액티브 패킷 기술을 이용한 상황인지 네트워킹

숭실대학교 정수환 · 최재덕

1. 서 론

인터넷의 다양한 기술의 발전과 이용자의 급격한 증가로 네트워크에 대한 요구 사항이 증가하고 복잡해지고 있다. 또한 우리의 일상생활 속에서 언제, 어디서나 인터넷에 접속할 수 있는 유비쿼터스 네트워크 환경이 자리를 잡고 있다. 이와 같이 복잡하고 다양한 요구 사항이 발생할 수 있는 네트워크에서는 기존의 패킷 기술 보다 더 능동적이고 지능적인 패킷 기술이 필요하다.

액티브 패킷 기술은 스스로 상황인지를 통하여 유연한 네트워크 서비스를 제공할 수 있는 기술이다. 액티브 패킷 기술들은 패킷에 특정 기능을 수행할 수 있는 실행 모듈을 적재하여 모듈이 실행 될 수 있는 플랫폼을 갖춘 목적지 노드에서 모듈을 동작시켜 특정 기능을 수행한다. 이러한 액티브 패킷 기술을 통하여 능동적인 네트워크 관리, 자원 관리, 상황인지형 서비스 등을 수행할 수 있다.

본 고에서는 인터넷의 급속한 발전으로 방대해진 네트워크를 보다 능동적으로 제어할 수 있는 액티브 패킷 기술들에 대해서 알아본다. 액티브 패킷 기술로는 **액티브 네트워크 (Active Networks), 상황인식 네트워크 (Cognitive Networks), 모바일 코드 (Mobile Code), 모바일 에이전트 (Mobile Agent), 스마트 패킷 (Smart Packet)**이 있으며, 각각 기술에 대한 설명과 이해를 돕기 위해 적용 분야에 대해서 살펴본다.

2. 액티브 패킷 기술

이번 장에서는 현재 기존의 수동적인 패킷 기술의 한계 보강을 위해 제시된 다양한 액티브 패킷 기술들 Active networks, Cognitive networks, Mobile code, Mobil agent, Smart packet에 대해서 살펴본다.

† 본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 정보통신부의 유비쿼터스컴퓨팅 및 네트워크원천기반기술개발사업의 지원에 의한 것임.

2.1 Active Networks

액티브 네트워크는 1990년 중반에 미래의 네트워크 시스템 진화 방향 제안을 목적으로 DARPA에서 폭넓게 연구가 진행되었다. 액티브 네트워크는 기존 네트워크 노드가 단순히 패킷을 저장한 후 포워딩 (store-and-forward)하는 방식과는 달리 계산 능력이 추가된 라우터에서 수행 코드를 적재한 패킷을 연산, 처리, 전송할 수 있게 하는 (store-compute- forward) 방식으로 단순 패킷 포워딩 기능만 담당했던 라우터에 지능화된 요소를 추가하여 패킷을 좀 더 융통성 있게 처리 할 수 있는 패킷 기술이다[1][2]. 그림 1은 IP 네트워크에서 액티브 라우터와 기존의 라우터의 패킷 처리 방식을 보여준다. 기존의 라우터는 패킷의 전달을 위하여 패킷 헤더만을 수정하여 전송하는 반면, 액티브 라우터는 패킷을 이용하여 사용자가 특별히 요구하는 응용 서비스 및 프로세스를 노드에서 연산을 통해 수행한다. 액티브 라우터의 계산 능력은 기존의 라우터에 적재되어 있지 않은 새로운 프로토콜과 기능을 동적으로 적재하여 유연하게 동작하게 한다.

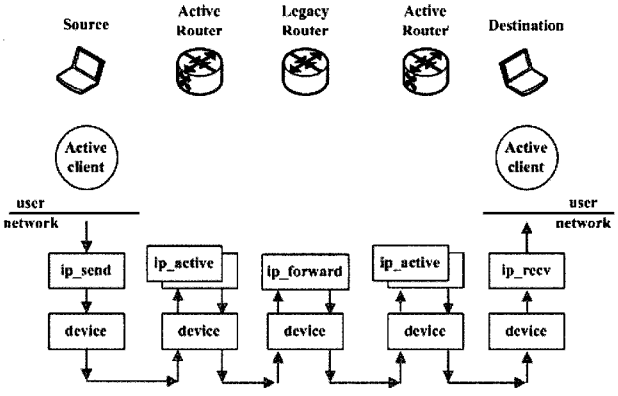


그림 1 액티브 네트워크에서 패킷 처리 방식

액티브 네트워크의 기본구조는 노드 운영체제, 수행환경, 액티브 응용 등의 세 가지 구조로 구분된다. 노드 운영체제는 액티브 노드의 자원과 자원들의 요청을 관리

한다. 수행 환경은 패킷으로 직접 전달되는 프로그램 코드에 의해 프로그램 되거나 제어되는 프로그래밍 인터페이스 또는 가상기계를 의미한다. 액티브 응용은 수행환경에 의해 공급된 프로그래밍 인터페이스를 사용하여 종단 사용자 응용을 위한 서비스를 구현한다. 이러한 기본 구조를 가진 액티브 네트워크는 액티브 네트워크 패킷에 포함된 프로그램을 통해 새로운 서비스들을 신속하게 액티브 노드로 전송하여 필요한 기능을 제공하게 할 수 있고, 액티브 노드에서는 액티브 네트워크 패킷에 포함된 프로그램을 실행하게 된다. 액티브 네트워크는 실행의 결과로 노드의 상태를 변경하고 패킷 정보를 변경하여 새로운 정보를 담은 패킷을 생성한 후 다음 노드에 전송하는 등 액티브 노드에 대해 프로그래밍이 가능하다.

액티브 네트워크에서 사용되는 액티브 패킷을 처리하기 위해 프로그램을 노드에 적재하는 방법은 프로그램 자체가 어떻게 패킷에 포함되어 운반 되는지에 따라서 크게 두 가지로 나뉜다. 첫째는 필요한 자원들과 프로그램 코드가 미리 액티브 노드에 상주하는 분리방식(discrete approach)으로 액티브 패킷을 처리하기 위해 필요로 하는 모든 프로그램 코드, 자원을 액티브 노드에 적재하여 두는 방식이다. 둘째는 프로그램 코드를 패킷에 포함시켜 전달하여 액티브 노드에서 실행되도록 하는 통합 방식(integrated approach)으로 프로그램 코드를 미리 액티브 노드에 적재하지 않고 패킷에 적재하여 전송하고 액티브 노드에서 수행하게 한다. 이 방식은 노드의 부담을 줄여 주고, 분리방식보다 유연하게 사용자가 필요로 하는 서비스 요청을 수행할 수 있지만, 전송하는 패킷에 적재되는 프로그램 코드, 자원이 커지게 되어 네트워크 트래픽의 증가, 패킷 손실 이외에 여러 가지 문제가 생길 수 있다. 이러한 두 가지 방법을 좀 더 효과적이고 안정적으로 사용하기 위해 혼합형 액티브 네트워크가 제안되었다. 혼합형 액티브 네트워크는경량화와 패킷 처리의 고속화 기능을 효과적으로 수행하기 위해 특성에 따라 다른 타입의 패킷을 정의하여 노드에서 수행되는 패킷을 고속으로 처리 할 수 있게 하고, 노드에 필요한 프로그램 코드를 동적으로 할당하여 노드를 경량화 할 수 있게 하였다.

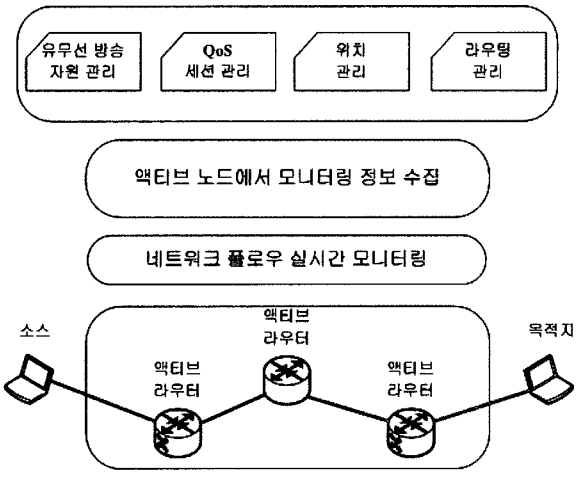


그림 2 능동형 자원 관리

액티브 네트워크는 네트워크의 중간 노드에서 필요한 처리를 할 수 있으므로 능동적으로 패킷을 처리할 수 있다. 이러한 기능은 그림 2와 같이 다양한 망 환경에서 실시간 모니터링을 통해 개별망간의 QoS 보장을 위한 위치, 라우팅, 과금 등 자원 관리 기능과 외부 공격을 실시간으로 탐지하고 대처하는 능동 지능형 보안 서비스를 제공할 수 있다[3].

2.2 Cognitive Networks

Cognitive Packet Network(CPN)은 전통적인 패킷 스위칭 방식에서 벗어나 네트워크의 지능적인 패킷 기술을 이용하여 상황인지 기능을 구체화한 액티브 패킷 기술이다. CPN은 기존의 IP 네트워크의 QoS 라우팅, 다양한 환경에서 라우팅 테이블 재구성 문제와 같은 여러 가지 문제점 등을 해결하고 있다. Cognitive packet은 기존의 고정된 라우팅 테이블을 사용하지 않고 네트워크 내에서 패킷 스스로가 학습을 통해서 최적의 라우팅을 구성한다. Cognitive packet은 각 기능에 따라 Cognitive Packet, Dumb Packet, ACK Packet 타입으로 구분된다. Cognitive packet은 네트워크 상황 정보를 수집과 업데이트 하는 기능을 갖고, ACK packet은 수집된 정보를 적재하여 소스 노드로 돌아오면서 각 노드의 Mail Box (MB)를 업데이트 하고, 마지막으로 dumb packet은 일반 데이터를 전송하는 패킷이다[4][5].

그림 3은 CPN에서 최적의 경로를 찾아 데이터를 전송하는 모습을 나타낸다. CPN에서 Cognitive packet은 소스에서 목적지로 데이터를 보내게 될 때, 기존의 라우팅 테이블을 이용하지 않고일반적인 소스와 목적지의 정보를 포함하고 있는 Cognitive Map (CM)을 참조하여 네트워크의 상황 정보에 맞는 최적의 경로를 찾기 위해 목적지로 보내지는 패킷이다. Cognitive packet이 목적지로 보내지면 네트워크 내의 노드들을 거치면서 노드들의 CM을 참조하고 자신이 노드에 도착한 시간과 지연시간, QoS 파라미터 값, 노드의 전력 정보 등을 수집하면서 네트워크를 이동한다. 이와 같이 정보를 수집하는 cognitive packet이 목적지에 도착하게 되면 자신이 수집해온 정보를 목적지 노드의 CM에 업데이트하고 그와 관련된 정보를 담아 ACK 패킷에 적재한다. ACK 패킷의 역할은 cognitive packet에 의해 수집된 정보를 가지고 목적지에서 소스로 cognitive

packet이 이동해 온 경로를 따라 이동하면서 각 노드들의 CM을 새롭게 업데이트 한다. 이렇게 네트워크의 정보를 적재한 ACK 패킷이 소스에 도착하면 소스 CM의 MB를 업데이트 하고 이때 업데이트 된 정보들을 바탕으로 하여 데이터를 보낼 최소 홉과, 최적의 QoS를 가진 경로를 결정하여 dumb 패킷에 데이터를 적재하여 정한 루트를 따라 목적지로 전송하게 된다.

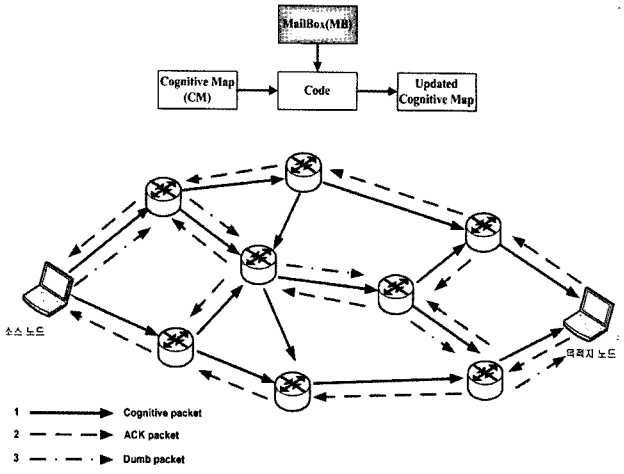


그림 3 CPN에서 최적의 경로 탐색 및 데이터 전송

Cognitive packet은 네트워크 경로에 대해 QoS를 보장하고 패킷 손실 등과 같은 문제를 해결하기 위한 트래픽 엔지니어링을 위해 사용된다. 네트워크의 혼잡 상황은 네트워크 리소스의 비효과적인 사용, 네트워크 내의 부적절한 트래픽 분산 등의 이유로 발생하며 종단 사용자는 네트워크에서 큐잉 지연, 패킷 손실을 통해 감지할 것이다. 이런 상황에서 트래픽 엔지니어링을 통해 네트워크의 성능을 증가시키고 네트워크에서 이용 가능한 리소스를 최대한 활용 할 수 있다.

CPN은 패킷 플로우에 QoS 파라미터 값인 지연, 손실, 지터, 대역을 기본적으로 제공하는 QoS 기반의 라우팅 프로토콜을 지원한다. 이러한 특성을 이용하여 CPN은 다량의 이상 트래픽을 이용한 DoS 공격에 대해서 효과적으로 대응할 수 있다. 웹서버를 통해서 클라이언트가 멀티미디어 서비스를 받고 있는 상황에서 멀티미디어 데이터는 cognitive packet과 ACK packet에 의해서 결정된 최적의 QoS를 지원하는 경로를 통해 전송된다. 여기서 공격자가 웹서버에게 다량의 패킷 전송을 통해 DoS 공격을 할 경우, 각 노드에서는 MB에 있는 QoS 할당 정보와 패킷 정보의 비교를 통해 이상 트래픽이라는 것을 감지하고 패킷을 버린다. 또한 이상 패킷이 감지 된 후에는 ACK 패킷을 통해 이러한 상황을 업스트림 상에 있는 노드들에게 알려 패킷을 폐기할 수 있도록 한다[6].

2.3 Mobile Code

모바일 코드는 일종의 소프트웨어로 우리가 알고 있는 다운로드 된 코드와 액티브 컨텐츠, 작은 규모의 소프트웨어 그리고 자동적으로 사용자의 워크스테이션에 다운로드 되는 프로그램들로 정의된다. 이것은 원격 소스에서 네트워크를 통해 로컬 시스템으로 보내지고 로컬 시스템에서 수행된다. 모바일 코드는 수행을 위한 사용자의 인스톨 과정이나 특별한 동작을 필요로 하지 않는 Active-X controls, 자바 애플릿, 브라우저 안의 스크립트, HTML 그리고 이메일 등으로 볼 수 있다. 이런 모바일 코드가 개인 컴퓨터, 소형 이동 단말 (PDA, 핸드폰, 기타 인터넷이 가능한 소형기기)에 보내지고 프로그램 코드들은 서버로부터 다운로드 되어 사용자의 초기화 없이도 자동으로 수행된다.

모바일 코드의 일반적인 구조를 보면 컴포넌트와 사이트로 구분할 수 있다. 컴포넌트에는 리소스 컴포넌트와 computational 컴포넌트로 구분된다. 리소스 컴포넌트는 데이터, 디바이스, 코드 등이 정의되고, computational 컴포넌트는 수행 상태, 데이터 그리고 코드로 정의된다.사이트는 코드가 수행될 수 있게 지원하고, 서버와 사용자의 워크스테이션간의 상호 작용을 유연하게 할 수 있도록 지원한다.

모바일 코드를 동작 형태에 따라 분류하면 그림 4와 같이 Client-Server, Remote Evaluation, Code on Demand, Mobile agent로 구분된다. Client-Server 모델은 클라이언트에 리소스가 있고 리소스를 사용하기 위한 코드와 다른 데이터가 소스에 있을 때 클라이언트에서 소스로 요청을 하면 소스에서 요청에 맞는 리소스를 이용할 수 있는 코드와 데이터를 전송하는 방식이다. Remote Evaluation은 리소스와 코드를 가지고 있는

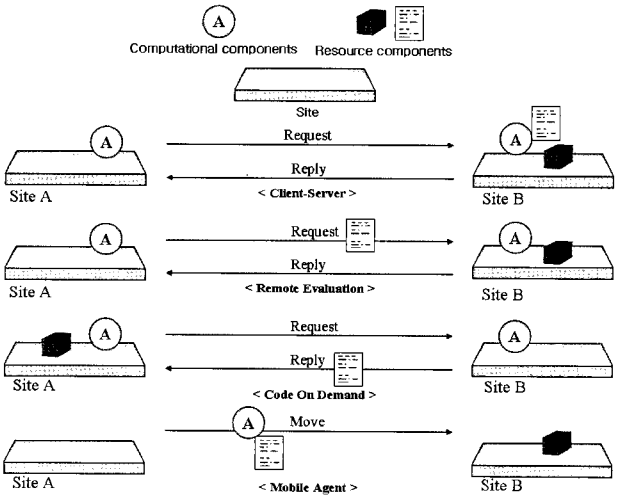


그림 4 모바일 코드 동작 모델

노드에서 또 다른 리소스와 데이터가 있는 노드로 코드를 전송하여 또 다른 리소스를 가진 노드에서 수행된 후 그 결과를 다시 응답해주는 방식이다. Code on Demand 방식은 요청하는 노드에 데이터와 리소스가 있고 서버격의 노드에는 코드와 리소스가 있을 때 요청 노드에서 리소스와 데이터를 사용하기 위해 서버격의 노드로 코드를 요청하는 방식이다. 즉 이 데이터와 리소스를 사용할 수 있는 설명서를 받는 것과 유사하다. 마지막으로 Mobile agent 방식은 리소스와 수행 코드가 함께 데이터가 있는 노드로 이동하여 수행되는 방식이다.

모바일 코드는 간단하면서도 다양한 응용에 적용할 수 있는 가능성을 많이 가지고 있기 때문에 유비쿼터스 네트워크와 새로운 응용에 상당 부분 적용되고 있다.

2.4 Mobile Agent

모바일 에이전트는 에이전트가 프로그램의 리소스가 있는 컴퓨터로 이동해 리소스를 검색하여 프로그램을 구동할 수 있는 패킷 기술로 현재 네트워크의 부하와 트래픽을 줄이고, 네트워크의 시간 지연에 따른 대기 시간을 줄이는 것에 대한 연구가 진행 중이다. 이 때 모바일 에이전트의 전송은 단순한 데이터나 프로그램의 이동만을 의미 하는 것이 아니며 프로세스 및 상태, 수행 데이터를 동시에 이동한다. 이런 모바일 에이전트는 자율성(Autonomy), 적응과 학습성(Adaptive and Learning), 코드 이동성(Mobility), 통신 및 협력성(Communicative and Collaborative)의 특징을 가지고 있다[8][9][10]. 모바일 에이전트의 특징을 자세히 살펴보면 첫째, 모바일 에이전트는 인공지능 분야와 분산처리 분야가 서로 유기적으로 영향을 미치며 발전된 기술로 모바일 에이전트 스스로 프로세스를 수행할 수 있는 기능을 가지고 있는 자율성이 있다. 둘째, 모바일 에이전트는 인공지능 분야에서의 학습, 추론 그리고 계획하는 기능을 통해 노드의 실행 환경에 적응하고, 네트워크의 상황을 학습한 결과를 이용하여 어떻게 수행할 것인지에 대한 계획을 미리 세우는 적응성과 학습성을 갖는다. 셋째, 이동 코드의 단순한 이동이 아닌 실행 코드, 메모리 상태 그리고 수행 후 얻은 데이터 등을 수반하여 이동하는 코드의 이동성을 지원한다. 이런 특징의 모바일 에이전트는 첫째, 네트워크의 부하와 트래픽을 줄여준다. 둘째, 네트워크의 시간 지연에 따른 대기 시간을 줄여준다. 셋째, 프로토콜을 캡슐화 하여 이동시키는 것이 가능하다. 넷째, 자율적이고 비동기적으로 동작 가능하다. 다섯째, 네트워크나 노드의 상황에 적응 가능하다. 여섯째, 이 기종들 간에서의 동작을 원활하게 할 수 있다.

모바일 에이전트의 가장 기본적인 구조는 그림 5와 같이 에이전트와 에이전트의 실행 환경인 플레이스(Place)로 구분된다. 에이전트는 state, implementation, interface, identifier 그리고 principals 등의 속성을 가진 객체로 네트워크를 통해 이동한다. 이동한 에이전트는 호스트에 미리 준비된 에이전트의 실행 환경인 플레이스에서 실행, 생성 그리고 소멸된다. 이런 플레이스는 engine (Virtual machine), resources, location, principals 등의 속성을 갖는다.

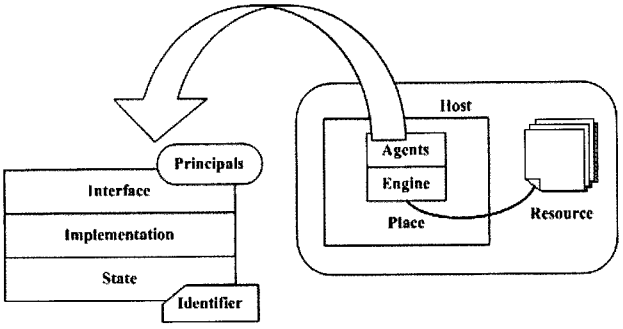


그림 5 모바일 에이젼트 구성 요소 (에이전트, 플레이스)

모바일 에이전트는 플레이스에서 생성되어 플레이스에서 소멸된다. 생성은 플레이스에서 인증을 받은 플레이스내의 다른 에이전트나 다른 시스템에 의해서 생성될 수 있고, 자신의 존속 시간이 끝나던가, 더 이상 사용되지 않을 때, 인증을 받지 못했을 때 등의 이유로 소멸 된다. 모바일 에이전트의 전송은 에이전트 자신이나 다른 에이전트, 플레이스 외부의에이전트나 시스템에 의해서 전송되며, 플레이스에서 플레이스로 전송된다.

기존의 네트워크 메니지먼트는 근본적으로 클라이언트 서버 메커니즘의 기능들을 사용하기 때문에, 네트워크 시스템들은 통신양의 증가에 따른 확장성에 문제가 있다. 그러나 모바일 에이전트 기술은 네트워크 메니지먼트에 대해서 다양한 변화에 유연하게 적응 할 수 있다. 모바일 에이전트는 멀티플 서브넷에서 네트워크 노드들 사이를 자동적인 이동 능력으로 노드를 방문하고 메니지먼트 태스크를 수행한다[11]. 메니지먼트 태스크를 수행하기 위해 모바일 에이전트는 정적으로 디자인되지 않아 여러 다른 이질적인 네트워크에서도 유연하게 동작 할 수 있고, 두 가지 타입의 모바일 에이전트로 구분된다. 먼저, 네비게이터 에이전트(Navigator agent)는 에이전트가 관할하는 서브넷의 토폴로지를 잘 알고 있어 태스크 에이전트 (Task agent)가 서브넷의 여러 목적지 노드에 효과적으로 이동할 수 있도록 한다. 태스크 에이전트는 각 노드로 이동하여 네트워크 메니지먼트에 관한 기능을 수행한다.

그림 6은 태스크 에이전트가 네비게이터 에이전트의 안내를 통해 여러 서브넷에서 네트워크 메니지먼트 기능

을 수행하는 것을 보여준다. 각 네비게이터 에이전트는 서브넷의 정의된 토폴로지를 가지고 있고 이 정보를 이용하여 태스크 에이전트를 관리한다. 에이전트 풀(Ageng Pool)은 네트워크에 적당한 네비게이터 에이전트를 태스크 에이전트가 선택할 수 있도록 해준다. 태스크 에이전트가 에이전트 풀에 도착하면 태스크 에이전트

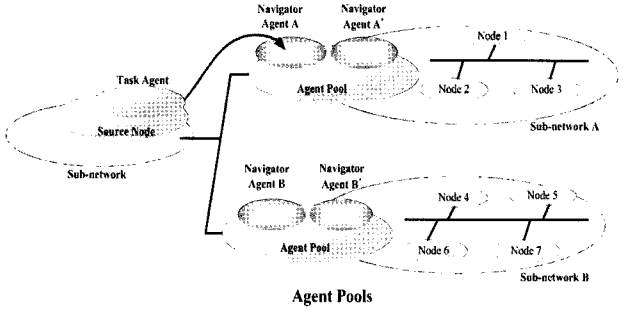


그림 6 모바일 에이젼트의 네트워크 메니지먼트

는 방문하기 원하는 노드들의 타입들이나 이름에 관련된 정보를 풀에 넘겨준다. 넘겨진 정보는 풀에서 적당히 선택된 네비게이터 에이전트에 의해 태스크 에이전트는 네비게이터 에이전트의 이동 스케줄과 서브넷에 의해 인증된 후 목적지 노드에 도착한다. 수행 코드와 함께 노드로 이동한 태스크 에이전트는 네트워크 트래픽 모니터링 또는 리소스를 측정, 노드의 CPU 사용 정보, 메모리 상태 등을 모니터링 한다. 이런 정보는 다시 수집되고 저장되어서 메니즈먼트 서버로 보내지고 다른 노드로 이동하는 에이전트를 통해 또 다른 노드에서 사용되기도 한다. 서브넷에서의 리소스 정보를 모니터링하여 서브넷 네트워크에서의 RTT를 줄이거나, 네트워크 트래픽을 분산하는데 사용될 수 도 있다.

2.5 Smart Packet

스마트 패킷은 유비쿼터스 네트워크의 상황인식 서비스를 지원하기 위한 패킷으로 self-executable, self-configurable, self-transferable 기능이 있다. 각각의 특성은 먼저, 스마트 패킷에 상황인식 서비스를 위한 바이너리 코드를 적재할 수 있으며, 이 코드는 목적지에서 자동으로 수행된다. 둘째, 스마트 패킷은 자동 설정 기능이 가능하다. 스마트 패킷에 적재된 수행 코드에 의해서 여러 가지 필요한 정보들이 수집되며, 이 정보들은 트래픽을 조절하고 라우팅 경로를 새롭게 구성하는 등의 특정 기능 수행을 위해 사용된다. 마지막으로 스마트 패킷은 마스터 노드라는 특정 노드에서 생성되고 각 노드들로 전파된다. 노드에서 스마트 패킷은 인터넷 웜과 같이 복제, 증식되어 이웃 노드로 전파된다. 스마트 패킷의 가장 큰 특징은 실행 코드를 실행할 수 있는 최소한의 플랫폼을 통해 실행코드를 유비쿼터스 네트워크의 노드들에게 전송하여 서비스 수행에 필요한 모듈 없이도 언제, 어디에서나 요청 서비스를 수행할 수 있게 지원한다. 유비쿼터스 네트워크에서는 언제, 어디서나 인터넷에 접속 가능하기 때문에, 다양한 환경에서 수많은 응용 서비스들이 제공될 수 있다. 그러나 이러한 응용 서비스들을 위해서 사용자 단말기에 서비스 수행에 필요한 모든 모듈들을 적재할 수 없기 때문에 스마트 패킷을 통해서 상황을 인지하여 필요한 모듈을 적절한 시기에 제공하는 것이다[12].

스마트 패킷은 그림 7과 같이 Smart Packet Header(SPH), Context-Aware Request(CAR), Executable Code(EC) 등의 세 부분으로 나뉜다. SPH는 스마트 패킷의 기본 정보를 포함하고, CAR은 실행 모듈을 수행하기 위한 부가 정보, 그리고 EC는 응용 서비스를 수행하기 위한 실행코드이다. 스마트 패킷 플랫폼은 패킷 인증과 기밀성을 제공하고 실행 코드를 수행하는 최소한의 플랫폼을 갖는다.

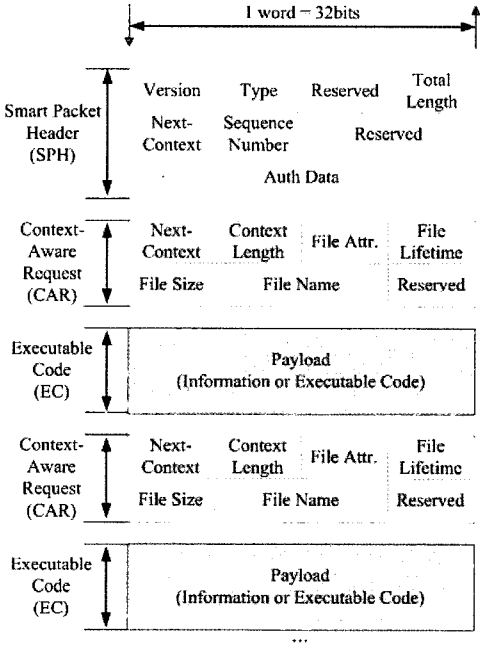


그림 7 스마트 패킷 형태

유비쿼터스 네트워크 안에서 동작하고 있는 노드들은 스마트 패킷을 받아 인증을 하고 실행할 수 있는 최소의 기능을 갖춘 에이전트를 갖고 있다. 또 스마트 패킷이 만들어지는 노드를 마스터 노드로 정의하고, 이 노드에서 스마트 패킷이 생성되고 수행코드가 적재된다. 생성된 스마트 패킷은 주변의 노드들로 전송되고 스마트 패킷을 받은 노드에서 다시 복제, 증식되어 이웃 노드로 전파 된다.

스마트 패킷은 화재, 해일 등의 위험 상황이 발생했을 때 사람들에게 신속하게 경고를 하고 대응 방안을 제공

하는 응용 서비스에 사용된다. 사람들은 위험 상황이 발생하면 그들이 위치한 곳 (지하철 역, 해변가, 컨퍼런스 회의장 등)에서 각기 다른 행동을 필요로 하기 때문에 스마트 패킷을 적용하는 것은 효과적이다. 스마트 패킷에 적재된 실행 모듈은 각각 장소에서 적절한 대피 경로를 단말기를 통해 사용자들에게 알려주어 위험 지역의 사람들이 각자 필요한 행동을 할 수 있도록 돕는다. 그림 8과 같이 각 단말들이 지하철역에 진입할 때마다 역 마스터 노드는 비상 상황이 발생 했을 경우의 대피 요령 및 안전한 장소로의 이동 경로를 포함한 모듈을 각 단말들에게 전송한다. 전송된 실행 모듈은 단말의 위치를 업데이트하여 언제든지 사용자를 안전한 곳으로 대피할 수 있도록 유도할 수 있다. 화재와 같은 비상 상황이 발생할 경우 마스터 노드는 위험 지역에 있는 사람들에게 EMERGENCY-NOTIFY 메시지를 브로드캐스트 하고 메시지를 받은 각 노드들은 사용자로 하여금 현재의 상황에 맞는 적절한 행동을 할 수 있게 안내한다.

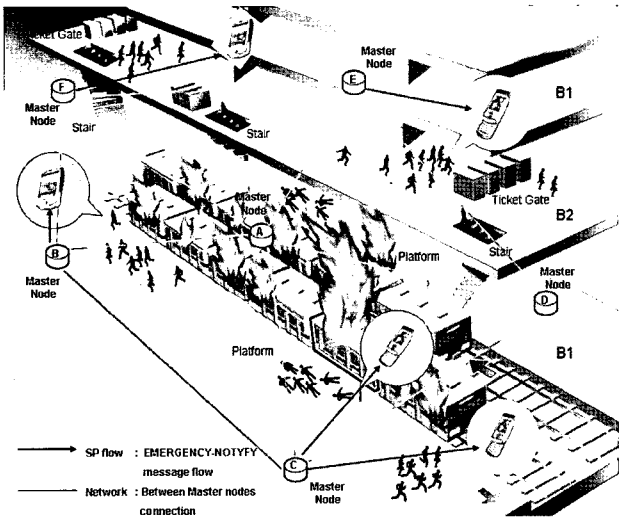


그림 8 스마트 패킷을 이용한 화재 알람 서비스

3. 결 론

지금까지 인터넷의 급격한 확산으로 네트워크에 대한 다양한 요구 기능과 유비쿼터스의 다양한 서비스 환경을 지원하기 위한 액티브 패킷 기술들에 대해서 살펴보았다. 액티브 네트워크는 기존 네트워크 노드가 단순히 패킷을 저장한 후 포워딩(store-and-forward)하는 방식과는 달리 패킷을 통하여 전송하여 실행(store-compute-forward)하는 방식으로 단순 패킷 포워딩 기능만 담당했던 라우터에 지능화된 요소를 추가하여 패킷을 좀 더 융통성 있게 처리 할 수 있는 패킷 기술이다. Cognitive Packet Network(CPN)은 전통적인 패킷 스위칭 방식에서 벗어나 네트워크의 지능적인 패킷 기술을 이용하여 상황인지 기능을 구체화한 액티브 패킷 기술이다. 모바일 코드는 일종의 소프트웨어로 우리가 알고 있는 다운로드 된 코드와 액티브 콘텐츠, 작은 규모의 소프트웨어 그리고 자동적으로 사용자의 워크스테이션에 다운로든 되는 프로그램들로 정의 된다. 모바일 에이전트는 에이전트가 프로그램의 리소스가 있는 컴퓨터로 이동해 리소스를 검색하여 프로그램을 구동할 수 있는 패킷 기술이다. 스마트 패킷은 유비쿼터스 네트워크의 상황인식 서비스를 지원하기 위한 실행 코드를 실행할 수 있는 최소한의 플랫폼을 통해 유비쿼터스 네트워크의 다양한 상황인지 서비스를 수행할 수 있도록 지원한다.

지금까지 살펴본 액티브 패킷 기술들은 인터넷의 급속한 발전과 유비쿼터스 환경의 다양한 환경을 충족시키기 위해 더욱더 지능적인 패킷 기술로 발전할 것이다.

참고문헌

[1] D. Tennenhouse, J.smith, W.Sincoskie, D. Wether all, and G. Minden, "A Survey of Active Network Research," IEEE Communications Magazine, pp. 80-86, January 1997.

[2] D. Wetherall, U. Legedza, and J. Guttag, "Introducing new internet services: why and how," IEEE Network Magazine, July/August 1998.

[3] 오행석, 남택용, "액티브 네트워크를 활용한 능동 지능형 서비스", 전자통신동향분석 제19권 제6호, 2004.

[4] E. Gelenbe, Z. Xu, and E. Seref, "Cognitive packet networks," Proceedings of the IEEE, pp. 47-54, 1999.

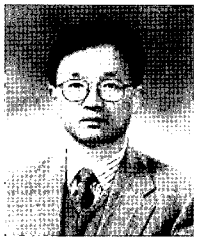
[5] E. Gelenbe, and R. Lent, "Power aware ad-hocCognitive Packet Networks," Ad Hoc Networks Journal, 2 (3), pp. 205-216, 2004.

[6] Erol Gelenbe, Michael Gellman, and George Loukas, "An Autonomic Approach to Denial of Service Defence," IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2005.

[7] Bieszczad A., and Pagurek B., "Network Management Application-Oriented Taxonomy of Mobile Code," Proc. of the EEE/IFIP Network Operations and Management Symposium(NOMS '98), New Orleans, Louisiana,

Feb. 15-20, 1998.
[8] A. Bieszczad, B. Pagurek, and T. White, "Mobile Agents for Network Managrment," IEEE Communications Surveys, Vol. 1, No. 1, 1998.
[9] M.Dalmeijer, E.Rietjens, M.Soede, D.K. Hammer, and A.T.M. Aerts, "A Reliable Mobile Agents Architecture," Proceeding of the 1st IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'98), pp. 64-72. 1998.
[10] Ichiro Satoh, "Selection of Mobile Agents." Proceeding of the 24th International Conference on Distributed Computing Systems (ICDCS'04), 2004.
[11] Ichiro Satoh, "Building Reusable Mobile Agents for Network Management." IEEE Transactions on Systems, Man, and Cybernetics, Part C 33(3): 350-357, 2003.
[12] Jaeduck Choi, Hyosun Roh, Souhwan Jung, Younghan Kim, "Support of Context-awareness in Ubiquitous Networks using Smart Packet." UbiCNS2005, 2005.

정 수 환



1985. 2 서울대학교 전자공학과(학사)
1987. 2 서울대학교 전자공학과(석사)
1998~1991 한국통신 전임연구원
1996. 6 미 워싱턴 주립대(시애틀) 박사
1996~1997 Stellar One SW Engineer
1997~현재 숭실대학교 정보통신전자공학부 부교수
관심분야 : 이동인터넷 보안, 네트워크 보안, VoIP 보안, RFID/USN 보안
E-mail : souhwanj@ssu.ac.kr

최 재 덕

2002. 2 숭실대학교 정보통신전자공학부 (학사)
2004. 2 숭실대학교 정보통신공학과(석사)
2004. 12 애드팍테크놀러지 연구원
2005~현재 숭실대학교 정보통신공학과 (박사과정)
관심분야 : 사용자 인증, 이동인터넷 보안, SIP 보안
E-mail : cjduck@cns.ssu.ac.kr