BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SĨ KHOA HỌC

NGHIÊN CỨU MẠNG CAMERA THÔNG MINH PHỤC VỤ GIÁM SÁT AN NINH

NGÀNH: CÔNG NGHỆ THÔNG TIN
MÃ SỐ:

NGUYỄN QUANG MINH

Người hướng dẫn khoa học: PGS.TS NGUYỄN NGỌC BÌNH

HÀ NỘI - 2006

LỜI CẢM ƠN

Để hoàn thành được luận văn này, em xin cảm ơn chân thành đến thầy giáo PGS. TS Nguyễn Ngọc Bình, người đã định hướng khoa học, thu thập kiến thức và hướng dẫn em trong suốt quá trình làm việc.

Nguyễn Quang Minh

Hà nội, 11 - 2006

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

BS	*Base Station*, trạm gốc. Điểm gắn kết giữa hệ thống camera giám sát với người dùng. Tại đây, tác tử di động giao tiếp với người dùng và chuyển yêu cầu người dùng thành nhiệm vụ giám sát tương ứng và trao đổi thông tin với hệ thống. Thuật ngữ tương đương OCU (*Operator/ Control Unit*)
SC	*Smart Camera*, camera thông minh. Ngoài bộ phận cảm biến ghi hình khung cảnh và biến đổi thành dữ liệu số, SC còn có các khối chức năng khác như lưu trữ, truyền thông, xử lý, điều khiển PTZ ...
SCN	*Smart Camera Network*, mạng liên kết các camera thông minh. Là mạng liên kết các SC, không hướng cấu trúc mà hướng các sự kiện hệ thống phục vụ cho mục đích giám sát an ninh. SCN là một đại diện của hệ thống xử lý hình toàn năng, hệ thống đa phương tiện nhúng phân tán.
s_clu	*Surveillance Cluster*, nhóm các camera giám sát. Một nhóm được tạo bởi các SC có quan hệ trong sự kiện, nhiệm vụ.
proxy	Trong SCN, khái niệm này dùng để chỉ những SC hoạt động ở chế độ trung gian giao tiếp giữa ứng dụng tra cứu, *BS* với các SC khác. Tên gọi khác: AGM (*Archive/ Gateway Module*)

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH VẼ

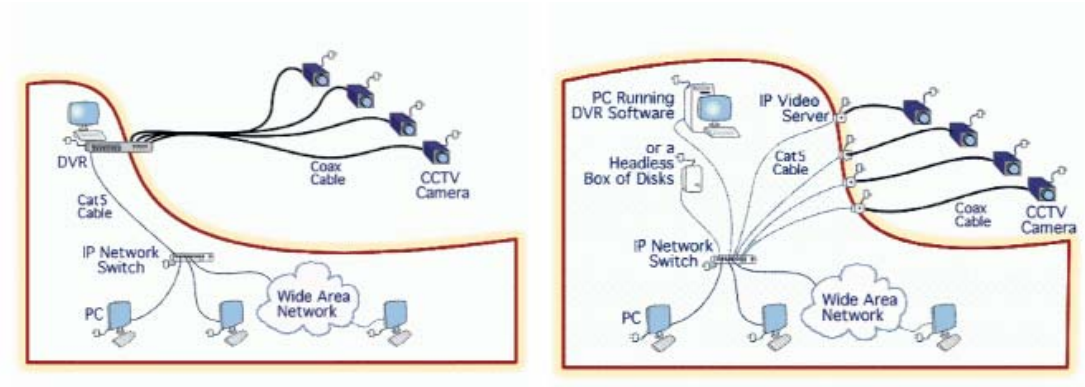
CHƯƠNG 1 : MỞ ĐẦU

1.1 DẪN NHẬP

Việc ứng dụng mạng camera để giám sát an ninh khu vực đã được đưa vào thực tế từ rất lâu. Theo dòng phát triển khoa học công nghệ, các mạng camera giám sát phát triển không ngừng, đến nay đã trải qua ba thế hệ công nghệ

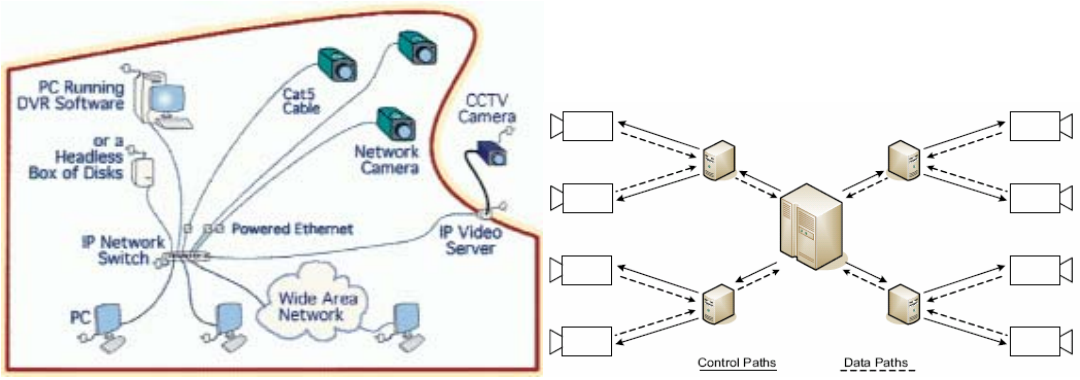
Thế hệ đầu tiên, là giai đoạn sử dụng các camera tương tự CCTV, tín hiệu hình ảnh được truyền từ về trung tâm, nơi có đặt thiết bị xuất hình hay lưu trữ ra băng từ.

Thế hệ thứ hai, đã có sự tiến chuyển là xuất hiện các thiết bị số đặt tại trung tâm, các dòng dữ liệu hình truyền tải về đây được phân tích và xử lý tự động theo thời gian thực. Hệ thống có khả năng đưa ra cảnh báo dựa trên phân tích tự động dữ liệu nhận được do các camera cung cấp.



Hình 1. Các hệ thống camera giám sát thế hệ thứ nhất và thứ hai

Thế hệ thứ ba, là thế hệ mạng giám sát ngày nay, đã có sự thay thế hoàn toàn các camera tương tự bởi các camera số nên dòng dữ liệu hình truyền về trung tâm là dòng video đã qua nén để tối ưu băng thông cũng như sử dụng trực tiếp hạ tầng mạng *IP* như *Ethernet* hay *Wireless LAN*. Tại trung tâm các thiết bị cũng có hiệu năng cao hơn nhiều so với thế hệ trước.



Hình 2. Hệ thống camera giám sát thế hệ thứ ba

Khái niệm camera thông minh *intelligent camera* bắt đầu được đưa ra vào thời điểm này, với định nghĩa đơn giản là camera có thể tiền xử lý các hình ảnh thu nhận được[1].

Tuy nhiên hệ thống thế hệ thứ ba chưa thật sự đáp ứng được nhu cầu người dùng trong nhiều trường hợp. Nguyên nhân sâu xa nằm tại kiến trúc của hệ thống. Kiến trúc này có nhiều nhược điểm, cụ thể như:

- ***Tính chịu lỗi thấp*** Khi có sự cố tại trung tâm điều khiển dễ dẫn đến điều khiển hệ thống, các phân tích và xử lý dữ liệu hình bị đình trệ đến khi sự cố này được khắc phục.
- ***Thông tin chưa như mong muốn*** Để có các kết quả phân tích dữ liệu hình chất lượng cao, các dữ liệu hình truyền tải về trung tâm phải chọn phương pháp nén không mất thông tin *lossless* và tốc độ dòng bit cao. Bài toán đặt ra ở đây là cân nhắc giữa băng thông và tỷ số nén. Những phương pháp nén hiện nay như JPEG, MPEG hay MJPEG cho tỷ số nén tốt nhưng thuộc loại nén mất thông tin. Như vậy có thể xảy ra trường hợp là có dữ liệu truyền về trung tâm nhưng chất lượng dữ liệu đó không đáp ứng được nhu cầu của ứng dụng.

[1] Các tác vụ như trích chọn đặc trưng, phát hiện chuyển động và thông tin cảnh báo truyền về trung tâm trước song song với việc truyền dòng dữ liệu hình về ở các tốc độ khung và chất lượng ảnh khác nhau.

- ***Thiếu tính tự chủ*** Thông tin điều khiển luôn theo hướng từ trung tâm đến camera, giữa các camera không có khả năng trao đổi thông tin trực tiếp.
- ***Không có khả năng tái cấu trúc*** kiến trúc phân tầng và phân chia chức năng của từng vùng trong hệ thống dẫn đến khả năng thích nghi của hệ thống là không cao. Điều này dẫn đến việc lai ghép hay phân tách hệ thống rất khó khăn. Hệ thống là hầu như không phân tách tùy ý được do tồn tại trung tâm điều khiển.

Về tổng quan, một hệ thống giám sát an ninh gồm có những thành phần sau:

1. Kiến trúc cảm biến.
2. Các thuật toán phát hiện và xử lý cấp thấp.
3. Kiến trúc xử lý tính toán phần cứng.
4. Kiến trúc xử lý tính toán phần mềm.
5. Giao diện người dùng.
6. Các thuật toán cấp cao để hợp nhất dữ liệu và loại bỏ những sự kiện không mong muốn.

Trong những năm gần đây, người ta đã tập trung nghiên cứu thay đổi kiến trúc hệ thống trong các thành phần 3, 4 từ xử lý tập trung sang phân tán. Tất nhiên những thay đổi về kiến trúc đó sẽ dẫn đến những thay đổi tương ứng ở những thanh phần còn lại. Hệ thống mới được xếp loại là thế hệ thứ 3+.

Luận văn này được xây dựng nhằm mục đích nghiên cứu và xây dựng mới một sản phẩm là hệ thống trong nhóm thế hệ 3+ này. Sản phẩm này được đặt tên là HỆ THỐNG CAMERA THÔNG MINH - *Smart Camera Networks* (SCN).

1.2 GIỚI HẠN HỆ THỐNG VÀ CÁC HỆ THỐNG TƯƠNG TỰ

Các hệ thống xử lý hình toàn năng *ubiquitous vision system* (UVS)[2], là mong muốn đạt được của các nhà khoa học máy tính trên thế giới. Trên một lĩnh vực cụ thể là giám sát an ninh khu vực thì SCN có thể coi là đại diện tiêu biểu của UVS, do vậy tôi chọn lựa và xây dựng SCN trong phạm vi các ràng buộc về công nghệ và ứng dụng nhất định.

SCN tổng quát được định nghĩa là mạng của các camera phân tán thực sự và phân tải phân tán xử lý tính toán[3].

Trên thế giới, các hệ thống gần tương tự như SCN cũng đã được nhiều nhà khoa học nghiên cứu hoặc đã được ứng dụng trong an ninh quốc phòng. Mục tiêu xây dựng hệ thống SCN cho thị trường dân sự và an ninh khu vực, nên tôi tập trung xây dựng và giải quyết hai bài toán cơ bản nhất của một mạng giám sát an ninh phân tán, cụ thể là:

1. Bài toán CB1: Đánh giá tác động và cơ chế điều chỉnh phân tán nhiệm vụ giám sát cho mỗi SC trong SCN.
2. Bài toán CB2: Tìm kiếm thông tin, dữ liệu hình đã lưu trữ trong SCN.

Các ứng dụng phát hiện và trích chọn đặc trưng cục bộ có thể xử lý bởi một camera đơn nhất được coi là đơn giản và không trình bày trong luận văn này.

Tuy phân tích hành vi đối tượng, phân tích tình huống phát hiện chuyển động bất thường là những ứng dụng phức tạp nhưng có thể giải quyết bởi việc phát triển bài toán CB1, và cũng nằm ngoài phạm vi nghiên cứu của luận văn này nên cũng không trình bày tại đây mà dành cho các nghiên cứu mở rộng tiếp theo.

---

[2] Capture and maintain an awareness of dynamic events of variable spatiotemporal resolution and of multiple levels of abstraction.

[3] Physically distributed cameras and distributed computing.

Do điều kiện kỹ thuật chưa có điều kiện triển khai thực tế tại Việt Nam nên hai bài toán cơ bản nêu trên được xây dựng và giải quyết trên cơ sở phân tích, đánh giá và thử nghiệm trên mô hình mô phỏng và phòng thí nghiệm.

1.3 Ý NGHĨA KHOA HỌC VÀ THỰC TIỄN

Luận văn này là một ứng dụng nhỏ của ngành khoa học máy tính vào trong lĩnh vực giám sát an ninh khu vực và là cơ sở để xây dựng các hệ thống thương mại mới, phù hợp với mặt bằng khoa học và công nghệ tại Việt Nam hiện nay. Những đóng góp chính về mặt khoa học của luận văn là:

- Đề xuất chuyển đổi những thuật toán xử lý ảnh và thông tin hình tập trung thành những liên kết nhiệm vụ phân tán trong một mạng phân tán thực sự của các thiết bị nhúng đáp ứng thời gian thực.
- Đề xuất sử dụng *framework* mềm dẻo cho phần mềm và sử dụng các tác tử thông minh di động khi liên kết nhiệm vụ là những hướng đi đúng khi phát triển ứng dụng cho các thiết bị nhúng.
- Đề xuất sử dụng cấu trúc lưu trữ trong hệ thiết bị nhúng phân tán với hai lớp trong suốt đối với ứng dụng và người dùng cho bài toán lưu trữ và tra cứu dữ liệu hình.
- Đề xuất sử dụng phương pháp truyền thông vô tuyến phi cấu trúc cho hệ thống giám sát an ninh trong các trường hợp khẩn cấp và đặc biệt.

Từ bài toán CB1 có thể dễ dàng phát triển thành các bài toán tương tự giải quyết được những vấn đề phức tạp hơn. Bài toán CB2 là khuôn mẫu và ví dụ cho việc xây dựng các hệ thống lưu trữ nhúng phân tán khác.

Các ứng dụng, nghiên cứu trong SCN có thể tái sử dụng và phát triển cho các hệ thống đa phương tiện phân tán khác ví dụ như *Smart Audio Network*, hay tổng quát hơn ví dụ như *Smart Sensor Network*.

Bố cục

Chương 2 nêu Giới hạn hệ thống; các vấn đề và định hướng chọn lựa kiến trúc phần cứng và phần mềm cho các SC là các phần tử cơ bản của hệ thống SCN.

Chương 3,4,5,6 là các chủ đề cơ bản nhất trong hệ thống SCN là vấn đề đánh địa chỉ SC, vấn đề đồng bộ bộ đếm, vấn đề định tuyến, vấn đề an ninh truyền thông trong SCN. Những chủ đề này là cơ sở và có tác động đến mô hình và giải pháp cho hai bài toán CB1 và CB2 được giải quyết trong chương 7 và 8.

Phần cuối cùng là kết luận và những hướng nghiên cứu tiếp theo khi phát triển hệ thống tương tự và kế thừa SCN.

Chương 2 : MÔ HÌNH THIẾT KẾ SC & SCN

Hệ thống SCN được xây dựng dựa trên nền tảng là các *SmartCamera* (SC) phân tán và truyền thông không dây ngang hàng (*ad-hoc*) giữa chúng.

- Mỗi SC là một hệ thống nhúng có đáp ứng thời gian thực RTES[4].
- Truyền thông trong SCN sử dụng chuẩn 802.11[5].

Trong những ứng dụng, hoàn cảnh cụ thể, một SC bình thường có thể hoạt động ở các chế độ khác nhau và hiện diện trong hệ thống như là một thiết bị khác.

- ***BS*** Điểm gắn kết với người dùng, tương tác với hệ thống khác hoặc từ đó là nguồn phát sinh điều khiển, dịch vụ thu thập số liệu.
- ***Proxy*** Điểm trung gian giữa *BS* và các SC khác hoạt động như một *Router*, *Data Proxy*, *RBS* ...
- ***active_SC*** Hoạt động ở chế độ bình thường, có xử lý cả hình ảnh và truyền thông.
- ***sleep_SC*** Hoạt động ở chế độ tiết kiệm năng lượng, giảm tải xử lý và truyền thông.

2.1 ĐỊNH HƯỚNG THIẾT KẾ SCN

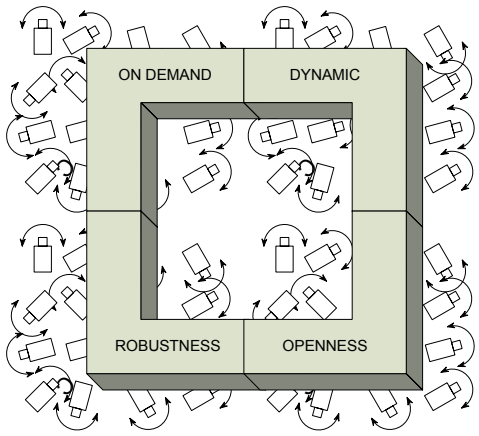
Khi lựa chọn kiến trúc phần cứng và phần mềm cho SC trong SCN cần tuân thủ các định hướng thiết kế sau: hướng mở, theo nhu cầu, hướng động và mạnh mẽ.

---

[4] Các SC được chọn phải đủ nhỏ; tối ưu năng lượng để có thể hoạt động mà không cần thiết bị cấp nguồn ngoài trong một khoảng thời gian nhất định; không bắt buộc phải cố định và khung nhìn giám sát có thể thay đổi PTZ tùy ý hoặc theo ứng dụng.

[5] Trong trường hợp một nhóm các SC được kết nối bởi một phương thức truyền thông tốc độ cao, hướng cấu trúc thì nhóm SC đó được coi như là một SC đặc biệt trong hệ thống và các SC khác giao tiếp với nhóm SC đó như một SC độc lập, phân biệt với *s_clu*, *microcluster*.

Trong hệ thống có thể tại một thời điểm hay một phạm vi nhất định hoặc một ứng dụng nhất định có tồn tại điểm truy nhập tập trung *AP* ví dụ như vệ tinh địa tĩnh, cluster *AP* tuy nhiên các ứng dụng được xây dựng nhằm đảm bảo phụ thuộc ít nhất vào các *AP* trên khi hoạt động.



Hình 3. Định hướng thiết kế SCN

Phục vụ theo nhu cầu - *on demand*

Khác với hướng tiếp cận của các hệ thống trước, các thông tin và dữ liệu hình được tập trung về trung tâm, và sẽ phân phối thông tin, tham chiếu đến những người dùng quan tâm.

Trong SCN với mỗi người dùng, mỗi ngữ cảnh, hệ thống sẽ có đáp ứng thích hợp. Định hướng này còn góp phần đảm bảo đáp ứng thời gian thực của hệ thống. Việc chia các nhiệm vụ giám sát theo các mức QoS khác nhau, và điều chỉnh mức QoS của ứng dụng tùy theo nhu cầu, sự kiện sẽ đảm bảo hệ thống luôn trong tầm kiểm soát của tải xử lý, tải truyền thông.

Khái niệm *on-demand* sẽ xuất hiện trong nhiều vấn đề của hệ thống SCN, ví dụ như:

- ***Routing on-demand*** Định tuyến theo nhu cầu.
- ***TimeSync on-demand*** Đồng bộ thời gian theo nhu cầu.
- ***Video on-demand*** Phát hình theo nhu cầu.

Hướng động - *dynamic*

Đây là một đặc tính chung thường gặp ở các hệ phân tán. Đặc tính này giúp SCN có khả năng tái cấu trúc khi có sự cố hoặc nhằm thích nghi với ứng dụng, môi trường.

Tính hướng động cần được đề cao trong tất cả các vấn đề trong SCN, ví dụ như:

- ***Dynamic Routing***. đảm bảo việc truyền thông tốt trong các điều kiện hoạt động khác nhau của hệ thống, thích nghi tốt với mô hình mạng không hướng cấu trúc như SCN.
- ***Address-Free Naming Architecture***. đảm bảo địa chỉ đơn nhất cho thiết bị trong phạm vi ứng dụng nhưng vẫn hỗ trợ tăng tốc tìm kiếm thiết bị dù không tồn tại một điểm tập trung và phân phối thông tin trong SCN.
- ***Dynamic Task Distribution***. đảm bảo các nhiệm vụ giám sát được truyền tải và chuyển giao tự động giữa các SC trong SCN mà không cần điểm tập trung và phân phối thông tin điều khiển.

Hướng mở - *openness*

Hệ thống SCN phải là hệ thống hướng mở hoàn toàn. Dĩ nhiên, khi thiết kế SCN và tuân thủ định hướng *on-demand* và *dynamic* thì hệ thống đã đảm bảo được tính thích nghi và khả mở. Tuy nhiên định hướng mở cần được tách riêng để lưu ý người thiết kế rằng SCN cần đảm bảo hỗ trợ khả năng lai ghép với các hệ thống khác. Các hệ thống đó không bắt buộc phải là SCN mà có thể là những hệ *multimedia* hoặc những hệ thu thập số liệu khác.

Mạnh mẽ - *robustness*

Điểm yếu của các hệ thống cũ là năng lực tính toán của hệ thống tập trung hoàn toàn tại trung tâm điều khiển và hầu như là cố định nếu như không có sự nâng cấp về thiết bị. Một hệ thống đã được thiết kế cho 100 camera sẽ gặp vấn đề về xử lý khi có bổ sung thêm 1000, 10000 camera.

Hệ thống SCN được xây dựng trên cơ sở phân tải tính toán, lưu trữ sẽ giải quyết vấn đề này chọn vẹn, đảm bảo nếu có gia tăng về số lượng điểm

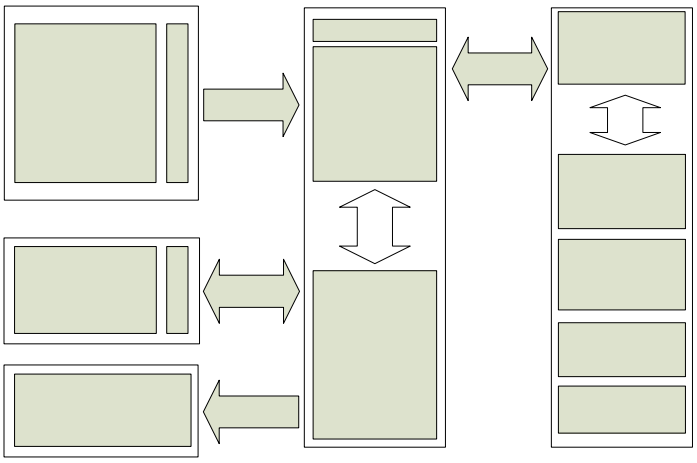
tham gia hệ thống, số lượng nhiệm vụ giám sát thì vẫn kiểm soát được tài nguyên và hiệu năng chung của hệ.

Sau đây là các kiến trúc điển hình của phần cứng và phần mềm của một SC trong SCN.

2.2 KIẾN TRÚC PHẦN CỨNG VÀ KHỐI CHỨC NĂNG CỦA MỘT SC

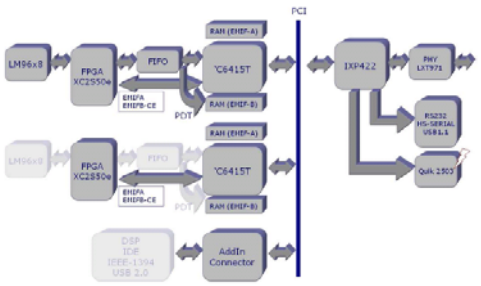
Tại những hệ thống tương tự SCN, người ta thường chia SC thành ba khối là cảm biến, xử lý và truyền thông.

- ***Sensor Unit*** Thu thập các dữ liệu hình
- ***Processing Unit*** Thực thi các tác vụ và xuất lệnh điều khiển camera
- ***Communication Unit*** Trao đổi thông tin giữa các SC với nhau, với hệ thống khác bao gồm cả dữ liệu và thông tin điều khiển.



Hình 4. Sơ đồ khối chức năng phần cứng trong SC

Hình 5 là sơ đồ khối kiến trúc phần cứng một SC với hạt nhân là DSP TMS320C6415T và đánh giá mức tiêu thụ năng lượng trung bình của SC đó.



Qty	Part	Power	Overall
Sensing unit			
1	Image sensor (KAC-9618)	2.1 W	2.1 W
1	FIFO memory (IDT-72V3664)	0.2 W	0.2 W
Overall power consumption			2.3 W
Processing block			
1	DSP (TMS320C6415T @ 1 GHz)	1.4 W	1.4 W
1	DSP (TMS320C6415T @ 600 MHz)	1.0 W	1.0 W
4	RAM (32 MB)	0.48 W	1.92 W
1	FPGA (XC2S50e)	1.8 W	1.8 W
Overall power consumption @ 1 GHz			5.12 W
Overall power consumption @ 600 MHz			4.72 W
Communication unit			
1	Network processor (IXP422 @ 533 MHz)	1.2 W	1.2 W
8	RAM (32 MB)	0.48 W	3.36 W
1	Ethernet PHY (LXT971)	0.33 W	0.33 W
Overall power consumption			4.89 W

Hình 5. Kiến trúc phần cứng và đánh giá mức tiêu thụ năng lượng một SC

Dưới đây là các khối chức năng của SC như trong hình 5.

Khối cảm biến

Trung tâm của khối cảm biến là cảm biến hình *CMOS* với giao tiếp *FPGA* là giao diện chung với khối xử lý. Tuy phần lớn cảm biến hình có phân giải lên đến 12 bit điểm, nhưng khi qua *FPGA* giảm xuống còn 8 bit. *FPGA* truyền dữ liệu đến khối xử lý thông qua vùng nhớ đệm *FIFO*. Ngoài ra còn có các thông số khác được quyết định bởi khối cảm biến là:

- ***dynamic range*** vùng động, trong các ứng dụng đòi hỏi chất lượng ảnh cao, khử mờ cùng với sự thích nghi với sự thay đổi của điều kiện chiếu sáng thì cần có các cảm biến hình ảnh xử lý chất lượng cao.
- ***resolution & frame rate*** độ phân giải và tốc độ truyền khung hình, nhiều cảm biến hình chỉ xuất những khung hình nhỏ chất lượng thấp *CIF* hay *QCIF*. Những định dạng này chỉ phù hợp với thiết bị *monitor* là điện thoại hoặc *PDA*. Có nhiều ứng dụng đòi hỏi độ phân giải cao hơn ví dụ như *PAL* (720x576 điểm). Phần lớn các thuật toán xử lý thường gặp dụng ảnh đầu vào là ảnh đa mức xám tuy nhiên với *BS* thì ảnh mầu vẫn là ưu tiên hơn trong trường hợp xuất trình diễn cho người giám sát. Tốc độ truyền khung tối đa cũng là một thông số rất quan trọng, thường thì 2 fps là có thể đáp ứng được cho việc theo dõi giám sát an ninh.

- ***digital interface*** giao tiếp số, trong các cảm biến hình có bao gồm bộ khuếch đại tương tự và các biến đổi ADC.

Khối xử lý

Do đặc thù công việc xử lý dòng dữ liệu hình cùng với yêu cầu tối ưu năng lượng nên DSP là chọn lựa hợp lý cho hạt nhân của hệ thống[6]. Một vài tác vụ đặc thù có thể xử lý trực tiếp trong khối xử lý đơn lẻ trên như nén hình, phân tích hình, tính toán đơn, điều khiển camera và các ứng dụng.

Khối xử lý giao tiếp với khối truyền thông qua bus, ví dụ bus PCI chạy ở xung nhịp 133MHz.

Khối truyền thông

Khối truyền thông đảm nhận việc trao đổi thông tin giữa SC với thế giới ngoài. Tổng quát thì việc truyền thông trong SC gồm có hai phần.

- ***Nội truyền thông*** giao tiếp với khối xử lý thông qua bus PCI, giao tiếp với khối lưu trữ qua các kênh DMA trực tiếp và vùng đệm.
- ***Ngoại truyền thông*** thiết lập các kênh giao tiếp như *Ethernet*, *wireless LAN* hay *GPRS*.

Để tăng hiệu năng và phân tách chức năng, trong SC có sử dụng bộ xử lý mạng chuyên dụng[7]. Đây là bộ xử lý hiệu năng cao dạng *System on Chip* (SoC) cung cấp được tất cả các giao diện cần thiết như *Ethernet*, *USB* và cổng tuần tự. GSM/GPRS được đáp ứng bởi mô đun *WaveCom* kết nối với cổng tuần tự của bộ xử lý trên. Các giao diện khác như *UMTS, IDE, WLAN* được tích hợp chung bởi bus *PCI*, *USB* hoặc cổng tuần tự.

2.3 KIẾN TRÚC PHẦN MỀM TRONG SC

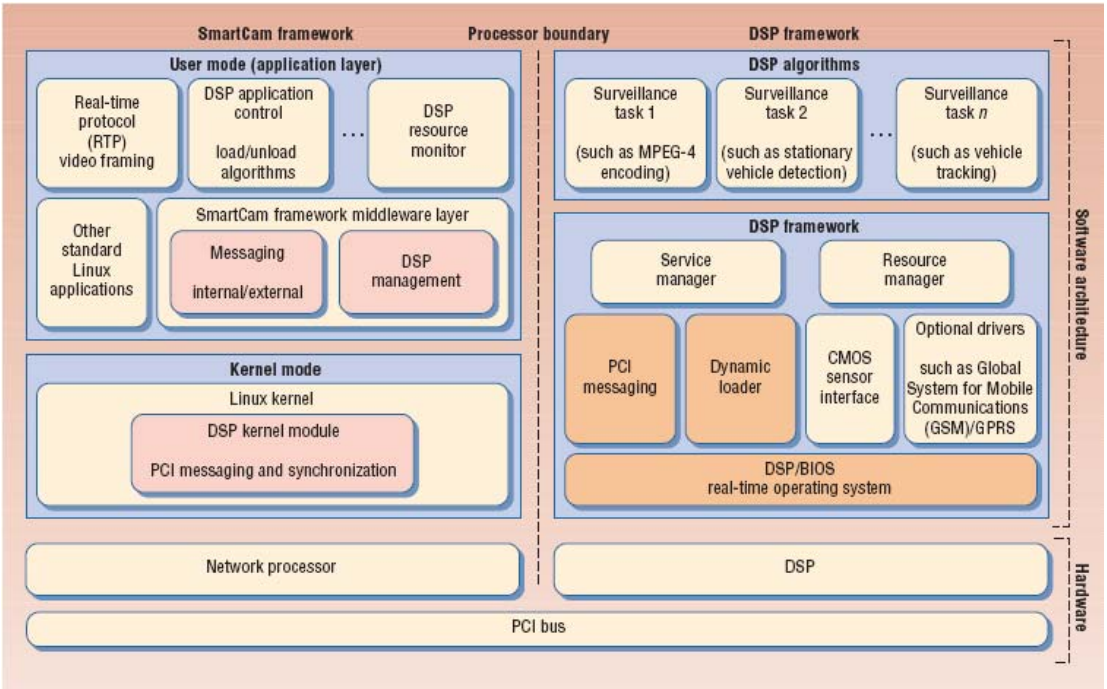
Kiến trúc phần mềm trong SC được xây dựng dựa trên kiến trúc phần cứng, tuy nhiên nhằm phục vụ cho các ứng dụng khác nhau, tránh để người

[6] DSP TMS320-C6415T có 1 Mb bộ nhớ và đạt đến 8000 MIPS với xung nhịp 1GHz.
[7] Intel XScale IXP422 có bộ nhớ 256 Mb và chạy ở tốc độ 533 MHz.

phát triển can thiệp quá sâu vào phần cứng hệ thống nên các đặc tả phần mềm đều cố gắng phát huy khả năng trừu tượng hóa các giao tiếp ứng dụng API. API giúp người phát triển tập trung vào nhiệm vụ chính mà không mất nhiều thời gian cho những vấn đề hiện thực cụ thể. Ví dụ như phát triển hàm *GetVideo()* toàn năng giúp người lập trình truy nhập nhiều nguồn video khác nhau (có khác biệt về cách thực hiện) mà không phải thay đổi bất kỳ dòng lệnh chương trình nào.

Trong một SC cụ thể, kiến trúc phần mềm được thiết kế nhằm đảm bảo tính mềm dẻo, linh động và hiệu năng cao [DESC_06]. Cũng như kiến trúc phần cứng, tương ứng kiến trúc phần mềm gồm có 2 phần.

- ***DSP Framework*** (DSP-FW): hỗ trợ cho các DSP và cung cấp môi trường ứng dụng cho các tác vụ thuật toán cũng như đặc tả phần cứng và quản lý tài nguyên để tái cấu hình và khả chuyển.
- ***SmartCam Framework*** (SC-FW): hỗ trợ bộ xử lý mạng và đóng vai trò cầu nối giữa các DSP và cung cấp truy nhập thế giới ngoài. Thêm vào đó, SC-FW còn thu thập các thông tin trạng thái của DSP được cung cấp bởi DSP-FW.



Hình 6. Kiến trúc phần mềm trong SC điển hình

DSP-FW

DSP-FW được xây dựng dựa trên DSP/BIOS, một hệ điều hành thời gian thực được cung cấp bởi Texas Instruments (TI), nó cung cấp các tác vụ bắt tay tĩnh, đồng bộ và đối tượng truyền thông, lớp giao tiếp phần cứng cơ bản như là một phần của *chip-support library* (CSL). DSP-FW cung cấp môi trường hoạt động cho các chức năng của hệ điều hành và các chức năng chuyển tiếp mức thấp.

Trong trường hợp kích hoạt hoặc tái cấu hình phần mềm DSP, chỉ các trình điều khiển và các chức năng cần thiết để bắt đầu DSP được lưu trong lớp trình điều khiển thiết bị cơ sở. Các trình điều khiển này liên kết với hạt nhân của hệ điều hành DSP/BIOS. Do vậy, không cần nâng cấp hay thay thế mô đun này khi sử dụng.

Tuy nhiên, hệ truyền thông điệp và tải hướng động tạo nên mức thấp nhất của hệ thống và là phần chính của lớp trình điều khiển cơ sở.

- ***Basic Driver***

- *Messaging* là trái tim của hệ thống truyền thông các DSP. Nhằm tiếp cận các kênh truyền thông, việc trao đổi dữ liệu và luồng điều khiển giữa các tác vụ được đảm nhận bởi hệ truyền thông điệp. Phụ thuộc vào đích đến, thông điệp có thể chuyển đến cùng DSP hay theo PCI đến DSP khác hoặc Bộ xử lý mạng. Khi thiết kế hệ thống truyền thông điệp cần bám sát kênh truyền thông và đạt hiệu năng cao nhằm tránh sụt giảm hiệu năng bởi tốc độ truyền dữ liệu thấp.
- *Dynamic Loading* Nhằm việc chuyển đổi qua lại giữa các ứng dụng hiện chưa sẵn sàng, hệ thống cần tạm dừng, tải về mã chương trình mới và hệ thống mới được bắt đầu. Tuy nhiên việc này thừa hưởng sự mềm dẻo của hệ thống bởi việc tải và gỡ bỏ ứng dụng, trình điều khiển ngay khi đang hoạt động. Do vậy mô đun tải động được tích hợp vào trong DSP-FW. *Dynamic loader* nằm trong lớp lõi và liên kết với hệ truyền thông điệp đến PCI và tích hợp trực tuyến với phần mềm ứng dụng đang hoạt động. Để tăng cường khả năng tái cấu hình, mọi mô đun trừ mô đun cấp thấp được tải động khi bắt đầu, hoặc theo yêu cầu. Bởi cách này dịch vụ hoặc ứng dụng có thể tải về, gỡ bỏ hoặc thay thế ngay khi đang hoạt động.

- ***Optional Driver*** Các trình điều khiển phần cứng không cần thiết trong quá trình khởi động được tải động khi có nhu cầu. Những trình này gồm có trình điều khiển cảm biến hình, trình điều khiển âm thanh, trình điều khiển video tương tự phải tuân theo giao diện chương trình *DSP-FW*.

- ***Services Layer*** Bao gồm một vài mô đun (dịch vụ) nhằm đáp ứng các mục đích theo dõi và định vị tài nguyên, dữ liệu phân tán và các thuật toán tích hợp theo chuẩn thuật toán do *TI* đề xuất.
    - *Resource Management*: việc quản lý các tài nguyên mức thấp như các kênh DMA, ngắt DMA và ngắt phần cứng bởi CSL như là một phần của DSP/BIOS. Tính toán thời gian thực dữ liệu hình yêu cầu khối lượng lớn dữ liệu truyền nhận do đó mức sử dụng DMA là cao. Vì thế một trong những tác vụ của quản lý tài nghiên là giám sát lưu thông của các kênh và ngắt DMA.
    - *Data Services*: khối dịch vụ dịch vụ cung cấp các dịch vụ *publisher/ subscriber* cho các trình điều khiển thiết bị phần cứng và ứng dụng. Đây là cách một *client* có thể xuất trình dữ liệu hoặc chính bản thân nó như là dữ liệu ẩn dấu một cách tách biệt. Do các dịch vụ dữ liệu nằm ngay trong mỗi DSP nên dữ liệu ban bố có thể được yêu cầu từ mọi DSP trong SC. Do đó, các ứng dụng yêu cầu dữ liệu nhập hoặc cung cấp dữ liệu cho vùng dữ liệu từ các dịch vụ dữ liệu thay vì trình điều khiển thiết bị phần cứng. Ứng dụng đăng ký như là một dịch vụ thành phần và các dịch vụ dữ liệu thiết lập kết nối với nguồn dữ liệu và cung cấp cho thuê bao dữ liệu được yêu cầu.
    - *Extended RF-5*: nhằm chuẩn hóa giao diện các ứng dụng Texas Instruments cung cấp *Reference Framework 5* (RF5), ở đó định nghĩa cách mà các ứng dụng cho phép truy vấn tài nguyên như thế nào và các ứng dụng có thể sử dụng bộ nhớ như thế nào. Do đó, DSP-FW cung cấp giao diện và phương thức để hỗ trợ các ứng dụng tương thích RF5. Tuy nhiên RF5 chỉ định nghĩa những

cấu hình tĩnh. Vì vậy những ứng dụng này cần được gắn nối vào khung ứng dụng DSP-FW để có thể tải động.

- ***Application Layer*** các ứng dụng thực thi trên đỉnh của các lớp được mô tả. Các ứng dụng có thể thực thi trên các mã tương thích SC bởi việc dùng kỹ thuật trừu tượng hóa phần cứng. Nhằm hỗ trợ việc chuyển đổi các ứng dụng giữa các DSP và SC, mọi ứng dụng phải hỗ trợ việc tuần tự hóa dữ liệu tức thời. Với cách này các ứng dụng có thể tái thực thi trên DSP khác bởi việc tải dữ liệu tuần tự hóa trước đó và tiếp tục quá trình tính toán.

Các ứng dụng có thể đơn thuần được xây dựng từ các thuật toán, ví dụ như trong hình 6 là các đoạn thuật toán *motion detection, mpeg-4 encoder, vehicle detection*.

Khi phát triển ứng dụng các thuật toán trên tương ứng với các chế độ hoạt động khác nhau để đảm bảo QoS chung của hệ thống do tầm quan trọng của các ứng dụng này tại các thời điểm là khác nhau.

SC-FW

SC-FW là thành phần quan trọng thứ hai trong kiến trúc phần mềm của SC. SC-FW chạy trên bộ xử lý mạng, được điều hành bởi Linux. SC-FW đáp ứng cho việc quản trị và đồng bộ các DSP cũng như truyền thông điệp giữa các DSP và giữa bộ xử lý mạng với DSP. Một SC-FW được coi là có 3 lớp gồm:

- ***DSP kernel module*** theo kiến trúc Linux, mô đun hạt nhân DSP chỉ cung cấp các chức năng cơ bản nhằm giữ cho hạt nhân của hệ điều hành nhỏ và hoạt động nhanh. Tuy nhiên để cung cấp các chức năng yêu cầu cho *DSP-FW* và các ứng dụng, thì mô đun hạt nhân DSP được chia thành ba khối chức năng.

- *DSP Services* trong hệ thống đa xử lý, nó thường đảm bảo các truy nhập qua lại đến các bộ xử lý. Do đó mô đun hạt nhân DSP cung cấp một giao diện truyền thông điệp cho các DSP nhằm bật chế độ khóa hay mở khóa các bộ xử lý để gán truy nhập đến bộ xử lý đó. Để đáp ứng tốc độ truyền dữ liệu cao, mô đun hạt nhân DSP còn quản lý truy nhập tài nguyên DMA của các DSP.
- *Low-Level Routines* quá trình truyền thông với các DSP đơn giản được nhìn nhận như là trao đổi vùng nhớ. Do đó mô đun hạt nhân cung cấp phương thức đọc và ghi vùng nhớ trong các DSP. Thêm vào đó, mô đun hạt nhân cung cấp thông tin đến các DSP và ứng dụng về số lượng DSP, các đặc trưng và các chức năng để khởi tạo hoặc reset DSP.
- *Message Pre-Dispatch* Nhằm truyền thông điệp hiệu quả giữa các ứng dụng và DSP, mô đun hạt nhân truyền thông điệp đến từ các DSP đến nhóm các ứng dụng, nơi mà cuối cùng sẽ truyền đến ứng dụng đăng ký cần nhận.

- ***DSP Access Library*** (DSPlib) bổ sung các đặc tả phần cứng và chức năng truyền thông cho lớp *kernel-mode*. DSPlib cung cấp giao diện mức cao cho các ứng dụng để tương tác với DSP. Các chức năng của DSPlib bao gồm hệ thống xuất bản/đăng ký thông điệp, điều này cho phép các ứng dụng đăng ký các thông điệp được nhận và gửi đi từ thư viện truy nhập DSP. Thêm vào đó, việc quản lý các ứng dụng tải động bao gồm việc giám sát và điều khiển (bắt đầu/ dừng) các tác vụ cũng là một phần của thư viện này.
- ***Applications*** các ứng dụng tương tác với các DSP bằng cách sử dụng thư viện truy nhập DSP. SC-FW bao gồm tập các ứng dụng nhằm hỗ

trợ DSP-FW, điều khiển và giám sát các DSP và mô đun hạt nhân DSP. Gồm có

- Dịch vụ cầu nối: nghe ngóng các thông báo dịch vụ từ quản lý dịch vụ và chuyển tiếp yêu cầu nhận được đến đúng quản lý dịch vụ cần thiết.
- Thành phần giám sát hiệu năng: thu thập các thông tin hiện trạng của các DSP và bộ xử lý mạng.
- Công cụ quản trị các mô đun tải động.

2.4 TỔNG KẾT VÀ BÀN LUẬN

Chương 2 nêu những giới hạn và định hướng thiết kế SCN, phân tích các khối chức năng phần cứng và phần mềm trong một SC điển hình.

Chỉ với 4 định hướng thiết kế đã nêu, gồm *on-demand*, *dynamic*, *openness*, *robustness* đã đảm bảo hệ thống SCN sẽ có nhiều điểm ưu việt hơn các hệ thống trước, tuy nhiên việc tuân thủ các định hướng này là không dễ dàng do các rào cản về công nghệ và lớp bài toán các nhiệm vụ giám sát là rất đa dạng và phức tạp.

Trong kiến trúc phần cứng của SC phần 2.3 chưa đề cập đến một bổ sung quan trọng và hữu dụng là khối lưu trữ. Việc bổ sung những thiết bị lưu trữ Flash dung lượng rất lớn cỡ 1 Gb, 4 Gb, thậm chí 16 Gb cho các SC là hoàn toàn khả thi và trong mức kinh phí chấp nhận được mà lại mang đến rất nhiều lợi ích cho hệ thống SCN. Nếu như trong khối xử lý xuất hiện nhiều DSP thì tại khối cảm biến đã có những đề xuất sử dụng hai cảm biến hình phục vụ cho những ứng dụng chuyên biệt hoặc những hệ ống kính khác nhau.

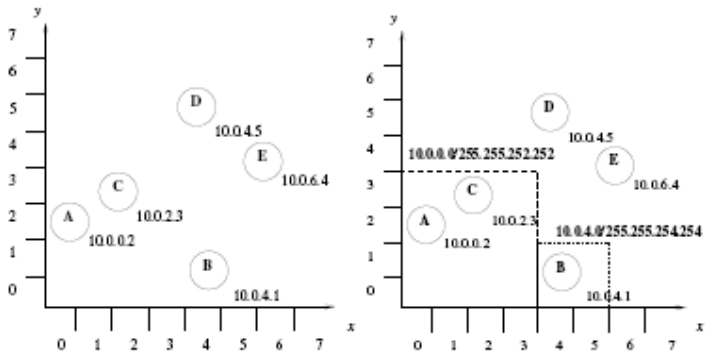
Tuy nhiên những thay đổi thuần túy về số lượng này không ảnh hưởng đến khuôn mẫu *framework* khi xây dựng phần mềm ứng dụng. Hiện nay, kiến trúc phần mềm đã nêu với DSP-FW và SC-FW vẫn đang là khuôn mẫu khi

thiết kế các SC. Cùng với sự phát triển của các vi xử lý, các *firmware* và *framework* mới cũng không ngừng được các nhà sản xuất đưa ra theo hướng đa nhân, đa ứng dụng và tăng cường hiệu năng trên cùng mức tiêu thụ năng lượng (định luật *Moore* mở rộng). Tại thời điểm hiện nay, kiến trúc mới nhất là công nghệ *DaVinci* cùng với *OS MontaVista* do *TexasInstrument* phát triển[8]. *DaVinci Framework* mạng đến nhiều cải thiện cho người lập trình ứng dụng bởi khả năng trừu tượng hóa thông qua các giao tiếp ứng dụng API [PCW_06].

[8] TMS320DM6443 dùng cho ứng dụng giải mã video và TMS320DM6446 nhắm đến các ứng dụng chuyển mã video.

CHƯƠNG 3 : KIẾN TRÚC ĐÁNH ĐỊA CHỈ TỰ DO TRONG SCN

Như là một tín đồ của mạng IP, việc đánh địa chỉ các SC dựa trên dải địa chỉ *IPv4* hay *IPv6* là điều được nghĩ đến đầu tiên khi tiếp cận vấn đề này. Nguyên do là sự xuất hiện của *BS* trong SCN có thể coi như là gốc của dải địa chỉ IP trong lớp. Bên cạnh đó kiến trúc sử dụng *TCP/IP* còn cho phép có thể tác động trực tiếp đến từng SC thông qua địa chỉ toàn cục. [TCP_04]



Hình 7. Cách đánh địa chỉ IP theo vị trí SC

Hình 7 là một ví dụ về việc gán địa chỉ IP cho mạng dựa vào vị trí không gian của nút. Địa chỉ của mỗi SC được cấu thành từ tọa độ (*x*,*y*) của nó và coi như là 2 *octet* thấp của địa chỉ IP.

SC C(*2*,*3*) có địa chỉ là 10.0.2.3. Như vậy có nghĩa là thông tin vị trí của SC đã được mã hóa vào trong địa chỉ IP. Do vậy việc phân miền con trong mạng cũng dễ dàng thông qua việc *subneting* dải địa chỉ 10.0.0.0/8.

Như trong hình thì A(*0*,*2*) và C(*2*,*3*) được xếp vào cùng *subnet* 10.0.0.0/255.255.252.252. *Subnet* này dễ dàng cho việc kiểm soát các gói tin quảng bá trong miền trên.

Các giao diện mạng của khối truyền thông trong SC đã được định sẵn địa chỉ MAC từ nhà sản xuất. Kỹ thuật này cho phép ánh xạ qua lại giữa địa chỉ MAC và địa chỉ IP.

Thoạt nhìn, kỹ thuật này tạo cảm giác là có thể dễ dàng đánh địa chỉ các nút SC dựa trên địa chỉ IP và tận dụng được những ưu điểm của truyền thông gói tin có *IP header*. Tuy nhiên kỹ thuật này không hiệu quả trong SCN.

Thứ nhất, SCN là hệ thống của các SC có quan hệ không gian thực 3D, trong khi lưới IP là 2D. do vậy khi phẳng hóa không đảm bảo tại một vị trí trong lưới chỉ có một SC.

Thứ hai, các SC trong SCN không bắt buộc phải có thiết vị cố định. Trong trường hợp SC dịch chuyển giữa các vùng thì địa chỉ IP của nó cũng thay đổi theo dẫn đến khả năng xung đột địa chỉ hoặc phải cập nhật lại địa chỉ SC đó cho các SC có liên quan.

Thứ ba, nếu chọn thang chia không tốt sẽ dẫn đến dư thừa dải địa chỉ hoặc có quá nhiều SC trong cùng *subnet*.

Thứ tư, hệ thống SCN là *dynamic* và *on-demand* do vậy một địa chỉ toàn cục trong toàn hệ thống là không cần thiết trong nhiều trường hợp. Việc truyền thông giữa hai SC ở tương đối xa nhau là không thường xuyên nên không cần chúng phải biết được địa chỉ toàn cục và *subnet* của nhau để làm gì.

Cuối cùng, với mô hình trên, hệ thống coi như là chỉ có một *BS* quản lý 10.0.0.0. Việc bổ sung hay chuyển biến SC thành *BS* gây ảnh hưởng đến toàn hệ thống.

Do vậy tôi đề xuất giải pháp cho nhiệm vụ đánh địa chỉ SC trong SCN là chọn lựa một trong hai phương pháp là:

- Sử dụng phương pháp cấu hình *Zeroconf* cho các mạng quy mô nhỏ và ít biến động.
- Phương pháp sử dụng địa chỉ cục bộ và đánh địa chỉ tự do kiến trúc AFA. Nền tảng của kỹ thuật này là thay vì các địa chỉ tuyệt đối và toàn

cục được cung cấp trực tiếp cho ứng dụng thì sẽ qua một bước trung gian là biến đổi thành địa chỉ tự do.

3.1 ZEROCONF

Zeroconf được đề xuất áp dụng nhằm tăng tính dễ sử dụng cho các thiết bị mạng dựa trên công nghệ IP.

Các sản phẩm có hỗ trợ *Zeroconf* hiện nay vẫn còn trong giai đoạn phát triển. Tuy nhiên, có hai lợi điểm của giải pháp này là có thể sử dụng mã nguồn mở để phát triển thêm tính năng *Zeroconf* cho các sản phẩm hiện tại và các chip nhúng ngày nay cũng có thể bổ sung thêm chức năng *Zeroconf* bằng cách cập nhật lại *firmware*.

Triển khai *Zeroconf* trong SCN

Nhìn nhận SC như một thiết bị mạng IP thông thường, việc triển khai *Zeroconf* không phải là một công nghệ mới mà là một kỹ thuật liên kết ba công nghệ hiện có để tạo nên. *Zeroconf* bao gồm công nghệ đánh địa chỉ liên kết cục bộ, *multicast DNS* (*mDNS*) và phát hiện dịch vụ thông qua *DNS* [ZERO_01].

Zeroconf sử dụng phương pháp đặt địa chỉ liên kết cục bộ để thiết lập địa chỉ[9]. Tuy nhiên còn có một ràng buộc khác là SCN sử dụng truyền thông *ad-hoc* như là giải pháp chính để trao đổi thông tin, nên việc gói tin ARP chậm có phản hồi do phải đi qua nhiều *hop* là có thể xảy ra. Vì vậy giới hạn số lượng SC là dưới 100 là hợp lý trong SCN khi sử dụng giải pháp này.

---

[9] Theo RFC3927 thì *link-local address* là: "Cấp IP trong khoảng 169.254.1.0 đến 169.254.254.255. Nó được dùng để gán địa chỉ IP cho thiết bị trong mạng IP mà không có một phương thức gán nào được sử dụng trước đó, ví dụ như *DHCP* server. Sau khi chọn ngẫu nhiên một địa chỉ, một gói tin ARP với IP đó sẽ được truyền đi trong mạng để kiểm tra xem nó đã tồn tại chưa. Nếu không có phản hồi thì IP đó được gán cho thiết bị, bằng không một IP khác được lựa chọn và lặp lại quá trình gửi ARP".
Cũng theo RFC 3927 thì Không gian địa chỉ này dùng tốt cho các mạng có số lượng đến 100, Khi quy mô lên đến 1000 thiết bị thì phương thức này vẫn làm việc tốt. Nguyên nhân là do nếu trong mạng có đến 1000 thiết bị thì máy của chúng ta vẫn có 98% cơ hội chọn được địa chỉ trống trong lần đầu tiên, và có đến 99,96% cơ hội cho lần thứ hai. Và xác suất khả năng chọn 10 lần mà không tìm được địa chỉ là 1 trong 10^{17}.

Zeroconf sử dụng *mDNS* thay thế cho *DNS* khi dịch vụ này không tồn tại trong mạng[10]. Để phân biệt các tên miền cục bộ so với các tên miền đã tồn tại khác, *Zeroconf* cung cấp một tên miền cấp cao giả có tên là *.local*[11]. Quy tắc đặt tên đề nghị là chọn một tên sao cho đó có thể là mã tổng hợp hoặc gợi nhớ đến chức năng của các SC đó trong mạng (ví dụ tên nhiệm vụ giám sát có QoS cao nhất trong tại SC đó hoặc góc hướng, số *hop* truyền đến *BS*).

Cơ chế tìm kiếm các tên trong nhóm *.local* luôn là *multicast*. Các SC có thể chọn cơ chế *multicast* để truy vấn trực tiếp đến các thiết bị có tên tương ứng. Các thiết bị giao tiếp ngang hàng với nhau và điểm đặc biệt là vẫn có thể truy cập đến một thiết bị dựa trên tên đã đặt ngầm định cho nó trước đó[12].

Họ giao thức DNS có định nghĩa một số dạng câu truy vấn để dành cung cấp thông tin cho các dịch vụ chạy trên đó có tên gọi là SRV. Việc bổ sung thông tin này cho phép chúng ta tìm kiếm nhanh về các dịch vụ hiện có trên mạng.

Ví dụ thông tin về dịch vụ web có dạng như sau *_http._tcp.example.com* thay vì sử dụng một tên giả cho dịch vụ này như là *www.example.com*. Phần _tcp trong tên chỉ ra đó là dịch vụ này chạy trên giao thức TCP, không phải UDP. Việc thêm giao thức vận chuyển vào tên giúp cho chúng ta xác định được lưu lượng yêu cầu và và các chính sách về cân bằng tải thích hợp.

---

[10] Các thiết bị có hỗ trợ *mDNS* sẽ liên lạc với nhau thông qua tên tham chiếu của chúng. Về cơ bản, triển khai *mDNS* giống như cơ chế thiết lập địa chỉ liên kết cục bộ, đầu tiên *mDNS* sẽ chọn một tên và truy vấn trong mạng cho đến khi không còn sự xung đột thì lấy tên này làm tên thiết bị.

[11] Giống như giá trị địa chỉ *169.254.* là các địa chỉ cục bộ và không phải là địa chỉ duy nhất, tên miền thuộc miền *.local* cũng chỉ mang ý nghĩa cục bộ. Điểm thuận lợi của các tên miền này là không cần phải có *host* đứng ra phân phối. Tuy nhiên, cũng chính vì điều này mà một SC không có quyền chiếm giữ một tên nào tùy thích và không cho SC khác sử dụng nó, nhưng nó cũng bao gồm một vài quy tắc cho phép các thiết bị phát hiện ra sự xung đột khi mà có hai thiết bị trùng tên tại cùng một thời điểm.

[12] Bất cứ truy vấn nào kết thúc bằng *.local* đều gửi đến địa chỉ 224.0.0.251 là địa chỉ IPv4 dành riêng cho *mDNS*. Các yêu cầu được gửi đến địa chỉ này và nếu thiết bị nào trùng tên với tên yêu cầu thì sẽ trả lời thông tin phản hồi cho phía gửi.Các truy vấn *mDNS* gồm có ba loại là: một câu truy vấn có một câu trả lời, một câu truy vấn có nhiều câu trả lời và câu truy vấn liên tục.

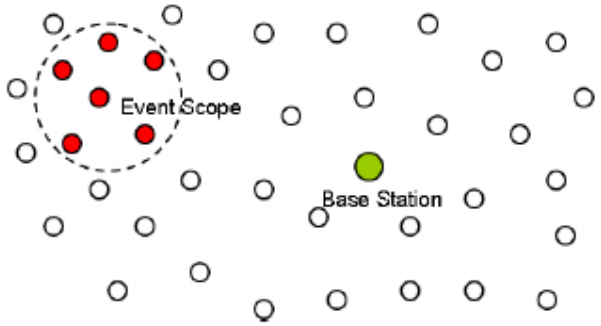
Tóm lại, *Zeroconf* cung cấp cho các thiết bị phần cứng và phần mềm các chức năng sau:

- Chắc chắn có được một địa chỉ IP để liên lạc trong mạng IP.
- Khả năng phát hiện các dịch vụ tồn tại trong mạng.
- Quảng bá rộng rãi dịch vụ hiện có của thiết bị với các thiết bị khác.

Zeroconf không yêu cầu người lập trình hay các nhà sản xuất phần cứng phải tuân theo một chuẩn các giao thức định trước hay là sử dụng các kiểu dữ liệu định trước nào. Do *Zeroconf* là một giao thức ở mức tổng quát, cung cấp cơ sở cho các ứng dụng dựa trên mạng IP hoạt động, nên khi áp dụng cách đánh địa chỉ này trong SCN thì các vấn đề có liên quan như đồng bộ, định tuyến, an ninh ... có thể áp dụng các thuật toán và ứng dụng tương tự như của môi trường PC.

3.2 KIẾN TRÚC ĐÁNH ĐỊA CHỈ TỰ DO AFA

Trong nghiên cứu và công nghiệp, người ta đã phát triển nhiều lớp hệ thống sử dụng cách đặt tên theo thuộc tính. Các kiểu đặt tên này thường là để phục vụ định tuyến *end-to-end*, nên không hiệu quả và không tuân thủ với định hướng thiết kế SCN. Do vậy kỹ thuật AFA[13] được đề xuất sử dụng cho những hệ thống phân tán như SCN.



Hình 8. Mô hình hệ thống hướng sự kiện [CG_06]

[13] An Address Free Naming Architecture

Cũng tương tự như việc sử dụng dịch vụ tên miền DNS, kỹ thuật AFA SCN được đề xuất dựa trên hai cơ sở là:

- ***Attribute-based naming*** là một biến hóa phù hợp với từng ngữ cảnh. Nó được sử dụng trong SCN như gợi nhớ một đề xuất giải quyết bài toán[14]. Loại tên dữ liệu này được đặt theo đặc tả của ứng dụng. Một định nghĩa hiệu quả chuyển tên, địa chỉ thậm chí cả định tuyến từ lớp mạng đến ứng dụng. Điều này là khác với các mô hình dịch vụ dạng cơ sở Internet khi mà ở đó chuyển tiếp gói *end-to-end*. Tuy thế nó mô tả gần tiếp cận với ngữ cảnh trong SCN.
- ***Randomized transaction identifiers*** nội dung ý tưởng được phát triển rất đơn giản là khi một nhận dạng đảm bảo đơn nhất là cần thiết, tại một thời điểm chọn lựa ngẫu nhiên có khả năng một nhận dạng đơn nhất được sử dụng. Dĩ nhiên sẽ có khả năng hai SC có cùng một nhận dạng tại cùng một thời điểm tuy nhiên người ta sẽ không cố gắng giải quyết những xung đột loại này mà thay vào đó là chọn một nhận dạng ngẫu nhiên mới cho giao dịch. Mỗi ứng dụng hay dịch vụ mạng sẽ có kiểu định nghĩa riêng của nó cho mỗi loại giao dịch. Xung đột nhận dạng sẽ dẫn đến mất giao dịch và được coi như là một tổn thất bình thường khác. Việc phát hiện xung đột sẽ do ứng dụng đảm nhận hoặc có thể đơn thuần chỉ là so sánh số tuần tự trong các gói dữ liệu[15].

Người ta đã chỉ ra là việc áp dụng AFA mang lại rất nhiều hiệu quả trong SCN [BEWS_01] và phát triển kỹ thuật tùy biến tên theo ứng dụng trong hệ thống SCN như sau:

Chọn xác thực trong AFA

[14] Thay vì đặt câu hỏi dạng như: "Có chuyển động được phát hiện ở SC số #27.201.3.97" thì câu hỏi có thể được đặt ra là "Có đối tượng chuyển động được phát hiện ở góc đông bắc?' hay "Ở đâu có đối tượng chuyển động mới được phát hiện".

[15] Tương tự với câu lệnh "#27.201.3.97 hãy truyền dữ liệu" thì ứng dụng đưa ra câu lệnh là "SC nào vừa truyền có xác thực là A thì hãy tiếp tục".

Trong thực tế, phần lớn các truyền thông trong SCN thường chỉ xảy ra trong nhóm cục bộ, với sự tham gia của số ít SC. Các tương tác thường có xu hướng lân cận và chỉ có số ít luồng dữ liệu truyền qua nhiều *hop*.

Việc sử dụng AFA có lợi điểm là các SC ở xa nhau có thể sử dụng trùng xác thực tại cùng thời điểm và các SC ở lân cận có thể sử dụng trùng xác thực tại thời điểm khác nhau. Nghĩa là xác thực địa chỉ AFA chỉ là đơn nhất tại một vùng lân cận hoặc một khoảng thời gian. Chính nhờ điều này mà xác thực địa chỉ AFA ngắn hơn rất nhiều so với địa chỉ đơn nhất toàn cục.

Một *heuristic* có thể được sử dụng để làm tăng hiệu năng đó là kỹ thuật *listening*. Có nghĩa là thay vì chọn ngẫu nhiên xác thực, SC có thể sử dụng xác thực vừa được sử dụng bởi việc nghe ngóng gói tin vừa truyền qua. Điều này thì không đảm bảo sẽ làm việc hoàn hảo, dĩ nhiên do hai SC không cùng quãng cách có thể cùng chọn một xác thực địa chỉ do một SC thứ ba giữa chúng truyền đi. Việc mất gói cũng có thể gây nên cản trở *listening*. Thêm vào đó một vài SC có thể chọn tối thiểu thời gian nghe ngóng do yêu cầu tín hiệu năng lượng của sóng vô tuyến. Vì những hạn chế này, việc nghe ngóng không giúp làm giảm bảng lưu số xác thực nhưng giúp sử dụng hiệu quả hơn nguồn tài nguyên hạn chế này.

Có những tình huống mà một SC chỉ định cần được xác thực, ví dụ như mục đích gỡ lỗi hoặc bảo trì. Khi đó một địa chỉ đơn nhất toàn cục cần được sử dụng ví dụ như địa chỉ MAC. Trong AFA người ta cũng không phản đối việc gán địa chỉ toàn cục cho SC, mà thay vào đó người ta đề xuất là nên sử địa chỉ đơn nhất một cách tiết kiệm [AFA_00]. Một nút có xác thực đơn nhất có thể gửi dữ liệu theo yêu cầu, thay vì phải đọc *header* của mọi gói tin.

Hiệu năng truyền E được tính theo công thức:

$$E = (\textit{Số bít truyền hữu dụng}) / (\textit{Tổng số bít truyền}) \qquad (3.2.1)$$

Trong mô hình ở đây bít truyền là gói tin bao gồm phần *header* và *data*. Chi phí của truyền gói tin là giá truyền cả *header* và *data*.

Mọi gói là một phần của giao dịch, chúng ta giả định mật độ giao dịch *T* là số trung bình của các giao dịch đồng thời thu nhận được tại một điểm đơn lẻ trong mạng. Một giới hạn của mô hình ở đây là tham số đơn lẻ *T* không đủ để mô tả mọi trạng thái - hai giao dịch dài có đặc điểm xung đột khác một giao dịch dài với một loại các giao dịch ngắn, thậm chí cả trường hợp *T = 2* trong cả hai trường hợp. Để đơn giản hóa phân tích, và phải chấp nhận là mô hình ở đây thiếu tính tổng quát, ta giả định mọi giao dịch phát sinh tại cùng thời điểm.

Header của gói chỉ có xác thực giao dịch. Cốt lõi trong mô hình ở đây là giả định giao dịch thành công (hữu dụng) khi và chỉ khi nguồn sử dụng một xác thực là đơn nhất đối với mội giao dịch khác tại cùng điểm trong mạng trong suốt quá trình giao dịch. Giao dịch thất bại dựa vào xung đột xác thực gây giảm hiệu năng bởi vì giá của bít truyền phải sử dụng mà có bút hữu dụng.

$$E_{static} = \frac{D}{D+H} \tag{3.2.2}$$

Công thức trên chỉ ra tỷ lệ của số bít data trên toàn giao dịch chứ không phải gói đơn lẻ.

Trong kiến trúc AFA, các giao dịch không được đảm bảo luôn thành công. Giao dịch chỉ có cơ may thành công nhất định do còn phụ thuộc vào các xung đột xác thực. Giả định rằng toàn bộ giao dịch dù thành công hay thất bại cũng chỉ phụ thuộc vào việc mất gói và coi các giao dịch có cùng độ dài thì

$$E_{afa} = \frac{D \times P(Success)}{D+H} \tag{3.2.3}$$

Trong đó cơ may thành công *P* được tính theo công thức:

$$P(Success) = \left(1 - \frac{1}{2^H}\right)^{2(T-1)} \quad (3.2.4)$$

Sự tác động của *heuristic* nghe ngóng đóng vai trò quyết định giá trị của *P*.

Như vậy, thay vì sử dụng xác thực có độ dài bit cố định 32 bit (IPv4) hay 48 bit (MAC), với kiến trúc đánh địa chỉ tự do độ dài bít xác thực có thể ngắn hơn nhiều. Tương ứng với mỗi độ dài bit đó là cơ may thành công của giao dịch.

3.3 TỔNG KẾT VÀ BÀN LUẬN

Chương 3 tập trung bàn luận về vấn đề đánh địa chỉ cho các SC trong mạng SCN. Tuân thủ định hướng thiết kế đã đề ra trong chương 2, hai hướng tiếp cận giải quyết là *Zeroconf* và AFA đã được nêu ra tại đây.

Zeroconf đã được chuẩn hóa trong RFC3927 và phù hợp với những SCN quy mô nhỏ và có ít biến động về thiết vị. Lợi điểm của phương pháp này là tận dụng được các ưu thế của gói tin và các dịch vụ có sẵn của IPv4.

Cách tiếp cận AFA là giải pháp cho bài toán tổng quát, rất hiệu quả trong các SCN quy mô lớn và trải rộng. AFA là hướng tiếp cận đánh địa chỉ dựa trên sự kiện, thay vì thiết vị. Nền tảng của AFA là *direct-diffusion* và *local-address*.

Từ các nhận định trên, hướng nghiên cứu đề xuất là khả năng phối hợp cả hai cách đánh địa chỉ này trong SCN. Vấn đề tương tự cũng được đặt ra khi giải quyết các bài toán về đồng bộ, định tuyến ... trong SCN.

CHƯƠNG 4 : ĐỒNG BỘ BỘ ĐỆM TRONG SCN

Vấn đề đồng bộ bộ đệm luôn được đặt ra trong các hệ thống phân tán. Trong những hệ thống như SCN, quá trình truyền thông có những điểm khác biệt so với hệ thống phân tán cổ điển [OGS_03]:

- Các truyền thông trong SCN có xu hướng quảng bá gần, thay vì truyền thông điểm - điểm. Điều này có nghĩa là mỗi truyền thông có thể có vài nút nhận. Đây là một đặc điểm của truyền thông không dây, cần được lưu ý khi phát triển ứng dụng.
- Vùng phủ sóng vô tuyến của mỗi SC là nhỏ so với quy mô không gian SCN.
- Trễ giữa nhãn thời gian và việc gửi một gói mang nhiều ý nghĩa hơn trễ giữa nhận và nhãn thời gian của nó. Điều này có nghĩa là gói có xu hướng được gửi xa nhất có thể hơn là mong muốn nó được nhận chính thức.

Điều này có ảnh hưởng không nhỏ khi chọn lựa giải pháp thiết kế, giải pháp đồng bộ các SC trong SCN. Đặc điểm truyền thông dẫn đến những rào cản sau:

- ***Những rào cản về năng lượng*** do đặc thù của dữ liệu hình và bài toán giám sát, trong nhiều trường hợp số lượng ít *active* SC, các SC còn lại cần chuyển trạng thái sang chế độ *sleep* SC tiết kiệm năng lượng để kéo dài thời gian hoạt động của hệ thống. Việc liên tục đồng bộ bộ đệm cho những SC này là lãng phí.
- ***Đặc tính động của hệ thống*** có nhiều sư kiện xảy đến với điểm nút camera ví dụ như: thêm, bớt, lỗi ... nên chọn phương pháp đồng bộ bộ đệm trong hay ngoài; chủ động hay bị động; để thích nghi với các sự kiện động cũng là một vấn đề được đặt ra.

- ***Đặc điểm truyền thông của hệ thống*** do đã chọn lựa truyền thông không dây kiểu *ad-hoc* là chế độ truyền thông chính trong hệ thống, nên các sự thay đổi của môi trường có tác động mạnh đến tương tác giữa các nút[16].

Dẫu vậy nhưng đồng bộ bộ đếm giữa các SC trong SCN là không thể bỏ qua[RTS_02], vì các nguyên nhân sau:

- ***Nhiệm vụ của s_clu*** chia sẻ để xử lý thông tin hình theo thời gian thực.
  - *Object tracking* theo dõi đối tượng dựa trên các yếu tố kích thước, hình dạng, góc hướng, vị trí, vận tốc và gia tốc của đối tượng được xác định bởi các SC đặt tại những vị trí khác nhau.
  - *Consistent state updates* trạng thái gần đây nhất của đối tượng sẽ thu nhận được tại SC bắt hình gần hiện tại nhất.
  - *Duplicate detection* thời gian xảy ra sự kiện giúp các SC phát hiện được là chúng quan sát được hai góc hướng của cùng một đối tượng hay hai đối tượng khác nhau.
- ***An ninh truyền thông*** Đồng bộ bộ đếm là điều kiện tiên quyết để áp dụng *μTESLA* trong xác thực các gói tin truyền quảng bá.
- ***Hợp nhất dữ liệu*** với việc khai phá và hợp nhất dữ liệu trong hệ thống thì việc đồng bộ thời gian để xác định chính xác các dữ liệu nào cần liên kết khi có yêu cầu thám sát mức cao.

Các giải pháp đồng bộ thời gian trong mạng truyền thống và hệ phân tán đã được nghiên cứu và xây ứng dụng từ lâu. Ví dụ như NTP được sử dụng rộng rãi trong môi trường Internet. Tuy nhiên áp dụng trực tiếp các phương pháp đó vào mạng SCN là vấn đề không đơn giản.

---

[16] Theo số liệu thực nghiệm, người ta đã chỉ ra là có đến hơn 20% thông điệp trong mạng không dây mật độ cao bị *miss* có bao gồm cả tín hiệu truyền đồng bộ thời gian.

4.1 CÁC GIẢI PHÁP TRUYỀN THỐNG

Sau nhiều năm, đã có nhiều giao thức được thiết kế cho đồng bộ bộ đếm vật lý trong mạng máy tính. Các giao thức này có điểm chung là: giao thức truyền thông điệp đơn giản không hướng kết nối dạng như truyền thông tin thời gian giữa *client* và một vài *server* và sử dụng thuật toán phía *client* để cập nhật thông tin thời gian nhận được từ phía *server*. *Server* được đề cập đến ở đây có thể nằm trong hoặc nằm ngoài hệ thống.

Giao thức NTP là khả mở, tự cấu hình để tạo nên một thang thời gian toàn cục trong mạng *multi-hop*, chịu lỗi, an ninh và đa năng. Trong hàng thập kỷ NTP là phương thức chính để duy trì nhịp thời gian của Internet. Do vậy xây dựng một giao thức tựa như NTP dùng cho SCN là điều mà nhiều người hay đặt ra đầu tiên. Tuy nhiên trên thực tế thì Internet và SCN là không giống nhau nên có thể NTP là hoàn hảo trên Internet nhưng không thực sự tối ưu trong SCN do gặp phải các rào cản đã nêu ở trên. Ví dụ như:

- ***Vấn đề tiêu thụ năng lượng*** NTP giả định là CPU luôn sẵn sàng và luôn nghe ngóng thông tin đồng bộ.
- ***Sự khác biệt giữa Single-Hop và Multi-Hop*** Truyền thông trong SCN qua nhiều *hop* nên trễ *end-to-end* là lớn hơn rất nhiều so với *single-hop*. Điều này gây nên khó khăn khi áp dụng phương pháp mà giả định ban đầu là kết nối toàn bộ hoặc topo trễ ít như NTP.
- ***Vấn đề hướng cấu trúc và phi cấu trúc*** NTP cho phép thiết lập cây đồng bộ thời gian có cấu trúc với gốc của các và nút có thể ở ngoài hệ thống mạng. Tuy nhiên SCN cần đồng bộ bộ đếm khi có tương tác truyền thông giữa các nút chứ không bắt buộc phải liên tục đồng bộ thời gian cho toàn hệ thống.
- ***Vấn đề kết nối và không kết nối*** nút chuyển vị, nút lỗi là những đặc tính động trong SCN. Điều này có nghĩa là sơ đồ mạng cũng biến động.

Dữ liệu có thể chuyển dịch theo cách nằm trong SC và di chuyển vật lý đến gần hoặc xa điểm cần nhận hơn, điều này có nghĩa là khó kiểm soát trễ truyền thông ứng với đồng bộ thời gian.

Do vậy nếu bắt buộc phải sử dụng NTP cần phải cân nhắc chiến lược phù hợp.

4.2 THIẾT KẾ GIẢI PHÁP ĐỒNG BỘ BỘ ĐẾM TRONG SCN

Giải pháp tối ưu cho đáp ứng mọi trường hợp trong mạng SCN hiện nay vẫn chưa được tìm ra, tuy nhiên có một vài phương pháp cho kết quả khả quan trong những nhóm trường hợp riêng. Mỗi phương pháp đều có ưu và nhược điểm riêng.

Kỹ thuật *Global clock syncronization*: đồng bộ nhóm [GCS_04]

Các SC có nhu cầu truyền thông được sắp xếp thành một chu trình. Một gói tin tham chiếu sẽ được truyền đi trên chu trình đó (lần thứ nhất), qua mỗi SC sẽ cập nhật thêm giá trị bộ đếm tại đó và khi quay trở lại SC xuất phát thì sẽ được tính toán và thông báo lại theo chu trình đó (lần thứ hai). Các SC cập nhật bộ đếm phụ thuộc vào vị trí trong chu trình và tổng thời gian xoay vòng.

Một cải tiến được đề nghị là chia nhỏ SCN thành các nhóm *s_clu*, mỗi *s_clu* chọn một SC đại diện và đồng bộ theo cách trên và các SC khác trong *s_clu* đồng bộ với SC đại diện đó bằng một vài kỹ thuật bổ trợ khác.

Kỹ thuật *No global timescale*: không lưu bộ đếm toàn cục

Như đã đề cập, việc lưu giữ thời gian toàn cục trong một mạng lớn chỉ có giá trị khi có nhiều nguồn sinh thời gian và phân tán trong toàn mạng. Trong những mạng phi cấu trúc dạng SCN, không từ chối việc có một vài những nguồn sinh thời gian trong mạng để sử dụng những thuật toán cổ điển. Tuy nhiên giải pháp tốt nhất là mỗi nút tự giữ *timescale*.

Mỗi nút không đặt đồng hồ cưỡng bức hoặc điều chỉnh tần số của nó mà để nó chạy ở tốc độ tự nhiên, Đồng bộ thời gian bởi việc xây dựng bảng tham số quan hệ giữa pha và tần số của bộ đếm cục bộ và bộ đếm khác trong mạng.

Kỹ thuật này có một số lợi điểm. Thứ nhất: lỗi đồng bộ giữa hai nút liên quan đến khoảng cách giữa chúng chứ không phải dùng bộ đếm khách chủ. Thứ hai, mỗi nút có bộ đếm riêng được tận dụng cho nhiều thuật toán xử lý tín hiệu nội tại. Cuối cùng, không cần liên tục hiệu chỉnh bộ đếm bởi CPU hoặc *kernel* và khi hiệu chỉnh có thể sử dụng NTP.

Kỹ thuật *post facto syncronization*: đồng bộ muộn [TS_01]

Kỹ thuật đồng bộ truyền thống đồng bộ bộ đếm tại nút theo *priori*: các bộ đếm được đồng bộ trước khi có sự kiện xảy ra và xuất hiện nhãn thời gian. Điều này dẫn đến việc trễ truyền do chờ đợi đồng bộ thời gian. Người ta xây dựng phương pháp "đồng bộ sau" tức là bộ đếm chạy không đồng bộ ở tốc độ tự nhiên của chúng. Khi có nhãn thời gian từ bộ đếm khác cần so sánh chúng mới được đồng nhất sau sự kiện. Hướng tiếp cận này có thể hỗ trợ cho việc truyền thông điệp chuyển tiếp, nơi mà không cần có kết nối mạng liên tục với điểm phát sinh sự kiện.

Kỹ thuật đa giao tiếp

Trong SCN không hạn chế việc tồn tại một vài SC được đồng bộ thời gian ngoài, ví dụ sử dụng NTPv3 để lấy thời gian từ Internet, thu tín hiệu pps[17] từ vệ tinh GPS. Các SC như thế này có thể dùng là điểm tham chiếu phục vụ đồng bộ thời gian theo phương thức RBS, mặc dù không cần sử dụng đến giá trị thời gian toàn cục đó.

Kỹ thuật RBS

[17] *pulse per second*

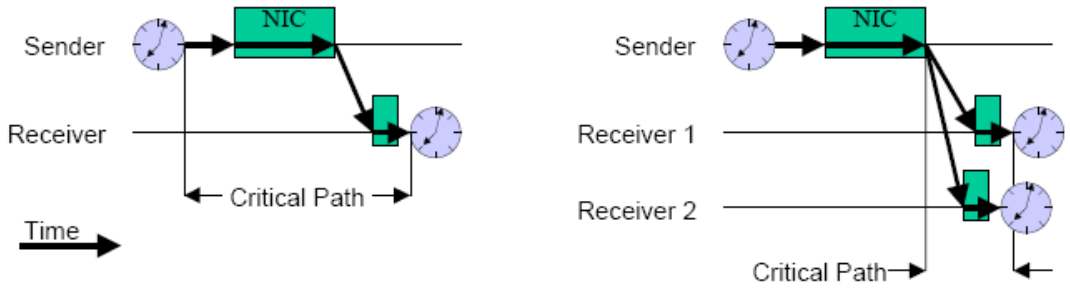
Đây là kỹ thuật được tập trung bàn luận trong SCN vì những ưu điểm khi áp dụng trong hệ thống này.

Các nguồn gây lỗi đồng bộ thời gian thường là ngẫu nhiên. Do các dự đoán bị trễ bởi những sự kiện ngẫu nhiên dẫn tới việc bất đối xứng của vòng truyền thông điệp. Kopetz và Schwabl phân tích và chia ra 4 khối thời gian xảy ra trong quá trình truyền thông [RBS_02]:

- ***Send Time*** thời gian cần thiết để nút gửi hoàn thành thông điệp.
- ***Access Time*** trễ xảy ra do việc chờ đợi truy nhập kênh truyền.
- ***Propagation Time*** khoảng thời gian cần thiết để truyền từ nút gửi đến nút nhận khi chia sẻ cùng một môi trường truyền dẫn.
- ***Receive Time*** quá trình xử lý yêu cầu giao tiếp mạng của nút nhận nhận thông điệp từ kênh và thông báo đến *host*.

Trong cả bốn khối thời gian trên đều có khả năng phát sinh lỗi gây mất đồng bộ bộ đếm. Phương pháp RBS đưa ra một hướng tiếp cận vấn đề khác là thay vì cố gắng dự đoán lỗi, người ta sử dụng các kênh truyền quảng bá có thể trong nhiều lớp vật lý mạng khác nhau để loại bỏ các đường găng do nhận thấy thông điệp được quảng bá ở lớp vật lý sẽ đến một tập các nút nhận mà trễ chênh lệch giữa chúng là rất nhỏ. Nền tảng của kỹ thuật RBS là thông điệp quảng bá chỉ được sử dụng để đồng bộ một tập các nút nhận với một nút khác, thay vì các phương pháp truyền thống là đồng bộ trực tiếp giữa nút nhận và nút gửi.

Trong quảng bá RBS luôn sử dụng khái niệm "quan hệ tham chiếu thời gian" chứ không sử dụng "giá trị thời gian tuyệt đối", tại đó các nút nhận được đồng bộ bởi gói "tham chiếu"



Hình 9. Đường găng trong đồng bộ thời gian truyền thống và RBS

Trong những giao thức truyền thống làm việc trong mạng LAN, thời gian trễ không dự đoán được phần lớn nằm ở *Send Time* (khi đọc bộ đếm gửi để chuyển gói đến NIC) và *Access Time* (trễ tại NIC chờ đến khi kênh truyền rỗi). *Receive Time* là khá nhỏ so với *Send Time* do chỉ cần đọc ngắt thời gian. Trong RBS, đường găng được thu hẹp lại là do chỉ tính thời gian từ khi có gói trong kênh truyền so với lần đọc bộ đếm trước.

Các bàn luận trên chỉ bao gồm các đồng bộ trong nội bộ hệ thống. Tuy nhiên, có nhiều ứng dụng độc lập đòi hỏi thời gian tuyệt đối được tham chiếu từ những nguồn bên ngoài ví dụ như UTC, những trạm phát sóng ngắn WWVB hay từ vệ tinh GPS. Bộ thu tín hiệu vệ tinh có thể tiếp nhận những xung *pps* từ những hệ thống này tại thời điểm bắt đầu mỗi giây.

RBS cung cấp khả năng đồng bộ mạng lưới chính xác với những nguồn thời gian ngoài loại này. Việc gắn bộ tiếp nhận tín hiệu GPS kết nối với một trong các nút của mạng *multi-hop* thì *pps* xuất phát từ nút đó có thể coi thành tham chiếu quảng bá cho các nút khác.

4.3 TỔNG KẾT VÀ BÀN LUẬN

Chương 4 bàn luận về vấn đề đồng bộ bộ đếm trong hệ thống SCN. Đây là một trong những vấn đề cơ bản nhất trong các hệ thống phân tán không riêng gì SCN.

Hiện nay, giải pháp đồng bộ toàn năng và hiệu quả cho SCN vẫn chưa được tìm thấy do những đặc điểm động của hệ thống này.

Ý tưởng đồng bộ RTS là không mới (được đề xuất từ năm 1992) nhưng tường minh, đơn giản và dễ dàng triển khai ứng dụng. Phương pháp này thuộc loại chấp nhận đồng bộ muộn. Tuy nhiên RTS không cung cấp được giải pháp đồng bộ toàn hệ thống và khó triển khai được trong các mạng trải rộng. Điều này gây một số trở ngại cho các ứng dụng có tra cứu thông tin quá khứ được lưu trữ cục bộ tại các SC.

Giải pháp đồng bộ theo nhóm là một hướng tiếp cận tổng hợp. Tuy nhiên đây hoàn toàn là ý tưởng và việc triển khai trong một hệ thống có biến động về thiết vị và thiết bị như SCN thì việc đảm bảo chu trình truyền tham chiếu đồng bộ đi hai lượt là như nhau là điều không chắc chắn. Đề xuất sử dụng bộ đếm cục bộ trong nhóm SC liên kết cũng chưa khẳng định được sự ưu việt hơn so với sử dụng NTP (NTPv3 còn được hỗ trợ từ trong *kernel* của *Linux OS*).

Do vậy, việc phân chia chức năng cho SC thành *BS*, proxy cũng là một gợi ý cho việc đồng bộ bộ đếm trong SCN. Các *BS* có thể là điểm tiếp nhận tín hiệu đồng bộ ngoài (như từ UTC, GPS, Internet ...) và các *proxy* đóng vai trò phát tham chiếu quảng bá cho các SC. Việc phối hợp đồng bộ RBS cho mức giữa SC và *proxy*, đa đồng bộ cho các mức còn lại và toàn hệ thống dẫn đến nhiều ưu điểm và khá hiệu quả trong các ứng dụng tra cứu.

Ngoài ra cùng xu hướng phát triển công nghệ như đã bàn trong chương 2, khả năng có nhiều hơn một bộ đếm trong một SC đã được đặt ra. Trong các mô hình thí nghiệm mô phỏng như Em-Star, Stargate [EM_04] có chia ra các loại đồng bộ giữa nút với nút, và trong nội bộ nút. Giải pháp được chọn là RBS do việc phát tín hiệu đồng bộ là độc lập với ứng dụng và hỗ trợ từ lớp MAC, sử dụng được khối truyền thông 802.11.

CHƯƠNG 5 : ĐỊNH TUYẾN VÀ LỊCH TRUYỀN THÔNG TRONG SCN

Định tuyến là vấn đề luôn được quan tâm trong các mạng đa bước truyền. Các phương pháp định tuyến được áp dụng trong SCN có những khác biệt với những phương pháp truyền thống hiện đang được áp dụng cho mạng có cấu trúc.

Có nhiều dự án nghiên cứu và xây dựng cơ chế định tuyến trong mạng *ad-hoc* [AHN_00], trong đó luận văn này tôi tập trung bàn luận về AODV và ZRP vì chúng có những điểm phù hợp với các bài toán cụ thể được đặt ra trong SCN.

Bảng 1. Các dự án nghiên cứu định tuyến trong mạng ad-hoc

DSR (Dave Johnson, CMU)	**AODV** (refinement of DSDV)
WINGs (JJ Garcia/UCSC)	AOMDV (Multipath - Das/Marina)
ROAM (JJ Garcia/UCSC)	Hiearachical (Akyildiz/ Georgia Tech)
WAMIS (Gerla/UCLA)	GPSR (Karp/Harvard)
ODMRP (Gerla et.al/UCLA)	CBRP (singapore)
TRAVLR (Kleinrock/UCLA)	Terminodes (EPFL)
Tora/IMEP (Park, Corson/UMD)	MMWN (Steenstrup/BBN)
Link Quality (Rohit Dube/ UMD)	ABR (C.K.Toh)
LAR (Texas A&M)	STAR (JJ Garcia/UCSC)
TBRPF (SRI)	**ZRP** (Zygmunt Haas/Cornell)
OLSR (Inria: Clausen./Jacket)	Fisheye/Hiararchical (UCLA)
DSDV (Dest. Sequence #'s)	CEDAR (Urbana-Champaign)

Các kiểu định tuyến trong mạng ngang hàng có thể chia vào hai nhóm chính là:

- ***Nhóm pro-active*** hướng tiếp cận gần giống với mạng truyền thống, trong đó liên tục tính toán và phát hiện định tuyến mới để cập nhật topo mạng. Điều này cho phép có thể chuyển tiếp gói, như là tuyến đã được dự đoán trước tại thời điểm nhận gói. Giao thức *pro-active* hoặc là giao thức *table-driven* cập nhật sự thay đổi của topo mạng bởi việc thêm, chuyển, xóa nút đòi hỏi phải cập nhật liên tục dẫn đến chiếm dụng băng thông vốn đã không được rộng trong mạng không dây.
- ***Nhóm re-active*** xây dựng định tuyến chỉ khi có nhu cầu, tức là khi gói tin có nhu cầu truyền. Khi đó bảng định tuyến theo nhu cầu sẽ được xây dựng dựa vào phản hồi của điểm cần đến. Kỹ thuật này cho phép không cần băng thông quảng bá và thám truyền cố định nhưng có bất lợi là có trễ truyền do đến thời điểm cần truyền mới xây dựng định tuyến. Các thuật toán loang cổ điển có thể áp dụng tại đây.

5.1 ĐỊNH TUYẾN AODV

Ngay từ tên gọi "*Ad-hoc on demand distance vertor*" đã cho biết đây là phương pháp tìm đường động dựa theo nhu cầu với những lợi điểm do được xây dựng riêng cho đặc thù của mạng *ad-hoc* gồm có:

- Tối thiểu hóa truyền quảng bá bởi kỹ thuật phát hiện định tuyến quảng bá DSR.
- Tách biệt việc quản trị kết nối nhóm cục bộ với bảo trì topo toàn hệ thống với bảng định tuyến động được. Tức là chỉ giữ lại thông số *next-hop* chứ không giữ lại thông tin toàn bộ tuyến.
- Các thông tin định tuyến mới được sử dụng sẽ được đánh số tuần tự.

Các loại thông điệp quảng bá trong AODV gồm có *RREQ*, *RREP*, *RERR* và *RREP-Ack*.

Trong AODV thì việc phát hiện đường được khởi tạo khi có nhu cầu truyền thông. Nút nguồn quảng bá thông tin tìm đường RREQ cho các nút lân cận. Trong gói tin RREQ sẽ bao gồm những thông tin sau [AODV_97]:

<source_addr, source_sequence_#, broadcast_id, dest_addr, dest_sequence_#, hop_cnt>

Cặp *<source_addr, broadcast_id>* xác thực duy nhất thông tin RREQ.

Mỗi nút sẽ bao gồm 2 thông số là Số tuần tự và id-quảng bá. Nút lân cận đó sẽ quảng bá *RREQ* cho lân cận của nó hoặc nếu thỏa mãn *RREQ* bởi gửi *RREP* ngược lại nguồn. Các bản sao của cùng *RREQ* sẽ bị loại bỏ.

Cùng với việc định tuyến thuận thì đường truyền ngược cũng được xây dựng tự động bởi việc thiết lập bản ghi nút truyền *RREQ*. Thực thể này sẽ được xóa tự động sau khoảng thời gian nhất định.

Nút thiết lập tuyến hồi gói tin *RREP* đến các lân cận mà nó nhận được *RREQ* theo tuyến đã được thiết lập bởi đường truyền ngược.

Mỗi nút trong hành trình *RREP* sẽ đặt con trỏ xuôi, cập nhật thời gian trễ, ghi số đích tuần tự của yêu cầu đích.

Nhằm đánh số thứ tự nguồn và đích, các thông tin hữu dụng được lưu trữ trong bảng định tuyến tạm thời, cùng với tuyến ngược lại do không có gì đảm bảo là tuyến ngược cũng sẽ qua đúng số *hop* và trùng với tuyến xuôi. Bảng định tuyến tạm thời này được bảo trì bởi nhiều tham số hỗ trợ ví dụ như *route caching timeout, active route timeout, route request expiration timer* và mã hóa các thông tin như:

- Đích đến.
- Bước truyền tiếp theo.
- Số bước truyền (*metric*).
- Số tuần tự hướng đích.
- Các lân cận tham gia tuyến.

- Thời gian hết hạn sử dụng của tuyến trong bảng định tuyến tạm thời.

Việc triển khai định tuyến AODV có những điểm giống với EIGRP do vậy ta có thể xây dựng giải thuật tương tự giải thuật DUAL của EIGRP để tăng tốc hội tụ cho định tuyến này khi áp dụng vào SCN[18].

5.2 ĐỊNH TUYẾN ZRP

Trên thực tế thì định tuyến AODV là rất tổng quát, do vậy hoàn toàn có thể thu hẹp phạm vi để có những hướng tiếp cận vấn đề tốt hơn trong SCN. Người ta lợi dụng sự khác biệt giữa *Link State* và *Distance Vector* để phát triển giao thức định tuyến ZRP[19] [ZRP_02].

Link state và *distance vector* khác nhau cơ bản về cách chúng quảng bá thông tin định tuyến. *Link state* quảng bá thông tin về trạng thái của các liên kết, trong khi đó *distance vector* quảng bá thông tin về các tuyến đường. Điều này dẫn đến việc cập nhật *routing table* cũng khác nhau. *Link state router* tự tính *route* từ *link state database* trong khi đó *distance vector router* tính *route* bằng cách so sánh các *route* mà nó nhận được từ lân cận quảng bá.

Việc nắm hết các thông tin trạng thái của các tuyến đường trong mạng phân tán diện rộng như SCN, nếu không giảm quy mô để thu hẹp không gian lời giải là không khả thi. Tuy nhiên khi triển khai thì *Link state* không bị *loop* trong khi việc này có thể xảy ra với *distance vector*.

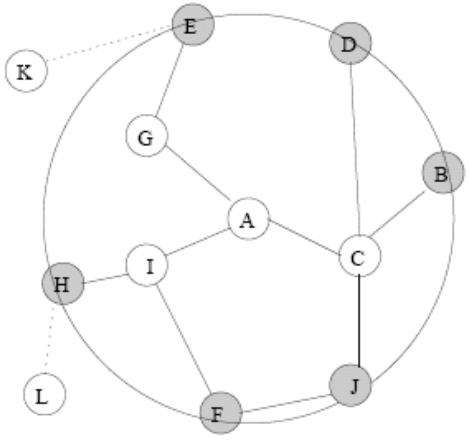
Vì vậy, ý tưởng nảy sinh là ứng dụng các phương thức định tuyến giống *link state* cho các nút gần lân cận và giống *distance vector* cho các nút xa nhau. Một cách hình dung đơn giản là sử dụng phương pháp định tuyến truyền thống cho những nút lân cận nhau và sử dụng phương pháp định tuyến theo nhu cầu cho những nút thuộc các nhóm khác nhau.

---

[18] Xem phụ lục 2.
[19] *Zone Routing Protocol*

Việc giả thiết là phần lớn các truyền thông diễn ra giữa các nút lân cận là hợp lý. Các thay đổi, chuyển vị nếu có chỉ có tác động lớn đến các nút lân cận tức là có ảnh hưởng cục bộ nhiều hơn so với tác động đến toàn hệ thống. Bằng việc tách biệt các lân cận cục bộ của một nút khỏi topo chung của toàn mạng cho phép người dùng áp dụng hướng tiếp cận khác tiên tiến hơn. Lân cận cục bộ này được gọi là *zone*. Theo như tên gọi của nó thì mỗi nút có thể là giao kết của nhiều *zone* và kích cỡ của các *zone* không nhất thiết phải như nhau. Thường thì kích cỡ này được đo bằng bán kính dài ρ, tính bằng số *hop* theo đường chu vi của *zone*.

Bởi việc chia mạng ra thành các *zone* chồng chất, không áp đặt những giá trị kích cỡ, phương pháp ZRP tránh được việc mô tả mạng theo cấu trúc chặt, dễ dàng cho việc bảo trì hệ thống, rất phù hợp với mô hình SCN. Vì trên thực tế thì hệ thống mạng có thể phẳng về mặt hình học nhưng các tuyến không đi theo hướng lân cận vị trí mà phụ thuộc vào các *zone* chồng chất.



Hình 10. Tuyến zone đối với nút A trong trường hợp ρ = 2

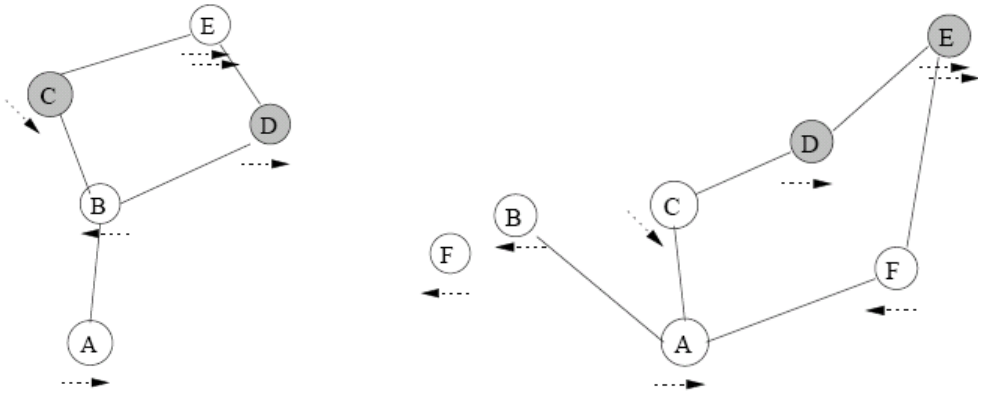
Trong hình minh họa trên thì từ nút A có nhiều tuyến để đi đến nút F, bao gồm cả những tuyến có *hop-count* > ρ. Tuy nhiên do tồn tại tuyến có *hop-count* <= ρ (tuyến AIF) nên F vẫn là nút nằm trong *zone*. Những nút

nằm trên đường chu vi của zone có *hop-count* = ρ và những nút nằm trong zone hoàn toàn có *hop-count* < ρ.

Để xác định lân cận trực tiếp của nút trước khi xây dựng định tuyến zone và phát hiện đặc tính của zone thì nút có sử dụng giao thức phát hiện địa chỉ MAC hay giao thức NDP. Một lần nữa khẳng định lại là ZRP là *framework*, trên cơ sở đó các kỹ thuật, giao thức có thể xây dựng theo đặc thù riêng để tận dụng lợi điểm của việc quy hoạch cục bộ nút.

Bảng 2. Các loại giao thức trong ZRP

Intrazone Routing Protocol	*IARP*	giao thức định tuyến trong zone, thuộc loại *proactive*, *table-driven* do các tác động thay đổi có ảnh hưởng rất lớn đến các nút lân cận. Do có giới hạn bởi bán kính ρ nên có thể đưa vào sử dụng thông số TTL và áp dụng giống như mạng IP có cấu trúc
Interzone Routing Protocol	*IERP*	giao thức định tuyến giữa các zone, thuộc loại *reactive*, chấp nhận trễ
Bordercast Resolution Protocol	*BRP*	giao thức quyết định biên truyền, xác định nút biên sẽ truyền quảng bá phục vụ IERP



Hình 11. Tái cấu trúc zone khi các nút chuyển vị

Các hướng mũi tên chỉ hướng chuyển dịch của nút và đại diện cho vận tốc. Điểm giám sát đặt tại nút A. Nút D và B dịch chuyển theo xu hướng xa nhau, còn nút E dịch chuyển theo hướng lại gần và xuất hiện khả năng nối kế trực tiếp với A và D. Nút F từ ở ngoài *zone* bắt đầu tham gia *zone* với truyền thông trực tiếp với A và E. Từ hình1 sang hình 2 là *zone* đã tự cấu hình và thích nghi với điều kiện mới. Cũng theo hình trên thì nút D vẫn còn quan hệ với nút A mặc dù định tuyến đã có thay đổi. Điều này chỉ ra rằng sự cần thiết của *pro-active* IERP: sự thay đổi topo mạng phải có điều chỉnh trong một thời gian nhất định.

Trong các bài toán giám sát thực, việc thay đổi thiết vị của các SC trong quá trình hoạt động là có xảy ra (ví dụ như SC đặt trên xe tuần tra, di động trong khu vực giám sát). Điều này dẫn đến các *zone* trong ZRP có biến động. Trong trường hợp có nhiều SC di chuyển thì việc tái lập *zone* là không dễ dàng và có thể vi phạm đáp ứng thời gian thực chung của hệ thống. Để cải thiện vấn đề này khi triển khai SCN cần hạn chế các SC di chuyển và trong trường hợp có nhiều SC chuyển vị thì cần phân biệt là SC đó có đóng vai trò *BS* hay không để thiết lập các *proxy* phù hợp hoặc sử dụng một giải pháp truyền thông hỗ trợ khác ví dụ như phát truyền hình số định hướng DVB-T hay *wimax*.

Khi áp dụng vào SCN thì ngoài cách xây dựng *zone* bởi bán kính dài ρ, *zone* thường được hiểu như là một *s_clu* có quan hệ trong nhiệm vụ giám sát.

Cải thiện hiệu năng mạng

Do SCN là mạng trải rộng nên việc đánh giá hiệu năng chung toàn mạng để cải thiện là khó khăn nên việc cải thiện tập trung ở truyền thông trong nội bộ *zone* hoặc *s_clu*. Ba tham số chính ảnh hưởng tới việc tối ưu hóa *zone* là thông lượng, độ trễ và độ tin cậy. Từ góc độ phát triển ứng dụng thì để cải thiện hiệu năng ta phải lưu ý các vấn đề sau:

- Tối ưu phân tán nhiệm vụ, giảm thiểu truyền thông. Thuộc lớp bài toán CSP sẽ được đề cập trong chương 7.
- Trong trường hợp truyền thông xảy ra là tiên đoán được hoặc theo chu kỳ ứng dụng thì đề xuất lập lịch truyền thông *multi-hop* nhằm giảm thiểu xung đột và chiếm hữu kênh truyền. Vấn đề này được trình bày trong phần tiếp theo.

5.3 LỊCH TRUYỀN THÔNG CỦA THÔNG ĐIỆP PHÁT SINH THEO CHU KỲ

Các thông điệp phát sinh theo chu kỳ đóng vai trò rất quan trọng trong hệ thống SCN. Khi đã xác định được tuyến thích hợp giưa *source* và *sink* còn cần đảm bảo hạn mức trễ của thông điệp truyền trong tuyến để tăng hiệu năng chung, QoS dịch vụ bằng phương pháp lập lịch.

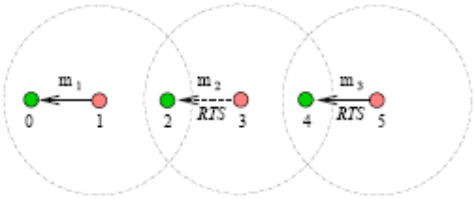
Do chọn lựa phương thức truyền thông *ad-hoc* là phương thức truyền thông chính trong SCN, nên ngoài việc theo dõi và bảo trì tuyến bởi các giao thức như AODV, ZRP cần phải để ý đến đặc điểm của kiểu truyền thông này. Sử dụng AODV hay ZRP đều chấp nhận phải có trễ truyền, nhưng trễ truyền phải tiên đoán được. Truyền thông *ad-hoc* có đặc điểm là chiếm hữu kênh

truyền khi SC đóng vai trò trạm phát, tức là nhất thời gây nghẽn truyền thông cho các SC lân cận[20].

Người ta đã chứng minh được rằng việc thỏa mãn được trễ hạn định của thông điệp dù trong đơn bước truyền là bài toán NP khó[21]. Do vậy, cách tiếp cận tại đây là phát hiện hướng sắp lịch truyền thông điệp trực tiếp với ràng buộc thời gian giới hạn [SMD_05]. Cụ thể kỹ thuật gồm:

- Đặt lịch truyền thông điệp dựa trên đúng thời điểm tại mỗi bước truyền của thông điệp đó.
- Phát hiện việc khả năng tái sử dụng kênh truyền tại các bước truyền và các thời điểm bằng cách truyền đồng thời những trạm phát không ảnh hưởng lẫn nhau khi truy xuất đường truyền.

Truyền nội dung thông điệp và tái sử dụng kênh truyền



Hình 12. Truyền thông điệp qua một bước truyền.

Vị trí và quãng cách đến nơi gửi đã được xác định. Chúng ta giả định khoảng truyền thông trùng với quãng các truyền thông. Trong hình 12 thì ba thông điệp m_1, m_2, m_3 truyền một bước cần được sắp lịch truyền thông.

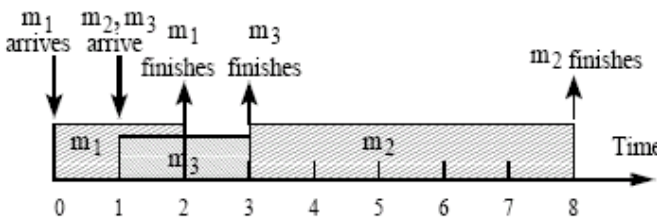
---

[20] Thực tế cho thấy việc tái sử dụng kênh truyền có hiệu quả với kiểu truyền thông CSMA/CA đặc biệt là khi kênh truyền được tận dụng nhiều, khoảng nhiễu rộng và khả năng xung đột cao. Mạng không dây trong họ 802.11 sử dụng phương thức CSMA/CA ở lớp MAC. Thứ nhất, do đặc điểm của kỹ thuật tránh xung đột nên mạng CSMA/CA không loại trừ hoàn toàn khả năng xung đột. Thứ hai, nơi gửi có thể tăng truyền lại theo hàm số mũ khi chúng nghe ngóng trên đường truyền. Thứ ba CSMA/CA sử dụng giao thức CTS/RTS để tránh xung đột. Trong những mạng như thế này, nút phải xác nhận quyền được truyền bởi gửi RTS và nhận đồng ý CTS trước khi truyền dữ liệu. Tại thời điểm đó các nút khác lân cận, có khả năng tiếp nhận thông điệp phải ngừng truyền thông, hay gọi là tạm khóa.

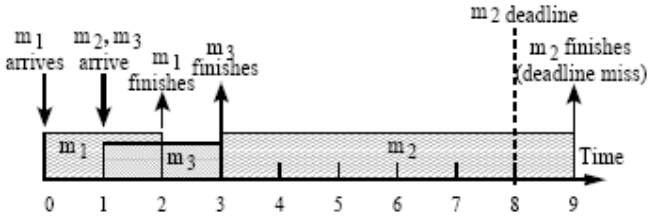
[21] Điều này có thể được chứng minh bởi việc đơn giản đồ thị *k-color*, trong đó đồ thị nội dung $G = (V, E)$ biểu thị xung đột giữa các truyền thông. Mỗi cạnh V đại diện cho việc truyền thông, tồn tại đỉnh E giữa 2 cạnh nếu truyền thông đó có thể diễn ra đồng thời. Nếu đồ thị đó là *k-color* thì mọi truyền thông có thể hoàn thành với k đơn vị thời gian.

Coi thời gian đến là AT, thời gian truyền là TT. ED là giới hạn trễ cho mỗi thông điệp và LST là thời gian bắt đầu truyền muộn nhất có thể.

Message	AT	TT	ED	LST
m_1	0	2	6	4
m_2	1	5	8	3
m_3	1	2	8	6



Message	AT	TT	ED	LST
m_1	0	2	6	4
m_2	1	6	8	2
m_3	1	2	10	8



Hình 13. Hai kiểu sắp lịch truyền thông.

Coi m_1 đến tại mốc thời gian 0, đây là thông điệp trong hệ thống tại thời điểm đó, nút 1 truyền RTS và nhận xác nhận RTS từ nút 0. Kể từ khi nút 2 nhận được RTS nó bị khóa trong thời điểm truyền m_1. Khi thông điệp m_2 truyền đến tại thời điểm 2, nút 3 gửi RTS về nút 2 nhưng không nhận được phản hồi. Nút 4 cũng nhận thông điệp RTS và cũng bị khóa truyền, khi nút 5 truyền đến nút 4 một RTS cho thông điệp m_3, nhưng nó không nhận được CTS nên chưa thể truyền m_3. Qua quan sát trong hình thì m_1 và m_3 không có quan hệ với nhau nên chúng có thể truyền độc lập. Đây là kết quả của việc truyền tuần tự m_1, m_2, m_3 dẫn đến việc tái sử dụng đường truyền thấp. Một điều quan trọng nữa là m_3 bị trễ quá hạn.

Tuy nhiên nếu đặt lịch truyền thông theo hướng tận dụng kênh truyền bằng các truyền m_3 đồng thời với m_1 sau đó là m_2 thì mọi thông điệp đều đến đích trong hạn tương ứng.

Mặc dù trong cách đặt lịch trên với việc tái sử dụng kênh truyền nhằm thỏa mãn hạn thời gian, truyền song song tuy nhiên cũng có trường hợp lại gây trễ quá han. Cũng trong hình 13, m_1 được truyền tại thời điểm $t=0$. Khi m_1

và m_2 truyền m_2 sẽ bị block. Khi m_3 tại thời điểm $t=1$, nó có thể truyền đồng thời với m_1. Giả sử tình huống này xảy ra m_2 phải đợi đến khi m_3 kết thúc tại $t=3$ rồi bắt đầu truyền. Khi m_2 yêu cầu 6 đơn vị thời gian để truyền thì nó sẽ kết thúc tại $t=9$ và trễ quá hạn như hình dưới. Trong trường hợp này truyền tuần tự m_1, m_2, rồi đến m_3 lại giải quyết được bài toán.

Điều này chứng tỏ việc tái sử dụng không gian truyền thuần túy không đảm bảo mọi thông điệp luôn tới hạn. Do vậy chiến thuật đặt lịch phải tính tới khả năng tác động của lịch đó với những thông điệp tiếp theo. Do bản chất tự nhiên của bài toán, chúng ta phải chọn lựa giải pháp để đặt lịch truyền thông điệp đa bước trong mạng. Dưới đây là 2 *heuristic* thường dùng.

Per-Hop Smallest LST First (PH-SLF)

PH-SLF là kỹ thuật đặt lịch phân tán trong đó mỗi nút tạo lịch truyền cục bộ độc lập với các nút khác. Trong cách tiếp cận này, một tập các thông điệp được sắp đợi tại nút. Nút đặt lịch chọn ra thông điệp nào có LST nhỏ nhất để truyền.Với giao thức MAC loại CSMA/CA thì việc xung đột và *back-off*, *false blocking* có thể được loại bỏ. Lợi điểm của phương pháp này là nó có thể được sử dụng như là một bổ sung cho mạng 802.11 vì PH-LSF có thể được xây dựng bằng phần mềm trong OS. Điều cần là các hướng tiếp cận bổ sung khác nhằm tránh xung đột và tăng hiệu quả tái sử dụng kênh truyền.

Channel Reuse-based Smallest LST First (CR-SLF)

Mục đích của CR-SLF làm tiếp cận nhằm nắm bắt tới hạn của thông điệp tại mỗi bước truyền, để tránh xung đột và tái sử dụng kênh.

Giả định ban đầu:

m_x^i thông điệp m_x tại *hop* thứ i

$T(m_x^i)$ truyền thông của m_x^i

$a(m_x^i)$ thời gian đến của m_x tại hop thứ i

$d(m_x^i)$ trễ tới hạn của m_x^i

$l(m_x^i)$ thời điểm bắt đầu muộn nhất (LST) của $T(m_x^i)$

$s(m_x^i)$ thời điểm bắt đầu truyền của $T(m_x^i)$

$f(m_x^i)$ thời điểm kết thúc của m_x^i. thời điểm m_x đến được *hop* tiếp theo

$e(m_x^i)$ thời gian sinh ra m_x (giống nhau trong mọi *hop* truyền của m_x)

Khi các SC cập nhật dữ liệu hình theo chu kỳ, thời gian đến của thông điệp tại *source* là thời gian mà dữ liệu được khởi tạo. Thời gian đến của nút trung gian là thời gian mà bit cuối cùng của thông điệp qua bước truyền trên đường đến đích. Theo dõi thời gian đến tại những nút trung gian phụ thuộc vào thông điệp được đặt lịch tại bước truyền trước. Thời gian bắt đầu tính từ khi thông điệp được đặt lịch truyền và kết thúc khi thông điệp được nhận toàn bộ bởi bước truyền tiếp theo (tức là lúc đó kênh truyền lại được giải phóng). Thời gian thực thi là thời gian mà kênh truyền bị chiếm hữu và trễ (sự khác biệt giữa thời gian tức thời của bit đầu tiên được truyền đi và bít cuối cùng được nhận bởi bước truyền tiếp theo). Theo đó thì:

$$a\left(m_x^{i+1}\right) = f\left(m_x^i\right) = s\left(m_x^i\right) + e\left(m_x^i\right) \quad (5.3.1)$$

Việc xếp lịch cần tuân thủ những định hướng sau khi sinh lịch truyền thông:

- Cần tối đa tái sử dụng không gian bởi đặt lịch cho những thông điệp không liên quan được truyền song song.
- Cần giám sát việc truyền thông bởi LST của chúng. LST là trễ hạn cục bộ của truyền thông tại mỗi bước truyền, kể từ khi thời gian cuối cùng trong mỗi thông điệp phải được đặt lịch để đảm bảo điều kiện tới hạn *end-to-end*.

Ý tưởng đề xuất là chia tập các thông điệp cần truyền thành các tập hợp con phân tách mà việc truyền thông chỉ xảy ra trong nội bộ chúng không có liên quan đến bên ngoài. Do vậy chúng có thể truyền song song. Những tập

hợp này được sắp xếp thứ tự và mọi truyền thông trong tập phải kết thúc trước khi truyền tập hợp tiếp theo.

Để xây dựng những tập hợp này, việc lập lịch dựa trên các LST của chúng. Tại mỗi bước, truyền thông có LST nhỏ nhất được chọn và chương trình kiểm tra xem có thể gán truyền thông này cho tập nào sẵn có không. Bằng cách kiểm tra truyền thông này có ảnh hưởng gì đến việc truyền các thông điệp trong tập đó hay không. Thông điệp có thể được xếp lịch truyền thông nếu thời gian kết thúc của nó không vi phạm trễ tới hạn. Và việc bổ sung thông điệp này vào tập hợp đó không gây vi phạm trễ tới hạn của việc truyền thông hiện thời. Nếu không có một tập hợp nào thỏa mãn thì một tập hợp mới được xây dựng. Khi thông điệp được lập lịch ở bước truyền thứ *i*, nó có thể được lập lịch tại bước truyền *i+1*. Theo dõi thông điệp cần được truyền *hop-by-hop*, do thời gian đến tại bước truyền tiếp theo là chưa biết cho đến khi nó đã được sắp lịch ở bước truyền trước. Quá trình trên tiếp tục cho mọi thông điệp trong hàng đợi đến khi chúng được sắp lịch trong mọi bước truyền từ nguồn đến đích. Các tập hợp xây dựng được định nghĩa lịch truyền thông cho những thông điệp nhiều bước truyền này.

Thuật toán được xây dựng như sau

1. $S = \phi$

   //Khởi tạo. Mục đích là xây dựng tập hợp $S = \{S_1, S_2, ..., S_n\}$ trong đó các phần tử là tách biệt và các thông điệp được truyền giữa chúng là không có quan hệ (trực giao).
2. Bước 1: Chọn truyền thông đưa vào lịch. Từ danh sách hiện thời của các thông điệp truyền thông, chọn ra cái có LST nhỏ nhất, ứng với truyền thông cần thiết nhất.
3. Bước 2: Gán truyền thông đó vào một tập hợp. Giả sử n tập hợp đã

được xây dựng trong quá trình sắp lịch trước gồm S_1, S_2, ..., S_n. $T(m_x^i)$ đã được chọn và gán vào tập hợp thỏa mãn điều kiện đầu tiên được tìm thấy. Nếu không tồn tại tập hợp này (hoặc lịch truyền đang là rỗng) thì một tập hợp mới S_{n+1} được thêm vào danh sách và $T(m_x^i)$ được gán vào đó để đảm bảo ràng buộc về trễ hạn thời gian không bị xâm phạm.

Tập S_j được coi là phù hợp với $T(m_x^i)$ nếu các điều kiện sau được thỏa mãn:

- Thời gian kết thúc $f(S_j)$ thì muộn hơn thời gian đến của thông điệp $a(m_x^i)$.
- Thời gian kết thúc của m_x^i không trễ hơn tới hạn thời gian hiện thời trong tập hợp.

$$f(m_x^i) = \max(s(S_j), a(m_x^i)) + e(m_x^i) \le d(m_x^i). \qquad (5.3.2)$$

- $T(m_x^i)$ không có quan hệ gì với những truyền thông hiện có trong S_j.
- Việc bổ sung truyền thông $T(m_x^i)$ vào S_j không vi phạm tới hạn thời gian của thông điệp theo thứ tự $S_k, j < k \le n$.

4. Bước 3: Cập nhật thời điểm kết thúc của tập hợp thỏa mãn điều kiện và chèn truyền thông mới cho bước truyền tiếp theo. Các bước này được lặp cho đến khi không còn thông điệp nào cần được sắp lịch.

5.4 TỔNG KẾT VÀ BÀN LUẬN

Chương 5 tập trung bàn luận về vấn đề định tuyến và sắp lich truyền thông trong SCN. Hai định tuyến chuẩn được trình bày tại đây là AODV và ZRP. Định tuyến AODV thuộc dạng định tuyến chấp nhận có trễ, giống như những vấn đề tương tự đã bàn luận trong các phần 3 và 4 về truyền tin định

hướng và đồng bộ bộ đếm muộn. Do vậy phương pháp này trên phương diện lý thuyết là hiệu quả và toàn năng trong SCN. Phương pháp luận khi đề xuất AODV có nhiều điểm giống với EIRGP của Cisco nên người lập trình có thể dễ dàng tiếp nhận, viết mã và thử nghiệm bởi những công cụ mô phỏng[22]. AODV thích hợp cho các ứng dụng truyền thông từ *source* đến *sink* do dễ áp dụng QoS và khả năng chịu lỗi cao[23].

Tuy nhiên trên thực tế, các nhiệm vụ giám sát có xu hướng cục bộ và các SC có xu hướng trao đổi thông tin trong nội bộ *s_clu* (thường là những SC có quan hệ về mặt không gian và góc hướng giám sát tạo nên *microcluster*) nên áp dụng ZRP tỏ ra có ưu thế hơn trong bài toán phân tải tính toán và ứng dụng nhóm *s_clu*[24].

Một vấn đề khác, chưa được đề cập đến ở đây là vấn đề xuất hiện những vùng xám (chất lượng truyền dẫn tín hiệu trong vùng thấp do nhiều nguyên nhân), vùng đen (mật độ truyền thông cao), vùng trắng (mật độ truyền thông thấp) trong SCN. Trong trường hợp tải hệ thống là không cao thì các vấn đề QoS không quá nổi bật, tuy nhiên trong trường hợp có nhiều SC tham gia truyền thông thì *metric* của tuyến được thiết lập phải đảm bảo đã bao gồm chi phí QoS trong đó.

Trong SCN thì vấn đề QoS truyền thông có hai trường hợp chính là:

- ***Đơn hướng*** SC_s là nguồn , SC_t là đích và C là các ràng buộc về tài nguyên. Cần xây dựng tuyến khả dụng từ SC_s đến SC_t thỏa mãn C[25].
- ***Đa hướng*** SC_s là nguồn, tập các SC_t là đích, các ràng buộc C và một tiêu chí tối ưu[26], xây dựng cây khả dụng phủ hết SC_s và các SC_t thỏa mãn ràng buộc C.

---

[22] *nsnam* (http://www.isi.edu/nsnam/ns/)
[23] Bài toán cơ bản 2 trong SCN.
[24] Bài toán cơ bản 1 trong SCN.
[25] Các yêu cầu bài toán thường gặp là định tuyến tối ưu hóa kênh và định tuyến ràng buộc kênh.
[26] Các yêu cầu bài toán thường gặp là định tuyến tối ưu hoá cây và định tuyến ràng buộc cây.

Phát triển thuật toán định tuyến theo tiêu chí QoS sẽ góp phần cân bằng tải truyền thông vốn không quá rộng của SCN.

Giới hạn trễ truyền là một dạng ngưỡng QoS truyền thông trong hệ thống, để đảm bảo tới hạn trễ của thông điệp phát sinh theo chu kỳ bài toán cần giải quyết là bài toán đặt lịch truyền thông trên tuyến. Dạng bài toán NP-khó này có liên quan đến việc tối ưu thời gian truyền thông trong điều kiện ràng buộc về tài nguyên như kênh truyền, thời hiệu dịch vụ... Trong hai phương pháp được đề xuất thì *PH-SLF* thích hợp trong các mạng không dây chuẩn 802.11 do dễ dàng thiết lập ngay từ trong OS, tuy nhiên *CR-SLF* lại tỏ ra hiệu quả hơn bởi nắm bắt được tới hạn của thông điệp ở mỗi bước truyền để tái sử dụng kênh truyền.

CHƯƠNG 6 : AN NINH TRUYỀN THÔNG TRONG SCN

Hệ thống SCN được thiết kế nhằm mục đích giám sát an ninh, do vậy nhằm tăng tính chịu lỗi, tránh các tác động từ bên ngoài vào hệ thống, cần có cơ chế bảo đảm an ninh truyền thông. Đối với hệ thống được xây dựng dựa trên nguyên tắc truyền thông *ad-hoc* như SCN thì, những vấn đề an ninh sau được đặt ra [SAN_99]:

- ***Availbility*** các dịch vụ mạng tồn tại dù bị tấn công từ chối dịch vụ DoS.
- ***Confidentiality*** thông tin không được tiết lộ cho những thực thể chưa được xác thực.
- ***Integrity*** đảm bảo thông điệp được truyền không gián đoạn.
- ***Authentication*** nút có khả năng xác thực nút đang tương tác, tránh giả mạo nút.
- ***Non-repudiation*** phát hiện và cô lập những nút lỗi. Khi nút A nhận được thông điệp lỗi từ nút B thì thông tin này được phản hồi ngược về B và thông báo đến các nút khác.

Các vấn đề được trình bày ở đây là:

- Tập giao thức an toàn SPINs
- Tấn công từ chối dịch vụ DoS

6.1 TẬP GIAO THỨC SPINs

Một điều có thể khẳng định là những phương thức tăng cường an ninh truyền thông đang sử dụng như SSL/TLS, IPSEC hay mã hóa gói tin bởi các thuật toán mã với khóa không đối xứng không thể sử dụng trực tiếp trong SCN do các bất lợi sau:

- Chi phí khởi tạo dịch vụ cao. Thời gian thực thi là O(*phút*)
- Chi phí thẩm tra cao. Thời gian thực thi là O(*giây*)
- Yêu cầu bộ nhớ cao

- Chi phí truyền thông cao. Yêu cầu từ *128 bytes* truyền thông trở lên.

Do những vấn đề mang tính bản chất của hệ thống như mạng phát quảng bá, cần tối ưu năng lượng, định tuyến nội vùng và *clustering* nên việc mã hóa đường truyền cũng như cơ chế IDS phải có thiết kế độc lập. Tập giao thức SPINs ra đời để đáp ứng nhu cầu trên.

Giao thức an toàn trong SCN cần được xây dựng trên cơ sở cân nhắc và đảm bảo hai vấn đề là độ tin cậy và tối thiểu năng lượng cần dùng.

Việc đảm bảo an ninh truyền thông trong SCN tập trung vào xây dựng các kênh truyền an toàn với bốn đảm bảo là:

- Độ tin cậy của dữ liệu.
- Xác thực dữ liệu.
- Toàn vẹn dữ liệu.
- Dữ liệu tươi mới.

Giả định ban đầu như sau:

A, B là những điểm nút truyền thông

N_A được sinh bởi nút A (chuỗi bít không định trước nhằm xác định độ tươi của dữ liệu)

$M_1|M_2$ biểu hiện chuỗi xích liên kết M_1 và M_2.

K_{AB} biểu hiện khóa mã (đối xứng) được chia sẻ giữa A và B

$\{M\}_{K_{AB}}$ là thông điệp mã của M với khóa đối xứng dùng chung A và B

$\{M\}_{(K_{AB},IV)}$ biểu hiện là thông điệp mã của M với khóa K_{AB} và vectơ IV được sử dụng trong chế độ mã như *CBC*, *OFB* hay *CTR*

Tập giao thức SPIN được chia thành hai nhóm là SNEP và $\mu TESLA$, trong đó SNEP dùng trong truyền điểm - điểm và $\mu TESLA$ dùng trong xác thực dữ liệu quảng bá đa điểm.

SNEP

SNEP có những tiên tiến như:

- chỉ bổ sung 8 byte vào mỗi thông điệp.
- cũng như những hệ mã hóa khác, trong SNEP cũng có sử dụng bộ đếm nhưng tránh truyền số đếm bởi việc giữ trạng thái bộ đếm tại cả 2 đầu truyền thông chứ không đính kèm trong thông điệp.

Một khung mẫu thường thấy để bảo vệ nội dung dữ liệu là mã hóa, nhưng mã hóa cũng không phải là toàn bộ của của vấn đề. Một trong những thuộc tính quan trọng khác của tính an toàn đó là *semantic security* - an ninh ngữ nghĩa, tức là một phương tiện thám sát phải không tìm được thông tin nội dung của cùng một *plaintext* dù được duyệt nhiều lần cùng một đoạn văn bản mã. Kỹ thuật cơ sở ở đây là lưu trữ ngẫu nhiên: trước khi mã thông điệp với loạt chuỗi (kiểu như mã hóa DES) thì phía gửi thêm vào đầu thông điệp một chuỗi ngẫu nhiên các bit. Điều này giúp tránh kẻ tấn công xác định thông điệp gốc từ thông điệp mã hóa dù biết được cặp thông điệp/ thông điệp mã của cùng một khóa.

Dĩ nhiên việc truyền ngẫu nhiên dữ liệu qua kênh vô tuyến yêu cầu nhiều năng lượng. Do vậy người ta xây dựng một kỹ thuật mã trong đó lưu trữ ngữ nghĩa an toàn bằng cách. từ chối việc chia sẻ bộ đếm giữa nút truyền và nhận cho việc mã hóa khối theo thời gian. Tức là thông tin về bộ đếm khối sẽ không được đính kèm trong thông điệp mã. Để kết gắn xác thực hai chiều và tin cậy dữ liệu, người ta dùng khái niệm MAC[27]..

Sự kết hợp của những kỹ thuật này tạo thành SNEP. Dữ liệu mã có khung như sau:

$$E = \{D\}_{(K_{cner}, C)} \tag{6.1.1}$$

trong đó D là dữ liệu, khóa mã là K_{encr}, và bộ đếm là C và

$$M = MAC(K_{mac}, C \mid E) \tag{6.1.2}$$

[27] *Message authen control*

Người ta sinh khóa K_{encr} và K_{mac} từ khóa gốc K. Như vậy một thông điệp hoàn chỉnh từ A gửi đến B là

$$A \to B : \{D\}_{(K_{encr},C)}, MAC\left(K_{mac}, C \mid \{D\}_{(K_{mac},C)}\right) \quad (6.1.3)$$

Sự kết hợp này mang lại những cách tân:

- ***Semantic security*** an toàn ngữ nghĩa, do bộ đếm được gia tăng sau mỗi thông điệp, nên cùng một thông điệp sẽ sinh mã khác nhau tại mỗi thời điểm. Đặt giá trị bộ đếm là đủ lớn để không lặp lại trong vòng đời của nút và truyền thông.
- ***Data authentication*** xác thực dữ liệu, nếu giá trị so sánh của MAC chính xác, thì nút nhận có thể giả định rằng thông điệp gốc được gửi từ chính gốc.
- ***Replay protection*** bảo vệ lặp, giá trị bộ đếm trong MAC tránh gây nên truyền lặp thông điệp cũ. Để ý rằng nếu giá trị bộ đếm không hiện diện trong MAC thì có thể dễ dàng tạo lặp thông điệp cũ.
- ***Weak freshness*** tươi, Nếu giá trị so sánh của thông điệp chính xác, nút nhận xác định được thông điệp vừa mới được gửi do thông điệp trước nhận được chính xác (giá trị bộ đếm thay đổi sai khác so với tiên đoán đủ nhỏ).
- ***Low communication overhead*** Trạng thái bộ đếm được lưu tại mỗi nút và không cần được truyền kèm trong mỗi thông điệp. Trong trường hợp MAC không khớp, phía nhận có thể sử dụng một số nhỏ gia tăng bởi bộ đếm để khôi phục lại thông điệp bị mất. Trong trường hợp tái đồng bộ không được hai phía sẽ trao đổi lại thông tin bộ đếm, điều này được đề cập trong phần dưới đây.

SNEP đơn giản chỉ có *weak freshness data* do mới chỉ đưa ra một chiều xác thực là thông điệp gửi đến nút *B* chứ không bảo đảm chính xác cho nút A rằng thông điệp được tạo từ nút *B* để đáp lại sự kiện ở nút *A*.

Để có *strong freshness data* thì Nút A sinh N_A ngẫu nhiên và gửi nó kèm theo thông điệp R_A đến nút B. Từ nút B phản hồi thông điệp R_B và thay vì N_B thì để tối ưu tiến trình thì ngầm đưa thông tin khi tính MAC. Khi đó thì SNEP với *strong freshness* cho phản hồi của nút B sẽ là:

$$\begin{aligned} &A \to B : N_A, R_A \\ &B \to A : \{R_B\}_{(K_{encr}, C)}, MAC\left(K_{mac}, N_A | C | \{R_B\}_{(K_{encr}, C)}\right) \end{aligned} \tag{6.1.4}$$

Nếu so sánh MAC chính xác thì nút A sẽ xác định được nút B đã sinh phản hồi sau khi nó gửi yêu cầu. Thông điệp đầu tiên có thể sử dụng SNEP đơn giản nếu tin tưởng được và xác thực thông tin là cần thiết.

μTESLA

Việc xác thực nguồn các thông điệp quảng bá trong SCN bằng xác thực số bất đối xứng, hay khóa mã bẫy một chiều đòi hỏi phải có khóa dài và phải bổ sung từ 50 đến 1000 byte cho mỗi gói tin. Điều này ảnh hưởng đến hiệu năng truyền thông và xử lý số học của các điểm nút.

TESLA là kỹ thuật được phát triển để xác thực thông tin quảng bá với xác thực gói dựa trên chữ ký số. Tận dụng những ưu điểm của *TESLA*, người ta phát triển *μTESLA* và có thể sử dụng trong những hệ thống như SCN.

- *TESLA* xác thực gói khởi đầu với chữ ký số còn *μTESLA* sử dụng kỹ thuật đối xứng.
- Hủy bỏ chữ ký xác thực sau mỗi gói là quá lãng phí nên *μTESLA* sử dụng cùng chữ ký xác thực trong một thời kỳ nhất định.
- Lưu trữ chuỗi khóa một chiều tại nút là không cần thiết, thay vào đó *μTESLA* lưu số của các nút gửi đã xác thực.

Như đã phân tích, các yêu cầu xác thực quảng bá đòi hỏi kỹ thuật bất đối xứng nhưng do kỹ thuật mã này đòi hỏi xử lý tính toán cao và nhiều bất lợi khác nên *μTESLA* vận dụng linh hoạt để vượt qua trở ngại của việc tạo sự

bất đối xứng bởi cách chậm tiết lộ khóa đối xứng trong việc xác thực quảng bá. Điều này mang lại thuận tiện trong nhiều kịch bản xác thực quảng bá.

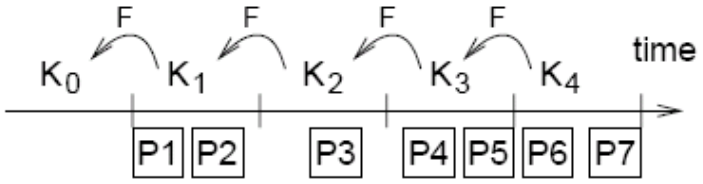
Tại một khoảng thời gian đủ nhỏ, coi như là không có sự kiện biến động về nút và tuyến thì với một nút được coi là trạm phát người ta có thể xây dựng cây truyền thông, xác định được trễ truyền thông từ trạm gốc đến trạm thu tương ứng với bao nhiêu khe thời gian.

μTESLA yêu cầu trạm phát cơ sở và các nút nhận phải được đồng bộ thời gian và mỗi nút phải biết và lưu được chặn trên của của độ lệch tối đa của lỗi đồng bộ. Để truyền gói xác thực, trạm phát đơn giản chỉ tạo *MAC* bởi việc sinh mã với khóa là thời gian quy ước đó, khi nhận được gói tin, trạm thu kiểm tra khóa *MAC* có chính xác chưa được gửi đi từ trạm phát không và chưa được nhận từ trước đó (dựa trên độ tương đồng về thời gian đồng bộ, giới hạn lỗi đồng bộ và thời gian trong lịch trình mà khóa không tiết lộ). Từ khi nút thu chắc rằng khóa *MAC* chỉ được biết bởi trạm phát thì nó sẽ lưu gói tin vào vùng đệm. Vào thời điểm mà khóa được bộc lộ thì trạm phát quảng bá thông tin xác thực khóa đến mọi trạm thu. Khi nút nhận được khóa nó sẽ dễ dang kiểm tra được tính đúng đắn của khóa đó (đã có trong vùng đệm chưa?) Nếu khóa được chấp nhận, nút có thể sử dụng nó để xác thực gói lưu trong vùng đệm của nó.

Mỗi khóa *MAC* là một khóa trong chuỗi khóa được sinh bởi hàm một chiều *F*. Để sinh chuỗi khóa một chiều, nơi gửi chọn khóa mới nhất K_n trong chuỗi ngẫu nhiên và lặp áp dụng F cho mọi khóa khác: $K_i = F(K_{i+1})$. Mỗi nút có thể đồng bộ thời gian và nhận khóa xác thực trong chuỗi khóa an toàn và xác thực nguồn gốc bởi cách xây dựng khối SNEP.

Điều này có nghĩa là xác thực khóa được chuyển đến trễ hơn sau một số khe thời gian nhất định để làm cơ sở đối sánh với khóa đang được lưu

trong vùng đệm của nút nhận. Việc kiểm tra là rất đơn giản bởi việc thực thi hàm một chiều không đòi hỏi nhiều năng lực tính toán.



Hình 14. Sử dụng chuỗi khóa theo khe thời gian để xác thực gốc truyền tin

Trong hình 14, mỗi khóa trong chuỗi khóa phù hợp với khoảng thời gian và mọi gói được truyền đến trong khe thời gian đó đều được xác thực bởi cùng khóa. Thời gian cho đến khi khóa gốc của khe kết thúc được chuyển đến là là 2 khe trong hình trên.

Giả định nút nhận có đồng bộ thời gian gần và biết khóa K_0 trong chuỗi xác thực. Gói P_1 và P_2 đã gửi trong khe 1 chứa *MAC* với khóa K_1. Gói P_3 có MAC sử dụng khóa K_2. Nếu giả sử gói P_4, P_5, P_6 đều mất cũng như gói tiết lộ khóa K_1 thì nút nhận không thể xác thực được gói P_1, P_2, P_3. Trong khe 4, trạm phát quảng bá khóa K_2 mà nút có thể xác thực được bởi $K_0 = F(F(K_2))$ và suy ra được $K_1 = F(K_2)$ để rồi từ đó xác thực được gói P_1, P_2 với khóa K_1 và P_3 với khóa K_2.

Việc triển khai *μTESLA* được chia ra thành các pha sau:

- ***Sender setup*** nút gửi đầu tiên sinh một chuỗi các khóa bí mật. Để sinh khóa 1 chiều độ dài n, nút gửi chọn khóa mới nhất K_n ngẫu nhiên, và sinh những giá trị còn lại bởi việc áp dụng hàm một chiều F[28]. Vì F là hàm một chiều nên mọi điểm đều dễ dàng sinh mã tiến ví dụ như tính toán $K_0, \ldots, K_j$ bởi K_{j+1} nhưng tính toán theo chiều ngược dạng tính K_{j+1} từ $K_0, \ldots, K_j$ là rất khó khăn. Điều này tương tự như hệ thống sinh mật khẩu dùng một lần *S/Key*.

[28] Ví dụ như hàm mã băm MD5: $K_j = F(K_{j+1})$

- ***Broadcasting authenticated packet*** thời gian được chia thành các khe thời gian và nút gửi gán mỗi khóa trong chuỗi khóa vào một khe thời gian. Tại khe thời gian *t*, nút gửi chọn khóa K_t để sinh *MAC* của gói tại thời điểm đó. Sau một khoảng thời gian δ, cụ thể là sau một vài khe thời gian khóa K_t sẽ được tiết lộ và thường thì δ này lớn hơn khoảng vài lần so với thời gian truyền dẫn giữa nút truyền và nút nhận.
- ***Bootstraping a new receiver*** thuộc tính quan trong của chuỗi khóa một chiều là khi nút nhận có khóa gốc xác thực trong chuỗi, thì các khóa sinh tuần tự trong chuỗi đó có thể tự xác thực. Ví dụ như khi nút nhận nhận được khóa K_i trong chuỗi khóa, nó có thể dễ dàng xác thực K_{i+1} bởi việc kiểm tra $?(K_i=F(K_{i+1}))$. Đây chính là *bootstrap* của $\mu TESLA$, và yêu cầu quan trọng nhất là nút truyền và nhận có đồng bộ thời gian *loosedly time synchronized*. Vì khi đó nút nhận biết được lịch biểu công khai khóa trong chuỗi khóa một chiều. Việc đồng bộ thời gian cũng như xác thực chuỗi khóa tạo thành kỹ thuật *strong freshness* và xác thực điểm - điểm. Nút nhận trả lời với thông điệp chứa thời gian T_s của nó (để đồng bộ thời gian), một khóa K_i trong chuỗi khóa thời gian ở khe thời gian quá khứ *i*, và thời gian T_i bắt đầu của khe thời gian i, quãng thời gian T_{int} và trễ công khai khóa δ.

$$\begin{aligned} &M \rightarrow S : N_M \\ &S \rightarrow M : T_S \mid K_i \mid T_i \mid T_{\text{int}} \mid \delta \\ &MAC\left(K_{MS}, N_M \mid T_S \mid K_i \mid T_i \mid T_{\text{int}} \mid \delta\right) \end{aligned} \qquad (6.1.5)$$

Nút gửi không cần mã hóa dữ liệu. *MAC* sử dụng khóa chia sẻ giữa nút và trạm phát để xác thực dữ liệu, N_M cho phép nút kiểm tra *freshness*. Thay vì dùng kịch bản xác thực số như trong *TESLA*, người ta dùng kênh xác thực *bootstrap* để xác thực thông tin quảng bá.

- ***Authenticating broadcasted packet*** khi nút nhận nhận được gói tin với *MAC*, nó cần bảo đảm rằng gói tin đó không phải được phát chuyển tiếp từ một điểm khác (*middle attack*). Mối đe dọa ở đây là điểm tấn công đó đã biết khóa công bố tương ứng với khe thời gian và nó có thể giả mạo gói tin nếu biết khóa được dùng để tính ra MAC. Nút nhận cần chắc rằng nút gửi không công bố khóa cho gói tin tương ứng với gói tin đến, nếu không điểm tấn công có thể giả mạo nội dung. Điều này được gọi là trạng thái an toàn - *security condition*, khi mà nút nhận phải kiểm tra mọi gói tin đến. Chính vì thế nút nhận và nút gửi cần đồng bộ thời gian và nút nhận cần biết lịch biểu sinh khóa. Nếu gói tin đến thỏa mãn điều kiện an toàn, nút nhận sẽ lưu gói (phục vụ cho việc kiểm tra nó khi khóa được công bố). Nếu điều kiện an toàn bị xâm phạm (gói tin có trễ hơn bình thường) thì nút nhận cần bỏ gói tin đó. Ngay khi nút nhận nhận được khóa K_j của khe thời gian cũ, nó xác thực khóa bởi kiểm tra xem khóa đó có liên quan với khóa K_i đã được biết trong lần xác thực trước không bởi một vài ứng dụng của hàm một chiều $F:K_i=F^{j-1}(K_j)$. Nếu việc kiểm tra thỏa mãn thì khóa K_j mới sẽ được dùng để xác thực mọi gói tin nhận được từ khe thời gian *i* đến *j*. Nút nhận sẽ lưu khóa K_j mới thay vì khóa K_i cũ.
- ***Node broadcast authenticated data*** Những thách thức mới nảy sinh nếu nút quảng bá thông tin xác thực. Do giới hạn bộ nhớ của nút, nó có thể không lưu được hết các khóa trong chuỗi khóa một chiều. Thêm vào đó, việc tái tính toán các khóa từ khóa sinh gốc K_n cũng đòi hỏi khối lượng tính toán lớn. Hậu quả là nút phát có thể không công bố khóa với từng nút nhận, do đó việc gửi đi cam kết xác thực khóa đòi hỏi thỏa thuận khóa nút đến nút - *node to node key agreement*. Một phương pháp đơn giản là khóa mã công khai để thiết lập khóa đối xứng.

Tuy nhiên trong SCN, người ta cũng không ưu tiên sử dụng những tính toán sinh khóa công khai quá cồng kềnh do phải dành năng lực xử lý cho mục đích chính của hệ thống. Vì vậy người ta thiết lập một giao thức riêng nhất cho thuật toán sinh khóa đối xứng bằng cách sử dụng trạm phát và một tác tử tin cậy để thiết lập khóa.

$$\begin{aligned} &A \to B : N_A, A \\ &B \to S : N_A, N_B, A, B, MAC(K_{BS}, N_A \mid N_B \mid A \mid B) \\ &S \to A : \{SK_{AB}\}_{K_{AS}}, MAC(K'_{AS}, N_A \mid B \mid \{SK_{AB}\}_{K_{AS}}) \\ &S \to B : \{SK_{AB}\}_{K_{BS}}, MAC(K'_{BS}, N_B \mid A \mid \{SK_{AB}\}_{K_{BS}}) \end{aligned} \quad (6.1.6)$$

Giao thức này sử dụng chính giao thức SNEP với *strong freshness*. N_A và N_B bảo đảm khóa *strong freshness* cho cả A và B. Giao thức SNEP đảm bảo tin tưởng (bằng cách mã hóa khóa K_{AS} và K_{BS}) cho việc thiết lập phiên khóa SK_{AB} cũng như xác thực thông điệp (bằng cách dùng khóa K'_{AS} và K'_{BS} cho *MAC*). Để ý rằng *MAC* trong giao thức thứ hai còn nhằm bảo vệ trạm phát khỏi tấn công từ chối dịch vụ bởi vì trạm phát chỉ cần gửi 2 thông điệp đến A và B dù có nhận được hàng loạt yêu cầu từ một nút.

Cuối cùng, việc công bố quảng bá khóa đến mọi nút nhận có thể gây tiêu hao năng lượng nút nghiêm trọng.

6.2 TẤN CÔNG TỪ CHỐI DỊCH VỤ DoS

Vấn đề an ninh truyền thông trong mạng SCN bao gồm cả việc xây dựng hệ thống tự phòng vệ trước các tấn công gây cô lập, phá hoại mạng. Tấn công từ chối dịch vụ DoS là dạng sự kiện có tác động đến hiệu năng chung của hệ thống. Các lỗi phần cứng, lỗi phần mềm, cạn kiệt tài nguyên, tác động môi trường hoặc các tương tác qua lại giữa những yếu tố này cũng có thể gây DoS. Các tấn công có thể lợi dụng phát tán lỗi phần mềm gây nên DoS, do đó

vấn đề này cần được lưu tâm và đánh giá đúng mức khi xem xét yếu tố an ninh hệ thống [DOS_02].

Bảng 3. Các lớp mạng và phòng chống tấn công từ chối dịch vụ

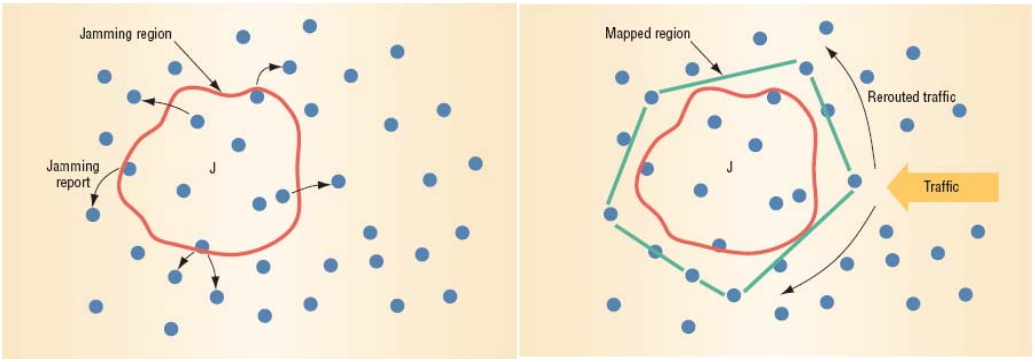
Lớp mạng	Kiểu tấn công	Cách phòng chống
Vật lý	*Jamming*	*Spread spectrum, priority message, lower duty cycle, region mapping, mode change*
	Tampering	*Tamper proofing, hiding*
Liên kết	*Collision*	*Error correcting code*
	Exhaustion	*Rate limittation*
	Unfairness	*Small frames*
Mạng và định tuyến	*Neglect and greed*	*Redundancy, probing*
	Homing	*Encryption*
	Misdirection	*Egress filtering, authorization, monitoring*
	Black holes	*Authorization, monitoring, redundancy*
Chuyển vận	*Flooding*	*Client puzzles*
	Desynchronization	*Authentication*

Hệ thống SCN được xây dựng nhằm mục đích giám sát an ninh có khả năng thích nghi trong các điều kiện ngoại cảnh khác nhau do vậy cần được thiết kế để có thể cung cấp dịch vụ cả khi tồn tại lỗi. Định hướng *robustness* khi thiết kế chỉ ra rằng cần chống tấn công DoS ngay từ mức vật lý. Đặc tính chịu lỗi tại SC cùng các giao thức phù hợp sẽ góp phần hạn chế khả năng gây cạn kiệt tài nguyên dẫn đến hệ thống bị ngừng trệ hay tê liệt.

Những kẻ tấn công luôn có hàng loạt công cụ tấn công khác nhau và việc chiếm hữu hay cô lập 1 vùng SC trong SCN là khả năng hoàn toàn có thể xảy ra và những SC này có thể trở thành những điểm tấn công chuyển tiếp mà không có biểu hiện gì khác thường so với các SC bình thường.

Một vài cách thức triển khai mạng có thể gây gia tăng nguy cơ bị tấn công. Triển khai SCN trong môi trường đã tồn tại sẵn mạng có dây và lưới năng lượng có khả năng tương tác trực tiếp với từng SC sẽ phải chịu những tác động mạnh mẽ khi gặp các tấn công có chủ đích.

Bảng 3 liệt kê các tầng trong SCN và mô tả sự tấn công và phòng vệ tại mỗi tầng mạng.



Hình 15. Phòng chống tấn công DoS kiểu gây nghẽn.

Ví dụ Tấn công gây tắc nghẽn *Jamming*

Là một phương thức tấn công thường gặp trong truyền thông không dây, *jamming* gây nhiễu dải tần số vô tuyến mà điểm nút đó đang sử dụng. Kẻ tấn công có thể gây phân tách hệ thống với *k* ngẫu nhiên nút *jamming* và dẫn đến N nút bị từ chối dịch vụ. Thường thì *k* khá nhỏ so với N. Nếu sử dụng mạng có một dải tần cơ sở thì kiểu tấn công này đơn giản mà khá hiệu quả.

Một nút có thể dễ dàng nhận biết *jamming* từ lỗi của những lân cận nó bởi việc kiểm tra các hằng số năng lượng, không phản hồi, cản trở truyền thông. Các tác động này có kết quả tương tự nhau, tuy nhiên *jamming* ngăn cản nút truyền dữ liệu hoặc thông báo có tấn công về *BS*. Thậm chí cả *jamming* không liên tục có thể gây nên phân tách mạng do dữ liệu truyền thông có thể hữu dụng trong một khoảng thời gian nhất định.

Cách chống *jamming* thông thường là sử dụng truyền thông trải phổ, đa tần số. Ví dụ như NIC tương thích chuẩn 802.11 có khả năng truyền thông

trong 11 kênh truyền khác nhau chút ít về tần số phát (thường dùng kênh 1, 6, 11).

Nếu kẻ tấn công có thể gây tắc nghẽn hoàn toàn mạng, có tác động DoS toàn bộ SCN thì các SC phải có chiến lược thích nghi với tấn công *jamming*, giả dụ như chuyển sang hoạt động ở chế độ dự phòng *lower duty cycle* và tiết kiệm năng lượng tối đa. Sau giai đoạn đó SC có thể kích hoạt và kiểm tra xem việc *jamming* đã kết thúc hay chưa. Bởi việc tiêu thụ năng lượng tiết kiệm, các nút có thể tồn tại qua thời gian bị tấn công do các SC là điểm tấn công gây tắc nghẽn phải tiêu thụ năng lượng liên tục và lớn hơn nhiều lần. Với SCN triển khai rộng rãi thì kẻ tấn công ít có khả năng gây tắc nghẽn toàn mạng mà chỉ có thể gây nghẽn cục bộ. Như trong hình 16 thì miền bị tấn công được cô lập bởi các cập nhật biên vùng tấn công bởi việc chuyển chế chế độ các SC lân cận điểm bị tấn công.

6.3 TỔNG KẾT VÀ BÀN LUẬN

Các chủ đề liên quan đến an ninh trong các hệ thống phân tán luôn là các vấn đề khó, trong SCN cũng vậy. Tuân thủ tiêu chí hướng mở khi thiết kế, việc giả mạo một SC là không quá khó khăn. Để đảm bảo dữ liệu hình truyền về *BS* là dữ liệu sạch, tránh bị sao chép; tránh các tấn công cố ý đến các *s_clu* trong những nhiệm vụ giám sát ... thì điều cơ bản nhất là xây dựng cơ chế an ninh truyền thông.

Tập giao thức SPINs là những nền tảng cho các nghi thức trao đổi thông tin trong mạng truyền thông *ad-hoc* như SCN. SNEP dùng cho truyền thông điểm nối điểm và *μTESLA* dùng trong xác thực gói tin quảng bá. Tất nhiên khi xây dựng ứng dụng thì những phần việc này đã do *kernel* đảm nhận. Trên phương diện người lập trình hay tích hợp hệ thống thì vấn đề thiết thực là xây dựng và quản lý cơ chế phòng chống tấn công từ chối dịch vụ DoS vì

ảnh hưởng của loại tấn công này có tác động rộng hơn các loại xâm nhập khác.

Ví dụ về phòng chống *jamming* có liên quan chặt chẽ với cơ chế QoS dịch vụ và *metric* các tuyến đã được bàn luận trong chương 5 do vậy được chọn lựa để trình bày tại đây.

Cũng như với PC, các kiểu tấn công DoS là thay đổi không ngừng và khả năng xử lý triệt để là không khả thi, tuy nhiên các kiến thức được nêu ở đây sẽ giúp người lập trình chủ động hơn khi xử lý các tình huống tấn công mới.

CHƯƠNG 7 : VẤN ĐỀ PHÂN TẢI, LIÊN KẾT NHIỆM VỤ GIÁM SÁT TRONG SCN

Mô tả bài toán CB1

Đây là dạng bài toán cơ bản thường gặp khi triển khai các giám sát. Vấn đề đặt ra là cần đánh giá tác động khi bổ sung (loại trừ) nhiệm vụ giám sát cho SC và ảnh hưởng như thế nào đến hiệu năng chung của SC, *s_clu* hay SCN.

Điểm khác biệt khi chọn lựa giải pháp cho bài toán CB1 trong SCN so với phương pháp trong hệ thống thế hệ cũ là: trong SCN không tồn tại trung tâm điều khiển ra quyết định tập trung. Tức là việc cân bằng tải hay liên kết nhiệm vụ hoàn toàn được phân tán cho mỗi SC. Bài toán CB1 thuộc lớp bài toán CSP[29] phân tán.

Giải pháp đề xuất

Tiếp cận vấn đề theo hướng xây dựng giải pháp cân bằng tải tính toán xử lý, truyền thông phân tán tại các SC và sử dụng các tác tử thông minh cho mục đích trên.

Các yêu cầu cơ bản của phân tải trong SCN

- Các SC thường được triển khai trong một khoảng cách rộng lớn, do đó việc phân tải cần được thực thi ở mức tác vụ hoặc do nhu cầu ứng dụng quản lý.
- Các nhiệm vụ phân tích dữ liệu hình được xếp vào loại nhiệm vụ thời gian thực, có yêu cầu cao về tài nguyên, yêu cầu đảm bảo về thời gian đáp ứng trong tình huống xấu nhất *WCET*, phân kỳ xử lý dựa trên tốc độ khung hình. Để đảm bảo nhiệm vụ phân tích dữ liệu hình đáp ứng *WCET* thì những yêu cầu tài nguyên như năng lực xử lý, bộ nhớ, thông

[29] *Constraint satifaction problems*

lượng dữ liệu cần được đáp ứng nhằm tránh gây quá tải hay cạn kiệt tài nguyên.

- Nhằm triển khai thuộc tính chịu lỗi mức cao và thời gian đáp ứng thấp, việc phân tán các nhiệm vụ phải được thực hiện như là một hệ đóng nằm trong hệ mở SCN.
- Việc tái phân bố hoặc sắp xếp lại nhiệm vụ do tác động của các sự kiện phần cứng hoặc phần mềm. Sự kiện phần mềm có thể phát sinh do kết quả của nhiệm vụ phân tích dữ liệu hình, gây nên thay đổi nhu cầu sử dụng tài nguyên hoặc thêm bớt các tác vụ khác. Sự kiện phần cứng có thể xuất phát điểm từ những thay đổi liên quan đến tài nguyên sử dụng như năng lực xử lý, băng thông mạng hoặc do sự thêm bớt SC trong hệ thống.
- Nhằm phát triển hướng sự kiện, việc phân chia nhiệm vụ phải *dynamic* trong quá trình hoạt động. Điều này có nghĩa là việc đề xuất phân tải mới hoặc tái phân chia nhiệm vụ phải đáp ứng hoàn toàn đặc tính thời gian thực, là định hướng vươn tới của các nhiệm vụ giám sát. Hai tình huống phân tải thường gặp là: Tăng tải hệ thống và Giảm tải hệ thống. Tuy nhiên chúng đều cùng hướng đến một mục tiêu là đảm bảo QoS dịch vụ trong hệ thống.
- Nhằm đáp ứng các nhiệm vụ giám sát cao cấp, thực thi nhiều nhiệm vụ giám sát cùng lúc thì cần tìm ra hướng liên kết nhiệm vụ tốt nhất có thể thay vì chỉ tìm ra một hướng duy nhất.
- Việc phân tích dữ liệu hình thường yêu cầu nhiều thời gian khởi tạo do phụ thuộc các pha học và pha chuẩn bị. Do đó cần giảm số chuyển tiếp giữa các *host* trong nhiệm vụ, nhằm tối thiểu thời gian khởi động những liên kết không đưa đến kết quả hữu dụng.

<u>Các yêu cầu cơ bản của cân bằng tải trong SCN</u>

Việc cân bằng tải có thể phân rã thành 4 chính sách:

- ***Information policy*** Thường thì *host* điều khiển trung tâm thu thập các thông tin trạng thái và điều tiết cân bằng tải. Tuy nhiên, trong trường hợp SCN việc cân bằng tải cần được thực hiện mà không có *host* điều khiển với sự hỗ trợ của một vài quá trình đồng bộ.
- ***Selection policy*** chỉ ra rằng khi nào một nhiệm vụ được sử dụng. Điều này không phải là vấn đề trong các nhiệm vụ không thời gian thực nhưng nó chỉ ra được là nếu thực thi nhiệm vụ đó thì có đáp ứng thời gian thực trên SC đó hay không.
- ***Activation policy*** Hướng tiếp cận cân bằng tải thường sử dụng ngưỡng công việc để cân bằng việc xử lý. Chính sách này đảm bảo quản lý trạng thái phần cứng và gia tăng những sự kiện phần cứng cũng như phần mềm sẽ này sinh trong nhiệm vụ phân tích.
- ***Location policy*** Chỉ ra được *host* đáp ứng được nhiệm vụ mà thoả mãn những yêu cầu về tài nguyên Việc tìm kiếm này thường được thực hiện bởi việc sắp xếp các tiêu chí, có thể không ước lượng trước được thời gian.

Ràng buộc CSP trong SCN

Trong phần trước đã chỉ ra vấn đề khi áp dụng hướng tiếp cận cân bằng tải cho việc phân chia nhiệm vụ. Việc thỏa mãn những ràng buộc liên quan đến thời gian thực, tuần tự hoặc tích hợp tài nguyên mức cao hoặc tìm ra ánh xạ tĩnh các nhiệm vụ cho các bộ xử lý trong môi trường đa xử lý được gọi là vấn đề CSP.

Một CSP được định nghĩa như là giá trị tổng hợp của $<V,D,P>$ trong SCN là:

- tập $V = \{v_1, ..., v_n\}$ các giá trị, đại diện cho các nhiệm vụ giám sát.

- tập $D = \{D_1, ..., D_n\}$ các miền trong đó mỗi miền D_i là tập các giá trị cho v_i , ở đây đại diện cho các SC, mỗi SC thực thi các nhiệm vụ v_i.
- tập $P = \{P_1, ..., P_m\}$ các tương quan ràng buộc. Trong đó mỗi P_j tương ứng với một vài tập con của V. Ràng buộc P phải nhỏ hơn ngưỡng giới hạn tài nguyên trong SC_j.

Việc gán một giá trị miền đơn nhất cho mỗi thành viên trong nhóm các giá trị được coi là hợp lệ hoặc cục bộ đáp ứng nếu nó không xâm phạm một ràng buộc nào. Nếu có phép gán cho mọi giá trị V thì đó được gọi là một lời giải của CSP.

Nếu giải quyết được CSP thì các liên kết nhiệm vụ trong SCN được giải quyết. Tuy nhiên các tiếp cận trực tiếp để giải quyết CSP là không khả thi do số các kiểm tra yêu cầu tìm lời giải trong không gian lớn những miền giá trị. Số lần kiểm tra là d^V cho V nhiệm vụ và d là kích thước của miền (số lượng các SC). Do vậy bài toán CSP được xếp vào nhóm NP - đầy đủ.

7.1 PHÂN TÁN NHIỆM VỤ CHO SC TRONG SCN

Quá trình tìm lời giải cho bài toán phân tán CSP trong hệ thống được thực hiện theo trình tự sau đây[DDTA_05].

Gán C,V,P trong CSP

Giả thiết đặt ra có m camera trong SCN và n nhiệm vụ.

Khi áp dụng vào bài toán phân tán nhiệm vụ trong SCN các giá trị V, D, P mang các ý nghĩa khác nhau như sau:

- Giá trị nội dung V đại diện cho số lượng nhiệm vụ giám sát được gán cho *surveillance cluster* (*s_clu*). Trong đó để tăng cường hiệu năng chung thì mỗi nhiệm vụ giám sát tương ứng với một giá trị QoS và sẽ không có hai nhiệm vụ nào được xếp ở cùng một mức QoS.

$$V = \{V_1, V_2, ..., V_n\} \quad (7.1.1)$$

- Giá trị vùng miền *D* đại diện cho SC trong *s_clu*.

$$D = \{\Delta_1, \Delta_2, ..., \Delta_n\}; \Delta = \{1,2,...,m\} : \forall \Delta_i = D \quad (7.1.2)$$

- Giá trị ràng buộc *P* đại diện cho ràng buộc tài nguyên chung của các camera. Giá trị này được tổng hợp từ việc lượng hóa các tài nguyên cần có để tạo thành CSP, ví dụ như:
  - Tải tính toán của bộ xử lý.
  - Số lượng bộ nhớ cần dùng.
  - Số kênh DMA cần dùng.
  - Chiếm dụng bus bộ nhớ.

$$\begin{aligned} &A = \prod^{R}(m,n) = \{A_1, A_2, ..., A_p\}; p = m^n \\ &A_j = \{a_1, ..., a_n\}; \forall a_i \in \Delta \\ &R = \{CPU, MEM, DMA, BUS\} \\ &P = \left\{ \forall A_j \in A : \forall d \in \Delta : \forall r \in R : \forall a_i \in A_j : \forall a_i = d : \left(\sum_i req(r,i) < avail(r,d) \right) \right\} \end{aligned} \quad (7.1.3)$$

Tìm lời giải cho bài toán CSP cục bộ

Nhằm tìm ra mọi cách phân chia khả dụng, thỏa mãn ràng buộc, cây giải pháp được xây dựng và duyệt. Cây giải pháp này chứa mọi kết hợp có thể của nhiệm vụ (nút) và cách kết hợp (cành). Tuy nhiên, để tăng hiệu quả duyệt, các cành thừa sẽ được lược bỏ. Với 2^n - *1* cành và 2^n nút đã được kiểm tra thỏa mãn ràng buộc, thì độ phức tạp của CSP cục bộ của mọi camera là $O(2^n)$. Dưới đây là mã thuật toán cho việc tìm lời giải cho CSP cục bộ.

Bảng 4. Thuật toán CSP cục bộ

	Thuật toán CSP cục bộ
1.	**function** solveCSP
2.	*resetLevel (0)*
3.	**for** (*allLevels >0*) *l* **do**
4.	*resetLevel(l)*

	Thuật toán CSP cục bộ
5.	**for** (*allAllocations on level l -1*) A **do**
6.	**for** (*allTaks rightside from A*) T **do**
7.	newAllocation ← A + T
8.	**if** (*isFeasible* (newAllocation)) **then**
9.	*accept* (newAllocation)
10.	**end if**
11.	**end for**
12.	**end for**
13.	**end for**

Giảm độ phức tạp tính toán bằng phương pháp tỉa sớm

Để áp dụng phương pháp tỉa cành sớm, nhằm tăng hiệu năng của thuật toán, thuật toán trên cần được hiệu chỉnh một chút với việc đặt ngưỡng T_c. Ngưỡng T_c có giá trị tương ứng với trọng số của cách phân chia đã được phát hiện. Và bản thân hệ thống tự thay đổi trạng thái của nó.

Bảng 5. Thuật toán CSP cục bộ có tỉa sớm

	Thuật toán CSP cục bộ có tỉa sớm
1.	**function** solveCSP
2.	*resetLevel (0)*
3.	**for** (*allLevels >0*) *l* **do**
4.	*resetLevel(l)*
5.	**for** (*allAllocations on level l -1*) A **do**
6.	**for** (*allTaks rightside from A*) T **do**
7.	newAllocation ← A + T
8.	actCosts ← *calculateCosts* (newAllocation)
9.	**if** (*isFeasible* (newAllocation)) **AND**
	actCosts < T_C **then**

Thuật toán CSP cục bộ có tỉa sớm	
	accept (newAllocation)
10.	**end if**
11.	**end for**
12.	**end for**
13.	**end for**

Hợp nhất các lời giải thành phần

Sau đó là thao tác tổng hợp các lời giải thành phần để có giải pháp tổng thể. Trong hệ thống phân tán thì việc hợp nhất cũng được tiến hành phân tán. Tuy nhiên một cách hợp nhất song song cũng có thể được sử dụng. Khi m là số camera thì *ld(m)* số bước kết hợp tuần tự cần thiết.

Số bước kết hợp phù hợp với kích cỡ thực của *s_clu* và liên quan đến việc khởi tạo camera. Thứ tự hợp phải không gây ảnh hưởng đến giải pháp tổng thể. Toán tử hợp này cần tuân thủ quy tắc:

- Giao hoán: $A \oplus B = B \oplus A$
- Không thứ tự: $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
- Không phân phối: $(A \circ B) \oplus C \neq (A \oplus C) \circ (B \oplus C)$

Phép hợp đơn giản gồm có hai bước.

- Bước 1: phát hiện các kết hợp có thể.
- Bước 2: kiểm tra các kết hợp này và loại bỏ những kết hợp bị vượt ngưỡng.

Việc kết hợp lời giải cũng giống như tạo cây. Việc duyệt cây theo thứ tự trái sang phải tại mỗi mức chính là tìm kiếm lời giải cho việc kết hợp các SC với các nhiệm vụ. Một trong những tiêu chí để tỉa sớm là khi duyệt cây thì tìm kiếm các cách phân chia bị xung đột và loại bỏ nó. Ví dụ như:

- $\{A, B\}^1 \oplus \{C\}^2$ nhiệm vụ A, B triển khai trên SC_1 và nhiệm vụ C triển khai trên SC_2 là không có xung đột.

- $\{A,B\}^1 \oplus \{A\}^2$ nhiệm vụ A, B triển khai trên SC_1 và nhiệm vụ A cũng triển khai trên SC_2 là có xung đột xảy ra.

Bảng 6. Thuật toán trộn hai thành phần

	Thuật toán trộn hai thành phần
1.	**function** mergeAllocations
2.	**for** (*all levels*) L1 **do**
3.	**for** (*all levels* < (*number of levels* - L1)) L2 **do**
4.	Result ← *mergeLevel* (L1, L2)
5.	*addResult* (Result, L1 + L2)
6.	**end for**
7.	**end for**
1.	**function** merge Level (L1, L2)
2.	**for** (*all allocations on level* L1) A1 **do**
3.	**for** (*all allocations on level* L2) A2 **do**
4.	**if** (A1 *not intersecting* A2) **then**
5.	Result ← A1 *union* A2
6.	**if** (costs(Result) < T_C) **then**
7.	MergedResults ← MergedResults + Result
8.	**end if**
9.	**end if**
10.	**end for**
11.	**end for**

Tìm hướng liên kết khả dụng

Các thao tác trên đã tìm ra tập các lời giải con, để tìm ra các lời giải khả dụng cần lượng giá tài nguyên. Tài nguyên chung sẽ bao gồm tất cả các chi

phí con có đặc điểm khác nhau, đại diện cho các hạn chế hiện có của hệ thống.

- ***Resoure Costs*** (C_R) chi phí tài nguyên. được tính bởi tỷ lệ của tài nguyên cần dùng và tài nguyên hiện có. Do có nhiều loại tài nguyên nên công thức tổng quát là:

$$C_R = \sum_{\forall R_i} C_{R_i} * k_{R_i} \quad (7.1.4)$$

  trong đó k_{R_i} là trọng số để phân chia ưu tiên các tài nguyên.

- ***Data-Transfer Costs*** (C_T) thường thì có hai loại chuyển vận dữ liệu là chuyển vận ngoài giữa các SC thông qua giao diện mạng và chuyển vận nội tại SC như tại bus PCI hoặc giao diện bộ nhớ. Chi phí này được tính theo công thức:

$$C_T = \sum_i \left(Datatransfer_{Interface\,i} * Slowness_{Interface\,i} \right) \quad (7.1.5)$$

- ***Migration Costs*** (C_M) chi phí này gồm thời gian chuyển vận dữ liệu giữa các SC C_{MT} (= 0 nếu không cần trao đổi dữ liệu) và thời gian trễ chờ do thứ tự phụ thuộc của các công việc C_{MI} (= 0 nếu không cần trao đổi dữ liệu).

$$C_M = k_{MT} * C_{MT} + k_{MI} * C_{MI} \quad (7.1.6)$$

  trong đó k là trọng số

- ***Affinity Cost*** (C_A) chi phí quan hệ. Gồm có quan hệ C_{AC} và C_{AS}
  - *Cluster Affinity Cost* (C_{AC}) chi phí quan hệ trông *s_clu* đặt ra giá trị ưu tiên chỉ ra những nhiệm vụ nào được gán cho SC trong trường hợp hệ thống quá tải.
  - *Scene Affinity Cost* (C_{AS}) yêu cầu các liên kết đến những camera trong những khung cảnh. Chi phí này càng cao càng chỉ ra khả năng cao việc phân chia nhiệm vụ cho các camera. Nhằm ngăn chặn việc phân chia các nhiệm vụ có quan hệ khung cảnh cho

một camera khác thay vì camera được lựa chọn thì chi phí trên được nhân với hệ số điều chỉnh k_L. Hệ số này được gán bằng k_{RP}, gọi là *Remote Penalty* nếu nhiệm vụ được gán cho camera không mong muốn.

Chi phí quan hệ chung được gán bằng số cao hơn trong *cluster-affine* và *scene-affine*. Tuy nhiên nếu cả hai giá trị này đều triệt tiêu thì chi phí quan hệ được gán bằng ∞.

$$C_A = \begin{cases} \infty & C_{AC} = C_{AS} = 0 \\ \max\left(k_{AC} * C_{AC}, k_{AS} * C_{AS}\right) & else \end{cases} \quad (7.1.7)$$

- ***QoS Costs*** - (C_{QoS}) Nhiệm vụ giám sát được vận hành ở các mức ưu tiên khác nhau. Thông thường thì nó được thiết kế để mọi nhiệm vụ hoạt động với chất lượng cao nhất có thể, tuy nhiên với sự xuất hiện của C_{QoS} thì các nhiệm vụ có thể hoạt động ở mức ưu tiên thấp hơn. Mức QoS cấp 1 là cấp cao nhất, ưu tiên nhất và các cấp tiếp theo có mức ưu tiên kém dần.

$$\begin{aligned} QoS_{cost} &= \{C_{QoS1}, ..., C_{QoSq}\} \\ C_{QoS1} &< C_{QoS2} < ... < C_{QoSq} \end{aligned} \quad (7.1.8)$$

Nhằm đảm bảo duy trì dịch vụ không vượt quá mức tối thiểu, cần đưa ra giá trị ngưỡng QoS_{min}. Ngưỡng này đảm bảo việc gán thêm nhiệm vụ không làm vi phạm chi phí QoS chung của hệ thống.

$$C_{QoS} = \left| C_{QoS_{current}} - C_{QoS_{desired}} \right| \quad (7.1.9)$$

Từ đó lượng giá tài nguyên tổng hợp sẽ được tính theo công thức:

$$C_{Total} = k_R * C_R + k_T * C_T + k_M * C_M + k_A * C_A + k_{QoS} * C_{QoS} \quad (7.1.10)$$

Việc xuất hiện hệ số tỷ lệ (k_x) là do các giá trị thành phần có thể ở những thang độ khác nhau cần chuẩn hóa, và để dễ dàng cho việc hiệu chỉnh và tối ưu đóng góp của các giá trị thành phần này. Điều chỉnh các hệ số này mang lại những biến đổi chung cho toàn hệ thống.

- Chống trễ khi truyền nhiệm vụ ($k_M \uparrow$) Có thể dẫn đến các nhiệm vụ phải hoạt động ở mức QoS thấp.
- Yêu cầu mức QoS cao ($k_{QoS} \uparrow$) Các nhiệm vụ có mức ưu tiên thấp có thể không được thực thi.
- Đảm bảo liên kết các nhiệm vụ có quan hệ khung cảnh đến đúng SC mong muốn ($k_A \uparrow$)
- Liên kết nhiệm vụ khi có kết nối mạng tốc độ cao hơn ($k_T \uparrow$)

Để tìm ra lời giải tốt nhất, cần tìm trong tập hợp lời giải đã tìm được lời giải có chi phí thấp nhất. Thường thì thao tac này được chọn như là một liên kết mới và ứng dụng sẽ được lưu ý về những SC mới này hoặc thay đổi mức QoS cho phù hợp.

Tái sử dụng lời giải, liên kết cũ *reusing allocations*

Việc phát hiện các lời giải có thể cho CSP phân tán là phức tạp và tốn nhiều thời gian. Thêm vào đó, thời gian để giải quyết CSP phân tán thì không ước lượg được thậm chí cả khi áp dụng các kỹ thuật tối ưu hiện thời. Do vậy những lời giải đã được tìm ra cần làm cơ sở, tái sử dụng để tăng hiệu năng chung và giảm thời gian tìm kiếm.

Đơn giản nhất, việc tái sử dụng các liên kết cũ có thể sử dụng khi tải hệ thống gia tăng, do đó tập các liên kết cần giảm trừ. Nhằm phát hiện những liên kết cũ nào có thể sử dụng lại, mỗi liên kết cũ sẽ được kiểm tra lại tính thích nghi. Độ phức tạp của quá trình này là $O(k)$ với k liên kết. Hai trường hợp thường gặp dẫn đến tăng tải tính toán hệ thống là:

- gia tăng các yêu cầu nhiệm vụ (mức QoS cao hơn)

  các yêu cầu tài nguyên có thể thay đổi tăng khi nhiệm vụ giám sát được nâng mức QoS. Việc thêm các nhiệm vụ mới vào *s_clu* cũng làm tăng tải hệ thống. Tuy nhiên trường hợp này không vận dụng được *reusing allocations*.

- giảm tài nguyên hiện có trong SC, hay loại bỏ SC.

7.2 ỨNG DỤNG TÁC TỬ THÔNG MINH

Phù hợp với mô hình kiến trúc phần cứng và phần mềm đã đề cập đến trong chương 2, kỹ thuật sử dụng tác tử di động[30] được lựa chọn cho bài toán này. Các tác tử này có thể hoạt động độc lập, sử dụng truyền thông theo thiết kế và di động. Quá trình triển khai được chia thành 4 bước:

1. Cài đặt các dịch vụ tác tử cơ bản để hỗ trợ phân chia nhiệm vụ.
2. Thực thi các thuật toán CSP cục bộ.
3. Thuật toán kết hợp và ghép nối.
4. Tổng hợp mọi thành phần vào một hệ thống đơn nhất.

Đề xuất sử dụng tác tử *Diet*[31] cho hệ thống SCN do nó chiếm dụng không quá nhiều tài nguyên và cho hiệu năng sử dụng chấp nhận được. *Diet-agent* nguyên bản được thiết kế với các truyền thông đơn giản nhưng cũng đã hỗ trợ các truyền thông đồng bộ nên cần được hiệu chỉnh để có thể hoạt động trong bài toán phân chia nhiệm vụ. Trong trường hợp cần thiết có thể xây dựng bổ sung tác tử truyền thông trung tâm. Để tạo môi trường hoạt động cho các tác tử này được nâng cấp và bổ sung để quản lý các chức năng chính trong SC. Trong SCN, cần xây dựng các tác tử mở rộng sau:

<u>Tác tử DSP</u>

Là mở rộng của tác tử *SmartCam* quản lý việc triển khai các nhiệm vụ giám sát trong hệ thống, bao gồm cả mã quản lý tác tử *SmartCam*; tải mã nhị

[30] Mobile-Agent là một phần mềm (chương trình) tồn tại trong một môi trường nhất định, có khả năng thay đổi vị trí, tự động hành động phản ứng lại sự thay đổi của môi trường nhằm đáp ứng mục tiêu đã được thiết kế trước. Tác tử có các tính chất chung như: tự động, bền bỉ, phản ứng tức thì, hướng mục tiêu, khả năng giao tiếp với tác tử khác hoặc con người.
[31] Decentralized Information Ecosystem Technologies http://diet-agents.df.net

phân cho DSP; đề xuất các mức QoS ứng dụng cũng như tính toán chi phí cho mỗi nhiệm vụ giám sát.

Do đặc tính di động nên tác tử DSP phải có khả năng di dời giữa các *host*. Trong trường hợp có dịch chuyển thì tác tử DSP sẽ tạm dừng xử lý tính toán của DSP này, chuyển trạng thái và tái hoạt động phía bên kia.

Theo phần việc đảm nhận mà các tác tử DSP hoạt động ở các chế độ như:

- ***Node Information Agent*** (NIA) làm nhiệm vụ truyền thông tin về nút, SC cho các nút, SC khác. NIA được sử dụng trong quá trình khởi tạo hoặc hiệu chỉnh *s_clu* để cung cấp thông tin cho các SC thành viên và nghe ngóng dịch vụ của các tác tử khác.
- ***Cluster Manager Agent & Proxy*** (CMA CMP) điều khiển các SC thành viên trong *s_clu*. CMA được liên kết đến mọi SC trong *s_clu* tuy nhiên nó chỉ truyền thông tương tác đến CMP. CMP cũng liên kết đến mọi SC trong *s_clu* và chỉ chuyển tiếp thông tin điều khiển hoặc hàng đợi đến CMA, nơi mà phần lớn các lệnh điều khiển liên quan đến bổ sung hay gỡ bỏ SC trong nhóm. Cấu trúc CMA/CMP là trong suốt đối với người dùng và các điều khiển vận hành hay hàng đợi luôn được hướng tới NIA.
- ***DSP Interaction Agent*** (DIA) nhằm tích hợp các DSP trong hệ thống. DIA đơn giản ghép nối các DSP-lib, cung cấp các chức năng tải mã nhị phân xuống DSP và quản lý các mã đó. Hơn thế DIA còn xuất trình và đăng ký dịch vụ cho hệ truyền thông điệp giữa các DSP.

Ứng dụng trong DSCP

Quá trình phân bổ nhiệm vụ giám sát được bắt đầu dựa vào sự kiện phần cứng hoặc phần mềm xảy ra, trong trường hợp có SC thành viên trong *s_clu* bị thay đổi hoặc khi có *s_clu* mới được tạo lập.

Khi đó SC nắm bắt được sự kiện sẽ chủ trì việc phân bổ gọi là *initiator* thông qua các thủ tục liên kết. Hành động đầu tiên là kiểm tra các SC khác xem chúng đã đang được phân chia nhiệm vụ hiện thời như thế nào.

Initiator tạo các tác tử công việc[32] trong mọi SC trong *s_clu*. Những tác tử công việc này bao gồm danh sách các nhiệm vụ giám sát được gán cho *s_clu*, mức QoS của chúng và các yêu cầu ràng buộc.

Các tác tử công việc lập thành hàng đợi cho DIA nhằm phát hiện số lượng và tải hiện thời của các DSP trong SC. Dựa trên những hiểu biết này, mọi tác tử công việc giải quyết vấn đề CSP cục bộ và thông báo về SC *initiator*, đến đây là hoàn thành phần việc cục bộ.

Nhằm hợp nhất các liên kết thành phần, *initiator* tạo nhóm hợp nhất các SC hoặc các tác tử công việc trong đó. Một tác tử công việc trong nhóm được coi là *master* trong khi cái còn lại được coi là *slave* của quá trình hợp nhất.

Sau khi *initiator* đã tạo được các nhóm hợp nhất thì các tác tử *slave* di trú sang tác tử *master*, nơi mà chúng hợp nhất các liên kết thành phần. Khi quá trình hợp nhất kết thúc *master* thông báo điều này đến *initiator*. Tác tử *slave* tạm dừng tại đây.

Initiator lại tiếp tục tạo các nhóm hợp nhất khác và khởi tạo quá trình kết hợp. Những bước này được thực hiện cho đến khi mọi liên kết thành phần được kết hợp toàn bộ. Nhằm đảm bảo cho kết quả của phép hợp nhất được lưu giữ tại Initiator SC thì *initiator* luôn là tác tử *master* trong các thủ tục hợp nhất mà có sự tham gia của nó.

Cuối cùng, tác tử *master* trong *Initiator* SC đưa ra danh sách cuối cùng của các liên kết, đã được kiểm tra về tính toàn vẹn vì việc trộn không đảm bảo mọi phương án phù hợp với mọi nhiệm vụ. Trong trường hợp không tìm

[32] Load Distribution Worker agent LDW

ra liên kết trọn vẹn thì tác tử *master* thông báo liên kết gần toàn vẹn nhất cho *initiator*. Tác tử *master* tạm dừng ở đây.

Initiator cuối cùng quyết định chọn hướng liên kết có giá thấp nhất và bắt đầu triển khai. Khi đó *initiator* mới thông báo đến các tác tử DSP để chúng thay đổi liên kết. Các tác tử có thay đổi mức QoS thì chỉ cần đổi mức xử lý trong khi các tác tử khác có thể phải di trú sang SC mới. Vào thời điểm này *initiator* quảng bá phương án được chọn cho mọi SC trong *s_clu* và trả quyền điều khiển cho SC.

Tái sử dụng các liên kết

Nhằm mục đích sử dụng lại các phương án liên kết không còn hữu dụng thì những phương án này thay vì bị loại bỏ thì được đánh dấu 'loại', ghi chú rõ nguyên nhân tại sao loại và lưu lại. Khi đó thì trong trường hợp hệ thống được giảm tải thì có khả năng sẽ dùng lại được những phương án đã bị loại đó.

Trong các nhiệm vụ cụ thể sẽ có khả năng xuất hiện tác tử chỉ tồn tại trong một khoảng thời gian ngắn. Ví dụ như trong bài toán theo vết đối tượng trong thị trường giám sát thì tác tử lần vết này chỉ tồn tại trong một thời gian ngắn tại SC nên các tác tử loại này được gọi là *transient agent*.

Bài toán ví dụ về theo vết đối tượng là người trong thị trường giám sát

Khi có đối tượng di chuyển trong thị trường quan sát của SC_A thì các thông tin lưu vết ví dụ như *color histogram* sẽ được sử dụng để tách biệt miền chuyển động, làm mịn khối và nhận dạng người theo tỷ lệ đầu/thân. Khi này SC_A được coi là *initiator*. Với việc dự đoán hướng chuyển động của đối tượng thì SC_A sẽ tạo ra các tác tử công việc cho các SC trong *s_clu* tại góc hướng đó là các nhiệm vụ giám sát như tính toán tốc độ di chuyển, loại hình chuyển động, chụp hình, quay đoạn hình, bắt tiêu điểm vào vùng đầu mặt... QoS của

các nhiệm vụ này được sử dụng là QoS chung tuy nhiên mức QoS có thể được nâng lên nếu giá trị đặc trưng tính toán bị vượt ngưỡng[33] cho phép.

Các nhiệm vụ giám sát trên lập thành hàng đợi cho DIA để xác định số lượng và tải hiện thời của các DSP trong các SC trong *s_clu* được dự đoán là sẽ có đối tượng di chuyển đến. Bản thân các SC này có thể cũng đang thực thi các nhiệm vụ giám sát khác hoặc đang ở chế độ tiết kiệm năng lượng. Do vậy không phải là nhiệm vụ sẽ được gán cho tất cả các SC. Những tính toán cục bộ tại các SC này sẽ được thông báo về *initiator* SC_A.

Chuyển sang giai đoạn hợp nhất các liên kết thành phần, dĩ nhiên tại thời điểm này *initiator* SC_A vẫn là *master*. Trong trường hợp không có liên kết đảm bảo QoS chung thì các tác động có thể của *master* là kích hoạt các SC đang hoạt động ở mức tiết kiêm năng lượng hoặc giảm tải[34] tại các SC trong *s_clu*.

Cuối cùng, khi đối tượng đã di chuyển ra khỏi thị trường quan sát của SC_A thì tác tử *master* này được chấm dứt khi có thông báo phản hồi từ các tác tử *slave* là đã có đối tượng trong thị trường quan sát. Các thông số trạng thái của tác tử này đã được di trú sang tác tử tại SC khác. SC này được coi là *initiator* mới.

Qua thử nghiệm đánh giá trên các hệ thống tương tự SCN thì quá trình CamShift có trễ dưới 3 giây [DSC_06]. Trễ này là chấp nhận được trong giám sát an ninh khu vực dân sự.

---

[33] Ví dụ như tốc độ di chuyển lớn hơn 2.5m/s trong vùng bảo vệ ngoài giờ HC; không đi thẳng mà cúi khom người hoặc có mang vác vật bất thường. Những giá trị đặc trưng này không thể chỉ nhận biết được từ một camera đơn lẻ.

[34] Ví dụ như chuyển độ phân giải quan sát hay tốc độ khung khi ghi hình. Đề xuất trường hợp xấu nhất là đưa nhiệm vụ snapshot lên mức QoS cao nhất và truyền cảnh báo về *BS*.

7.3 TỔNG KẾT VÀ BÀN LUẬN

Chương 7 tập trung bàn luận về vấn đề cân bằng tải và phân chia nhiệm vụ giam sát trong SCN. Dạng bài toán CB1 này là nền tảng cho việc phát triển ứng dụng trong SCN.

Thực tế cho thấy hệ thống phải có chiến lược xử lý phân tán CSP đáp ứng thời gian thực nên việc ứng dụng các tác tử di động là hoàn toàn hợp lý. Các kết quả thu nhận được tại đây là:

- Khi thiết kế: việc chia hệ thống thành các nhóm *s_clu* ngoài những lợi điểm có được khi định tuyến ZRP, đánh địa chỉ AFA, đồng bộ bộ đếm RBS còn góp phần giảm thiểu không gian lời giải của CSP. Các *s_clu* này có thể chồng lẫn nhau và liên kết trong đó có thể thay đổi khi hoạt động bởi sự đảm nhiệm của các tác tử di động.
- CSP phân tán là giải pháp chính tắc và hiệu quả cho *s_clu* khi bổ sung (loại bớt) nhiệm vụ giám sát. Nhằm phân chia nhiệm vụ cho các SC, cần sử dụng hướng tiếp cận phân tán theo vùng miền, để xử lý các CSP cục bộ mà vẫn tiết kiệm được chi phí truyền thông.
- Nhằm tìm ra lời giải khả dụng và lược bớt sớm trong cây lời giải, cần định nghĩa các hàm lượng giá, trong đó tích hợp các lớp chi phí khác nhau nhằm mô tả mục tiêu đúng nhất. Các chi phí liên quan đến tài nguyên phần cứng cần được lưu ý để đảm bảo đáp ứng thời gian thực của nhiệm vụ giám sát.
- Hàm lượng giá được tích hợp ngay từ những tầng thấp của thuật toán CSP để có thể giảm bớt không gian tìm kiếm ngay từ những tầng này nhằm tăng hiệu năng chung.
- Việc phân chia nhiệm vụ phải hướng sự kiện và tự động bởi các tác tử di động trong SC.

- Khi ước lượng: SC phải tiên đoán được khả năng đáp ứng về thời gian khi tham gia vào nhiệm vụ liên kết.

CHƯƠNG 8 : LƯU TRỮ NỘI DUNG TRONG SCN

Nếu bài toán CB1 liên quan đến hiệu năng hoạt động hiện thời của SCN thì bài toán CB2 liên quan đến quá khứ của hệ thống. Trong các hệ thống an ninh giám sát khả năng tra cứu lần vết, dò tìm thông tin trong kho lưu trữ cũng là một tiêu chí đánh giá quan trọng. Các dữ liệu trong quá khứ thường được sử dụng với mục đích tìm kiếm các hình ảnh bất thường trong sự kiện, phân tích hành vi ...

Các ứng dụng lưu trữ nội dung thông tin trong những hệ thống phân tán đã được nghiên cứu và triển khai trong nhiều năm gần đây. Trong SCN các SC thu thập số liệu hình từ môi trường và phát sinh dữ liệu cho việc xử lý lọc, chuyển đổi và lưu trữ cung cấp hạ tầng thông tin chung cho người dùng.

Lưu trữ nội dung thông tin bao gồm những điểm chính sau: nơi lưu trữ số liệu, các đánh chỉ mục dữ liệu và các ứng dụng truy xuất các thông tin này như thế nào để tối thiểu hóa trễ truyền thông.

Cách tiếp cận đơn giản nhất là dòng dữ liệu sự kiện được truyền về điểm tập trung *BS*, tại đó đánh chỉ mục dữ liệu để phục vụ trong những truy xuất sau này. Tuy nhiên trong hệ thống động, năng lượng hạn chế, đa bước truyền như SCN thì biện pháp này là không hiệu quả.

Cách tiếp cận khác là các SC lưu trữ dữ liệu và sự kiện cục bộ (ví dụ như ghi vào thẻ nhớ Flash) do đó các thao tác ghi là tại chỗ và không cần truyền thông. Yêu cầu truy xuất, ví dụ như một sự kiện tại một SC thành viên, dưới hình thức một thông điệp yêu cầu sẽ được gửi đến SC đó và đề nghị xử lý. Trong những hệ thống nhỏ, yêu cầu đơn lẻ, đơn giản thì mô hình này là khả thi tuy nhiên xác suất tìm kiếm thành công thấp trong môi trường quảng bá như SCN, dẫn đến trễ yêu cầu cao. Các nghiên cứu như *Directed Diffusion* có mang lại một số hiệu quả nhất định, cải thiện khả năng tìm đọc dữ liệu bởi

cơ chế định tuyến thông điệp thông minh nhưng chưa trọn vẹn cho vấn đề này.

Ngoài ra còn có các hướng tiếp cận lai giữa hai hướng trên như *Geographic Hash Table* (GHT) sử dụng cách đánh chỉ mục nội mạng cho lưu trữ nội dung phân tán hoàn toàn tại các SC. Trong phương pháp này, mỗi nội dung dữ liệu có khóa tương ứng và phân bổ trong bảng băm. Do đó, việc ghi dữ liệu sẽ được gửi đến bảng băm nút và cập nhật vào bảng băm nội mạng. Yêu cầu tìm đọc dữ liệu sẽ truy vấn trong bảng băm nội mạng để định vị nút lưu dữ liệu, từ đó tìm đến vị trí tương ứng của dữ liệu trong bảng băm nút.

Phần lớn các cách tiếp cận trên đều giả định hệ thống phẳng, thuần nhất và các SC đều có năng lượng tương đương. Trong hệ thống SCN thì đặc điểm dữ liệu hình có những điểm riêng khác các hệ khác là:

- Phương pháp nén lưu trữ khác nhau phù hợp cho các ứng dụng khác nhau.
- Dữ liệu đa phân giải: các khung hình CIF, QCIF, VGA ... phù hợp cho nhiều yêu cầu giám sát khác nhau.
- Tốc độ khung hình: 2, 10, 15, 25 .... khung hình trên giây phù hợp cho nhiều yêu cầu giám sát khác nhau.
- Ảnh tĩnh có thể coi là trường hợp đặc biệt của dòng dữ liệu hình.
- Ảnh màu và ảnh đa mức xám có ứng dụng khác nhau tùy theo ứng dụng.

Qua nhiều nghiên cứu và ứng dụng, người ta xây dựng nên kiến trúc lưu trữ TSAR[35] [TSAR_05] phản ánh và xác định được tính tự nhiên và đa lớp của các mạng cảm biến cảnh báo và có thể ứng dụng trong SCN. TSAR là thành phần của kiến trúc lưu trữ PRESTO là hợp nhất của lưu trữ nội dung với cơ chế đệm và sự tiên đoán.

[35] Two Tiers Sensor Storage Architecture

Một lợi điểm của những kiến trúc lưu trữ như thế này là dung lượng lưu trữ của bộ nhớ Flash cũng tăng không ngừng theo định luật Moore. Các thiết bị lưu trữ Flash hiện nay yêu cầu rất ít năng lượng cho việc ghi và xóa thông tin.

Kiến trúc TSAR lợi dụng việc lưu dữ liệu và sự kiện cục bộ trên thiết bị lưu trữ Flash tại mỗi SC. Các SC chỉ gửi thông tin định danh rút gọn, còn gọi là *metadata* đến *proxy*. Tùy theo ứng dụng mà *metadata* có thể nhỏ hơn nhiều lần so với dữ liệu gốc. Các *proxy* tương tác lẫn nhau để tạo thành hệ chỉ mục phân tán cho các dữ liệu trong hệ thống. Hệ chỉ mục này cung cấp khung nhìn rộng, logic của các dữ liệu lưu trữ và cho phép các ứng dụng xuất hàng đợi cũng như tìm đọc dữ liệu quá khứ. Cơ chế *look-up* này làm giảm bớt các truy vấn tìm đọc trong hệ thống và giảm mức tiêu thụ năng lượng chung.

Bảng 7. So sánh tính năng các hệ lưu trữ nội dung

HỆ THỐNG	DỮ LIỆU	CHỈ MỤC	ĐỌC	GHI	SẴN SÀNG
Lưu trữ tập trung	Tập trung	Chỉ mục tập trung	Tại điểm lưu trữ	Gửi đến điểm lưu trữ	Có
Lưu trữ cục bộ tại nút	Phân tán	Không	Tìm kiếm rộng, định hướng	Cục bộ	Không
GHT/DCS	Phân tán	Nội mạng	Bảng băm nút	Gửi đến vị trí băm trong nút	Không
TSAR/ PRESTO	Phân tán	Phân tán tại các proxy	Tìm kiếm Proxy, hàng đợi nút	Cục bộ và cập nhật chỉ mục	Có

Việc thiết kế theo mô hình TSAR mang lại bốn lợi điểm:

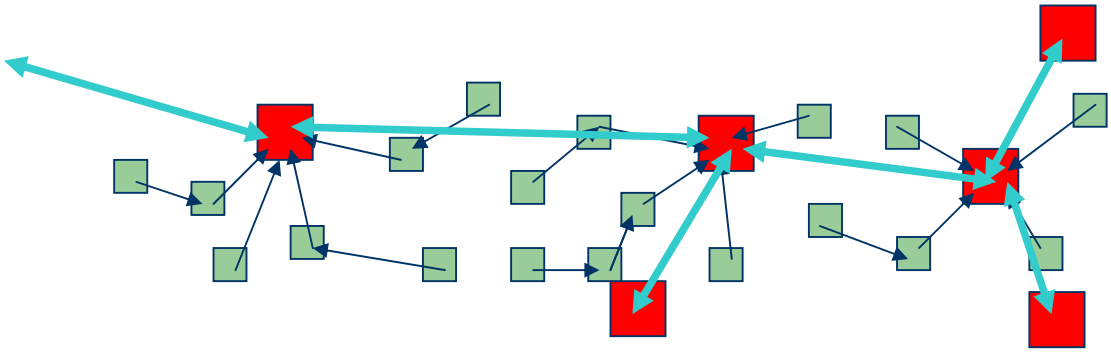
- Cốt lõi của kiến trúc TSAR là kiến trúc đánh chỉ mục phân tán dựa trên *interval skip graph*. Kiến trúc này có thể lưu trữ tổng hợp các dữ liệu của SC và tổ chức của chúng nhằm mục đích tìm kiếm dễ dàng. Cấu trúc dự liệu có độ phức tạp tìm kiếm và cập nhật là $O(\log n)$.
- Tại mức SC, mỗi SC tự bảo trì nội dung thông tin được lưu trữ trong thiết bị nhớ Flash. Trong kiến trúc lưu trữ này, các trạng thái tại SC

được lưu trong chỉ mục *metadata* theo luật xa gần, *multi-resolution*. Các cấu trúc chỉ mục được bảo trì tại các *proxy* và chỉ những yêu cầu trực tiếp hoặc các hàng đợi đơn được định hướng xuống SC. Lưu trữ tại SC ở xa có hiệu quả phụ thuộc vào *proxy*. Các lưu trữ cục bộ được tối ưu cho các truy nhập theo chuỗi thời gian hay phục vụ các ứng dụng có tần xuất sử dụng cao. Các SC theo chu kỳ sẽ gửi thông tin trạng thái tổng hợp của chúng về *proxy*. Tại đây với cơ chế đánh giá *cache-hit cache-miss* sẽ có điều chỉnh tương ứng thông báo ngược lại các SC.

- Xây dựng nguyên mẫu của TSAR dạng đa lớp là tổng hợp của các *proxy* dạng Stargate và các SC dạng Mote. Việc triển khai hỗ trợ nền không-thời gian, giá trị nội dung và hàng đợi dựa trên khoảng cách của dữ liệu.
- Thử nghiệm TSAR bởi kết hợp EmStar/EmTOS lên nguyên mẫu, đánh giá các trễ của hàng đợi *end-to-end* trong mạng đa bước truyền.

8.1 CHỌN LỰA THIẾT KẾ

Mô hình hệ thống



Hình 16. Kiến trúc TSAR với proxy và SC

Các SC đóng vai trò *proxy* quản lý chỉ mục cho nhiều SC lân cận và cũng không hạn chế việc một SC được đánh chỉ mục trong nhiều *proxy*. Các *proxy* trao đổi thông tin theo mô hình đồ thị.

Mô hình sử dụng

Việc thiết kế các hệ lưu trữ như TSAR phụ thuộc vào các truy vấn sẽ xuất hiện trong hệ thống. Các SC cung cấp dữ liệu hình và có hai thuộc tính chính của thông tin này là thời điểm và địa điểm phát sinh sự kiện. Các ứng dụng thông thường chỉ truy vấn các thông tin liên quan thời gian và địa điểm của sự kiện. Có vài cách sắp xếp và phân loại các truy vấn này ví dụ như xây dựng bảng băm dự phòng cục bộ như DIMS.

Các ứng dụng còn có thể xuất các truy vấn dựa trên giá trị nội dung. Và thường thì giá trị nội dung v này nằm trong một khoảng giá trị (v_1,v_2) thay vì là một giá trị nhất định. Việc đánh chỉ mục các truy vấn loại này dựa trên việc quản lý các thông tin tóm tắt từ SC gửi đến *proxy*.

Các truy vấn sử dụng các thông tin tóm tắt của các thời gian và nội dung cũng có thể xuất hiện trong hệ thống. Những truy vấn này yêu cầu được đánh chỉ mục dựa trên cả thời gian và giá trị nội dung.

Nguyên lý thiết kế

Việc thiết kế giải pháp lưu trữ cho SCN được xây dựng dựa trên các nguyên tắc sau:

- ***Store locally, access globally*** kỹ thuật lưu trữ cục bộ được xây dựng dựa trên so sánh mức tiết kiệm năng lượng so với lưu trữ mạng, và ước tính rào cản này không thể vượt qua trong tương lại gần. Tuy nhiên công tác lưu trữ phải được tối ưu cho các nhu cầu truy vấn của ứng dụng và tạo nên một giao diện lô gíc đơn nhất với ứng dụng và cân bằng giữa chi phí truyền thông và lưu trữ.

- ***Distinguish data from metadata*** dữ liệu cần được xác định như là tự giới thiệu với ứng dụng tránh để phải tìm kiếm vét cạn. Để thực hiện điều này, phải tổ chức *metadata* với mỗi bản ghi dữ liệu có các trường dữ liệu với quy tắc tường minh nhằm giúp cho việc xác định hoặc tổ chức hàng đợi trong hệ thống lưu trữ[36]. Việc cung cấp những *metadata* như vậy sẽ giúp *proxy* đánh chỉ mục *metadata* và có khung nhìn logic tổng quát của mọi dữ liệu trong hệ thống. Điều này làm tăng hiệu năng hệ thống, giảm trễ truy vấn và thêm nhiều chức năng hữu dụng.
- ***Provide data-centric query support*** trong các ứng dụng cần chỉ rõ vị trí bản ghi dữ liệu thì hệ thống phải cung cấp được dòng dữ liệu tuần tự như là thông tin được truyền tải liên tục từ điểm truyền thông theo thời gian và vị trí xác định. Để làm được điều này cần phải thường xuyên bảo trì thông tin *metadata* nhằm làm giảm thiểu giá tìm kiếm và theo dõi thông tin.

Thiết kế hệ thống

Với thiết kế TSAR, việc ghi các sự kiện được thực hiện được thực thi tại SC, tuy nhiên trong ứng dụng chỉ bắt được đặc tả nội dung trong *metadata*. *metadata* giúp ứng dụng xác định vị trí và dạng bản ghi dữ liệu cần truy xuất.

Tùy thuộc vào ứng dụng mà *metadata* có bao gồm 2 hoặc 3 yêu cầu soi phóng to thông tin từ bản thân dữ liệu, cũng giống như *metadata* trích chọn đặc trưng của ảnh hay dữ liệu hình hay cung cấp khung nhìn đa phân giải theo yêu cầu.

Nhằm bổ sung cho việc lưu trữ dữ liệu cục bộ, mỗi SC theo chu kỳ cần được cập nhật thông tin tóm tắt cho các *proxy* gần nó. Tổng hợp này bao gồm

[36] Trong *Metadata* ngoài các thông tin như địa điểm, thời gian cần phải có các thông tin đặc thù của dữ liệu hình hay giá trị đặc trưng được kết xuất từ dữ liệu (tọa độ khung nhìn, độ sáng bình quân, ngưỡng phát hiện chuyển động, *entropy*, mặt nạ *Law* ...).

thông tin như địa chỉ SC, khoảng thời điểm (t_1, t_2), con trỏ đến vị trí lưu trữ trên thiết bị nhớ Flash...

Proxy sử dụng thông tin tóm tắt trên để tạo ra bảng chỉ mục. Bảng chỉ mục là toàn cục và lưu trữ tất cả các thông tin về các SC tuy nhiên nó được lưu trữ bộ phận phân tán tại các *proxy*. Do vậy tuy hệ thống là phân tán nhưng ứng dụng nhìn nhận và truy nhập dữ liệu như là hàng đợi cổ điển. Đặc biệt mỗi hàng đợi là không đơn nhất mà là một dãy các hàng đợi thỏa mãn điều kiện tìm kiếm. Có một vài phương pháp thiết lập và duyệt chủ mục phù hợp cho những ứng dụng cụ thể và phương pháp trình bày dưới đây là phương pháp hiệu quả nhất.

Do việc các chỉ mục được xây dựng từ những tổng hợp chung thay vì dữ liệu thực nên việc lần tìm chỉ mục sẽ chỉ kết quả gần chính xác. Lợi điểm của kỹ thuật tổng hợp TSAR là đảm báo kết quả tìm kiếm không bao giờ thất bại hoàn toàn - *false negatives*. Tuy nhiên trường hợp lần chỉ mục bị *false positives* cũng có thể xảy ra khi đối sánh đúng thông tin tóm tắt nhưng lần tìm tại SC lại không thỏa mãn. Các SC có thể phân biệt các *false positive* từ hàng đợi của kết quả tìm kiếm và tính toán được tỷ lệ của chúng. Dựa trên tỷ lệ này TSAR sẽ phát triển kỹ thuật động thích hợp để cân bằng giữa *metadata* và *false positives*.

8.2 CẤU TRÚC DỮ LIỆU

Ở lớp *proxy*, TSAR thực hiện cấu trúc đánh chỉ mục gọi là *Interval Skip Graph* nhằm mục đích tìm kiếm nội tại cấu trúc dữ liệu phân tán có điểm hoặc khoảng giá trị thành phần. *Interval skip Graph* là sự kết hợp của *Interval Tree* - cây tìm kiếm nhị phân dựa trên thời gian, và *Skip Graph* - cấu trúc dữ liệu phân tán cho hệ thống ngang hàng [SG_03].

Đầu tiên, độ phức tạp tìm kiếm sẽ là $O(\log n)$ cho lần truy nhập khoảng thời gian đầu tiên phù hợp giá trị nội dung và độ phức tạp là hằng cho truy nhập các khoảng thời gian phù hợp tiếp theo. Tiếp theo, việc đánh chỉ mục dựa trên khoảng thời gian thì tốt hơn là đánh chỉ mục dựa trên giá trị tổng hợp nội dung.

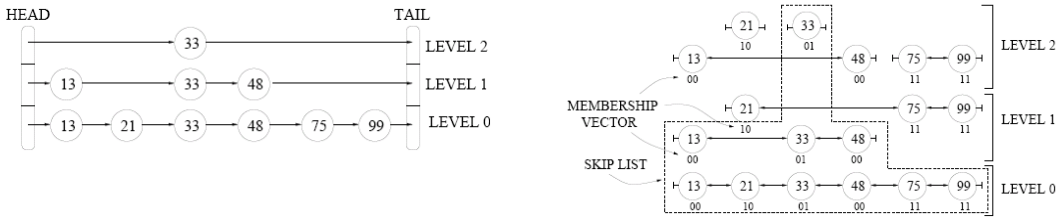
Giả định ban đầu có N_P *proxy* và N_S SC trong mạng SCN 2 lớp. Mỗi *proxy* phục vụ cho nhiều SC và không có giới hạn cụ thể cho số lượng này. Mỗi SC truyền nội dung tổng hợp theo khoảng thời gian của dữ liệu hay sự kiện đến một hoặc vài *proxy*, trong đó khoảng thời gian *i* được đại diện bởi [low_i, $high_i$]. Những khoảng thời gian này có thể có thể bao gồm thời điểm hoặc khoảng giá trị dùng cho đánh chỉ mục dữ liệu. Sẽ không có giả định gì về độ rộng của khoảng thời gian hoặc về quãng cách giữa các khoảng thời gian đó.

Phạm vi các hàng đợi dựa trên khoảng thời gian được đưa ra bởi người dùng đến mạng các *proxy* và SC. Mỗi hàng đợi *q* cần thiết để xác định mọi giá trị chỉ mục về khoảng thời gian [low_q, $high_q$]. Kết quả của *interval skip graph* là đánh chỉ mục mọi khoảng thời gian như là một danh mục tuần tự.

Skip Graph

Một *skip list* là một mở rộng của danh sách liên kết đơn thông thường, tuy nhiên để tăng hiệu quả tìm kiếm người ta sẽ bổ sung các con trỏ đến nút xa hơn thay vì chỉ duy trì con trỏ tuần tự [SL_90]. Mức *0* của một *skip list* là danh sách kết nối đến mọi nút theo sắp xếp tăng của khóa. Cho mỗi *i* lớn hơn không, mỗi nốt ở mức *i-1* xuất hiện trong mức *i* độc lập với một vài xác suất *p* cố định. Trong *doubly-linked skip list*, mỗi nút lưu con trỏ đến trước và trỏ đến sau trong mỗi danh sách nó xuất hiện, tức là có trung bình $\frac{2}{1-p}$ con trỏ cho mỗi nút. Các danh sách ở nút cao hơn giúp cho việc tìm duyệt nhanh hơn.

Quá trình tìm kiếm bắt đầu ở nút mức cao, và tuần tự xuống mức thấp hơn chỉ khi chắc chắn nút cần tìm không có ở mức đó. Do không có nhiều hơn $\frac{1}{1-p}$ nút tại mỗi mức nên thời gian tìm kiếm trung bình sẽ là $O\left(\log n \frac{1}{(1-p)\log\frac{1}{p}}\right)$ với *n* nút ở mức *0*.



Hình 17. Một skip list và skip graph với n = 6 nút và [log n] = 3 mức

Một *skip list* đơn thì thiếu dự phòng và chịu lỗi, nên phải được nâng cấp thành *skip graph* với *membership vector m(x). m(x)* là một từ ngẫu nhiên từ một vài chữ cái cố định và chỉ độ dài $O(\log n)$ tiền tố của *m(x)* cần được tạo bởi giá trị trung bình. Đề xuất sử dụng *membership vector* là liên kết mọi danh sách trong *skip graph* được đánh nhãn bởi một vài từ *w* xác định, và nốt *x* trong danh sách được đánh nhãn *w* khi và chỉ khi *w* là tiền tố của *m(x)*.

Các thuộc tính của *Skip Graph* gồm có:

- ***Ordered index*** các khóa là các số nguyên có thứ tự. Viêc tìm kiếm bao gồm việc so sánh khóa với chỉ mục hiện tại. Thêm vào đó, các con trỏ ở mức thấp nhất trỏ đến hạng mục kế tiếp trong bảng chỉ mục.
- ***In-place indexing*** các phần tử dữ liệu vẫn nằm trên SC khi nó được chèn vào, các thông điệp được truyền giữa các nút để thiết lập liên kết giữa những phần tử với nhau trong bảng chỉ mục.
- $\log n$ ***height*** có tất cả $\log n$ con trỏ liên kết với mỗi phần tử trong đó n là số các phần tử được đánh chỉ mục. Mỗi con trỏ thuộc một mức l trong

khoảng $[0...\log_2 n-1]$ và liên kết với các con trỏ khác ở cùng mức tạo thành chuỗi $n/2^l$ phần tử.

- ***Probabilistic balance*** thay vì sử dụng biện pháp tái cân bằng, *skip graph* sử dụng kỹ thuật cân bằng ngẫu nhiên đơn giản nhằm bảo trì gần với cân bằng hoàn hảo với sai khác nhỏ.
- ***Redundancy và resiliency*** mỗi phần tử dữ liệu nằm trong cây tìm kiếm độc lập, do vậy việc tìm kiếm có thể bắt đầu từ mọi nút trong mạng, loại bỏ có điểm chốt trong gốc tìm kiếm đơn. Thêm vào đó, nếu chỉ mục trỏ đến nút lỗi, có nghĩa là dữ liệu trong nút lỗi không thể truy nhập được, thì các phần tử dữ liệu còn lại vẫn có thể truy nhập trong qua cây tìm kiếm với gốc là nút khác.

Các con trỏ có thể đến từ những phần tử dữ liệu đơn trong cây nhị phân. Con trỏ duyệt từ mức cao nhất với *n/2* phần tử, *n/4* ở mức tiếp theo và tiếp tục. Việc tìm kiếm là duyệt giảm dần từ mức cao nhất đến mức 0, tại mỗi mức so sánh khóa với phần tử tiếp theo trong mức và quyết định khi nào dừng duyệt. Trong trường hợp cây cân bằng hoàn hảo như tại đây có $\log_2 n$ mức thì việc tìm kiếm chỉ có 0 hoặc 1 con trỏ tại mỗi mức. Do đã giả định là các dữ liệu nằm tại các nút khác nhau nên giá trị đo của tìm kiếm là số các thông điệp được truyền nhận và có độ phức tạp tìm kiếm là $O(\log n)$.

Quá trình cập nhật cây từ dưới lên như là B-Tree với gốc được nâng lên tại mỗi mức như là cây sinh trưởng. Bằng cách này, hai chuỗi tại mức *1* luôn bao gồm *n/2* thực thể và không bao giờ cần tách chuỗi khi bổ sung cấu trúc. Việc cập nhật bao gồm việc chọn lựa trong 2^l chuỗi để chèn vào một phần tử tại mỗi mức *l*.

Việc bảo trì *skip graph* cân bằng dựa trên việc theo dõi các phần tử thuộc chuỗi thành phần ở mức *l* chỉ có thể thuộc một trong 2 chuỗi mức *l*+1.

Để chèn một phần tử phải bắt đầu từ mức 0 và chọn ngẫu nhiên 1 trong 2 chuỗi có thể tại mỗi mức và dừng lại khi đến một chuỗi rỗng.

Ý nghĩa của việc triển khai trên là mỗi phần tử tương ứng với một chuỗi bút ngẫu nhiên. Mỗi chuỗi tại mức *l* được cấu thành bởi những bit có cùng *l* bit đầu tiên, nên có 2^l chuỗi có thể tại mỗi mức và chuỗi chia chính xác làm 2 tại mức tiếp theo.

Interval Skip Graph

Một *skip graph* được sử dụng để lưu trữ nội dung đơn, để lưu trữ khoảng thời gian [low_i, $high_i$] cho cho phép tìm kiếm hiệu quả mọi khoảng thời gian mọi giá trị *v*, thì cần phải nâng cấp. Cấu trúc dữ liệu có thể mở rộng phạm vi tìm kiếm theo cách trực tiếp.

Interval Skip Graph được xây dựng bởi việc ứng dụng phương pháp cây tìm kiếm có tham số cho cây tìm kiếm nhị phân tạo bởi *Interval Tree*. Đặc điểm của *Interval Tree* là mỗi nút của nó quản lý thông tin cho một đoạn con và mỗi nút sẽ có hai nút con quản lý 2 đoạn con cuả đoạn được quản lý bởi nút đó. Phương pháp này dựa trên việc quan sát cấu trúc tìm kiếm bởi việc so sánh khóa trong cây nhị phân có thể dùng tiếp để tìm khóa thứ hai không bé hơn khóa đầu tiên.

Với một tập các khoảng thời gian đã sắp xếp theo thứ tự không giảm $low_i \le low_{i+1}$, người ta định nghĩa khóa tiếp theo như là tổng tối đa $\max_i = \max_{k=0\ldots i}(high_k)$. Tập các khoảng thời gian có bao gồm thời điểm *v* có thể được tìm kiếm theo khoảng thời gian bắt đầu (khoảng thời gian có low_i thấp nhất) mà $\max_i \ge v$. Khi đó duyệt cây theo giá trị tăng của chặn dưới cho đến khi gặp khoảng thời gian đầu tiên mà $low_i > v$, và chọn những khoảng thời gian này cho *intersect* v.

Với cách tiếp cận này, người ta đã tham số hóa cấu trúc dữ liệu *skip graph* với mỗi đầu vào lưu trữ một khoảng (chặn dưới và chặn trên) và khóa

thứ hai (tính từ giá trị tối đa của chặn trên). Để tính khóa thứ hai max_i cho đầu vào i, người ta sử dụng $high_i$ và giá trị tối đa được thông báo bởi mỗi lân cận trái của i.

Để tìm kiếm những khoảng thời gian có chứa thời điểm v, đầu tiên người ta tìm kiếm v trong bảng chỉ mục thứ hai max_i, và định vị đường vào đầu tiên có $\max_i \geq v$.Nếu $\text{low}_i > v$ thì khoảng thời gian này không chứa v và các khoảng thời gian tiếp theo cũng vậy nên dừng lại.

- ***Lookup Complexity*** có độ phức tạp tìm kiếm là $O(\log n)$.
- ***Insert Complexity*** trong trường hợp xấu nhất là $O(n)$.

Sparse Interval Skip Graph

Trong phần trước đã chỉ ra rằng chi phí chèn và xóa trong *Interval Skip Graph* trong trường hợp xấu nhất có độ phức tạp là $O(n)$ so với $O(\log n)$ của *Interval tree*. Nguyên nhân chính của sự khác biệt này là *skip graph* có một cấu trúc tìm kiếm đầy đủ với gốc là mỗi phần tử để phân tải và chịu lỗi trong hệ thống phân tán. Trong TSAR thì số lượng nút *proxy* là nhỏ so với số SC (có dữ liệu tổng hợp được đánh chỉ mục) nên dẫn đến tiết kiệm đáng kể khi xây dựng *full search*.

- ***Implementation*** độ phức tạp là $\log_2 N_P + O(1)$, trường hợp xấu nhất là $N_P \log_2 n$
- ***Short-cut search*** trung bình là $\log_2 N_P + O(1)$ bước, ước tính độ phức tạp là $O(\log N_P)$

Bảng 8. So sánh các phương pháp đánh chỉ mục

	Range Query Support	Interval Representation	Re-balancing	Resilience	Small Networks	Large Network
DHT,GHT	no	no	no	yes	good	good
Local index, flood query	yes	yes	no	yes	good	bad

P-tree, RP* (distributed B-Tree	yes	possible	yes	no	good	good
DIMS	yes	no	yes	yes	yes	yes
Interval Skip Graph	yes	yes	no	yes	good	good

8.3 LƯU TRỮ DỮ LIỆU VÀ THÔNG TIN TÓM TẮT

Ở mức các SC bình thường, TSAR triển khai hai kỹ thuật quan trọng. Thứ nhất, là kỹ thuật lưu trữ cục bộ tại SC với tiêu chí tối ưu cho các thiết bị có dung lượng nhớ cố định. Thứ hai, kỹ thuật tổng hợp thông tin thích hợp cho phép SC có thể thay đổi tính chất của dữ liệu và hàng đợi.

Lưu trữ cục bộ tại SC

Interval skip graph cung cấp một kỹ thuật hiệu quả để tìm kiếm các nút SC chứa dữ liệu cần cho hàng đợi. Những hàng đợi này sẽ được định tuyến đến SC, nơi lưu giữ những bản ghi và phản hồi ngược đến *proxy*. Để cho phép những tìm kiếm kiểu này, mỗi SC trong TSAR tự bảo trì một các bản ghi cục bộ. Tất nhiên, do có giới hạn về tài nguyên và năng lượng nên việc bảo trì dữ liệu cục bộ phải được cân đối trong phạm vi đó.

Ngoài các dữ liệu hình thuần túy cần cân nhắc giữa đa phân giải, sắc màu và tốc độ khung thì các bản ghi còn bao hàm nhiều loại dữ liệu khác nhau như dòng dữ liệu theo thời gian *streaming media*, dữ liệu tạm, các dữ liệu đã qua xử lý như biến đổi FFT, biến đổi *Wavelet*, *clustering*, đối sánh hay trích chọn đặc trưng ...

Timestamp	Calibration Parameters	Data/Event Attributes	size	Opaque Data

Hình 18. Bản ghi lưu trữ đơn

Các dữ liệu lưu cục bộ được tập hợp thành các bản ghi và nối vòng logic. Các bản ghi mới được xếp vào đuôi của vùng đệm. Hình 21 là biểu diễn

khuôn dạng của mỗi bản ghi: gồm có trường *metadata* bao gồm các nhãn thời gian, chỉ số SC, thông số chung ... Các dữ liệu hình thuần túy được lưu trong trường dữ liệu của bản ghi này. Trường dữ liệu được coi là như là nguyên khối trong đặc tả ứng dụng vì các thông tin chung đã được đưa vào trong metadata do vậy hệ thống lưu trữ không cần quan tâm chi tiết đến trường này. Như trong các hệ xử lý ảnh camera thì các ảnh nhị phân là rất nhiều và có thể xếp vào trường dữ liệu vì những thông tin như *texture*, *histogram*, *bit length* ... đã được tính và đặt trong *metadata*. Cuối cùng, TSAR hỗ trợ độ dài biến đổi của trường dữ liệu, ví dụ kích cỡ của bản ghi có thể phụ thuộc vào giá trị trong *metadata* của bản ghi khác.

Ba toán tử có tác động trực tiếp lên hệ lưu trữ tại đây là: khởi tạo, đọc và xóa. Trong đó việc khởi tạo là đơn giản và tường minh, đơn giản chỉ cần tạo bản ghi mới và gắn vào cuối trong chuỗi lưu trữ. Do luôn ghi vào cuối nên hệ lưu trữ luôn phải giám sát và định vị danh sách chỗ trống. Mọi trường trong bản ghi cần được đặc tả ngay tại thời điểm khởi tạo, để có thể tính toán được kích thước chuẩn của bản ghi và số byte mất đi dành cho việc lưu trữ bản ghi này.

Thao tác đọc nhằm mục đích đáp ứng các hàng đợi được sinh ra từ ứng dụng. Trong các cơ sở dữ liệu truyền thông, việc tìm kiếm hiệu quả do việc luôn bảo trì B-tree các khóa chỉ mục. Tuy nhiên với những SC có giới hạn tài nguyên thì việc này là không đơn giản. Thường thì TSAR SC không bảo trì chỉ số nào trong danh sách lưu trữ cục bộ mà chỉ cung cấp *metadata* theo chu kỳ đến *proxy* và lưu một cảnh báo về vị trí của các bản ghi trong bộ nhớ flash.

Dù cho dung lượng lưu trữ cục bộ có thể mở rộng, nhưng vẫn có lúc cần ghi đè lên dữ liệu cũ, đặc biệt là trong các ứng dụng đòi hỏi tốc độ dữ liệu cao. Điều này được thực hiện bởi kỹ thuật lưu trữ đa phân giải sẽ được trình bày kỹ lưỡng trong phần dưới hoặc đơn thuần chỉ ghi đè dữ liệu cũ. Thao tác

xóa bao gồm xóa chỉ mục trên *interval skip graph* tại *proxy* và vùng lưu trữ tương ứng trong bộ nhớ flash của SC được giải phóng.

Tóm tắt theo nhu cầu - Adaptive Summarization

Việc tóm tắt dữ liệu như là hồ dính giữa lưu trữ tại SC với chỉ mục tại *proxy*. Mỗi cập nhật từ SC đến *proxy* bao gồm ba thông tin: tóm tắt nội dung, thời điểm tương ứng với tóm tắt đó và độ lệch bắt đầu/ kết thúc trên thiết bị nhớ flash. Nói chung thì *proxy* có thể đánh chỉ mục khoảng thời gian dựa trên thông tin tóm tắt của khoảng giá trị được ghi trong đó hoặc cả hai. Cách đánh chỉ mục cũ cho phép tìm kiếm nhanh mọi bản ghi trong thời điểm chắc chắn, trong khi cách đánh chỉ mục mới cho phép tìm mọi bản ghi đúng với giá trị đó.

Như đã mô tả trong phần trước, do có sử dụng năng lượng để gửi tóm tắt (tần xuất và độ cô đọng của các tóm tắt này) nên dẫn đến xuất hiện chi phí *false hit* trong các hàng đợi. Việc tần xuất gửi thông tin tóm tắt thưa thì tiêu tốn ít năng lượng, tuy nhiên việc *false hit* lại gây tăng năng lượng khi yêu cầu không truy xuất được dữ liệu.

Độ chi tiết của tóm tắt phụ thuộc vào hai tham số là thời gian khi tóm tắt dữ liệu được xây dựng và truyền đến *proxy* theo nhu cầu được đặc tả bởi ứng dụng. Việc thay đổi kích thước của tóm tắt này rất có giá trị trong nhiều ứng dụng. TSAR đưa ra một kịch bản tóm tắt đơn giản dựa trên tính toán tỷ lệ giữa *false*/*true hit*, giảm (tăng) *interval* giữa các tóm tắt khi tỷ lệ này tăng (giảm) dựa theo ngưỡng.

8.4 TỔNG KẾT VÀ BÀN LUẬN

Chương 8 bàn luận về vấn đề lưu trữ nội dung phục vụ tra cứu trong SCN. Phương pháp lưu trữ được trình bày ở đây là TSAR với sự phân chia chức năng và chế độ hoạt động trong các SC thành các SC, *proxy* và *BS*. Các

proxy là cầu nối giữa SC và *BS*. Điều này cũng tương tự như chia *zone* trong định tuyến ZRP, một SC có thể cung cấp *metadata* cho nhiều *proxy*, và từ *BS* có thể truy vấn thông tin từ nhiều *proxy*.

Các dữ liệu tại SC được lưu trữ trên thiết bị nhớ Flash, và cũng được xây dựng bảng chỉ mục cục bộ. Cũng do tài nguyên bị giới hạn nên việc đánh chỉ mục tại đây cũng không quá tinh vi có thể sử dụng phương pháp băm bình thường. Bảng băm này được bảo trì bằng cách khi có dữ liệu mới cần chèn vô chỉ mục thì sẽ kiểm tra con trỏ trong danh sách hiện thời trỏ đến dữ liệu, sử kiện có tần xuất truy nhập, thời gian lưu trữ có vượt ngưỡng đề ra hay không? Từ đó quyết định thay thế con trỏ cũ đó hay lập con trỏ mới móc nối đến dữ liệu mới.

Cơ chế đánh chỉ mục tại *proxy* được xây dựng dựa trên *Interval Tree* và *Skip Graph*. Việc sử dụng và bảo trì các con trỏ *skip* dựa trên xác suất và tần suất giúp đảm bảo tăng tốc tìm kiếm trong quá trình hoạt động. Thông tin tóm tắt của một SC có thể xuất hiện ở nhiều *proxy* đảm bảo cho BS có khả năng chọn lựa và chịu lỗi. Việc xây dựng *metadata* trên có sở lưu các đặc trưng của dữ liệu hình và đa phân giải theo luật xa gần đảm bảo hệ thống luôn có phản hồi thỏa đáng đến yêu cầu ứng dụng.

KẾT LUẬN

Những bàn luận trên đây đã chỉ ra sự ưu việt của hệ thống SCN so với các hệ thống giám sát an ninh thế hệ cũ trong những ứng dụng đòi hỏ tính chịu lỗi và thích nghi cao. Thiết kế trong chương 2 là thiết kế chuẩn chung cho các SC và các thực thể trong các mạng *multimedia* phân tán khác. Các vấn đề trong chương 3, 4, 5, 6 là những vấn đề cơ bản nhất của các hệ thống phân tán sử dụng truyền thông không dây kiểu *ad-hoc*. Chương 7, 8 đề cập đến các mô hình ứng dụng, lưu trữ và tra cứu trong hệ thống SCN.

Khả năng ứng dụng SCN trong công tác giám sát an ninh đã và đang được triển khai trên thế giới. Trong giai đoạn này, đó là các ứng dụng trong quân sự và quốc phòng, hy vọng rằng trong tương lai gần sẽ có những hệ thống SCN thương mại phục vụ cho dân sự.

Việc phát triển các hệ thống giám sát an ninh cho các công trình xây dựng dân dụng và các khu đô thị việc nghiên cứu SCN là cần thiết và khả thi trong xu hướng phát triển chung của các mạng nội bộ và mạng khu vực. Trong điều kiện hiện nay, hướng ứng dụng các sản phẩm thương mại có sẵn và tự phát triển các ứng dụng, kịch bản giám sát là phù hợp với nhân lực và trình độ kỹ thuật chung ở Việt nam. Để có thể hoàn toàn làm chủ được công nghệ và chủ động triển khai hệ thống cần có sự phối hợp của các chuyên gia từ nhiều nhóm ngành khác nhau và tham khảo thêm các hệ thống tương tự hiện đã có trên thế giới.

Kiến nghị về những nghiên cứu tiếp theo

Tìm kiếm các phương pháp đánh chỉ mục dữ liệu, tổng hợp thông tin *metadata* khác không dựa trên kỹ thuật *interval*.

Xây dựng SCN mới với truyền thông *hybird* nhằm tận dụng những công nghệ truyền thông vô tuyến mới sẵn có hoặc đang thử nghiệm ở Việt nam.

TÀI LIỆU THAM KHẢO

Tiếng Anh

[TCP_04] Adam Dunkels, Juan Alonso, Thiemo Voigt (2004), *Making TCP/IP Viable for Wireless Sensor Networks*,

[SPI_01] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, J.D. Tygar (2001), *SPINS: Security Protocols for Sensor Networks*, Mobile Computing and Networking 2001 Rome, Italy.

[DOS_02] Anthony D.Wood, John A.Stankovic (2002), *Denial of Service in wireless Sensor Networks*, University of Virginia.

[SS_03] Arun Hampapur, Lisa Brown, Jonathan Connell, Sharat Pankanti, Andrew Senior, Yingli Tian (2003), *Smart Surveillance: Applications, Technologies and Implications* , IEEE Pacific-Rim Conference On Multimedia. Singapore.

[AHN_00] Charles E. Perkins (2000), *Ad Hoc Networking with AODV*, Nokia Research Center - powerpoint slide.

[AODV_97] Charles E. Perkins, Elizabeth M. Royer (), *Ad-hoc On-demand Distance Vector Routing* , MILCOM '97 panel on Ad Hoc Networks, Nov. 1997.

[ZERO_01] Erik Guttman (05 - 2002), *Autoconfiguration for IP Networking: Enabling Local Communication*, IEEE Internet Computing 1089-7801/01.

[SMD_05] Huan Li, Prashant Shenoy, Krithi Ramamritham (03-2005), *Scheduling Messages with Deadlines in Multi-hop Real-time Sensor Networks*, **IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2005)**

[SG_03] James Aspnes, Gauri Shah (2003), *Skip Graphs*, In Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms, Jan. 2003.

[ZRP_02] Jan Schaumann (12 - 2002), *Analysis of the Zone Routing Protocol*,

[AFA_00] Jeremy Elson, Deborah Estrin (2000), *An Address-Free Architecture for Dynamic Sensor Networks*, Tech. Rep. 00-724, Computer Science Department USC, January 2000.

[TS_01] Jeremy Elson, Deborah Estrin (2001), *Time Synchronization for Wireless Sensor Networks* , UCLA CS Technical Report 200028

[RTS_02] Jeremy Elson, Kay Romer (2002), *Wireless Sensor Networks: A new Regime for Time Synchronization* , Proceeding of the First Workshop on Hot Topics in Networks - Princeton, New Jersey, USA. http://www.acm.org/sigcomm/HotNets-I

[RBS_02] Jeremy Elson, Lewis Girod, Deborah Estrin (2002), *Fine-Grained Network Time Synchronization using Reference Broadcasrt*, UCLA Computer Science Technical Report 020008.

[BEWS_01] John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govidan, Deborah Estrin, Deepak Ganesan (2001), *Building Efficient Wireless Sensor Networks with Low-Level Naming*, as part of the SCAADS project, SCOWR project, due to support from Cisco Systems.

[EM_04] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, D. Estrin (2004), *EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks*, in the Published as CENS Technical Report #34

[CG_06] Laura Savidge, Huang Lee, Hamid Aghajan, Andrea Goldsmith (03-2006), *Event-Driven Geographic Routing for Wireless Image Sensor Networks*, In Proc. of Cognitive Systems and Interactive Sensors - Paris.

[SAN_99] Lidong Zhou, Zygmunt J. Hass (1999), *Secure Ad Hoc Networks*, IEEE network, special issus on network security, November/ December 1999.

[DSC_06] Markus Quaritsch, Markus Kreuzthaler, Bernhard Rinner, Bernhard Strobl (2006), *Decentralized Object Tracking in a Network of Embedded Smart Cameras*, Workshop on Distributed Smart Cameras, ACM SenSys 2006. http://www.iti.tugraz.at/dsc06

[DESC_06] Michael Bramberger, Andreas Doblander, Arnold Maier, Bernhard Rinner, Helmut Schwabach (2006), *Distributed Embedded Smart Cameras for Surveillance Applications*, Published by the IEEE Computer Society 0018-9162/06 p68-75.

[DDTA_05] Michael Bramberger, B. Rinner, H. Schwabach (06-2005), *Distributed Dynamic Task Allocation in Clusters of Embedded Smart Cameras*, Proc. Int'l Conf. Systems, Man and Cybernetics, IEEE Press. 3005.

[TSAR_05] Peter Desnoyers, Deepak Ganesan, Prashant Shenoy (năm), *TSAR: A Two Tier Sensor Storage Architecture Using Interval Skip Graphs*, National Science Foundation grants EEC-0313747, CNS-0325868 and EIA-0098060.

[GCS_04] Qun Li, Daniela Rus (2004), *Global Clock Synchronization in Sensor Networks*, IEEE Infocom 2004

[OGS_03] Richard Karp, Jeremy Elson, Deborah Estrin, Scott Shenker (2003), *Optimal and Global Time Synchronization in Sensornets*, CENS Technical Report #12 , April 2003

[DCNL_04] William E. Mantzel (11-2004), *Distributed Camera Network Localization*, Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on page(s): 1381- 1386 Vol.2

[SL_90] William Pugh (06-1990), *Skip lists: a probabilistic alternative to balanced trees*, Communications of the ACM, June 1990, 33(6) 668-676.

Tiếng Việt

[PCW_06] Công nghệ máy tính và mạng (10-2006), *DaVinci đi cùng MontaVista*, Thế giới vi tính PCWORLD p85-86.

PHỤ LỤC

1. QUAN HỆ VỊ TRÍ KHÔNG GIAN, THỊ TRƯỜNG QUAN SÁT CỦA CÁC CAMERA VÀ THUẬT TOÁN DALT

Các định nghĩa và định lý dưới đây [DCNL_04] áp dụng cho camera thông thường và cũng được sử dụng khi tính toán quan hệ vị trí và thị trường quan sát của các SC trong SCN.

Định nghĩa 1

Hai camera f_i và f_j là có liên kết *linked* (biểu diễn $f_1 \leftrightarrow f_2$) nếu khung nhìn của chúng có có cùng tập hợp 6 hoặc nhiều hơn điểm đặc trưng.

Định nghĩa 2

Một mạng camera trải rộng là mạng có ít nhất một cặp camera không có liên kết.

Định nghĩa 3

Một liên kết các camera tạo thành bộ ba Δ khi các cặp camera trong chúng được liên kết với nhau đôi một.

Định lý 4

Khi các camera tạo thành liên kết bộ ba Δ thì khi biết vị trí và góc hướng của một camera (ví dụ f_1) và khoảng cách từ đó đến các camera khác (ví dụ f_2, f_3) được biết thì có thể tính được vị trí và góc hướng đến cả ba f_1, f_2, f_3.

Định nghĩa 5 hai camera f_i, f_j là *triple-wise connected* nếu tồn tại các liên kết tam giác $\Delta_1, \Delta_2, ..., \Delta_n$ dạng như Δ_r và Δ_{r+1} có hai camera chung và $f_i \in \Delta_1$ và $f_j \in \Delta_n$.

Định nghĩa 6

Một *microcluster* là tập các camera M với đặc điểm thuộc tính là khi $f_i \in M$ thì $f_j \in M$ khi và chỉ khi f_i là *triple-wise connected* với f_j.

Định lý 7

Mọi camera nằm trong một *microcluster* có thể có quan hệ vị trí toàn cục trong một hệ quy chiếu với thang độ toàn cục và với ít nhất bảy bậc tự do.

Theo (ĐN 2) thì SCN được coi là một mạng camera trải rộng. Theo định lý 4, tác vụ *pan* khung nhìn của camera có thể tạo thành liên kết bộ ba camera mới hoặc phá hủy liên kết bộ ba cũ. Định nghĩa 5 gợi ý hướng truyền tải nội dung xử lý theo vết đối tượng trong nhiệm vụ giám sát. Định lý 7 giúp xây dựng lưới vị trí camera cho các nhiệm vụ giám sát.

Các kết quả trên hữu dụng khi khởi tạo hệ thống hoặc tái cấu trúc SC trong *s_clu* khi đang vận hành. Thuật toán DALT được đề xuất cho việc xây dựng và tìm kiếm các camera có quan hệ vị trí và thị trường quan sát.

Thuật toán DALT *(distributed alternating localization triangulation)*
1. **if** đã biết thiết vị **then**
2. *quảng bá thông tin thiết vị này đến các camera có liên kết*
3. **else**
4. *đợi đến khi có thông tin thiết vị được quảng bá đến từ 2 hoặc hơn camera có liên kết*
5. *sử dụng những thông tin thiết vị của những lân cận có quan hệ bộ ba với 6 hoặc hơn điểm đặc trưng*
6. *khởi tạo, tính toán thiết bị của mình từ những thông tin trên*
7. **end if**
8. **repeat**
9. *tự quảng bả thông tin ước đoán thiết vị của mình*
10 *sử dụng mọi thiết vị của quan hệ bộ ba cũng như các điểm đặc trưng có thể*
11 *với góc hướng cố định, ước đoán chuyển dịch tối ưu*
12 *với chuyển dịch cố định, ước đoán góc hướng tối ưu*
13 **until** ước đoán được hết thiết vị các camera

2. ĐỊNH TUYẾN EIRGP CỦA CISCO VÀ GIẢI THUẬT DUAL

EIRGP là giao thức định tuyến độc quyền của Cisco trong mạng truyền thống và NGN.

Trong EIGRP có các bảng *Neighbor*, bảng *Topology* và bảng *Routing*. Bảng *neighbor* để lưu giữ các lân cận, bảng *topology* chứa thông tin về tất cả các *route* do các lân cận gửi đến và bảng *routing* thì chỉ chứa tuyến đường tốt nhất. EIGRP chỉ gửi cập nhật mỗi khi có sự thay đổi và chỉ gửi thay đổi này đến những *router* cần thông tin đó.

Sau khi các *router* đã nhận đầy đủ thông tin về các *router* khác lân cận xong thì nó sẽ sử dụng giải thuật *Diffusing Update Algorithm* để xác định *successor* và *feasible successor*. *Successor* là *next hop router* mà từ *local router* đến mạng đích thông qua *router* này có *metric* là thấp nhất (gọi là *feasible distance*). Còn *feasible successor* là *next hop router* mà có *metric* cao hơn ở trên (tất nhiên, nếu không nó đã là *successor*) nhưng có khoảng cách từ nó đến mạng đích (gọi là *advertised distance* hoặc *reported distance*) là thấp hơn *feasible distance* của *successor* ở trên.

Successor được đặt vào *routing table*, còn *feasiable successor* thì vẫn để ở trong *topology table*. Khi một *successor* là *invalid* thì một *feasible successor* thích hợp nhất sẽ được chọn và được làm *successor* (mà *router* không cần tính lại bảng *topology*).

Giải thuật DUAL

Đây là thuật toán cập nhật lan truyền (*diffusing*) và có tốc độ hội tụ nhanh. Nó là lan truyền ở chỗ mỗi một *router* sẽ chọn tuyến đường tốt nhất đến một mạng đích bằng cách chọn một *successor*, rồi *successor* này lại chọn tuyến đường tốt nhất đến đích dựa vào *successor* của nó. Có nghĩa là mỗi một *router* chỉ cần tìm *router neighbor* gần nhất, rồi *router neighbor* gần nhất đó lại tìm tiếp một *neighbor* khác gần nhất để đến mạng đích. Khi một tuyến đường qua *successor* mà *invalid* mà không có *feasible successor* để thay thì các *router* cũng *query* theo kiểu lan truyền như vậy để tìm ra *successor* mới. Vì có khái niệm *feasible successor* nên các router *không* phải tính toán lại bảng *topology* để tìm ra tuyến đường mới sau khi *invalid*. Còn cơ chế chống *loop* là do trong quá trình lan truyền để tìm tuyến đường, DUAL sẽ đảm bảo không có sự lặp lại của mỗi một *router*.

TÓM TẮT LUẬN VĂN

Ngày nay, việc phát triển các hệ thống mạng camera phục vụ giám sát an ninh khu vực trên nền công nghệ IP đang là chuẩn hóa khi thiết kế các giải pháp phụ trợ cho các công trình xây dựng dân dụng. Tuy nhiên kiến trúc các hệ thống hiện tại là hướng cấu trúc, thiếu tự chủ, tính chịu lỗi và thích nghi chưa cao do đặc điểm hệ thống là cần có trung tâm điều khiển tập trung. Trong một vài ứng dụng cụ thể thì việc xây dựng thuật toán hoạt động trên nền kiến trúc này gặp nhiều khó khăn khi có sự biến động về số lượng camera và nhiệm vụ giám sát trong hệ thống. Luận văn này tập trung nghiên cứu về hướng xây dựng hệ thống mạng mới trên cơ sở các camera thông minh phân tán thực sự và phân tải phân tán nhiệm vụ giám sát để tạo thành hạ tầng phát triển các thuật toán trên. Hệ thống này được gọi là MẠNG CAMERA THÔNG MINH - Smart Camera Network (SCN). Trên cơ sở phân tích các vấn đề cơ bản là vấn đề đánh địa chỉ camera, đồng bộ bộ đếm phân tán, định tuyến và an ninh truyền thông, luận văn đề xuất và đưa ra giải pháp giải quyết hai bài toán cơ bản .

3. Bài toán CB1: Đánh giá tác động và cơ chế điều chỉnh phân tán nhiệm vụ giám sát cho mỗi SC trong SCN.
4. Bài toán CB2: Tìm kiếm thông tin, dữ liệu hình đã lưu trữ trong SCN.

Những bài toán này là cơ sở cho việc phát triển các ứng dụng gia tăng và nghiên cứu về sau trong SCN.

TỪ KHÓA

mạng camera thông minh, camera phân tán, phân tán nhiệm vụ giám sát, tác tử di động, lịch truyền thông điệp

ABSTRACT

Nowadays, the development of camera system for the sake of security supervision basing on IP technology is becoming standardized for selection on backing solution of civil construction. However, architecture of recent system is structure driven, fault tolerance and less adaptiveness because of the short of central controlling system. In some specific application, the building of algorithm based on this architectural ground faces many difficulties when there is a change in the number of cameras as well as surveillance tasks. This thesis (dissertation) focuses on carrying out study on building new network system based on real distributed smart cameras and distributing surveillance tasks to create an developed infrastructure for above algorithm. This system is named Smart Camera Network (SCN). On the ground of basis analysis of naming camera, time synchronization, routing, semantic security, the paper puts forth and gives out suggestions to solve the two research problems.

5. Problem 1: evaluation of impact and the adjustment scatter mechanism of surveillance task for each SC in SCN.
6. Problem 2: data and information gathering archived in SCN.

The mentioned problems lay out the foundation for the development of increasing application and following research on SCN.

KEYWORD

smart camera network, distributed camera, distributed surveillance task, mobile agent, scheduling messages