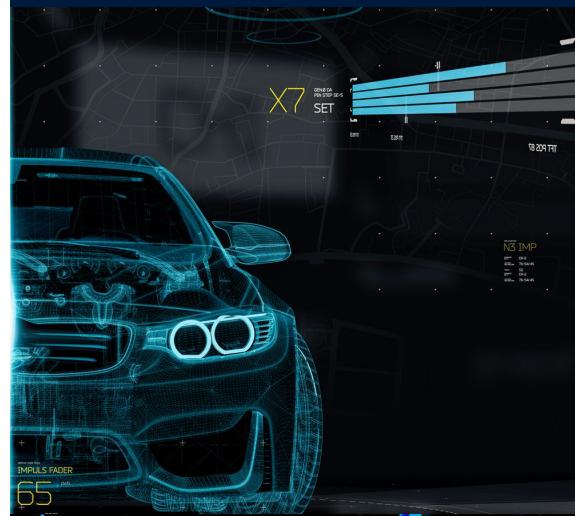
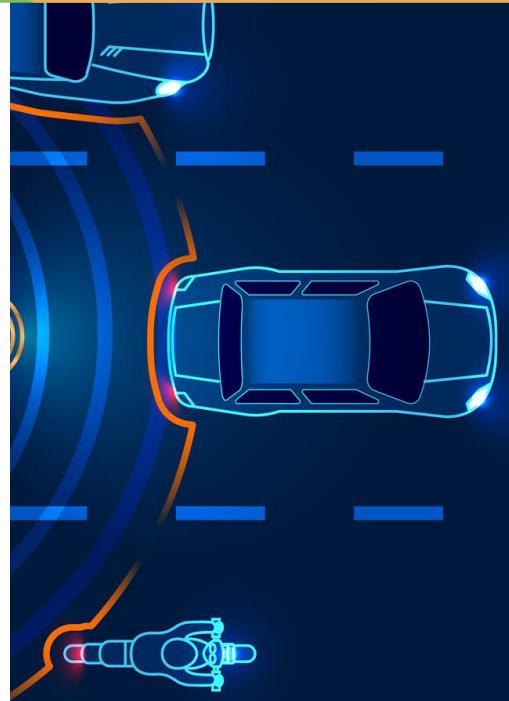




SYNAPSE

by CUBE Intelligence

자율자동차를 위한
Decentralized Security Platform



자율자동차를 위한 Decentralized Security Platform

자율자동차와 지금까지의 자동차가 다른 점은 자율주행차는 네트워크에 연결되어 있다는 점입니다. 이는 마치 인터넷에 연결 안된 컴퓨터에 인터넷이 연결되는 상황이라고 할 수 있습니다. 컴퓨터가 인터넷에 연결되는 순간부터 해킹의 위험에 시달리듯이, 자율자동차는 해킹의 위험에 빠지게 됩니다.

특히 자율자동차는 거의 대부분 통신에 의존합니다. 자율자동차는 통신에 의한 경로와 목적지를 다운받지 않으면 움직일 수가 없다. 또한 신호등, 도로변 IoT 등과 연결하는 V2X, 그리고 주변의 다른 자율자동차와의 통신 (V2V) 등 통신이 필수적입니다. 이렇게 통신에 의존하는 자율자동차에 해킹이 발생하면 탑승자들은 치명적이 됩니다. 운행에 대한 해킹과 달리, 자동차에 대한 해킹은 생명과 직결되게 되기 때문입니다. 따라서 통신에 의존하는 자율자동차는 해킹에 대하여 완전무결한 준비를 해야 합니다.

기존의 시큐리티 보안으로는 DDoS 공격 등에 대하여 철저하게 방어할 수 없었습니다. Cube는 블록체인의 핵심요소인 머클트리의 방식을 사용하여 자율자동차 네트워크 보안에 기여를 합니다.

큐브 블록체인 시큐리티 layer의 1차버전은 블록체인의 핵심개념인 Merkle Tree (Hash Tree) 기반과 Secret Sharing에 의한 key refreshment로 효율적이고 무결한 전송으로 요약할 수 있습니다.

큐브의 첫번째 프로젝트는 서버에서 endpoint까지 도달하는 자동차의 업데이트, 그리고 동시에 자동차의 운행정보를 서버까지 도달하는데 생길 수 있는 security 문제를 서버의 decentralization으로 비잔틴 공격자로부터 악의적인 공격을 차단하는 것입니다.

SYNAPSE의 구조

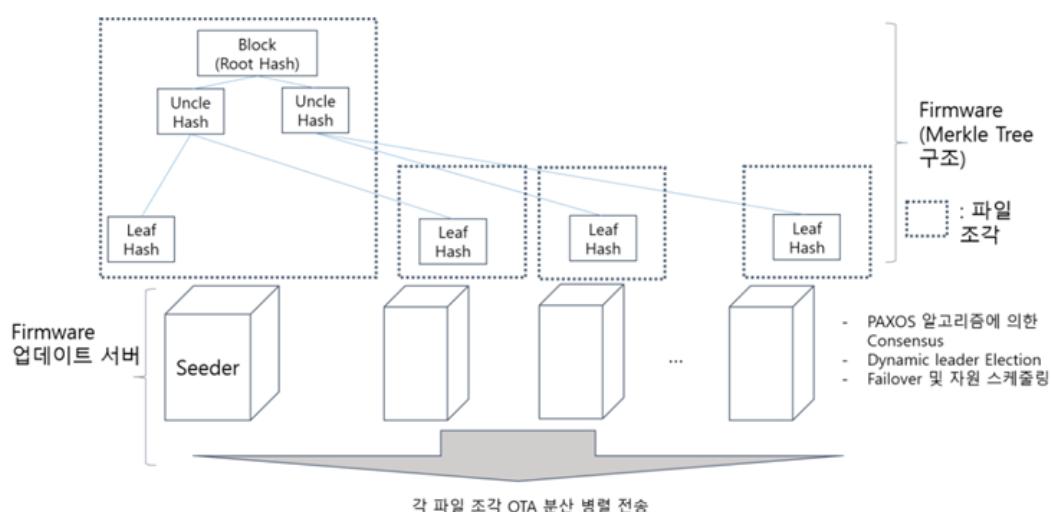


그림1) Firmware를 Merkle Tree 구조로 변환



—이더리움의 창시자Vitalik Buterin과 University of Illinois at Urbana-Champaign의Ravi Kiran Raman and Lav R. Varshney교수는 블록체인은 Merkle Tree 형태로 데이터를 보관하므로 암호학적으로 안전하다고 제안하고 있습니다.

- Seeder는 Uncle Layer의 hash list, root hash와 담당 leaf hash 전송 - 나머지 참여 서버들은 그 외 leaf hash 담당하여 전송
- OTA로 encrypt된 파일 조각 전송 - 각 자율주행 차량 파일 조각 수령 후 파일 재구성 (파일 조각 수령 시, hash list와 root hash 이용하여 무결성 증명에서 검증)
- 병렬 서버간 Private 블록체인 망 형성
- 고가용성을 위하여, PAXOS Consensus, Dynamic Leader Election과 Active Active Failover 기능 구현 디자인 취지, 전제조건 등
- 자율주행 차량을 Peer로 삼아 Peer-assisted firmware 업데이트하는 것은 보안과 신뢰의 보장 차원에서 과한 복잡도가 발생하며, vehicle-to-vehicle 통신의 안정성, 효율성, 성능 등 선결 조건이 충족되기 전에는 효과를 보장할 수도 없습니다. 따라서, 자율자동차 네트워크, 운영자, vendor 들이 직접 고 신뢰 클라우드 기반 서버군을 두고 블록체인의 핵심인 Merkle Tree방식으로 firmware 업데이트하는 것이 현재로써 가장 이상적인 상용화 방안입니다.

OTA로 서버에서 자동차에 데이터를 보내어 자동차를 업데이트하는 것과 반대로 큐브박스를 통해 OBD 및 개인 정보 등의 판매 및 분석을 위한 클라우드 기반의 빅데이터 플랫폼이 필수인 상황에서, 인프라를 일부 확장을 통한 firmware update 서버 구동은 같은 방식으로 구현을 하였습니다.

자동차에서 OBD 및 개인정보 수집 분석 프레임워크

(1) 개요: 통상적 Big Data 처리 및 분석용 플랫폼 아키텍처를 따르되, 수집의 안정성/안정성 측면의 기술과 관리 솔루션 신규 디자인

(2) SYNAPSEBig Data 플랫폼 Reference 아키텍처

Ravi Kiran Raman and Lav R. Varshney, , Blockchain systems establish a cryptographically secure data structure for storing data in the form of a hash chain, Distributed Storage Meets Secret Sharing on the Blockchain

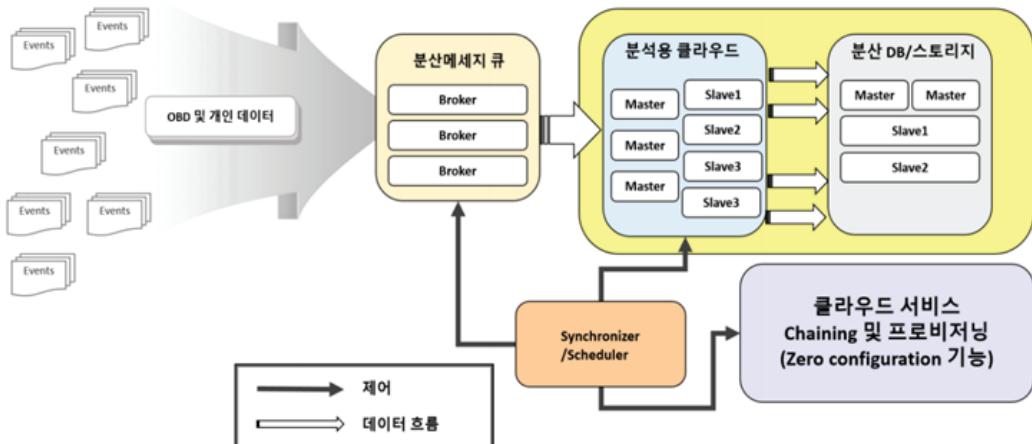


그림2) 큐브 빅데이터에 대한 구조

그림2) SYNAPSE 빅데이터에 대한 구조

- 클라우드 기반의 통상적인 구조 Figure2에 도식화
- 자율주행 차량과 병렬 서버 간 다중 채널 형성 후 Merkle Tree 구조의 개인 데이터 분할 전송
- Seed가 되는 Merkle Tree의 root hash, uncle layer hash list는 secret sharing 기술을 이용하여 추가 분할 전송하여 안전성 보장.

SYNAPSE 알고리즘은 기본적으로 블록체인의 핵심기술인 Merkle Tree를 바탕으로 하고 있습니다. 이더리움의 창시자 Vitalik Buterin 등이 제시하였듯이 Merkle Tree는 블록체인의 기본 구조이며 가능하도록 하는 핵심적인 부분입니다. .

비잔틴 중계인과 악의적 공격방지

SYNAPSE의 security 전송방법은 비잔틴 중개인이 있을 때조차도 신뢰할 수 있고 시크릿하게 전달을 보장하는 방법입니다. 우리 솔루션의 핵심은 해독 키를 안전하게 전달하기위한 secret sharing을 가능하도록 하는 것입니다. 비잔틴 중계인이란, 서로 짜고 메시지를 탈취 및 위변조를 시도하는 악의적 중계 노드를 의미합니다. Cube의 시큐리티 기술은 이러한 악의적인 비잔틴 중계인이 있더라도 불구하고, 무결하게, 효율적으로 자동차회사 또는 IoT로부터 자동차로 데이터를 전송하는 기술이라고 할 수 있습니다. 이 기술은 반대로 자동차에서 다른 IoT 쪽으로 데이터를 전송할 때도 적용됩니다.

특히 SYNAPSE의 security는 비잔틴 중계인이 있는 상황에서도 다음 요구 사항의 만족을 보장합니다. 첫째, 전송하는 데이터가 손실없이 자동차에게 전달되도록 합니다 (신뢰성 요구 사항). 둘째, Subscribers 또는 비잔틴 브로커가 액세스 권한없이 기밀 메시지에 액세스 할 수 없도록 합니다 (기밀 요구 사항).

Merkle trees are a fundamental part of what makes blockchains tick., Vitalik Buterin, Merkling in Ethereum, November 15, 2015



지금까지의 자동차의 보안은 네트워크의 보안에 대하여 매우 취약한 상태이며, end point에서만 일반적으로 복제 및 암호화 기술을 사용합니다. 이미 나와있는 네트워크 암호화 기술을 사용하고 하더라도 일반적인 복제 및 암호화 기술을 사용할 수밖에 없습니다. 그러나, 이러한 작업들은 비잔틴 중개인들에 의해 해독 키가 손상되고 남용될 가능성이 있습니다. 비잔틴 중개인은 관심있는 가입자가 게시 메시지의 암호를 해독하지 못하도록 키를 삭제할 수 있습니다. 손상된 키 사용 비잔틴 중개인은 비공개 게시 메시지를 해독하고 승인되지 않은 가입자에게 공개할 수 있습니다. 비잔틴 브로커는 암호화된 메시지와 암호 해독 키를 간단히 삭제할 수 있습니다. Cube는 이러한 위험을 secret sharing 기법을 적용해서 해결하고 있습니다.

Cube는 pub/sub secret overlay에 chain이 된 복제된 브로커들의 그룹에 대하여 secret sharing을 적용하는 기법을 개발하였습니다. cube 솔루션에 대한 높은 수준의 개요를 제공하기 위해 브로커 복제 본 (broker replicas)은 먼저 게시자와 가입자 간의 종단 간 경로를 따라 배치됩니다. 게시자는 secret sharing 구성표를 사용하여 암호 해독 키를 분할합니다. 우리가 secret sharing이라고 부르는 암호화된 게시 메시지와 함께 propagation됩니다. Secret sharing이 생성되어 원래의 암호 해독 키가 비잔틴 브로커에 의해 재구성될 수 없는 방식으로 복제 브로커에 전달됩니다. 이 방법을 사용하면 기밀 게시 메시지가 유출되는 것이 안전합니다.

복제본은 해커가 데이터와 키를 삭제하거나 허가 받지 않은 제3 자에게 데이터를 보내는 것을 막는 데 사용됩니다. Cube의 방법은 pub / sub 오버레이에 더 많은 브로커를 추가함으로써 성능 오버 헤드를 증가시킬 수 있습니다. 따라서 우리는 주어진 브로커 복제본을 가장 효율적인 방법으로 활용하려는 도전에 직면해 있습니다. 이러한 문제를 해결하기 위해 Pub / Sub 오버레이 관리자가 유연하게 정의할 수 있는 안정성 및 성능 기준을 기반으로 브로커 복제본을 동적으로, 전략적으로 할당하기 위한 프레임워크를 만들고 있습니다.

Pub/sub 아키텍처에서 브로커라는 중앙 소스가 모든 데이터를 수신하고 배포합니다. Pub/sub 클라이언트는 브로커에 데이터를 게시하거나 브로커에서 데이터를 가져 오기 위해 구독할 수 있습니다. 데이터를 게시하는 클라이언트는 데이터가 변경될 때만 데이터를 보냅니다 (Report by Exception 또는 RBE). 데이터를 구독하는 클라이언트는 브로커에서 자동으로 메시지를 수신하지만 메시지가 변경될 때만 다시 수신합니다. 브로커는 데이터를 저장하지 않습니다. 단순히 데이터를 게시자에서 구독자로 이동합니다. 데이터가 게시자로부터 들어오면 중개인은 해당 데이터를 구독한 모든 클라이언트에게 즉시 데이터를 보냅니다.

Request-response 아키텍처에서는 클라이언트가 서버에서 직접 데이터를 요청하기 때문에 각 클라이언트가 각 서버에 직접 연결을 엽니다. 클라이언트는 데이터가 언제 바뀔지 모르기 때문에 일정한 간격으로 데이터를 요청합니다. 서버가 클라이언트 요구에 응답할 수 있고 네트워크가 트래픽 볼륨을 처리할 수 있는 한 Request-response는 입증되고 신뢰할 수 있는 통신 방법입니다. 그러나 다수의 클라이언트와 서버가 있는 경우 Request-response 아키텍처의 트래픽 양은 빠르게 문제가 될 수 있습니다.

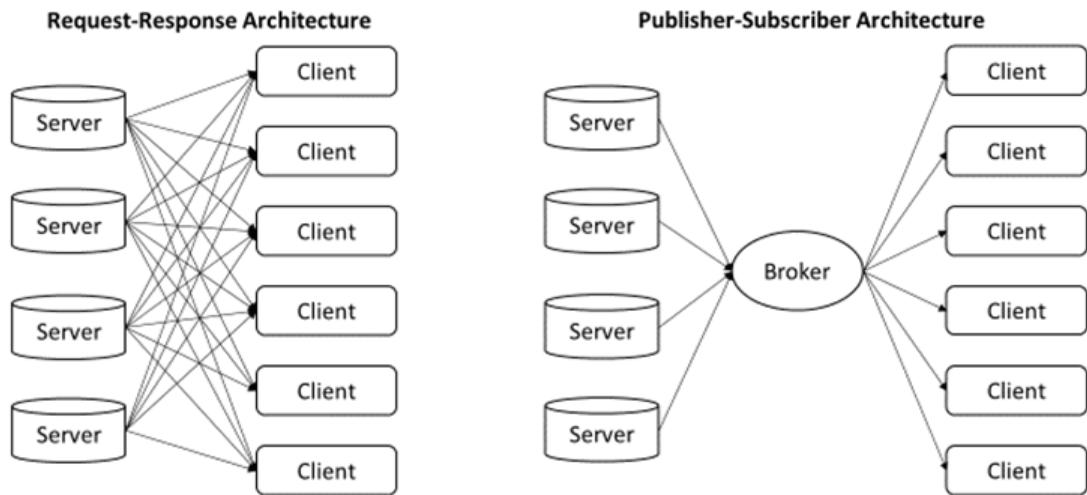


그림3) 직접 요청과 응답 구조 & Pub/Sub 구조

대조적으로, pub/sub 아키텍처는 통신을 단순화합니다. 데이터에 대한 직접 연결 및 반복적인 요청은 필요하지 않습니다. 연결망은 각 장치에서 브로커까지의 단일 링크로 대체됩니다. 클라이언트와 브로커 사이의 연결은 열려 있고 매우 가볍습니다. 이 연결을 통해 이동하는 것은 두 가지 뿐입니다. 변경된 데이터와 클라이언트가 아직 거기에 있음을 브로커가 알 수 있도록 하는 신호입니다.

따라서 다수의 서버와 클라이언트간 데이터 및 서비스를 공유해야 하는 모빌리티 데이터 거래에는 pub/sub 모델을 사용하는 것이 좋습니다. 브로커는 데이터의 중앙 처리 기관이기 때문에 개별 서버가 여러 클라이언트를 처리할 필요가 없으며 클라이언트가 여러 서버에 연결할 필요가 없습니다. 또한 데이터는 정기적인 간격이 아닌 데이터가 변경될 때만 RBE 기반으로 게시되고 전송되므로 네트워크 트래픽이 전반적으로 감소합니다.

이러한 이유로 Pub/sub은 소셜 네트워킹, 분산된 비즈니스 프로세스 및 실시간 업무 핵심 시스템과 같은 많은 응용 프로그램 영역에서 널리 보급됩니다. 많은 Pub/sub응용 프로그램은 메시지 손실 및 개인 정보 침해에 민감합니다. 이러한 문제를 극복하기 위해 Cube는 비밀 공유 및 복제 기술을 사용하는 새로운 방법을 개발하였습니다. 이는 Pub-/sub오버레이 네트워크에 있는 여러 비잔틴 중개인이 있는 경우에도 암호화된 게시와 함께 해독 키를 안정적이고 기밀로 제공하는 것입니다. 또한 안정성과 성능을 위해 유연하게 정의할 수 있는 기준에 따라 브로커 복제본을 동적 및 전략적으로 할당하기위한 프레임 워크를 개발하였습니다.

다음은 2018년과 2016년 IEEE학회지에 저희 연구팀에서 제출한 논문으로, 이 이론을 바탕으로 CUBE X는 상용화한 것입니다.



요약

Publish / Subscribe (short pub / sub)는 느슨하게 결합된 클라이언트가 비동기 방식으로 통신하는 통신 패러다임입니다. Subscribers는 특정 주제 및 / 또는 콘텐츠에 대한 관심을 나타내기 위해 Subscriptions 를 발행합니다. Publishers는 자신의 신원 및 / 또는 위치를 직접 알지 못하고 pub / sub 라우팅 시스템을 통해 Subscribers에게 자신의 Publications을 보급합니다.

이러한 비동기 특성으로 인해 pub / sub 패러다임은 대규모 이벤트 구동 및 유비쿼터스 시스템의 유연한 개발을 지원합니다. Pub / sub는 분산된 비즈니스 활동 모니터링, 알고리즘 트레이딩 시스템의 주가 모니터링, 복잡한 이벤트 처리 및 항공 교통 관제 시스템과 같은 미션 크리티컬 시스템이 있습니다. 다국적 연구 그룹은 인터넷의 아키텍처를 향상시키기 위해 pub / sub 라우팅 패러다임을 채택하여 최근에는 더 많은 콘텐츠 위주의 통신 패턴을 보여줍니다. 많은 소셜 네트워킹 서비스는 pub / sub 추상화를 중심으로 구축됩니다. 우리는 실제 시스템 [요소 사이의 연결의 패턴을 분석하여 복잡한 네트워크의 연구에 Pub / 서브 시스템의 잠재적인 응용 프로그램도 구상할 수 있습니다 (12 - (16)].

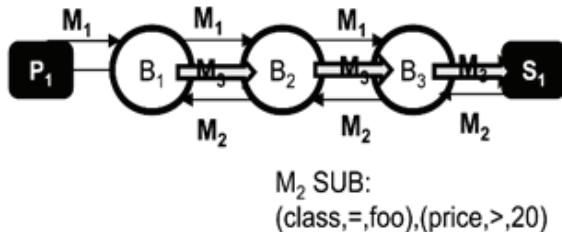
예를 들어, 다변량 신호는 분포된 전도 센서로부터 측정되어 오일 - 물 흐름 패턴을 분석하고, 커뮤니티 구조에 따라 시각화 됩니다 [17]. 이 센서는 pub / sub 시스템에 배치하여 관심있는 패턴을 필터링하고 보다 확장 가능하고 효율적인 방법으로 전달할 수 있습니다.

Pub/sub 시스템은 일반적으로 분산 이벤트 매칭 전달 브로커의 오버레이로 형성되는 8 - 11 , 18 , 19 스케일 러블 방식으로 이벤트 대용량을 처리하기 위해]. pub / sub broker 오버레이 [20 , 21]의 레퍼런스 구현에서 발행자는 먼저 이벤트를 게시하기 전에 모든 브로커에게 광고를 보급합니다. 게시된 이벤트를 게시라고 합니다. Publications은 특정 주제로 분류될 수 있으며 메시지 또는 내용을 포함할 수 있습니다. 서브 스크립션이 근본적으로 [advertisement, last hop]의 목록 인 SRT (Subscription Routing Table)의 광고와 일치하는 경우, 튜플의 경우 Subscriptions은 최종 흡 중개인에게 전달됩니다 .

이렇게 하면 Subscribers는 게시자를 향해 라우팅 됩니다. Subscriptions은 PRT (게시 라우팅 테이블)를 구성하는 데 사용됩니다. PRT는 [Subscriptions, 마지막 흡] 튜플의 목록으로 , publication을 라우팅하는 데 사용됩니다. 게시가 PRT의 Subscriptions과 일치 하면 Subscriptions이 제공된 마지막 흡 중개인에게 전달됩니다 . 이 프로세스는 게시가 최종적으로 Subscribers에 도달할 때까지 계속됩니다.

그림 3 은 콘텐츠 기반 라우팅의 예를 보여줍니다. 스텝 1 , 광고 (M 1)에 도달 B 1 . 단계도 2는 , 매칭 정액제 (M 2)에 도달 B 3 . 이후 M 2가 일치 M 1 브로커에 B (3) , M (2) 로 중계됩니다 B 1 은 IS 마지막 흡의 M 1 . 이 단계의 종료 후, PRT는 경로 (그에 따라 갱신됩니다)에서 B 1 행 B 3 . PRT상의 라우팅 정보에 기초하여 , 가입 M에 일치 하는 publication (예를 들어, M 3)(2)는 가입자에게 전달될 수 있는 S 1 통하여 . Subscribers는 M 2 에서 (class, =, bar) 와 같은 특정 주제에 대한 관심을 지정할 수 있습니다 . Subscribers는 또한 콘텐츠를 구체적으로 표현함으로써 보다 정교한 방식으로 관심사를 표현할 수 있습니다. 예를 들어, S 1 , 속성에 대한 특정 값 범위에 걸쳐 발현 관심 가격에 도시 된 바와 같습니다.

M_1 ADV: (class,=,foo),(price,>,15)
 M_3 PUB: (class,=,foo),(price,30)



SRT of B2	
If SUB intersects	Forward to
(class,=,foo),(price,>,15)	B_1
PRT of B2	
If PUB intersects	Forward to
(class,=,foo),(price,>,20)	B_3

그림 4. pub / sub 오버레이에서 라우팅 상태 업데이트의 예

큐브의 주요 관심사는 pub / sub 브로커가 실패하거나 손상되어 신뢰할 수 있고 안전한 이벤트 전달을 방해하도록 임의로 작동하는 것입니다. 우리는 임의적으로 행동하는 pub / sub 브로커를 비잔틴 중개인이라고 부릅니다.

비잔틴 브로커는 퍼블리셔와 가입자 간의 많은 앤드 - 투 - 앤드 전달 경로에 걸쳐 존재할 수 있습니다. 비잔틴 브로커의 임의의 행위로 인해 pub / sub 오버레이에서 실행되는 응용 프로그램이 파괴되어 최종 사용자에게 매우 위험한 결과가 발생할 수 있습니다. 따라서 우리는 효과적으로 이 문제를 해결할 수 있는 새로운 솔루션을 고안해야 합니다.

특히 우리는 비잔틴 중개인이 있는 상황에서도 다음 요구 사항의 만족을 보장하고자 합니다 [22]. 첫째, 게시 메시지가 손실없이 관심있는 가입자에게 전달되는지 확인해야합니다 (신뢰성 요구 사항). 둘째, Subscribers 또는 비잔틴 브로커가 액세스 권한없이 기밀 메시지에 액세스 할 수 없도록 해야 합니다 (기밀 요구 사항).

관련 작업 섹션에서 더 자세히 설명했듯이, 위의 요구 사항을 위반하는 기준의 작업은 일반적으로 복제 및 암호화 기술을 사용합니다. 복제된 브로커는 메시지 손실의 가능성을 줄일 수 있습니다. 암호화는 게시 메시지의 개인 부분을 보호할 수 있습니다. 그러나 우리가 알고 있는 한, 이러한 작업들은 비단틴 중개인들에 의해 해독 키가 손상되고 남용될 가능성을 간과하고 있습니다.

비잔틴 중개인은 관심있는 가입자가 게시 메시지의 암호를 해독하지 못하도록 키를 삭제할 수 있습니다. 손상된 키 사용 비잔틴 중개인은 비공개 게시 메시지를 해독하고 승인되지 않은 가입자에게 공개할 수 있습니다. 비잔틴 브로커는 암호화된 메시지와 암호 해독 키를 간단히 삭제할 수 있습니다.

본 백서에서는 secret sharing 기법을 적용하는 신규 방법을 제시합니다. 우리 솔루션에 대한 높은 수준의 개요를 제공하기 위해 브로커 복제본은 먼저 게시자와 가입자 간의 종단 간 경로를 따라 배치됩니다. 게시자는 secret sharing 구성표를 사용하여 암호 해독 키를 분할합니다. 우리가 secret sharing 라고 부르는 스피트 키암호화 된 게시 메시지와 함께 propagation됩니다.

secret sharing이 생성되어 원래의 암호 해독 키가 비잔틴 브로커에 의해 재구성될수 없는 방식으로 복제 브로커에 전달됩니다. 이 방법을 사용하면 기밀 게시 메시지가 유출되는 것이 안전합니다. 복제본은 비잔틴 중개인이 publication과 키를 삭제하거나 허가 받지 않은 Subscribers에게 publication을 보내는 것을 막는 데 사용됩니다.



우리의 방법은 pub / sub 오버레이에 더 많은 브로커를 추가함으로써 성능 오버 헤드를 증가시킬 수 있습니다. 따라서 우리는 주어진 브로커 복제본을 가장 효율적인 방법으로 활용하려는 도전에 직면해 있습니다. 이러한 문제를 해결하기 위해 Pub / Sub 오버레이 관리자가 유연하게 정의할 수 있는 안정성 및 성능 기준을 기반으로 브로커 복제본을 동적으로, 전략적으로 할당하기 위한 프레임 워크를 제안합니다.

우리는 소셜 네트워크 서비스에서 발생한 상호 작용을 게시하는 서비스에 적용하여 솔루션의 효율성을 평가했습니다. 특히, 우리는 익명의 페이스 북 사용자들 사이의 상호 작용의 실제적인 흔적을 찾아내고 우리의 새로운 솔루션을 갖춘 pub / sub 미들웨어에서 추적을 재연했습니다.

백서의 나머지 부분은 다음과 같이 구성됩니다. 첫째, 우리는 비밀 전달 방법의 세부 사항을 제시하고 다양한 적용을 논의합니다. 두 번째로, 우리는 신뢰성과 성능에 대한 동적으로 변화하는 요구에 따라 복제본을 할당하기 위한 프레임 워크를 설명합니다. 셋째, 성능 평가 결과를 분석합니다.

은밀한 전달 방법

이 섹션에서는 비잔틴 중개인이 있을 때 조차도 신뢰할 수 있고 비밀스러운 전달을 보장하는 방법을 소개합니다. 우리 솔루션의 핵심은 해독 키를 안전하게 전달하기 위한 secret sharing 계획을 시행하는 것입니다.

Pub / Sub Overlay에서의 secret sharing

우리의 주요 관심사 중 하나는 비잔틴 중개인이 해독된 해독 키를 사용하여 암호화된 게시 메시지를 임의로 처리할 수 있는 것입니다. 따라서 열쇠는 비잔틴 중개인으로부터 보호되어야 합니다. 여기서 우리는 Shamir의 secret sharing 방식을 사용합니다 [23]. 이 기법을 사용하면 비밀을 n 개의 공유로 분할하여 원본 비밀을 재구성하는 데 최소한 k 개의 공유가 필요합니다. 즉, 최대 $k-1$ 개의 공유가 손상되더라도 원래의 비밀은 다시 생성될 수 없습니다. 이것은 $k > 1$ 이고 $n \geq k$ 인 제약 조건을 갖는 (k, n) 임계 스킴이라고 불립니다.

우리의 맥락에서, 비밀은 Subscribers가 발행자에 의해 암호화된 발행 메시지를 해독하는 데 필요한 핵심입니다. 이 섹션에서는 비잔틴 중개인이 게시자와 Subscribers 간의 중단 간 경로를 따라 거주하는 경우를 중점적으로 다룹니다. 다음 즉시 흡에서 F 수의 비잔틴 중개인이 있다고 가정하고 게시자가 원래의 해독 키를 분할해야 하는 방법을 보여줍니다. 그런 다음 분할 secret sharing가 관심있는 가입자에게 propagation되어야 하는 방법을 설명합니다.

초기 키 분할

발행자 가정 P_1 , 게시 송출 P_1 , 브로커를 통해 $B_{(1)}$ 에 나타난 바와 같이, 도 2 . 합니다고 가정 B_1 비잔틴 브로커와 드랍 P_1 . 메시지 손실을 방지하기 위해 복제본을 통해 중복 경로를 설정할 수 있습니다 B'_1 . 이중 경로를 통해 중복된 서적을 보내 적어도 하나의 메시지가 등록자를 향해 전달되도록 보장할 수 있습니다. 복제본을 사용하면 pub / sub 오버레이가 안정적인 전달 실패에 대해 관대하게 됩니다. 일반화로서 오버레이의 각 흡에서 f 개의 실패가 있는 경우 적어도 $f+1$ 흡의 복제본이 필요합니다. 이러한 복제본은 가상 노드라고 하는 그룹을 형성합니다.

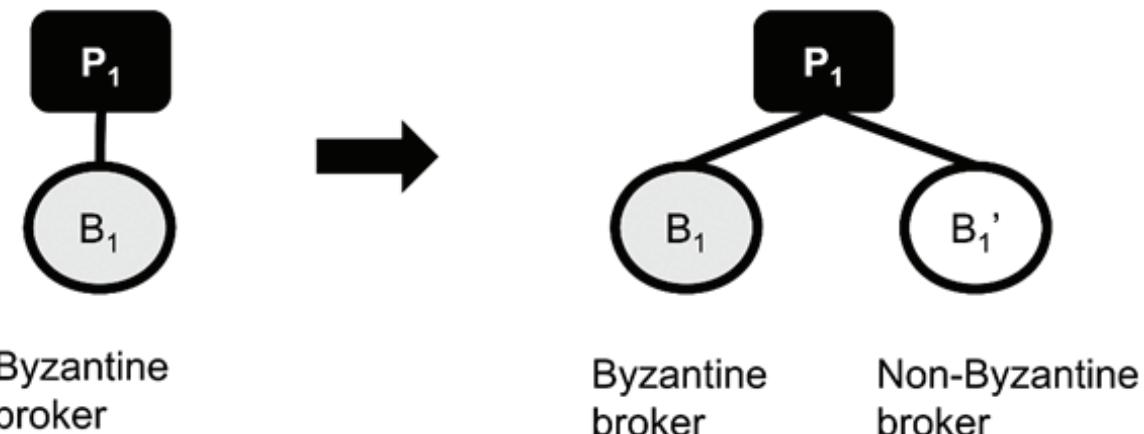


그림 5. 비잔틴 브로커가 신뢰할 수 있는 게시 전달 요구 사항을 위반하는 경우를 처리하기 위한 간단한 브로커 복제 예제.

암호 해독 키의 경우 보안 인 밴드 키 전달 전략을 사용합니다. 앞의 경우와 마찬가지로 노드 장애를 허용하기 위해 브로커 복제본을 추가해야 합니다. 그러나 앞의 예제에서 사용한 간단한 복제 기술은 암호 해독 키를 전송하는 경우에 충분히 안전하지 않습니다. 어려움은 중개인이 비잔틴인지 여부를 완벽하게 감지하는 것이 불가능하다는 사실에서 비롯됩니다. 비잔틴 중개인이 암호 해독 키를 얻지 못하게 하기 위해 암호 해독 기술을 사용하여 초기에 게시자에서 해독 키를 여러 공유로 안전하게 분리합니다. 퍼블리셔 P의 다음 흡인 가상 노드 V에 r 개의 복제 본이 있다고 가정합니다. 게시자가 직접 연결되어 있는 V의 브로커를 게시자 에지 브로커라고 합니다. 이 비밀로 분할할 수 R의 게시자가 주, 이를 주 균등하게 분산될 수 R에에서 복제 V. 단 하나의 secret sharing을 가지므로, 각 복제본은 원래의 비밀을 재구성할 수 없습니다. 그러나 여러 비잔틴 중개인이 원래 비밀의 재건축에 필요한 충분한 수의 share을 모을 수 있는 가능성을 배제할 수는 없습니다.

또한, 그 (가정 K, N)의 임계 하는 것은 방식 Publishers에 의해 사용하는 경우, F 게시자 에지 브로커 중 비잔틴 브로커 저하 될 k secret sharing. 다른 비잔틴 브로커가 k - 1 secret sharing를 승인된 가입자에게 정확하게 전달하더라도, 그 secret sharing는 비밀 암호 해독 키를 재구성하기에 충분하지 않습니다. 이러한 관찰에 기초하여, 우리는 우선 이러한 중개인이 (k, n) 임계 값 체계와 비밀 암호 해독 키를 인가된 사람에게 확실하게 전달하는 요구 사항을 위반하지 못하도록 비잔틴 중개인의 수가 k 보다 작아야 한다고 가정해야 합니다.

이 가정은 가정 1에서 다음과 같이 공식적으로 언급됩니다.

가정 1 모든 가상 노드 V에는 비잔틴이 아닌 중개인이 있습니다. $\lfloor \frac{|V|}{2} \rfloor + 1$

예를 들어, 가상 노드에 5 개의 브로커가 있는 경우, $= \lfloor \frac{5}{2} \rfloor$ 비잔틴 브로커가 2 개 있고 비잔틴

브로커가 3 개 이상 있다고 가정합니다 $= \lfloor \frac{5}{2} \rfloor + 1$

가정 1은 우리가 달성하고자 하는 최대 내결함성을 나타냅니다. 이는 다음과 같은 위협 모델을 고려할 때 합리적인 가정입니다. 브로커 복제본을 실행하는 각 서버는 독립적인 인증 및 권한 부여 프로세스를 따르므로 특정 서버의 보안 침해가 즉시 또는 다른 서버의 다른 보안 침해로 이어지지 않습니다.

가정 1과 (k, n) 임계치 방안을 감안할 때, 우리는 n 개의 secret sharing 중 k 개의 secret sharing을 k 개의 비-비잔틴 중개인에게만 전달해야 함을 보증해야 합니다. 이 있는지 즉, F 비잔틴 브로커 적어도 있을가 F 이외 비잔틴 브로커 + 1 개 추가 복제. 따라서, 발행자에서 사용되는 임계치는 ($f+1, 2f+1$)로 표현될 수 있습니다. 대안으로서, 우리는 ($n \lfloor \frac{n}{2} \rfloor + 1, n$)으로 임계치 방정식을 표현할 수 있는데, 여기서 n 는 다음 흡 가상 노드의 복제본 수입니다. 예를 들어, 발행자의 다음 흡에서 5 게시자 에지 브로커가 있습니다 P 후 (3, 5)의 임계 방식을 사용해야, P는 .

그러나 초기 키 분할만으로도 분주 공유가 게시자 에지 브로커를 초과하여 가입자에게 안전하게 제공될 수 있다고 보장할 수는 없습니다. 우리는 이 문제를 다음 하위 섹션에서 더 분명히 밝힙니다.

secret sharing의 propagation.

우리가 secret sharing를 가입자들에게 믿을 수 있게 전달하는 문제를 제시하기 전에, 우리는 다음과 같이 주요 표기법을 채택합니다.

- V : 가상 노드
- V_f : 전달 중개자가 있는 가상 노드
- V_r : 수신 브로커와 가상 노드
- $\text{prec}(V)$: V 에 선행하는 가상 노드. 예를 들어, V_f 는 V_r 에 선행합니다.
- $|V|$: 가상 노드 V 의 브로커 수
- 재구성 (S) : 분할 secret sharing의 현재 수신 세트의 경우는 true를 반환하는 술어 S 는 앞의 가상 노드에서 비밀 분할 재구성하는 데 사용할 수 있는 V .
- n_{Byz} : 비잔틴 브로커가 아닌
- Byz : 비잔틴 중개인
- $n_{\text{Byz}}(V)$: 비잔틴 브로커 중 V 세트
- $\text{Byz}(V)$ (V 비잔틴 브로커의 집합) : V
- $B(V)$: 모든 브로커의 집합 V

첫 번째 과제는 비잔틴 브로커가 이전 흡에서 받은 secret sharing을 변경하는 것을 방지하는 것입니다. 이러한 변조는 가입자 측에서 메시지의 무결성을 검사하기 위한 디지털 서명과 같은 잘 알려진 보안 방법으로 쉽게 예방할 수 있습니다

더 어려운 과제는 (k, n) 임계 값 체계가 시행될 때 단일 브로커에서 k 공유가 publication 전달 경로를 따라 가지 않도록 하는 것입니다. k 공유를 받은 브로커가 비잔틴이 되면 모든 공유를 삭제하여 가상 노드에 $k-1$ 공유만 남길 수 있습니다. 이 경우, 원래의 키는 가입자 측에서 신뢰성 있게 재구성될 수 없습니다. 여러 비잔틴 중개인이 적어도 k 개의 share을 모으기 위해 서로 공모하면 비잔틴 중개인이 비밀을 재구성하고 이를 남용할 수 있습니다. 따라서 모든 흡에서 비잔틴 브로커가 집합 적으로 k 개 이상을 받지 못하게 하는 것이 중요합니다.

첫 번째 단계로서 (k, n) 임계 propagation라는 기본 secret sharing propagation 체계를 적용합니다. 이 propagation 방식의 구현은 알고리즘 1에 제공됩니다. 이 알고리즘은 다음 흡 가상 노드의 브로커가 n 개의 분할 보안 공유 간에 k 개 이상의 공유를 수신하지 않도록 설계되었습니다.



알고리즘 1 : 결정론적 (k , n) 임계 값 propagation 방식

```
/* 입력 : ( k , n ) 비밀 공유 체계로 생성된 n 초기 공유 */  
  
1 foreach 전달 브로커  $B_f \in V_f$  do  
2    $x =$  총 secret sharing  $B_f$  가 있음;  
3   foreach 공유  $i \in x$  do  
4     foreach 수취 중개인  $B_r \in V_r$  do  
5        $t = B_r$ 의 총 수신 secret sharing ;  
6       만약  $t < K$  라면  
7         Send I to  $B_i$  .  
8        $t = t + 1$ ;
```

알고리즘 1은 심각한 제한요소를 가지고 있는데, 그 이유는 forwarding broker B_f 가 scheme 자체를 망가트릴 수 있고, 그 share 를 rm 다음번에 브로커로부터 받는 share 를 보낼 수 있기 때문입니다.

그림 5의 다음 예제를 고려하십시오 . 게시자 P_1 브로커에 공유 비밀을 전송 B_1, B_2 와 B_3 가상 노드에서 V_1 . 이제 secret sharing가 후속 가상 노드 V_2 로 중계되어야 한다고 가정합니다. 또한 3 개의 복제본이 있습니다. 이상적인 경우의 각 브로커 V_2 는 하나의 점유율을 받아야합니다. 그러나, 가정 $B_{(3)}$ 의 $V_{(1)}$ 및 $B_{(2)}$ 의 V_2 에 도시 된 바와 같이, 비잔틴 브로커 될 일이 도 3에 있습니다 . $B_{(3)}$ 의 V_1 secret sharing propagation 정책을 무시하고 해당 share를 전달할 $B_{(2)}$ 의 V_2 가 있습니다. 두 share의 수령시에 $B_{(2)}$ 의 V_2 비밀 키를 재구성하거나 의도적으로 키를 삭제할 수 있습니다. 이 경우를 방지하기 위해, 우리는 secret sharing 체계를 강화하는 것을 고려할 수 있습니다. V_1 은 다음과 같습니다. 그림 5 (A) 와 같이 임계 값이 (2, 3)에서 (3, 5)로 증가하고 두 개의 복제본 (B_4 와 B_5)이 V_1 에 배치됩니다 . 이와 같이, $B_{(2)}$ 의 V_1 비밀을 재구성할 수 없고, 다른 비잔틴 브로커 안전하게 원래 비밀 재구성 충분한 세 공유를 전달할 수 있습니다. 그러나 우리는 이 새 임계 값 체계도 실패하는 경우를 쉽게 발견할 수 있습니다.

도시 된 바와 같이 도 3 (B)가 있다고 가정 $B_{(1)}$ 하는 대신에 $B_{(3)}$ 의 V_2 비잔틴 브로커 밝혔습니다. 또한, V_1 의 다른 비잔틴 중개인 B_3 이 V_2 의 B_1 에 그 공유를 전달한다고 가정하십시오 . 이 예들은 k가 다음과 같은 상황을 방지할 수 없음을 보여줍니다. Share는 어떤 흙 (hop)에서도 임의의 단일 중개인에게 도착합니다. 이것에는 두 가지 이유가 있습니다. 첫째, 전달 노드의 중개인은 다음 흙의 복제본이 비잔틴인지 여부를 확실하게 감지 할 수 없습니다. 둘째, 비잔틴 중개인은 다음 흙의 복제본에 공유를 보내는 것과 같은 임의의 동작을 수행할 수 있습니다.

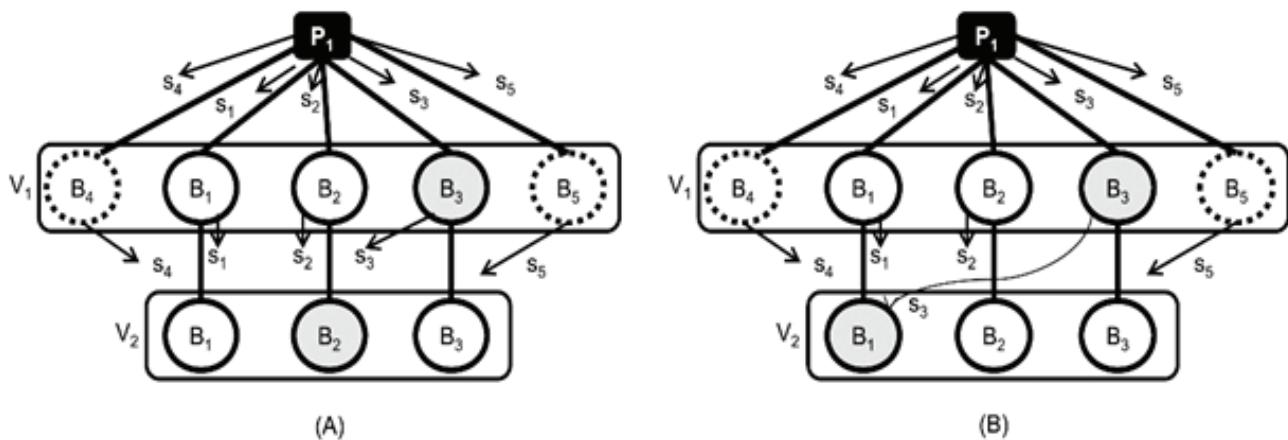


그림 6. 다중 흡을 통한 secret sharing propagation 문제

대안으로 선형 네트워크 코딩을 사용하여 원래의 비밀을 분할하는 방법을 고려했습니다 [25]. 선형 네트워크 코딩에서는 비밀이 n 개의 인코딩 된 블록으로 분할됩니다. 우리는 알고리즘 1의 propagation 스키마에 따라 n 개의 인코딩 된 블록을 배포합니다.

이것은 비잔틴 브로커가 원본 비밀을 디코딩하기 위해 전체 n 개의 인코딩 된 블록을 필요로 하기 때문에 비잔틴 브로커가 원래의 암호를 불법적으로 재구성하는 것을 방지하는 강력한 메커니즘이다. 한편, (k, n) 임계치 방식이 사용되면, k 비잔틴 중개인이 원래의 비밀을 재구성하려면 +1 주를 필요로 합니다. 가상 노드의 포워딩 중개자 대부분이 비잔틴이 아니라고

가정하면, 비잔틴 브로커가 비공식 중개인이 알고리즘 1에 설명된 전달 방식을 준수하는 한 비잔틴 중개인이 n 개의 인코딩 된 블록을 모두 수신하는 것은 불가능합니다 알고리즘 1과 쌍을 이루는 선형 네트워크 코딩은 더 높은 결합 허용을 나타낼 수 있습니다. 그러나 이 버전의 보안 propagation도 실패할 수 있습니다. 예를 들어, 도시 된 바와 같이, 도 4의 (A), 복호기가 세 개의 블록들로 인코딩 된다고 가정 $C_{(1)}, C_{(2)}$ 및 $(C)_3$. 한다고 가정 $C_{(2)}$ 및 $(C)_3$ 도달 B_2 의 V_2 비잔틴 브로커로서 $B_{(3)}$ 의 V_1 은 임의로 그의 코드 블록을 전송하는 $B_{(2)}$ 대신에 $B_{(3)}$ 의 V_2 . 아무 문제가 없다 B_2 의 $V_{(2)}$ 원래의 암호 해독 키를 재구성할 수 있을 것입니다. 그러나, 이 브로커는 임의로 앞쪽 부호화 블록 월 B_1 의 V_3 필요한 모든 부호화 된 블록들이 수집되는 상황에 이르게 다음 흡에서. 만약 B_1 의 V_3 가 비잔틴으로 밝혀지면 이 브로커는 키를 재구성하여 악의적인 의도로 사용할 수 있습니다.

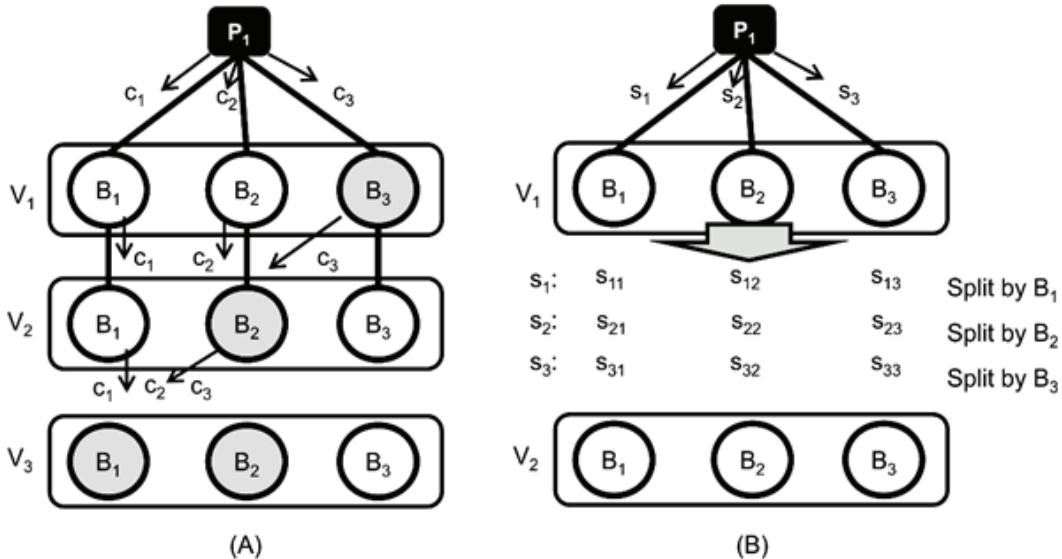


그림 7. 암호화된 게시 메시지 전달 실패 예 및 secret sharing의 신뢰할 수 있는 propagation propagation 예제.

앞서 언급 한 propagation 체계는 게시자에게 원래 비밀을 한 번만 분할합니다. 우리는 이제 경로 아래로 secret sharing을 분리하기로 선택하고 이것이 가장 실용적인 솔루션임을 입증합니다. 도시 된 바와 같이 예를 들어, 도 4의 (B) 이후, $B_{(1)}$ 의 V_1 secret sharing 수신의 1 게시자로부터 P_1 , 그것은 더욱 세 서브 주에 주 분할 이야 $(_{11})$, S 의 $(_{12})$ 과 $S_{(13)}$. 다음에, $B_{(1)}$ 의 V_1 다음 흡에 그 더 분할 secret sharing를 중계 V_2 를 합니다.

반복적 비밀 propagation 라 불리는 propagation 방식은 알고리즘 2에서 구현됩니다.

알고리즘 2 : 반복 secret sharing propagation 방식

```

/* 입력 : ( k , n ) 비밀 공유 체계로 생성된 n 초기 공유 */

1 foreach 전달 브로커  $B_f \in V_f$  do
2    $x =$  총 secret sharing  $B_f$  가 있음;
3   foreach 공유  $i \in x$  do
4      $S(i) = i$  를 ( $|V_r| - |Byz(V_r)|, |V_r|$ ) 임계 스킴으로 나눔 으로써 secret sharing 목록 ;
      도 5는 X의 첫번째 인덱스 =  $V$ 의  $(R)$  ;
6   foreach  $i \in S(i)$  do
7     보내고 난 '으로 X에서 브로커 번째'  $V$ 의  $(R)$  ;
8    $x = x + 1;$ 

```



우리는 알고리즘 2에서 가상 노드의 비잔틴 중개인이 원래의 비밀을 재구성하기에 충분한 수의 secret sharing을 받을 수 없다는 것을 증명합니다. 우리가 증명을 진행하기 전에 몇 가지 추가 핵심 가정을 설정합니다.

가정 2 게시자가 올바르게 작동합니다 .

가정 2와 달리 게시자는 게시 / Subscriptions 오버레이를 공격할 수 있습니다. 예를 들어, Wun et al. 서비스 거부 공격 (DoS) 공격에 참여하는 게시자의 가능성을 제시했습니다 [26]. 그러나 이 백서는 비잔틴 중개인 문제를 다루는 데 중점을 두고 있으며 악의적 인 게시자에 대한 보안 대책을 고안하는 것은 이 백서의 범위에 포함되지 않습니다.

가정 3 비잔틴 브로커는 메시지 propagation 규칙을 준수합니다 .

이제 우리는 알고리즘 2가 브로커에 의해 시행된다면 정리 1이 성립함을 증명하고,

정리 1 V_f 에서의 브로커 B_f , R_f 은 이상 수신할 수 V_f 에서의 secret sharing 집합 $\{S\}$ 각 세트에 대해 같은 것을 S_f 가 수신된 재구성을 (S_f) 보유하고 있습니다 . $\lfloor \frac{|V_f|}{2} \rfloor$

증명 1 V_f 에서의 가정합니다 B_f , R_f 은 수신된 브로커에서 secret sharing 집합 $\{S\}$, 각 세트에 대해, 그런 S_f 가 수신된 재구성 (S_f) 보유하고 있습니다. 이 경우에만 발생 V_f 의 비잔틴 브로커 f 는 브로커가 분할 공유 배포해야 합니다 알고리즘 2의 프로토콜 위반 ' S_f '를 재구성하는 방식으로 (S'_f) (: 8-10 알고리즘 2에 강제로) 유지하지 않습니다. 이것은 V_f 에 있는 브로커의 대다수가 비잔틴이 아니라는 것을 의미합니다 . 그러므로 가정 1과 모순됩니다 $\lfloor \frac{|V_f|}{2} \rfloor + 1 < \lfloor \frac{|V_f|}{2} \rfloor + 1$

정리 1은 비잔틴 중개인이 모든 분할 secret sharing을 받지 못하면 비밀을 재구성할 수 없다고 말하고있습니다 .

정리 2 V_f 의 비 비잔틴 브로커 R_f 은 V_f 에서의 비 비잔틴 브로커로부터 secret sharing를 수신 $\{S\}$ 를 원래 비밀 재구성 충분한 S_f 제작자에 의해 생성을 PUB .

우리는 다음과 같이 수학적 유도에 의한 정리 2를 증명합니다.

증명 2 기준 : PUB와 관심있는 가입자 사이에는 하나의 가상 노드 만 있습니다 .

V_f 에 속한 비잔틴 브로커가 받는 secret sharing의 수가보다 적다고 가정하십시오 $\lfloor \frac{|V_f|}{2} \rfloor + 1$. 이것은 게시자가 충분한 수의 secret sharing을 생성하지 않았다는 것을 의미합니다. 따라서 이것은 가정 2와 모순됩니다 .

귀납적 단계 : PUB의 i 번째 연속 가상 노드에 있는 비잔틴 브로커가 원래 비밀 S_f 를 재구성하기에 충분한 총 secret sharing를 받는다고 가정합니다 . 우리는 다음 가상 노드 V_{i+1} 에서 비잔틴 브로커가 S_f 를 재구성하기에 충분한 수의 secret sharing을 받는다는 것을 보여줍니다 .



nByz secret share의 총 수가 있다고 가정 (V_{i+1}) 수신이보다 작 ($\lfloor \frac{|m|}{2} \rfloor + 1$) ($\lfloor \frac{|V_{i+1}|}{2} \rfloor + 1$) 에서 바이츠 (V_i). V_i 에 속하는 모든 비잔틴 브로커 nByz는 총 $|V_{i+1}|$ B (V_{i-1})로부터 받은 모든 비밀 nByz에 대한 비밀 몫. 귀납적 인 단계로서 i 번째 가상 노드까지 비잔틴 브로커가 올바른 수의 secret sharing를 수신했다고 가정했습니다. 따라서, 가정이 V_i 에서 비 비잔틴

브로커 $\lfloor \frac{|m|}{2} \rfloor + 1$ 은 ($\lfloor \frac{|m|}{2} \rfloor + 1$) ($\lfloor \frac{|V_{i+1}|}{2} \rfloor + 1$) 보다 적게 받은 것은 V_i 적어도 하나의 비잔틴 브로커가 B (V_{i-1})로부터 받은 share 중 하나 보다 적게 생성되었음을 의미합니다. 이것은 또한 비잔틴 브로커가 아닌 사람이 알고리즘 2에 명시된 규칙을 위반 했으므로 가정 3과 모순된다는 의미입니다. $\lfloor \frac{|V_{i+1}|}{2} \rfloor + 1$

정리 2는 비잔틴 브로커가 공인된 수신자에게 원래의 비밀을 재구성하기에 충분한 secret sharing 집합을 항상 전달하도록 보장된다는 것을 명시하고있습니다. 마지막으로 정리 3을 도출할 수 있습니다.

정리 3 발행자 PUB에 의해 발행된 메시지의 자격이 없는 비인가 가입자 (SUB 2)는 PUB에 의해 생성된 원래의 키 S를 재구성하기 위해 브로커로부터 충분한 수의 secret sharing를 받을 수 없습니다. PUB에 의해 게시된 메시지를 받을 수 있는 권한이 부여 된 Subscribers (SUB 1)는 PUB에서 생성 한 원래 키를 다시 작성하기에 충분한 수의 secret sharing를 받아야합니다.

증명 3 기준 : PUB와 두 명의 가입자인 SUB 1과 SUB 2 사이에는 하나의 가상 노드 V 만 있습니다 .

secret sharing SUB 1의 개수가 S를 재구성하기에 충분하지 않다고 가정합니다 . 이 가정은 V에 속한 비잔틴 브로커가 충분한 수의 secret sharing을 보내지 못했음을 의미합니다. 그러므로, 이 가정은 정리 2와 모순됩니다 .

SUB secret sharing 수 있다고 가정 (2)가 수신하고 재 구축하기에 충분하다 S . 이 가정은 V의 비잔틴 중개인이 S를 재구성하기 위해 SUB 2에 충분한 수의 secret sharing를 보내기 위해 서로 공모할 수 있음을 의미합니다 . 그러나, PUB 가 Byz (V)에게 보낸 secret sharing의 수는 보다 $\lfloor \frac{|V|}{2} \rfloor + 1$

유도 단계 : 3 내가 연속 가상 PUB 사이의 노드와 두 가입자, SUB있을 때 보관 유지하는 정리 가정 (1) 과 SUB 2. 내가 있을 때 (3)가 보유하고 정리 $\lfloor \frac{|V|}{2} \rfloor + 1$ 개 PUB 사이의 가상 노드와 두 가입자, SUB 1 및 SUB 2.

secret sharing SUB 1의 수가 V_{i+1} 을 통해 수신하면 S를 재구성하기에 충분하지 않다고 가정합니다 . 이 가정은 V_{i+1} 의 비잔틴 브로커가 SUB 1에 충분한 수의 secret sharing을 전달하지 못했다는 것을 의미합니다 . 이것은 정리 2 와 모순됩니다 .

SUB secret sharing 수 있다고 가정 2 수신을 재구성하기에 충분한 S를 . 이것은 V_{i+1} 에 있는 브로커 가 SUB 2 로 자신의 share을 보냈을 때만 일어날 수 있습니다 . 이것은 V_{i+1} 에 있는 브로커의 대부분이 가정 1과 모순되는 비잔틴임을 나타냅니다 . $\lfloor \frac{|V_{i+2}|}{2} \rfloor + 1$

원래의 키를 재구성하기 위해서 가입자는 secret sharing이 통과 한 가상 노드의 수와 각 가상 노드에 할당된 복제본의 수를 알아야합니다. 가상 노드의 수는 수신된 secret sharing에 적용할 재구성의 수에 해당합니다. 모든 가상 노드에서의 복제본 수는 재구성을 실행할 때 적용할 임계 값 스키마에 대한 필요한 정보를 가입자에게 제공합니다. 발행인과 중개인은 이러한 정보를 secret sharing에 표시하고 브로커는 이를 가입자를 향한 엔드 투 엔드 경로로 전달합니다.

솔루션 분석 및 적용.

암호 해독 키가 모든 발행과 함께 전송된다고 가정하십시오. 그런 다음 가입자가 마지막에 받는 분할 보안 공유의 최대 수는 최대 p^{r_h} 일 것입니다. 여기서 p 는 게시자에서 Subscribers까지의 분리된 종단 간 경로의 수이고, r 은 각각의 복제본의 평균 수입니다 엔드 - 투 - 엔드 경로의 흡 H 는 흡 카운트로서 측정 경로 길이입니다. f 가 각 흡에서 비잔틴 브로커의 평균 개수라고 가정합니다. 그러면 가입자가 마지막에 받는 secret sharing의 최소 수는 $p^{r_h} - p(r-f)^{r_h}$ 입니다. 경로 길이가 증가하면 $\lfloor \frac{|m|}{2} \rfloor + 1$ $\text{shad}_{\frac{|T_{f+1}|}{2}}^{|\mathcal{T}_{f+1}|} + 1$ 수가 크게 늘어날 수 있습니다. 그러나 Cloud 기반 pub / sub 시스템 [27]의 등장과 함께 pub / sub overlay가 점점 더 평평해지고 있습니다. 즉, end-to-end 경로 길이는 최대 3입니다.

그러나 secret sharing의 수가 무시할 수 없는 우려라면 secret sharing을 줄이기 위한 두 가지 적용 기법이 있습니다. 하나는 모든 게시에 대해 암호를 생성하는 대신 대량의 게시에 대한 암호 해독 키를 새로 고치는 것입니다.

또 다른 적용 기법은 외부 저장소를 통해 대역 밖의 secret sharing를 전달하는 것입니다. 이 저장소는 또한 secret sharing를 별도로 보유하기 위해 복제될 수 있습니다. 이렇게 하면 비밀 대역의 대역 내 전달에 비해 트래픽 증가가 줄어들 수 있습니다. 그러나 Subscribers는 다른 커뮤니케이션 채널을 통해 저장소에서 키를 가져와야 하기 때문에 게시 내용을 열면 더 늦어 질 수 있습니다. 반대로 게시 내용을 게시에 피드백 되는 즉시 해독 키로 열 수 있으므로 대역 내 비밀 전송 방법에 추가 대기 시간이 필요하지 않습니다. 대역 내 전달 접근법은 또한 클라이언트가 시간과 공간에서 분리되어 확장 가능한 통신을 보장할 수 있는 pub / sub의 특성을 고수합니다 [1].

암호화된 콘텐츠 propagation

지금까지 복제된 브로커와 함께 pub / sub 오버레이에 적용되는 해독 키 propagation 방법을 소개했습니다. 복제본은 암호화된 컨텐트를 전달할 수 있는 여러 대체 경로를 소개합니다. 일반적으로 대체 경로는 트래픽 로드 균형 조정을 위한 기회를 제공합니다. 그러나 우리의 맥락에서 이러한 경로는 publication 자체의 안정적인 전달이라는 새로운 문제를 수반합니다.

n 개의 복제본을 가진 다음 흡 가상 노드가 주어지면 포워딩 가상 노드는 n 신뢰할 수 있는 배달을 보장하기 위해 암호화된 콘텐츠의 복제본이 생깁니다. 이러한 방식으로 관심있는 가입자를 향한 경로로 콘텐츠를 복제하면 네트워크 트래픽이 크게 증가할 수 있습니다. 이 문제를 피하기 위해 복제본 중 하나에 하나의 publication 만 전송하고 분실되었을 경우를 대비하여 publication을 다시 전송할 수 있습니다. 그러나 publication을 다시 전송하면 무시할 수 없는 지연이 발생할 수 있습니다.

중복 트래픽과 재전송으로 인한 지연을 줄이기 위해 파일을 인코딩하여 여러 블록으로 분할한 다음 여러 경로를 통해 동시에 전송할 수 있습니다. 누락된 블록만 재전송 해야 합니다. 모든 n 블록은 특정 가상 노드의 비잔틴 브로커의 손에 달려있습니다. 블록은 암호화되어 있고 이전 섹션에서 고안 한 secret sharing propagation 방법을 통해 안전하게 전송된 키로 만 해독할 수 있기 때문에 여전히 안전합니다. 블록을 손상시키는 비잔틴 중개인의 경우 게시자와 Subscribers는 디지털 서명 메커니즘을 사용하여 각 블록의 무결성을 검사할 수 있습니다. 이 접근 방식의 오버 헤드는 본 백서의 평가 절에서 측정됩니다.

브로커 복제본 관리

이전 섹션에서는 게시 propagation를 보호하기 위해 브로커 오버레이에서 복제본을 활용했습니다. 이 섹션에서는 이러한 복제본을 관리하기 위한 새로운 프레임 워크와 프로토콜을 제시합니다.

복제 배치 프레임 워크

실제로, pub / sub 오버레이의 모든 노드를 완전히 복제하는 것은 비용과 제한된 예산으로 인해 실현 불가능할 수 있습니다. 따라서 자원의 효율적인 사용을 위해 pub / sub 오버레이의 가장 적절한 위치에 전략적으로 복제본의 배치를 지시하는 프레임 워크를 고안합니다. Google의 프레임 워크에서 관리자는 복제본 게재 위치에 대한 기준을 명시적으로 지정할 수 있습니다. 이 기준은 주로 두 가지 범주로 나뉩니다. 첫 번째 기준은 신뢰성 계수 (R)를 지정합니다. 두 번 $\lceil \frac{m}{2} \rceil + 1$ 은 신 $\lceil \frac{r_{fail}}{2} \rceil + 1$ 소를 지정합니다 (P). 추가 복제본을 배치하면 pub / sub 오버레이가 내결합성이 향상됩니다. 반면 복제본을 추가하면 복제본의 secret sharing가 대기 시간과 트래픽을 증가시켜 잠재적으로 혼잡을 초래할 수 있으므로 성능이 저하될 수 있습니다. 위의 두 가지 상충되는 문제 (즉, 안정성 대 성능) 사이의 균형을 맞추기 위해 3 상 할당 방법을 고안했습니다. 이 방법에 대한 입력은 모든 게시자와 가입자 쌍 사이의 종단 간 경로 집합입니다.

오버레이에서 n 개의 노드가 주어지면 최대 n ($n - 1$) 개의 엔드 투 엔드 경로가 있을 수 있습니다. 첫 번째 단계에서 우리 프레임 워크는 신뢰성 기준에 따라 복제본을 할당합니다. 모든 엔드 - 투 - 엔드 경로에 우선 순위가 지정됩니다. 우선 순위 (ρ)는 엔드 - 투 - 엔드 경로에서 다음 메트릭의 곱으로 측정됩니다. (1) 흡수로 측정된 경로 길이. (2) 일정 기간 동안의 실패 빈도 비율 (γ) 및 (3) 사용자 정의 가중치 (ω). 실패 빈도 비율 (γ)은 모든 종단 간 경로에서 발생한 총 오류 수에 대해 종단 간 경로에서 발생한 오류 수의 비율입니다. 가중치 (ω)는 종단 간 경로의 중요도를 나타내며 사용자 (관리자)는 자유롭게 숫자 값을 지정할 수 있습니다. 복제본은 ρ 에 비례하여 할당됩니다.

두 번째 단계에서는 각 종단 간 경로에 할당된 복제본이 이제 종단 간 경로를 구성하는 노드 사이에 분산됩니다. 복제본은 엔드 투 엔드 경로 내에서 실패 빈도 비율에 비례하여 분배됩니다. 엔드 투 엔드 경로상의 노드의 주파수 비율은 엔드 투 엔드 경로 내의 모든 노드 중 실패의 총 수에 대한 노드의 실패 수의 비율로 측정됩니다. 도 5는 발행자 사이의 종단 대 종단 경로의 두 번째 단계의 완료 후에 복제본의 샘플 위치를 도시 P 및 가입자 S_1 , S_2 및 S_3 . 한다고 가정 ρ 의 경로에 대한 값, $P - S_{(1)}$, $P - S_{(2)}$ 및 $P - S_{(3)}$ 각각 부여되어 2, 3, 5됩니다. 총 10 개의 사용 가능한 복제본이 주어지면 각 경로의 복제본 수는 그림 7의 표에 표시된 것처럼 첫 번째 단계에서 결정됩니다. 두 번째 단계에서는 복제본이 개별 실패 빈도 비율에 따라 노드에 할당됩니다.

우리는 이 단계에 대한 두 가지 재미있는 것을 관찰했습니다. 첫째, 가상 노드가 단지 두 개의 복제본으로 구성된 경우가 있을 수 있습니다. 이 경우 대부분의 노드가 비잔틴이 아닐 것이라고 확신할 수 없기 때문에 secret sharing는 강제될 수 없습니다. 둘째, B에서 모든 3 개의 종단 간 경로가 교차합니다. B_1 및 B_2 . 따라서 두 브로커는 두 번째 단계를 실행하는 동안 한 번 이상 복제본을 받습니다. 두 번째 단계의 가능한 변형은 복제본 팩을 노드에 한 번만 할당하는 것입니다. 예를 들어, 복제의 2 개 팩으로부터 제거될 수 $B_{(1)}$ 및 비전하에 노드에 재 할당될 수 있습니다.

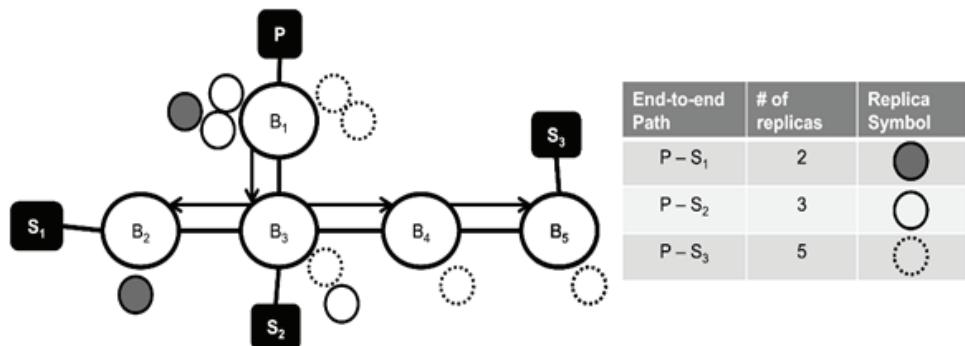


그림8. 게시자의 엔드 - 투 - 엔드 경로의 복제 배치의 예

P 가입자 S_1 , S_2 및 S_3

B는 중개인을 나타냅니다. 복제본을 사용하면 B는 가상 노드를 형성합니다.

세 번째 단계에서 각 엔드 투 엔드 경로의 복제본은 현재 할당된 복제본이 성능을 저하시키는 방식에 따라 할당이 해제됩니다. 우리가 중점적으로 다루는 QoS 메트릭은 secret sharing 계획에 의해 발생하는 대기 시간입니다. 대기 시간은 복제본 수에 비례하여 증가할 수 있습니다. 따라서 관리자는 분리된 종단 간 경로 당 할당할 수 있는 최대 복제본 수를 설정할 수 있습니다. 첫 번째 단계에서 엔드 투 엔드 경로에 할당된 복제본 수가 허용된 최대 복제본 수를 초과하면 복제본을 점진적으로 제거할 수 있습니다. 여기서는 장애 빈도 비율이 가장 낮은 노드에서 복제본을 먼저 제거합니다. 이는 복제본 추가와 대기 시간 메트릭 간의 상관 관계를 반영하는 기본 성능 모델을 기반으로 합니다.[28].

동적 복제 배포 프로토콜

이 절에서는 런타임에 pub / sub broker 오버레이에서 브로커를 유연하게 재배포하기 위한 프로토콜을 제공합니다. 이 재배포 프로토콜에는 브로커의 부착 및 / 또는 분리가 포함됩니다. 이 프로토콜은 게시 배달 서비스가 중단되는 것을 방지하도록 설계되었습니다. 복제본의 첨부 또는 분리 시 secret sharing 임계 값 체계는 가상 노드의 브로커 복제본 간에 업데이트됩니다.

재배포 프로토콜을 명확히 하기 전에 참조 pub / sub overlay 구현 [20 , 21]의 확장 브로커 아키텍처에 대해 설명합니다 . 그림 8에서 볼 수 있듯이 각 브로커에는 단일 입력 대기열과 여러 출력 대기열이 있습니다. 출력 대기열은 다음 흡의 각 가상 노드와 연관되도록 그룹화됩니다. 각 출력 대기열은 다음 흡 가상 노드의 브로커 복제본에 지정됩니다. 브로커는 입력 대기열을 통해 이전 가상 노드에서 secret sharing를 수신합니다. 이전 흡의 secret sharing 가 입력 대기열에서 대기열에서 제외되면 브로커는 주제를 실행합니다. 기반 공유를 통해 secret sharing 및 publication을 전달할 위치를 결정합니다. 개인 정보를 공개하지 않는 한 주제를 암호화할 필요가 없습니다.

그러나 화제가 암호화되어야 하지만, 동형 매칭 기술은 [29]에서 소개된 대로 사용되어야 하는데, 이는 CUBE가 이 연구 이후 가장 먼저 할 연구를 위한 주제입니다. 다음 가상 노드가 발견되면 이를 전달하기 위해 브로커는 먼저 수신된 secret sharing를 다시 분할합니다.

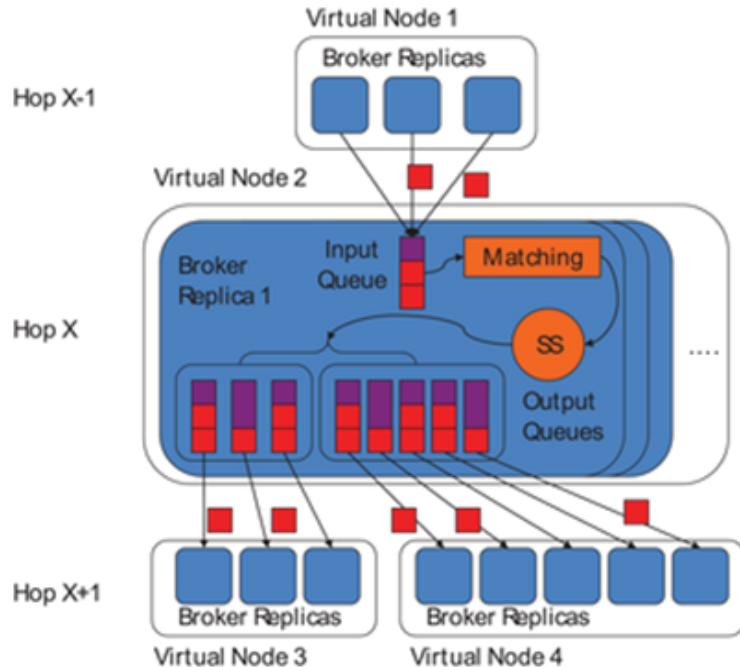


그림 9. 비밀 포워딩을 위한 확장된 pub / sub broker의 구조.

알고리즘 3 : 브로커 복제 분리

/* 분리 브로커를 들면 B를 D */

1 $B_d()$ 에 대해 모든 전달 가상 노드에 자체 ID를 보냅니다. \mathbb{N}

2 동안 모든 $V \in$ 와 입력 큐가 비어 있지 않은 ACK는 비어 있습니다. \mathbb{N}

3 처리중인 메시지를 입력 대기열에 보관합니다.

4 출력 메시지를 적절한 출력 대기열에 넣습니다.

5 모든 출력 대기열을 비웁니다.

6 $B_{d\text{의}}$ 모든 전달 및 수신 가상 노드 연결을 끊습니다.



알고리즘 4 : 브로커 복제본 첨부

```
/* 첨부 중개자 Ba */
```

1 입력 큐를 초기화하십시오.

2 라우팅 상태 복제;

3 출력 그룹 및 출력 대기열을 구성합니다.

4 개 커넥트 가상 노드 전송을 위한 B의 을 ;

5 전달 가상 노드에 B a의 ID 와 새로운 임계 값 체계를 통보합니다 .

알고리즘 5 : 브로커 B에 대한 전달 가상 노드의 브로커에서 업데이트

```
/* 알림 메시지를 받으면 */
```

detachment 통지 를 받은 경우는 1

2 임계 값 스킴 변경.

3 B에 매핑 된 출력 대기열을 플러시 합니다.

4 B에 매핑 된 출력 대기열을 제거합니다.

5 ACK를 다음 흡으로 보냅니다.

6 else

```
/* 알림 수신 시 : */
```

7 출력 대기열을 만들고 B에 매핑합니다.

8 임계 값 구성표 변경;

이제 알고리즘 3, 알고리즘 4 및 알고리즘 5에 명시된 대로 브로커를 재배포하기 위한 프로토콜을 설명합니다. 여기서는 가상 노드 간의 관계를 지정하는 몇 가지 정의를 제공합니다. 가상 노드 V의 $x_{\text{는}}$ A는 포워딩 가상 노드는 가상 노드를 위한 V의 y 경우 V의 y 에 서적 전송 V의 x . 한편, 가상 노드 V의 $x_{\text{는}}$ A는 수신 한 가상 노드를 가상 노드를 위한 V의 y 경우 V의 $X_{\text{가}}$ 으로부터 서적 수신 V의 x . 브로커 간에 교환되는 승인은 ACK로 표시됩니다.



브로커 B_d 의 분리를 위해 B_d 에 대한 가상 노드 전달의 브로커는 secret sharing를 위한 임계 값 스키마를 업데이트해야 합니다. 경우 B 에서 D 가 분리되는 가상 노드에서 브로커의 수 B 에 D 는 (표시 속하는 V 의 B 에 D) 임계 방식은 실행 가정 1 (즉 감소 도착) 전달하는 가상 노드들에 대한 B 의 D (원래 하였다, R) 가정 브로커 복제본의 수가 V_{B_d} 일 때 r 이라고 가정합니다. $\lfloor \frac{r}{2} \rfloor + 1$. B_d 가 분리되면 전달 가상 노드의 중개자는 ($\lfloor \frac{r}{2} \rfloor + 1, r - 1$) 임계 값 스키마를 새로 적용해야 합니다. 유일한 장애는 임계 값 체계가 업데이트 될 때 전달 가상 노드의 브로커가 수신 메시지 처리를 중단할 때 발생합니다. 그러나 업데이트 프로세스가 즉시 실행되므로 중단을 무시할 수 있습니다. 이 업데이트 작업은 비밀 키를 기밀로 제공하는 데 매우 중요합니다.

전달 가상 노드의 브로커가 들어오는 메시지의 처리를 일시 중지하지 않으면 들어오는 메시지가 이전 임계 값 스키마로 분할 될 수 있습니다. 이것은 비공개 share의 대부분이 단일 중개인에 도달할 수 있는 경우를 초래할 수 있으며 이는 중개인이 비잔틴 인 경우에 대비하여 비밀 재구성을 초래할 수 있습니다. 임계 값 체계가 업데이트 된 후 전달 가상 노드의 브로커는 수신 메시지를 계속 처리할 뿐만 아니라 분리 브로커 복제본에 매팅 된 출력 대기열을 플러시 합니다.

마찬가지로 브로커 B_a 를 연결하기 위해 전달 가상 노드에서 매우 짧은 일시 중지가 있습니다. 마찬가지로, 임계 값 방식은 (에서 변경해야 합니다, R 에 (), R 1). 알고리즘 4 : 2에서 설명한 바와 같이, 첨부 브로커 B_a 는 B_a 가 새로이 속한 가상 노드 (예 :)로 브로커 중 하나를 복제해야 합니다. REMUS [30] 및 Snow flock [31]과 같은 널리 사용되는 VM 복제 도구는 사용하지 않습니다. $\lfloor \frac{r}{2} \rfloor + 1$ $\lfloor \frac{r+1}{2} \rfloor + 1$ V_{B_a}]를 사용하여 복제할 브로커가 상주할 VM을 즉시 복제할 수 있습니다. 이는 VM 복제가 보안 공유와 같은 보안에 민감한 정보를 포함한 VM 상태를 복제하기 때문입니다. VM 복제 기술의 보안 허점 때문에 새로 연결된 브로커 [24]의 라우팅 상태를 구성하는 주문형 복제 기법을 채택합니다.

Subscriptions 및 광고 주제는 B_a 에서 전체 라우팅 상태를 구성하기 위한 구성 요소입니다. B_a 인접 가상 노드의 브로커에게 Subscriptions 및 광고를 전달하도록 요청할 수 있습니다. 그러나 이러한 이웃 가상 노드에는 비잔틴 브로커가 있을 수 있습니다. 비잔틴 중개인은 광고 및 Subscriptions을 임의로 삭제하거나 손상시킬 수 있으므로 B 에 의한 복제 노력을 방해합니다. 따라서 B_a 는 비잔틴 브로커가 보내는 Subscriptions 및 광고 만 받아들여야 합니다. 참고 B 의 A 는 어떤 브로커에서 식별할 수 V_{B_a} 비잔틴입니다. 그러나, 우리는 대부분의 가정 r 에 브로커가 아닌 비잔틴 있습니다. 따라서 B_a 는 최소한 수신된 경우에만 Subscriptions 또는 광고를 수락합니다 $\lfloor \frac{r}{2} \rfloor + 1$ times 여기서 r 은 복제본의 수입니다 V_{B_a} . Subscriptions 및 광고 주제의 유효성을 검사하는 이 절차는 B_a 에 대한 전체 라우팅 상태를 복제하는 데 지연을 초래할 수 있습니다. 그러나, 우리는 publication의 이웃 브로커에 전달될 수 있도록 하는 기술을 채택 B 일치하는 Subscriptions에 추가된 후 즉시 B 의 을 . 따라서 게시 전달이 매우 빠르게 재개됩니다. 동적 복제 배포 프로토콜의 비파괴 적 성격은 [24]에서 개발된 기술을 기반으로 합니다. 이전 섹션에서 설명한대로 필요에 따라 브로커 게재 위치를 동적으로 수정할 수.

secret sharing가 지연 및 트래픽에 미치는 영향

Shamir의 secret sharing 체계 [23]의 Java 구현은 PADRES pub / sub broker에 통합되었습니다. Intel Core2 Duo CPU T5550 (1.83 GHz 및 3GB 메모리)이 장착된 시스템에서 이 브로커를 실행했습니다. 게시자와 Subscribers 간의 경로 길이가 늘어남에 따라 먼저 secret sharing 수를 측정했습니다. 그림 9 (A)와 7 (B)에서 볼 수 있듯이 경로 길이와 노드 팬 아웃이 증가함에 따라 기밀 공유 수가 기하 급수적으로 증가합니다. 앞서 언급했듯이 게시자는 secret sharing 를 줄이기 위해 모든 게시마다 하나씩 생성하지 않고 대량의 게시에 대한 해독 키를 새로 고칠 수 있습니다. 예를 들면, 도 7 (C) 및 도 7 (D))를 사용하면 200MB의 데이터마다 키를 새로 고치는 것과는 달리 키가 1GB마다 새로 고쳐지면 트래픽 증가량이 약 10 배 정도 줄어들 수 있습니다.

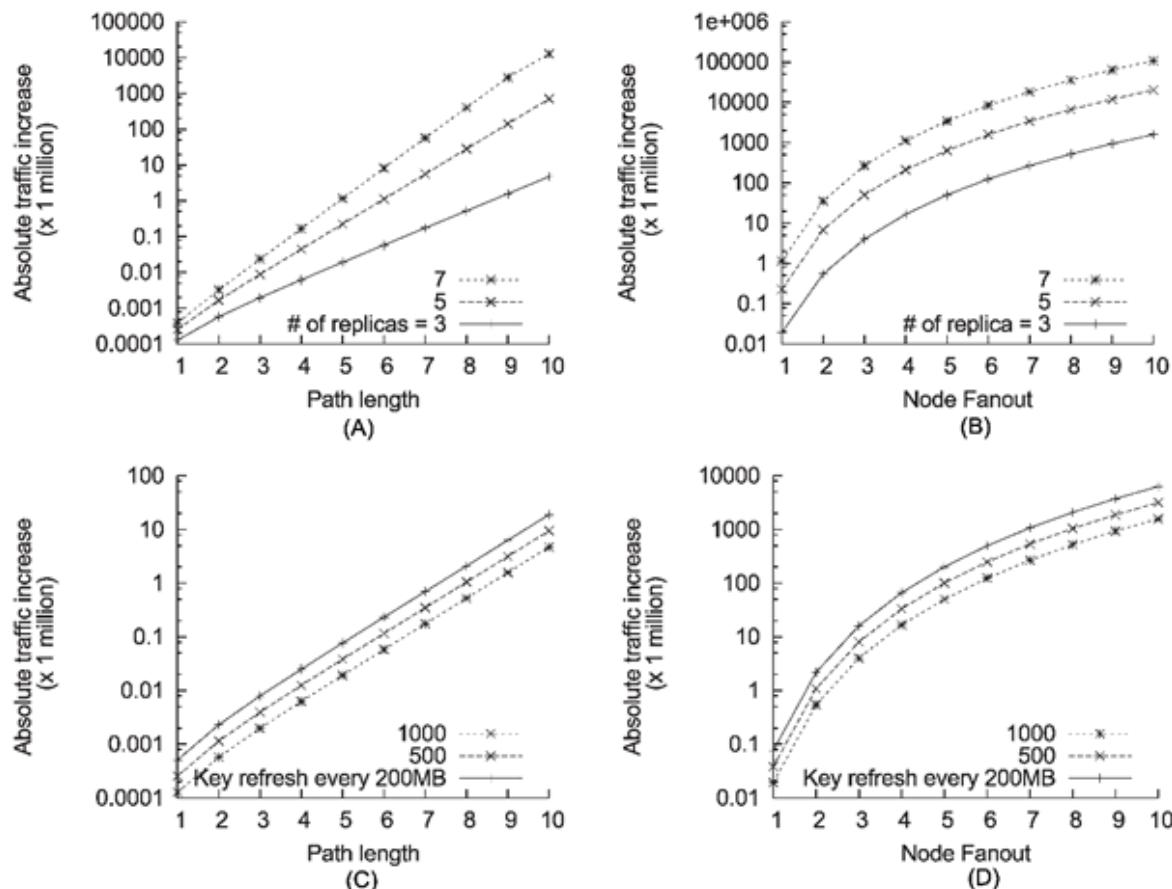


그림 10. 다양한 노드 팬 아웃 및 경로 길이로 레이던시 및 트래픽에 대한 비밀 전달 효과.

우리는 또한 비밀을 3 ~ 10 개의 share으로 나누는 데 걸리는 시간을 측정했습니다. 비밀을 재구성할 때 대기 시간도 측정했습니다. 비밀은 AES-128 또는 AES-256 키입니다. 우리는 threshold scheme (k, n)을 다음과 같이 설정합니다 $n = \lfloor \frac{k}{2} \rfloor + 1$ 도시 된 바와 같이, $k=8$, 비밀을 분열시키는 데 걸린 시간은 1 밀리 초 미만이었다. 비밀을 재구성하는 데 걸린 시간은 share의 수에 비례하여 증가했습니다. 키를 분할하는 것보다 재구성하는 것이 더 오래 걸립니다.

예를 들어, 최대 10 개의 공유에서 평균 5.2 ms 밖에 걸리지 않았습니다. 그러나 재구성은 가입자에 의해 단 한 번만 수행되며 엔드 투 엔드 경로를 따르는 브로커는 재구성에 관여하지 않습니다. 우리의 계획에는 모든 가상 흡에서 비밀 분할이 필요합니다. 따라서 모든 가상 흡에서 브로커 복제본에서 비밀 분할 오버 헤드가 발생합니다. 이로 인해 전체 종단 간 대기 시간이 흡 수에 따라 증가합니다. 그러나 비밀 분할은 브로커 복제본 간에 동시에 수행되므로 모든 가상 흡에서 복제본의 수에 따라 종단 간 대기 시간이 증가하지 않습니다.

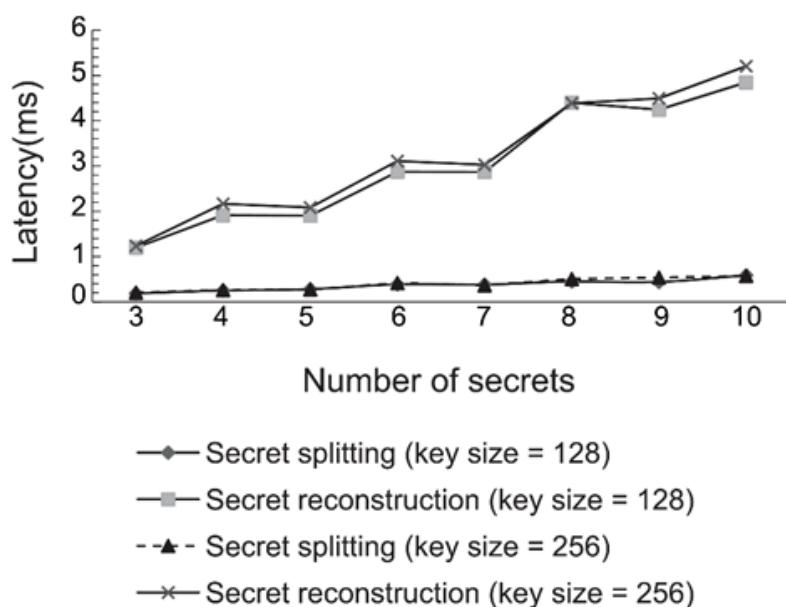


그림 11. 비밀 분할과 재구성의 성능 오버 헤드.

비잔틴 중개인의 효과

이 섹션에서는 Zipf 배포에 따라 브로커 오버레이 네트워크의 노드에서 무작위로 선택된 비잔틴 브로커의 존재 하에서 얼마나 많은 재전송이 발생할 수 있는지 보여줍니다. 우리는 Zipf 분포의 왜곡을 결정하는 α 를 변경하여 비잔틴 브로커의 클러스터링 정도를 변경했습니다. α 가 높을수록 비잔틴 중개인은 더 밀집됩니다. 이전의 테스트 케이스와 마찬가지로 값 c 로 제어되는 Zipf 배포에 따라 Facebook 사용자를 pub / sub broker 오버레이에 무작위로 배치했습니다. 우리는 비잔틴 브로커가 안정적인 전달 요구 사항을 위반하기 위해 메시지를 삭제하고 가정했습니다. 또한 장애 감지 및 장애 조치 메커니즘이 활성화되지 않았다고 가정합니다. 이 설정이 주어지면 노드 팬 아웃이 5 인 400 노드 브로커 오버레이에서 상호 작용을 재생하여 그 결과를 그림 11에 표시합니다.

우리가 처음으로 관찰한 것은 재전송의 증가가 비잔틴 중개인 수의 증가에 비례하지 않는다는 것입니다. 이는 브로커가 Facebook 사용자 간의 여러 종단 간 경로 교차로에 위치할 수 있기 때문입니다. 따라서 소수의 비잔틴 중개인조차도 pub / sub 상호 작용에 큰 영향을 줄 수 있습니다. Facebook 사용자가 서로 멀리 떨어져 있으면 비잔틴 브로커가 서로 다른 종단 간 경로에서 교차할 확률이 높아집니다. 그러나 많은 경우 비잔틴 중개인이 α 에 적당히 분산되었을 때 재전송이 가장 빈번하게 발생했습니다.

이것은 오버레이 네트워크의 핵심에 더 가까운 브로커가 선택되었기 때문에 비교적 많은 수의 교차에 영향을 주기 때문입니다. 전체 재전송 횟수는 우리가 재생한 총 상호 작용 수보다 큽니다. 예를 들어, $\alpha = 1$, $c = 0.5$ 및 $f = 15$ 인 경우 필요한 재전송 횟수가 250 만 회를 초과했습니다. 이는 Facebook 사용자 간의 종단 간 경로를 따라 둘 이상의 브로커가 상호 작용하는 동안 실패했음을 나타냅니다. 이 실험을 통해 브로커 중 일부는 게시 / 서브 오버레이 및 실행중인 서비스에 큰 영향을 미칠 수 있음을 알 수 있습니다. 이를 방지하기 위해 비잔틴 브로커 및 장애 조치 메커니즘을 신속하게 탐지해야 합니다. 그러나 완벽하게 비잔틴 중개인을 발견하는 것은 실제로 가능하지 않습니다. 실행 가능한 솔루션은 브로커가 secret sharing가 경로로 더 멀리 나뉘어 질수록 수신된 콘텐츠를 경로 아래로 복제하도록 강제하는 것입니다.

그러나 내용은 secret sharing보다 훨씬 클 수 있습니다. 따라서 이 솔루션으로 인한 트래픽 증가는 비실용적입니다. 따라서 불완전한 장애 조치 메커니즘과 엄격한 복제 계획 간의 균형을 해결하는 새로운 솔루션이 필요합니다. 우리는 앞으로 이 솔루션을 개발할 계획입니다.

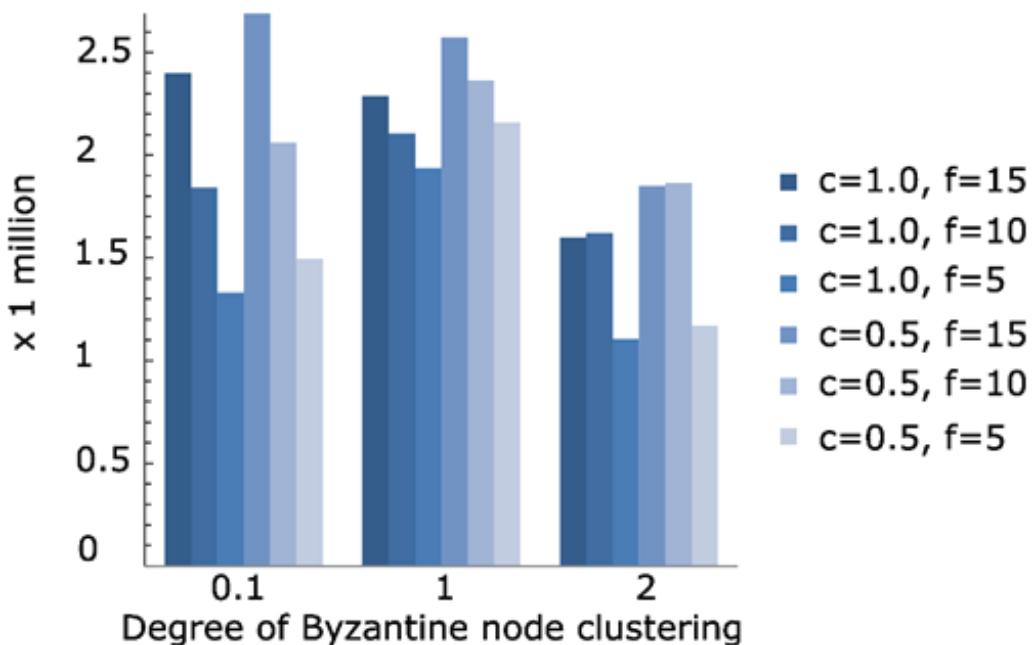


그림 12. 비잔틴 브로커가 있는 상황에서의 재전송 횟수.

관련된 일

이 섹션에서는 브로커 오버레이를 통한 pub / sub 메시징의 신뢰성 및 기밀성과 관련하여 기존의 작업이 어떻게 문제를 해결하는지 조사합니다. 우리는 이 최첨단 작품들과 비교함으로써 작품의 참신함을 제시합니다.

Gryphon [33]은 다중 중복 오버레이를 유지하는 pub / sub 시스템입니다. 오버레이에서 오류가 발생하면 게시자와 Subscribers가 백업 오버레이로 전환합니다. 그리폰 (Gryphon)은 정확히 한 번 배송됩니다. 그러나 Gryphon은 대부분의 시간 동안 과소 이용될 수 있는 백업 오버레이에 대한 리소스의 과도한 프로비저닝을 필요로 합니다. 대조적으로, Yoon et al. 결함이 있는 브로커를 요청에 따라 복제하는 기술을 고안했습니다 [24]. 브로커의 동적 복제 시 다양한 기능적 및 비 기능적 요구 사항을 충족시키기 위해 여러 방법으로 게시 전달을 다시 시작할 수 있습니다. 이 작업은 또한 정확히 한 번 배달 및 게시자 FIFO 순서를 지원합니다. 그러나 이 작업은 주로 오류가 많은 단일 브로커를 복제하는 데 초점을 맞춥니다. [34], Kazem Zadeh et al. 각 중개인이 이웃에 대해 구성 가능한 가시성을 갖는 pub / sub 시스템을 고안했습니다. 예를 들어 가시성을 3으로 설정하면 중개인은 최대 3 흡 거리에 있는 이웃의 상태에 액세스 할 수 있습니다. 이 시스템에서 브로커는 소프트 링크를 적절하게 설정하여 가시성 범위 내에 있는 여러 개의 오류가 있는 브로커를 우회 할 수 있습니다. 그러나, 앞의 두 작품 [24 , 34]은 기밀 문서 배포를 지원하지 않습니다. 따라서 원치 않는 가입자가 게시 내용을 수신 및 공개하는 것을 막을 수 없습니다.

우리의 pub / sub 시스템은 기밀 전달뿐만 아니라 안정적인 전달을 수행하기 위해 비밀 전달 기술을 실행합니다. [24], 우리의 pub / sub 시스템은 [33 , 34]의 시스템과 같이 중복 오버레이를 과다 공급하는 것과 반대 방향으로 탄력적으로 성장하고 축소할 수 있는 오버레이를 기반으로 합니다. 우리의 작업은 [24]의 pub / sub 시스템보다 더 진보적인데, 우리의 pub / sub 시스템은 하나 이상의 장애가 있는 브로커를 견딜 수 있습니다.

기존의 출판 / publication 중 일부는 출입 통제 메커니즘을 사용하여 publication의 내용 기밀을 보호합니다. 예를 들어 Gryphon은 누가 정보 공간의 일부를 게시하고 Subscriptions 할 수 있는지 제한하기 위한 액세스 제어 체계를 제공합니다. Event Guard [35]는 액세스 제어도 지원합니다. Event Guard는 게시자와 가입자가 신뢰할 수 있는 동안 라우팅 브로커가 메시지를 도청 및 삭제하거나 넘칠 수 있는 위협 모델을 가정합니다.

그러나 이러한 연구는 여러 비잔틴 중개인이 publication에 사적 내용을 공개하기 위해 서로 공모하는 경우를 다루지 않습니다. 우리 시스템은 브로커에 secret sharing 계획을 적용함으로써 그러한 사례를 방지합니다. [36]에서 역할 기반 액세스 제어는 브로커와 클라이언트 간에 투명하게 액세스 제어를 시행하는 데 사용됩니다. 그러나 이 작업은 브로커가 올바르게 작동하는 것을 신뢰하지만 브로커가 비잔틴이 될 수 있는 경우를 설명합니다.

출판 기밀성을 보호하기 위한 또 다른 작업은 publication을 암호화하기 위한 암호화 기술을 사용합니다. 게시가 암호화되어 있기 때문에 악의적 인 중개인이나 의도하지 않은 Subscribers가 publication을 수신하더라도 해독 키가 없는 경우 기밀 내용을 공개할 수 없습니다. 그러나 이 방법은 해결해야 할 사소한 딜레마에 직면합니다. 이 방법은 데이터 누출을 어느 정도 방지할 수 있지만 암호화된 콘텐츠에서는 실제 콘텐츠를 검사할 수 없기 때문에 콘텐츠 기반 라우팅을 어렵게 만듭니다. 이러한 딜레마를 해결하기 위해, Nabeel et al. [37]는 publication의 내용에서 일련의 속성을 추출하고 암호화된 페이로드 대신 이러한 속성에 대해 일치 알고리즘을 실행했습니다.



또한, Nabeelet al. 암호 해독의 필요없이 암호화된 출판 콘텐츠에 대해 일치 연산을 실행하기 위해 준 유사 암호화 기법을 사용했습니다 . 이 방법을 사용한 일치 결과는 암호화되지 않은 게시에 대한 일치 방법과 일치하도록 보장됩니다. [38], Choi et al. 변환을 보존 하는 스칼라 제품을 사용하여 브로커에서 Subscriptions과 일치하는 동형적으로 암호화 된 publication을 일치시키는 성능 오버 헤드를 줄이는 데 중점을 둡니다 . 그러나 작업은 게시자와 Subscribers가 대역 외 채널을 통해 암호 해독 키를 교환하기 위해 사전에 서로 연락해야 합니다. [39]. 우리의 접근 방식과는 달리 이것은 클라이언트가 시간과 공간에서 일반적으로 분리되어 있는 pub / sub의 고유 한 특성을 파고합니다 [1]. 이 작품들은 동형 적합성을 가진 중개인이 충돌에 시달리는 경우를 다루지 않습니다. 우리의 pub / sub 시스템은 장애 발생시에도 가상 노드에 동적으로 브로커를 추가하기 때문에 고장에 견딜 수 있습니다. 우리의 시스템은 이러한 안전한 콘텐츠 일치 기술과 직각을 이룹니다.

[40]에서는 신뢰할 수 없는 중개인 및 가입자로부터 게시의 신뢰성, 무결성 및 기밀성을 보호하기 위해 공유된 비밀이 제안됩니다. 그러나 작업은 공유된 비밀 정보를 관리하는 중앙 집중식 보안 인프라를 기반으로 합니다. 이러한 중앙 집중식 접근 방식은 pub / sub 시스템의 확장성을 제한할 수 있습니다. 또한 중앙 집중식 보안 관리자는 단일 실패 지점이 될 수 있습니다. 우리의 작업은 공유 비밀을 저장하기 위한 중앙 저장소를 가정하지 않습니다. 우리의 pub / sub 시스템에서 비밀은 사전 배포된 분산 브로커를 통해 전달됩니다. 따라서 비밀을 관리하기 위해 추가 인프라를 도입할 필요가 없습니다. 또한 비잔틴 브로커가 반복되는 비밀 propagation 기술을 통해 게시 propagation 경로를 따라 거주하는 경우에도 비밀을 보호합니다.

지금까지 최첨단 기술로 복제, 액세스 제어, 암호화와 같은 기존 보안 기술을 pub / sub 시스템에 적용해왔다. 그러나 우리가 알고 있는 한, 비잔틴 중개인이 암호 해독 키를 손상시킬 수 있는 사안을 다룬 기존 작품은 없으며 이는 민간 publication을 안전하게 전달하는 데 심각한 위협이 됩니다. 또한 종종 기존의 작품은 값 비싼 동기화 메커니즘과 중앙 집중식 코디네이터에 크게 의존하며, 우리의 작업은 분산된 브로커를 활용합니다. 기존의 작품들 중 어느 것도 비잔틴 중개인이 출판 propagation 경로를 따라 거주하는 경우를 다루지 않습니다. 간행 비밀 전달 기술을 게시 전달 경로를 통해 안전하게 전달 비밀에 적용합니다. 기존 작업은 보안 문제에만 초점을 맞추지 만, 우리 작업은 관리자가 보안 / 안정성과 성능 / 효율성 요구 사항 간의 균형을 맞추기 위한 최상의 사용자 지정 정책을 고안하는데 도움이 되는 프레임 워크를 제공합니다. 기존 작품의 대부분은 유연하게 확장하거나 축소할 수 없는 과도한 프로비저닝 중복 브로커 오버레이를 사용합니다. 우리의 업무는 구성 가능한 보안 및 성능 요구 사항을 기반으로 요청 시 브로커를 복제하고 통합하는 기술을 사용합니다.

결론

pub / sub broker 기반 오버레이에서 우리는 암호화된 컨텐트 및 암호 해독 키의 신뢰성 있고 비밀스러운 전달을 보장하기 위해 브로커 복제본에 비밀 전달 방법을 적용했습니다. 우리의 방법은 모든 엔드 투 엔드 경로에서 각 가상 노드의 브로커 복제본 중 절반 이상이 비잔틴이 아닌 한 publication 전달 경로에 따라 비잔틴 브로커의 존재를 허용합니다. 비밀 키는 모든 가상 노드에서 publication 전달 경로 아래로 더 분할됩니다. 이 방법은 비잔틴 중개인이 기밀 메시지를 해독하기 위한 비밀 키를 재구성하여 정리할 수 있는 상황을 방지하는 것으로 입증되었습니다. 이 방법은 또한 비잔틴 브로커에 의한 게시 메시지 삭제를 방지합니다. 우리는 PADRES pub / sub broker 오버레이에서 우리 계획의 성능 함의를 평가하고 우리의 계획에 대한 몇 가지 개정을 논의했습니다. 비밀 전달 기술 외에도 구성 가능한 신뢰성 및 성능 요구 사항에 따라 오버레이의 다른 부분에 전략적으로 브로커 복제본을 배치하기 위한 프레임 워크를 고안하여 자원의 효율적인 사용을 다루었습니다. 또한 브로커 복제본을 분리하고 첨부하여 브로커 복제본의 배치를 업데이트하는 무중단 프로토콜을 구현했습니다.

참고 문헌

1. Eugster PT, Felber PA, Guerraoui R, Kermarrec AM. The Many Faces of Publish/Subscribe. *ACM Comput Surv.* 2003 Jun;35(2):114-131.
2. Fawcett T, Provost F. Activity Monitoring: Noticing Interesting Changes in Behavior. In: *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD'99.* New York, NY, USA: ACM; 1999. p. 53-62.
3. Tock Y, Naaman N, Harpaz A, Gershinsky G. Hierarchical Clustering of Message Flows in a Multicast Data Dissemination System. *Parallel and Distributed Computing and Systems (PDCS 05).* 2005 Nov.
4. Koenig I. Event Processing as a Core Capability of Your Content Distribution Fabric. In: *Gartner Event Processing Summit;* 2007.
5. Corsaro A. *The Data Distribution Service for Real-Time Systems;* 2010.
6. Publish-Subscribe Internet Routing Paradigm.
7. Jokela P, Zahemszky A, Esteve Rothenberg C, Arianfar S, Nikander P. LIPSIN: Line Speed Publish/Subscribe Inter-networking. *SIGCOMM Comput Commun Rev.* 2009 Aug;39(4):195-206.
8. Apache Kafka.
9. Apache Storm.
10. MQTT OASIS Standard; 2014.
11. Shelby Z, Hartke K, Bormann C. The Constrained Application Platform (CoAP); 2014. Internet Engineering Task Force (IETF) RFC 7252.
12. Gao ZK, Yu-Xuan , Fang PC, Jin ND, Xia CY, Hu LD. Multi-frequency complex network from time series for uncovering oil-water flow structure. *Scientific Reports.* 2015;5(8222).
13. Gao ZK, Jin ND. A directed weighted complex network for characterizing chaotic dynamics from time series. *Nonlinear Analysis: Real World Applications.* 2012;13(2):947-952.
14. Gao ZK, Yang YX, Fang PC, Zou Y, Xia CY, Du M. Multiscale complex network for analyzing experimental multivariate time series. *EPL.* 2015;109(3):30005.
15. Xia CY, Meng XK, Wang Z. Heterogeneous Coupling between Interdependent Lattices Promotes the Cooperation in the Prisoner's Dilemma Game. *PLoS ONE.* 2015 06;10(6):1-13.
16. Xia CY, Meloni S, Perc M, Moreno Y. Dynamic instability of cooperation due to diverse activity patterns in evolutionary social dilemmas. *EPL.* 2015;109(5):58002.
17. Gao ZK, Fang PC, Ding MS, Jin ND. Multivariate weighted complex network analysis for characterizing nonlinear dynamic behavior in two-phase flow. *Experimental Thermal and Fluid Science.* 2015;60:157-164.
18. TIBCO Enterprise Service Bus:.
19. IBM Websphere MQ.
20. Jacobsen HA, Cheung AKY, Li G, Maniyaran B, Muthusamy V, Kazemzadeh RS. The PADRES Publish/Subscribe System. In: *Principles and Applications of Distributed Event-Based Systems;* 2010. p. 164-205.
21. Carzaniga A, Rosenblum DS, Wolf AL. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems.* 2001 Aug;19(3):332-383.
22. Mühl G, Jaeger MA, Herrmann K, Weis T, Ulbrich A, Fiege L. Self-stabilizing publish/subscribe systems: algorithms and evaluation. In: *Proceedings of the 11th international Euro-Par conference on Parallel Processing. Euro-Par'05.* Berlin, Heidelberg: Springer-Verlag; 2005. p. 664-674.

23. Shamir A. How to Share a Secret. *Communications of the ACM*. 1979;22(11):612-613.
24. Yoon Y, Muthusamy V, Jacobsen HA. Foundations for Highly Available Content-Based Publish/Subscribe Overlays. In: Proceedings of the 2011 31st International Conference on Distributed Computing Systems. ICDCS'11. Washington, DC, USA: IEEE Computer Society; 2011. p. 800-811.
25. Li SYR, Yeung RW, Cai N. Linear network coding. *Information Theory, IEEE Transactions on*. 2003 feb;49(2):371-381.
26. Wun A, Cheung A, Jacobsen HA. A taxonomy for denial of service attacks in content-based publish/subscribe systems. In: Proceedings of the 2007 inaugural international conference on Distributed event-based systems. DEBS'07. New York, NY, USA: ACM; 2007. p. 116-127.
27. Li M, Ye F, Kim M, Chen H, Lei H. A scalable and elastic publish/subscribe service. In: Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International. IEEE; 2011. p. 1254-1265.
28. Lazowska ED, Zahorjan J, Graham GS, Sevcik KC. Quantitative system performance: computer system analysis using queueing network models. Prentice-Hall, Inc.; 1984.
29. Popa RA, Redfield CMS, Zeldovich N, Balakrishnan H. CryptDB: protecting confidentiality with encrypted query processing. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. SOSP'11. New York, NY, USA: ACM; 2011. p. 85-100.
30. Cully B, Lefebvre G, Meyer D, Feeley M, Hutchinson N, Warfield A. Remus: high availability via asynchronous virtual machine replication. In: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation. NSDI'08. Berkeley, CA, USA: USENIX Association; 2008. p. 161-174.
31. Lagar-Cavilla HA, Whitney JA, Scannell AM, Patchin P, Rumble SM, de Lara E, et al. SnowFlock: rapid virtual machine cloning for cloud computing. In: Proceedings of the 4th ACM European conference on Computer systems. EuroSys'09. New York, NY, USA: ACM; 2009. p. 1-12.
32. Wilson C, Boe B, Sala A, Puttaswamy KPN, Zhao BY. User interactions in social networks and their implications. In: Proceedings of the 4th ACM European conference on Computer systems. EuroSys'09. New York, NY, USA: ACM; 2009. p. 205-218.
33. IBM. IBM Gryphon Project:.
34. Kazemzadeh RS, Jacobsen HA. Reliable and Highly Available Distributed Publish/Subscribe Service. In: Proceedings of the 2009 28th IEEE International Symposium on Reliable Distributed Systems. SRDS'09. Washington, DC, USA: IEEE Computer Society; 2009. p. 41-50.
35. Srivatsa M, Liu L, Iyengar A. EventGuard: A System Architecture for Securing Publish-Subscribe Networks. *ACM Transaction on Computer Systems*. 2011;29(4):10.
36. Bacon J, Evers DM, Singh J, Pietzuch PR. Access Control in Publish/Subscribe Systems. In: Proceedings of the second international conference on Distributed event-based systems. DEBS'08. New York, NY, USA: ACM; 2008. p. 23-34.
37. Nabeel M, Shang N, Bertino E. Efficient privacy preserving content based publish subscribe systems. In: Proceedings of the 17th ACM symposium on Access Control Models and Technologies. SAC MAT'12. New York, NY, USA: ACM; 2012. p. 133-144.
38. Choi S, Ghinita G, Bertino E. A Privacy-Enhancing Content-Based Publish/Subscribe System Using Scalar Product Preserving Transformations. In: Database and Expert Systems Applications, 21st International Conference, DEXA 2010; 2010. p. 368-384.



39. Ion M, Russello G, Crispo B. Design and implementation of a confidentiality and access control solution for publish/subscribe systems. *Computer Networks*. 2012;56(7):2014-2037.
40. Minami K, Lee AJ, Winslett M, Borisov N. Secure aggregation in a publish-subscribe system. In: Proceedings of the 7th ACM workshop on Privacy in the electronic society. WPES'08. New York, NY, USA: ACM; 2008. p. 95-104.