận diện lệnh bất hợp pháp.
Tấn công kênh kề	Che giấu (hiding) và tạo mặt nạ (masking)[93]	Nhận diện hiệu quả các mẫu tấn công đối với các sự kiện lấy mẫu thường xuyên. Cập nhật nhanh các mẫu so khớp và chẩn đoán các lỗi hỏng bộ nhớ đệm như Meltdown và loạt lỗi hỏng Spectre.
Sao chép trái phép IP	Xác minh sự tin cậy[94]	Kiểm tra kết quả tính toán của các tài sản then chốt (chẳng hạn khóa mật mã). Nhận diện lệnh bất hợp pháp.
Dịch ngược	Làm rối phần cứng[92]	Không có.

Tấn công nhằm vào phần cứng giám sát	Mã hóa, kiểm soát truy cập, xác thực thực thể, bảo vệ vật lý phần cứng và tự kiểm thử tại thời gian chạy	Chứng nhận cho phần cứng giám sát.
--------------------------------------	--	------------------------------------

9 Chứng nhận cho phần cứng giám sát

Chứng nhận cho phần cứng giám sát là điều thiết yếu để sử dụng công nghệ giám sát phần cứng một cách tin cậy. Hai trường hợp sử dụng được xem xét. Trong trường hợp sử dụng thứ nhất, phần cứng giám sát theo dõi phần cứng đích và phát hiện phần cứng độc hại. Trong trường hợp thứ hai, phần cứng giám sát phối hợp với các thiết bị bên ngoài để đạt hiệu năng tốt hơn.

Trong trường hợp sử dụng thứ nhất, phần cứng giám sát thu thập thông tin kênh kề (ví dụ: mức tiêu thụ điện năng, phát xạ điện từ và thời gian tính toán) của phần cứng đích để kiểm tra trạng thái của phần cứng. Một thuật toán phát hiện được thực thi trên thiết bị giám sát có thể phát hiện các hành vi độc hại bằng cách sử dụng thông tin kênh kề. Tuy nhiên, việc phát hiện chỉ bằng phần cứng giám sát có thể gây ra phát hiện sai do tài nguyên tính toán hạn chế.

Trong trường hợp sử dụng thứ hai, phần cứng giám sát cung cấp thông tin kênh kề cho một thiết bị bên ngoài. Thuật toán phát hiện có thể được thực thi một phần hoặc toàn bộ trên các tài nguyên bên ngoài (ví dụ: thiết bị hoặc máy tính cá nhân) nhằm cải thiện độ chính xác của việc phát hiện.

Ví dụ về các mối đe dọa nhằm vào phần cứng giám sát bao gồm:

— Truy cập trái phép

Kẻ tấn công có thể truy cập (đọc, sửa đổi hoặc xóa) dữ liệu (ví dụ: các vết kênh kề và khóa mật mã của đối tượng được giám sát) trong phần cứng giám sát.

Kẻ tấn công cũng có thể truy cập dữ liệu được truyền hoặc xâm phạm bảo mật của phần cứng giám sát bằng cách theo dõi hoặc thao túng việc truyền thông giữa phần cứng giám sát và các tài nguyên bên ngoài. Điều này cũng bao gồm các cuộc tấn công tiêm lỗi.

— Cập nhật trái phép

Kẻ tấn công có thể cài đặt phần mềm trái phép (ví dụ: phần sụn) lên phần cứng giám sát.

— Trực trặc

Một trực trặc trong hoạt động giám sát có thể làm giảm bảo mật của phần cứng giám sát.

Để giảm thiểu rủi ro từ các mối đe dọa nêu trên, có thể sử dụng các biện pháp đối phó sau:

— Kiểm soát truy cập

Kiểm soát truy cập theo các chính sách bảo mật có thể cải thiện bảo mật của phần cứng giám sát trước truy cập trái phép.

— Xác thực thực thể

Việc xác thực một thực thể bên ngoài (ví dụ: người dùng hoặc tài nguyên bên ngoài) có yêu cầu truy cập vào phần cứng giám sát cũng cải thiện bảo mật của phần cứng giám sát trước truy cập trái phép.

— Toàn vẹn và xác thực dữ liệu

Việc xác thực nguồn gốc dữ liệu và xác nhận tính toàn vẹn của dữ liệu có thể cải thiện bảo mật của phần cứng giám sát trước các cập nhật trái phép.

— Tính bí mật

Việc mã hóa dữ liệu được lưu trữ trong, gửi đi từ hoặc gửi đến phần cứng giám sát có thể bảo đảm tính bí mật và cải thiện bảo mật của phần cứng giám sát trước truy cập trái phép. Các mô-đun bảo mật phần cứng (HSM) và mô-đun nền tảng tin cậy (TPM) có thể được dùng để bảo vệ khóa mã hóa.

— Xác minh cập nhật phần mềm

Việc xác minh các bản cập nhật phần mềm có thể giảm thiểu rủi ro phần mềm độc hại bị cài đặt và có thể cải thiện bảo mật của phần cứng giám sát trước các cập nhật trái phép và trực trặc.

— Bảo vệ vật lý

TCVN XXXX:2026

Việc đặt phần cứng giám sát trong một môi trường vật lý được kiểm soát hoặc được theo dõi có thể cải thiện bảo mật của phần cứng giám sát trước truy cập trái phép.

— Tự kiểm thử

Việc tự kiểm thử các chức năng của phần cứng giám sát có thể cải thiện độ tin cậy và bảo mật của phần cứng giám sát trước các trục trặc.

Chứng nhận cho phần cứng giám sát bảo đảm rằng phần cứng giám sát đáp ứng các mục tiêu bảo mật này.

ISO/IEC 15408-2 có thể được áp dụng cho việc chứng nhận phần cứng giám sát. Các thành phần chức năng bảo mật liên quan được liệt kê trong [Bảng 4](#).

Bảng 4 — Các thành phần chức năng bảo mật liên quan trong ISO/IEC 15408-2

Mục tiêu bảo mật	Các thành phần chức năng bảo mật liên quan trong ISO/IEC 15408-2
Kiểm soát truy cập	FDP_ACF (chức năng kiểm soát truy cập)
Xác thực thực thể	FIA_UAU (xác thực người dùng), FDP_DAU (xác thực dữ liệu)
Toàn vẹn và xác thực dữ liệu	FDP_UIT (bảo vệ toàn vẹn khi truyền dữ liệu người dùng), FDP_DAU (xác thực dữ liệu)
Tính bí mật	FDP_UCT (bảo vệ tính bí mật khi truyền dữ liệu người dùng giữa các TSF), FCS_COP (hoạt động mật mã)
Xác minh cập nhật phần mềm	FDP_UIT (bảo vệ toàn vẹn khi truyền dữ liệu người dùng), FDP_DAU (xác thực dữ liệu)
Bảo vệ vật lý	FPT_PHP (bảo vệ vật lý TSF)
Tự kiểm thử	FPT_TST (tự kiểm thử TSF)

Các mối quan hệ nêu trên có thể được dùng để mô tả các hồ sơ bảo vệ (PP)/đích an toàn (ST) cho phần cứng giám sát.

Thư mục tài liệu tham khảo

- [1] Cassar I., Francalanza A., Aceto L., Ingólfssdóttir A., A Survey of Runtime Monitoring Instrumentation Techniques. Arxiv 254, 15–28 (2017).
- [2] Pirzadeh H., Hamou-Lhadj A., A View of Monitoring and Tracing Techniques and their Application to Service-Based Environments. Multimedia Services in Intelligent Environments 49–62 (2010).
- [3] Gao L., Lu M., Li L., Pan C., A Survey of Software Runtime Monitoring. 2017 8th IEEE Int Conf Softw Eng Serv Sci Icsess 308–313 (2024) doi:10.1109/icsess.2017.8342921.
- [4] Cornelis F. et al., A taxonomy of execution replay systems. Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet (2003).
- [5] Kornaros G., Pnevmatikatos D., A survey and taxonomy of on-chip monitoring of multicore systems-on-chip. Acm T Des Automat El 18, Article 17 (2013).
- [6] Abadi M., Budiu M., Erlingsson Ú., Ligatti J., Control-flow integrity principles, implementations, and applications. ACM Transactions on Information and System Security (TISSEC) 13, 1–40 (2009).
- [7] Garrison T., Intel CET Answers Call to Protect Against Common Malware Threats | Intel Newsroom. <https://newsroom.intel.com/editorials/intel-cet-answers-call-protect-common-malware-threats/> (n.d.).
- [8] Carlini N., Barresi A., Payer M., Wagner D., Gross T. R., Control-flow bending: On the effectiveness of control-flow integrity. in 161–176 (2015).
- [9] Bao C., Forte D., Srivastava A., Temperature Tracking: Toward Robust Run-Time Detection of Hardware Trojans. IEEE T Comput Aid D 34, 1577–1585 (2015).
- [10] Potkonjak M., Nahapetian A., Nelson M., Massey T., Hardware Trojan horse detection using gate-level characterization. 2009 46th Acm IEEE Des Automation Conf 688 (2009) doi:10.1145/1629911.1630091.
- [11] Mondal A., Mahalat M.H., Medapati A.R., Roy S., Sen B., XOR based Methodology to Detect Hardware Trojan utilizing the Transition Probability. 2018 8th Int Symposium Embed Comput Syst Des Ised 00, 215–219 (2018).
- [12] Hasegawa K., Yanagisawa M., Togawa N., Empirical Evaluation and Optimization of Hardware-Trojan Classification for Gate-Level Netlists Based on Multi-Layer Neural Networks. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences 101, 2320–2326 (2018).
- [13] Hasegawa K., Chikamatsu K., Togawa N., Empirical Evaluation on Anomaly Behavior Detection for Low-Cost Micro-Controllers Utilizing Accurate Power Analysis. in 54–57 (IEEE, 2019).
- [14] Hasegawa K., Yanagisawa M., Togawa N., Detecting the existence of malfunctions in microcontrollers utilizing power analysis. in 97–102 (IEEE, 2018).
- [15] Takasaki K., Hasegawa K., Kida R., Togawa N., An Anomalous Behavior Detection Method for IoT Devices by Extracting Application-Specific Power Behaviors. in 1–4 (IEEE, 2020).
- [16] Beard S. R., Trustguard: A Model for Practical Trust in Real Systems. (2019).
- [17] Leatherman R., Stollon N., An embedding debugging architecture for SOCs. IEEE Potentials 24, 12–16 (2005).
- [18] Vermeulen B., Goossens K., Debugging multi-core systems on chip. Multi-Core Embedded Systems 153–198 (2010).
- [19] Daoud E. A., Nicolici N., Embedded Debug Architecture for Bypassing Blocking Bugs During Post-Silicon Validation. IEEE Transactions Large Scale Integration Vlsi Syst 19, 559–570 (2011).
- [20] Leatherman R., On-chip instrumentation approach to system-on-chip development. First Silicon Solutions (1997).

- [21] Hopkins A. B. T., McDonald-Maier K. D., Debug support strategy for systems-on-chips with multiple processor cores. *Ieee T Comput* 55, 174–184 (2006).
- [22] Abramovici M. et al., A reconfigurable design-for-debug infrastructure for SoCs. in 7–12 (2006).
- [23] Paula F. M., de, Nahir A., Nevo Z., Orni A., Hu A. J., TAB-BackSpace: unlimited-length trace buffers with zero additional on-chip overhead. 411 (2011) doi:10.1145/2024724.2024821.
- [24] Park S.-B., Mitra S., Post-silicon bug localization for processors using IFRA. *Commun Acn* 53, 106–113 (2010).
- [25] Ojha A. K., Techniques in least-intrusive computer system performance monitoring. in 150–154 (IEEE, 2001).
- [26] Salapura V. et al., Next-Generation Performance Counters: Towards Monitoring Over Thousand Concurrent Events. *Ispass 2008 - Ieee Int Symposium Perform Analysis Syst Softw* 139–146 (2008) doi:10.1109/ispass.2008.4510746.
- [27] Sprunt B., The basics of performance-monitoring hardware. *Ieee Micro* 22, 64–71 (2002).
- [28] Mink A., Carpenter R. J., Nacht G., Roberts J. W., Multiprocessor performance-measurement instrumentation. *Computer* 23, 63–75 (1990).
- [29] Kyung H. et al., Performance monitor unit design for an AXI-based multi-core SoC platform. 1565 (2007) doi:10.1145/1244002.1244336.
- [30] Jägemar M., Utilizing Hardware Monitoring to Improve the Performance of Industrial Systems. (2016).
- [31] Eranian S., What can performance counters do for memory subsystem analysis? in 26–30 (2008).
- [32] Sylvester D., Blaauw D., Karl E., Elast IC, An Adaptive Self-Healing Architecture for Unpredictable Silicon. *Ieee Des Test Comput* 23, 484–490 (2006).
- [33] Atienza D. et al., Reliability-aware design for nanometer-scale devices. 2008 Asia South Pac Des Automation Conf 549–554 (2008) doi:10.1109/aspdac.2008.4484011.
- [34] Wells P. M., Chakraborty K., Sohi G. S., Mixed-mode multicore reliability. *Acn Comp Ar* 37, 169–180 (2009).
- [35] Lu C., Wang X., Koutsoukos X., Feedback utilization control in distributed real-time systems with end-to-end tasks. *IEEE Transactions on Parallel and Distributed Systems* 16, 550–561 (2005).
- [36] Yao J., Liu X., Gu Z., Wang X., Li J., Online adaptive utilization control for real-time embedded multiprocessor systems. *J Syst Architect* 56, 463–473 (2010).
- [37] Pastrnak M., With P. H., n D. & Meerbergen, J. V. Qos concept for scalable MPEG-4 video object decoding on multimedia (NoC) chips. *Ieee T Consum Electr* 52, 1418–1426 (2006).
- [38] Skadron K., Abdelzaher T., Stan M.R., Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management. *Proc Eighth Int Symposium High Perform Comput Archit* 17–28 (2002) doi:10.1109/hpca.2002.995695.
- [39] Shu L., Li X., Temperature-aware energy minimization technique through dynamic voltage frequency scaling for embedded systems. 2010 2nd International Conference on Education Technology and Computer 2, V2-515 (2010).
- [40] Choi K., Soma R., Pedram M., Fine-grained dynamic voltage and frequency scaling for precise energy and performance trade-off based on the ratio of off-chip access to on-chip computation times. *Proc Des Automation Test Europe Conf Exhib* 1, 4–9 (2004).
- [41] Schantz, R.E., Loyall, J.P., Rodrigues, C. and Schmidt, D.C. (2006), Controlling quality-of-service in distributed real-time and embedded systems via adaptive middleware. *Softw: Pract. Exper.*, 36: 1189-1208. <https://doi.org/10.1002/spe.761>
- [42] Ali, G., Hass, J., Sill, A., Hojati, E., Dang, T. and Chen, Y. Redfish-nagios: A scalable out-of-band data center monitoring framework based on redfish telemetry model. In Fifth International

Workshop on Systems and Network Telemetry and Analytics, 3-11 (2022).
<https://doi.org/10.1145/3526064.3534108>

- [43] Bas, F., Benedicte, P., Alcaide, S., Cabo, G., Mazzocchetti, F. and Abella, J. SafeDM: A Hardware Diversity Monitor for Redundant Execution on Non-Lockstepped Cores. In 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), 358-363, IEEE (2022).
<https://doi.org/10.23919/DATE54114.2022.9774540>
- [44] Bauer L., Shafique M., Henkel J., Efficient resource utilization for an extensible processor through dynamic instruction set adaptation. IEEE transactions on very large scale integration (VLSI) systems 16, 1295–1308 (2008).
- [45] Ragel R. G., Parameswaran S., Kia S. M., Micro embedded monitoring for security in application specific instruction-set processors. in 304–314 (2005).
- [46] Lisboa C., Kastensmidt F. L., Neto E. H., Wirth G., Carro L., Using built-in sensors to cope with long duration transient faults in future technologies. in 1–10 (IEEE, 2007).
- [47] Bastos R. P., Natale G. D., Flottes M.-L., Rouzeyre B., How to sample results of concurrent error detection schemes in transient fault scenarios? in 635–642 (IEEE, 2011).
- [48] Torres F.S., Bastos R.P., Robust modular Bulk Built-in Current Sensors for detection of transient faults. 2012 25th Symposium Integr Circuits Syst Des Sbcci 1, 1–6 (2012).
- [49] Bastos R. P., Torres F. S., Natale G. D., Flottes M., Rouzeyre B., Novel transient-fault detection circuit featuring enhanced bulk built-in current sensor with low-power sleep-mode. Microelectron Reliab 52, 1781–1786 (2012).
- [50] Bastos R. P. et al., A bulk built-in sensor for detection of fault attacks. in 51–54 (2AD). doi:10.1109/hst.2013.6581565.
- [51] Shobaki M.E., Lindh L., A hardware and software monitor for high-level system-on-chip verification. Proceedings of the IEEE 2001. 2nd International Symposium on Quality Electronic Design 56–61 (2001).
- [52] Dong C., He G., Liu X., Yang Y., Guo W., A Multi-Layer Hardware Trojan Protection Framework for IoT Chips. Ieee Access 7, 23628–23639 (2019).
- [53] Xu M., Bodik R., Hill M. D. A, "flight data recorder" for enabling full-system multiprocessor deterministic replay. Acm Sigarch Computer Architecture News 31, 122 (2003).
- [54] Zhang H. et al., Architectural Support for Containment-based Security. (Association for Computing Machinery, 2019). doi:10.1145/3297858.3304020.
- [55] LeBlanc T.J., Mellor-Crummey J.M., Debugging parallel programs with instant replay. IEEE Transactions on Computers 471–482 (1987).
- [56] Georges A., Christiaens M., Ronsse M., Bosschere K. D., JaRec: a portable record/replay environment for multi-threaded Java applications. Software: practice and experience 34, 523–547 (2004).
- [57] Dunlap G.W., King S.T., Cinar S., Basrai M.A., Chen P.M., ReVirt: Enabling intrusion analysis through virtual-machine logging and replay. ACM SIGOPS Operating Systems Review 36, 211–224 (2002).
- [58] Halsall F., Hui S., Performance monitoring and evaluation of large embedded systems. Software Engineering Journal 2, 184–192 (1987).
- [59] Narayanasamy S., Pokam G., Calder B., Bugnet: Continuously recording program execution for deterministic replay debugging. 32nd International Symposium on Computer Architecture (ISCA'05) 284–295 (2005).
- [60] Audenaert K. M., Levrouw L. J., Interrupt replay: a debugging method for parallel programs with interrupts. Microprocessors and Microsystems 18, 601–612 (1994).
- [61] Dodd P. S., Ravishankar C. V., Monitoring and debugging distributed realtime programs. Softw Pract Exp 22, 863–877 (1992).

- [62] Feldman S.I., Brown C.B., Igor: A system for program debugging via reversible execution. Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging 112–123 (1988).
- [63] Steven J., Chandra P., Fleck B., Podgurski A., jRapture: A capture/replay tool for observation-based testing. Proceedings of the 2000 ACM SIGSOFT international symposium on Software testing and analysis 158–167 (2000).
- [64] Pan D. Z., Linton M. A., Supporting reverse execution for parallel programs. *Acm Sigplan Notices* 24, 124–129 (1988).
- [65] Choi J.-D., Srinivasan H., Deterministic replay of Java multithreaded applications. Proceedings of the SIGMETRICS symposium on Parallel and distributed tools 48–59 (1998) doi:10.1145/281035.281041.
- [66] Sorin D.J., Martin M.M., Hill M.D., Wood D.A., SafetyNet: improving the availability of shared memory multiprocessors with global checkpoint/recovery. Proceedings 29th Annual International Symposium on Computer Architecture 123–134 (2002).
- [67] Shalabi Y., Yan M., Honarmand N., Lee R.B., Torrellas J., Record-replay architecture as a general security framework. 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA) 180–193 (2018).
- [68] Siegel H. J., Schwederski T., Kuehn J. T., IV, N. J. D. An overview of the PASM parallel processing system. *Computer Architecture* 387–407 (1987).
- [69] Tokuda H., Kotera M., Mercer C., A real-time monitor for a distributed real-time operating system. Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging 68–77 (1988).
- [70] Hofmann R., Klar R., Mohr B., Quick A., Siegle M., Distributed performance monitoring: methods, tools, and applications. *IEEE Transactions on Parallel and Distributed Systems* 5, 585–598 (1994).
- [71] Watterson C., Heffernan D., A monitoring approach to facilitate run-time verification of software in deeply embedded systems. Limerick, University of Limerick. PhD 201, (2010).
- [72] Faruque M.A.A., Ebi T., Henkel J., Run-time Adaptive on-chip Communication Scheme. 2007 Ieee *Acm Int Conf Comput Des* 26–31 (2007) doi:10.1109/iccad.2007.4397239.
- [73] Faruque M.A.A., Krist R., Henkel J., ADAM: run-time agent-based distributed application mapping for on-chip communication. 2008 45th ACM/IEEE Design Automation Conference 760–765 (2008).
- [74] Fick D. et al., Vicis: A reliable network for unreliable silicon. Proceedings of the 46th Annual Design Automation Conference 812–817 (2009).
- [75] Wen C.-N., Chou S.-H., Chen C.-C., Chen T.-F., NUDA: A non-uniform debugging architecture and nonintrusive race detection for many-core systems. *IEEE Transactions on Computers* 61, 199–212 (2010).
- [76] Zhou P., Teodorescu R., Zhou Y., HARD: Hardware-assisted lockset-based race detection. 2007 IEEE 13th International Symposium on High Performance Computer Architecture 121–132 (2007).
- [77] Tsai J. J. P., Fang K.-Y., Chen H.-Y., A noninvasive architecture to monitor real-time distributed systems. *Computer* 23, 11–23 (1990).
- [78] Liu L. et al., Jintide®: A Hardware Security Enhanced Server CPU with Xeon® Cores under Runtime Surveillance by an In-Package Dynamically Reconfigurable Processor. 2019 Ieee *Hot Chips 31 Symposium Hcs* 1–25 (2018) doi:10.1109/hotchips.2019.8875682.
- [79] Ciordas C., Hansson A., Goossens K., Basten T., A monitoring-aware network-on-chip design flow. *Journal of Systems Architecture* 54, 397–410 (2008).
- [80] Noraziah A., Hafizulrahim A.S., Ramiza R.M., Junaida S., Noorhuzaimi M.N., System Memory Monitoring through windows application programming interface. 2008 First Int Conf Appl Digital Information Web Technologies *Icadiwt* 354–359 (2008) doi:10.1109/icadiwt.2008.4664372.

- [81] Bartocci E., Falcone Y., Francalanza A., Reger G., Introduction to runtime verification. in 1–33 (Springer, 2018).
- [82] Rosu F. C. G., MOP: An Efficient and Generic Runtime Verification Framework. (2007).
- [83] Francalanza A., A theory of monitors. in 145–161 (Springer, 2016).
- [84] Giacomo G. D., Vardi M. Y., Linear temporal logic and linear dynamic logic on finite traces. in 854–860 (Association for Computing Machinery, 2013).
- [85] Huang R., Grawrock D., Doughty D. C., Suh G. E., Lecture Notes in Computer Science. 154–171 (2011) doi:10.1007/978-3-642-21599-5_12.
- [86] Sherman B., Wheeler D., Hardware security risk assessment: A case study. in 179–184 (3AD). doi:10.1109/hst.2016.7495579.
- [87] Guin U., Asadizanjani N., Tehranipoor M., Standards for Hardware Security. Getmobile Mob Comput Commun 23, 5–9 (2019).
- [88] Bhunia S., Tehranipoor M., Hardware security: a hands-on learning approach. (Morgan Kaufmann, 2018).
- [89] Austin T.M., DIVA: A reliable substrate for deep submicron microarchitecture design. MICRO-32. Proceedings of the 32nd Annual ACM/IEEE International Symposium on Microarchitecture 196–207 (1999).
- [90] Wei S., Li K., Koushanfar F., Potkonjak M., Hardware Trojan horse benchmark via optimal creation and placement of malicious circuitry. Proceedings of the 49th Annual Design Automation Conference 90–95 (2012).
- [91] Li J., Lach J., At-speed delay characterization for IC authentication and Trojan Horse detection. in 8–14 (9AD). doi:10.1109/hst.2008.4559038.
- [92] Chakraborty R. S., Bhunia S., HARPOON: an obfuscation-based SoC design methodology for hardware protection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 28, 1493–1502 (2009).
- [93] Popp T., Mangard S., Oswald E., Power analysis attacks and countermeasures. IEEE Design & test of Computers 24, 535–543 (2007).
- [94] Hoque T., Cruz J., Chakraborty P., Bhunia S., Hardware IP Trust Validation: Learn (the Untrustworthy), and Verify. in vol. 00 1–10 (29AD).
- [95] Ahmed, M.A., Aloufi, J. A Smart Memory Controller for System on Chip-Based Devices. Journal of Nanomaterials, 2022. <https://doi.org/10.1155/2022/4944335>
- [96] Skorobogatov S., Woods C., Breakthrough silicon scanning discovers backdoor in military chip. in 23–40 (Springer, 2012)
- [97] Lipp M. et al., Meltdown: Reading kernel memory from user space. in 973–990 (2018)
- [98] Kocher P. et al., Spectre attacks: Exploiting speculative execution. in 1–19 (IEEE, 2019)
- [99] Evtushkin D., Riley R., Abu-Ghazaleh N. C., and E. & Ponomarev, D. Branchscope: A new side-channel attack on directional branch predictor. ACM SIGPLAN Notices 53, 693–707 (2018)
- [100] Gras B., Razavi K., Bos H., Giuffrida C., Translation leak-aside buffer: Defeating cache side-channel protections with {TLB} attacks. in 955–972 (2018)
- [101] Qiu P., Wang D., Lyu Y., Qu G., Voltjockey: Breaching trustzone by software-controlled voltage manipulation over multi-core frequencies. in 195–209 (2019)
- [102] Xiao K., Nahiyan A., Tehranipoor M., Security rule checking in IC design. Computer 49, 54–61 (2016)
- [103] Rostami M., Koushanfar F., Karri R., A Primer on Hardware Security: Models, Methods, and Metrics. P IEEE 102, 1283–1295 (2014)

TCVN XXXX:2026

- [104] AS6171A, Test Methods Standard; General Requirements, Suspect/Counterfeit, Electrical, Electronic, and Electromechanical Parts
- [105] FIPS 140-2, Security Requirements for Cryptographic Modules (2001)
- [106] ISO/IEC 17825, Information technology — Security techniques — Testing methods for the mitigation of non-invasive attack classes against cryptographic modules
- [107] ISO/IEC 19790, Information technology — Security techniques — Security requirements for cryptographic modules
- [108] ISO/IEC TR 20004, Information technology — Security techniques — Refining software vulnerability analysis under ISO/IEC 15408 and ISO/IEC 18045
- [109] ISO/IEC 15408-2, Information security, cybersecurity and privacy protection — Evaluation criteria for IT security — Part 2: Security functional components
- [110] ISO/IEC 15408-3, Information security, cybersecurity and privacy protection — Evaluation criteria for IT security — Part 3: Security assurance components
- [112] Forti, S., Gaglianese, M. Brogi, A. Lightweight self-organising distributed monitoring of Fog infrastructures. Future Generation Computer Systems, 114, 605-618 (2021) <https://doi.org/10.1016/j.future.2020.08.011>
- [113] Rajan, J.G., Ganesh, R.S. Hardware Based Data Security Techniques in IOT: A Review. In 2022 3rd International Conference on Smart Electronics and Communication (ICOSEC), 408-413, IEEE (2022). <https://doi.org/10.1109/ICOSEC54921.2022.9952021>

