플랫폼 독립적인 환경에서 객체기반 키 분배 서버의 설계 및 구현

손태식[U] 서정택 이상하 김동규

아 주 대 학 교 정보통신공학과
{ tsshon, sjtgood, dkkim }@madang.ajou.ac.kr, shyi@netsec.ajou.ac.kr

Design and Implementation of Object Oriented Key Distribution Server on the Platform Independent Environments.

Tae-Shik Shon[U], Jung-Taek Seo, Sang-Ha Lee, Dong-Kyoo Kim
Dept. of Information and Communication, Ajou University

요 약

인터넷 사용 인구의 폭발적인 증가로 인한, 네트웍상의 정보 교류는 이제 실생활에서는 빼 놓을 수 없을 만큼 하나의 큰 부분으로 자리잡고 있다. 이러한 상황에서 정보보호의 중요성은 정보 교류의 필요성 이상으로 심각하게 대두되는 문제이다. 현재 정보보호 서비스 시스템에서 빠질 수 없는 부분이 양단간 기밀 통신을 위한 키의 분배이며, 키의 분배는 시스템의 효율을 위해 세션키를 공개키 방식으로 암호화하여 분배하는 방법을 사용한다. 본 논문에서는 서로 다른 기종간에도 쉽게 적용이 가능하도록, 자바로 구현된 키 분배 서버를 제안하며, 이 키 분배 서버는 객체 기반으로 설계되어 정보보호 서비스 시스템에 적용 한 후에도 쉽게 유지, 보수 및 확장이 가능하도록 되어 있다. 향후의 연구 과제로는 정보보호 서비스 시스템과의 더욱 원활한 이식을 위해 향상된 인터페이스의 개발은 물론이고, 키 분배 서버 및 정보보호 서비스 시스템 구성 요소에 관한 여러 관리 기능이 연구되어야 할 것이다.

1. 서론

현재의 컴퓨팅 환경은 전세계적으로 네트웍의 발전과 인터넷의 폭발적 증가로 인해 개방형 분산 시스템 환경으로 가고 있다. 분산 시스템에서는 컴퓨터가 네트워크에 연결되어 있고, 다양한 사용자가 공통적으로 서비스를 이용하기 때문에 고의적이든 아니든 심각한 보안 문제를 야기 시킬 수 있다. 이 문제를 해결하기 위하여 부분적인 시스템과 네트웍에 의존적인 많은 정보보호 기능들이 제시되어왔고 상용화되었지만, 현재와 같은 다양화된 분산 환경에 적합한 정보보호 기능을 수행하는 시스템의 개발은 아직 미약한 상태이다. 따라서 본 논문에서는 인증, 기밀성, 무결성, 접근 제어, 부인 봉쇄 등 정보보호 서비스에서 양단간 안전한 정보 교류에 필수적으로 사용되고 있는 기본적 요소인 세션 키 분배 과정의 경우에 있어서, 분산 환경과 시스템 특성에 의존하지 않으며, 키 분배를 가능하게 해주는 새로운 키 분배 서버를 설계 및 구현하며, 여기서 연구된 키 분배 서버는 사용 환경의 특성에 따라 때로는 통합된 정보보호 서비스의 일부로, 때로는 기존의 정보보호 서비스에 추가되어 또는 이기종간 그리고 서로 다른 서비스간의 공통 기능으로 삽입되어 사용 및 확장이 가능하다. 즉, 본 논문에서 자바 언어를 사용하여 설계하고 구현한 키 분배 서버는 자바의 플랫폼 독립적인 특징으로 인해 이기종 네트웍, 시스템간에도 적용이 가능하며, 객체 기반으로 구현되었으므로, 추후에도 키 분배 서버 기능의 유지, 보수 그리고 확장에 편리한 특성을 가지고 있다.

논문의 구성은 다음과 같다. 2장에서는 현재 사용되고 있는 키 분배 서버가 없는 경우의 세션 키 분배 프로토콜과 키 분배 서버가 있는 경우의 세션 키 분배 프로토콜에 대해서 기술하고, 3장에서는 플랫폼 독립적인 객체 기반의 키 분배 서버 설계에 대해서, 4장에서는 플랫폼 독립적인 객체 기반의 키 분배 서버의 구현에 대해서 기술하며, 5장에서는 본 논문의 결론과 향후 연구 방향에 대해서 기술한다.

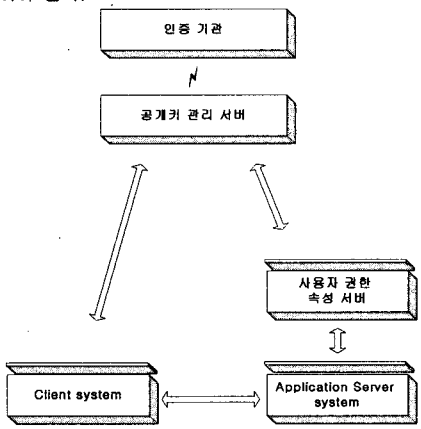
2.키 분배 서버의 유무에 따른 키 분배 과정

정보보호 서비스를 제공하기 위한 키 분배 서버의 일반적인 기능은 기밀 통신을 원하는 양단간 두 사용자에게 안전한 세션을 확립해 주기 위한 키를 분배하는 것이다. 일반적으로 키 분배 서버가 적용되는 경우는 이용 네트웍 환경에 따라, 소규모인 경우에는 안전한 채널을 유지하기 원하는 개개인에 대하여 각각의 세션 키 분배를 통해 암호화된 통신을 유지하도록 해주며, 대규모 네트웍 환경에서의 많은 사용자들에 대한 기밀 통신 서비스를 위해서는 사용자 각각에 대해서 세션키를 분배해야 되는데, 이렇게 되면 다량의 세션 키 관리에 있어 그 효율에 많은 문제를 가지게 되므로 보통은 공개키 암호화 시스템 방식을 혼합 적용하여, 세션 키 분배에 있어서는 공개키 암호화 시스템을 적용하고, 이렇게 분배된 세션 키를 사용하여 실제 기밀을 요구하는 정보를 암호화하는 방법을 사용한다. 그러나 여기에 사용되는 공개키가 누구의 공개인지에 따라서 세션키를 분배하는데 많은 영향을 미칠 수 있다.

정보보호 서비스를 제공하기 위한 키 분배는 두 가지 경우로 나누어 볼 수 있다. 우선 PGP, PEM 같은 응용 서비스에 해당하는 방법으로 키 분배 서버 없이, 각 사용자마다 공개 키 쌍을 적용하여 인증 당국과 사용자간의 직접적인 교류가 이루어지는 것과, Kerberos, SASAME와 같이 통합된 정보보호 서비스 시스템에서 인증 서버 및 권한 속성 관리 서버와 함께 일관된 정보 보호 서비스를 제공하기 위해 키 분배 서버를 이용하는 것이다.

2-1 키 분배 서버가 없는 세션 키 분배 프로토콜

키 분배 서버 없이 세션 키를 분배하는 경우에는 클라이언트와 응용서버 측의 사용자 권한 속성 관리 서버는 둘 다 자신의 비밀키를 가지고 있어야 한다.



[그림 1] 키 분배 서버 없는 세션키 분배 과정

첫 번째 단계 : 클라이언트는 사용자 권한 속성 관리 서버의 인증서를 가져온다.

두 번째 단계 : 인증서에서 사용자 권한 속성 관리 서버가 관리하는 응용 애플리케이션 서버의 리스트를 확인한다.

세 번째 단계 : 인증서에서 사용자 권한 속성 관리 서버의 공개키를 사용해 응용 서버에게 보낼 세션 키 패키지와 사용자 정보를 암호화하고 자신의 비밀키로 서명한다.

네 번째 단계 : 응용 서버는 클라이언트에게서 받은 정보를 사용자 권한 속성 관리 서버에게 보내고, 사용자 권한 속성 서버는 클라이언트의 인증서를 인증서 관리 서버에게서 얻어와, 정당한 사용자인지를 인증 한 후 세션 키 생성 정보를 응용 서버에게 넘겨준다.

마지막 단계 : 응용 서버의 하부 메카니즘은 수신한 세션 키 생성 정보를 이용해 세션 키를 생성하고, 클라이언트와의 기밀 통신을 시작한다.

위의 키 분배 서버가 없이 세션 키가 분배되는 메카니즘은, 세션 키 분배 프로토콜이 간단하긴 하지만, 세션 키 생성을 위해 각 클라이언트 및 응용 서버의 하부 레벨이 복잡해지며, 관리해야 할 공개키의 증가로 공개키 관리 서버의 과부하가 우려된다.

2-2 키 분배 서버를 이용한 세션 키 분배 프로토콜

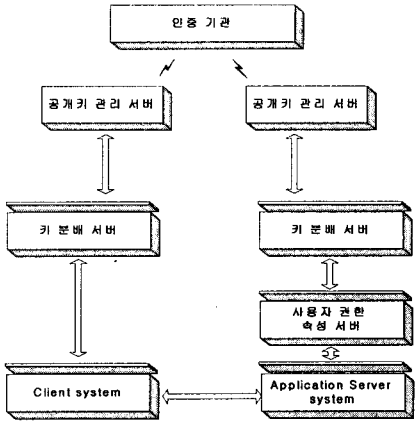
위의 키 분배 서버가 없는 세션 키 분배 과정에서는 공개키 관리 문제와 각 클라이언트 및 응용 서버의 하부 레벨의 복잡성 증가라는 문제가 발생하는데, 키 분배 서버를 사용하는 경우에는 세션 키 분배 과정에 있어 키 분배 서버의 공개키를 이용하므로 클라이언트 각각의 인증서를 통해서 공개키를 이용할 필요성이 없어지고, 클라이언트가 인증서를 받아와 처리하는 등의 공개키를 통한 키 분배 과정이 생략되므로 하부 레벨을 간단하게 구현 할 수 있게 된다.

첫 번째 단계 : 클라이언트는 키 분배 서버에게 분배해야 할 세션 키를 요구

두 번째 단계 : 키 분배 서버는 응용 서버의 서비스를 받기 위한 서비스 티켓과 응용 서버와 클라이언트 사이에서 사용할 세신 키 생성을 위한 세션키 패키지와 그 외 정보 등을 클라이언트에게 넘겨준다.

세 번째 단계 : 클라이언트의 하부 메카니즘은 클라이언트가 키 분배 서버로부터 수신한 정보를 응용 서버와 사용자 권한 속성 서버의 세션키로 암호화해 응용 서버에게 전송한다.

네 번째 단계 : 응용 서버는 수신한 정보를 사용자 권한 속성 서버에게 넘기고, 다시 사용자 권한 속성 서버는 수신한 서비스 티켓을 키 분배 서버에게 넘겨준다.



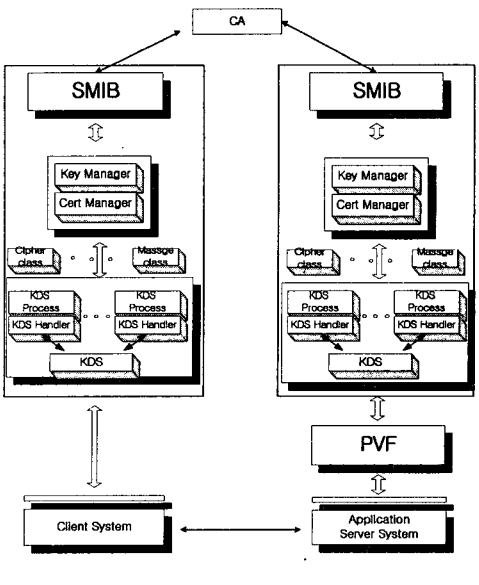
[그림 2] 키 분배 서버가 있는 세션키 분배 과정

다섯 번째 단계 : 키 분배 서버는 서비스티켓이 송신 측 키 분배 서버의 비밀키로 서명 된 것을 확인하고, 그 외 정보를 얻어 낸 후 사용자 권한 속성 서버에게 다시 넘겨준다.

여섯 번째 단계 : 사용자 권한 속성 서버는 실제 서비스를 받을 정당한 사용자인지 인증 한다.

마지막 단계 : 정당한 사용자로 인증이 끝나면, 응용 서버의 하부 메카니즘에서, 사용자 권한 속성 서버로부터 넘겨받은 세션 키 생성 정보를 가지고 세션 키를 생성한 후, 클라이언트와 기밀 통신에 이용한다.

3. 플랫폼 독립적인 객체기반 키 분배 서버의 설계



[그림 3] 키 분배 서버 구성 모듈의 설계

여기서 설계된 키 분배 서버는 객체 기반으로 설계되어, 키 분배 서

버의 기능과 특성에 따라 클래스별로 모듈화 되어 있으며, 실제적으로 자바 언어를 통해 구현되어 있어 플랫폼 독립적으로 작동하는 특징을 가진다. 즉, 객체를 기반으로 한 키 분배 서버가 정보보호 서비스 시스템의 한 구성요소로 적용되었을 때, 추후의 문제점 발생이나 기능의 확장 및 보완에 있어서 해당하는 클래스의 보수 및 필요로 하는 기능의 클래스만을 새로 추가하면 된다는, 쉽고 간편한 유지, 보수 성질을 지니고 있으며, 익히 알려져 있듯이 자바 가상 머신이 운영된다면 어떤 시스템에 상관없이 적용이 가능하다는 특징을 띤다. 다음에는 플랫폼 독립적인 객체 기반 키 분배 서버를 구성하는 각 클래스들의 기능과 특징에 따른 설계에 대해서 알아보겠다.

KDS 클래스 : 키 분배 서버의 기본 기능인 서버로서의 기능을 구현한 클래스로서, 멀티 쓰레드 서버로 구현된 클래스

KDS Handler 클래스 : 키 분배 서버에 접속을 원하는 클라이언트의 각 메시지 종류에 따른 실제 기능을 수행할 수 있도록 KDS 클래스와 KDS Process 클래스 사이에서 중계자 역할을 하는 클래스

KDS Process 클래스 : 클라이언트가 요청한 기능을 실제적으로 수행하는 클래스

Cert Manager 클래스 : 키 분배 서버에서 필요로 하는 인증서에 대한 요청을 받아들여, SMIB에서 실제로 인증서를 가져오는 등의 중간 기능을 구현한 클래스

Key Manager 클래스 : 키 분배 서버에서 사용하는 공개키, 개인키 및 세션키에 대한 요청을 받아들이고 SMIB로부터 실제로 해당 키를 얻어 오는 등의 중간 기능을 구현한 클래스

Message 클래스 : 키 분배 서버와 통신을 주고받는 해당 객체 사이의 모든 메시지를 정의해놓은 추상 클래스

Cipher 클래스 : 키 분배 서버 각 구성 모듈에서 일어나는 암호화/복호화에 관한 기능을 지원하는 클래스(자바 API 이용)

이 밖에도 각 메시지의 종류에 따라 메시지를 정의하고 있는 클래스들이 개별적으로 설계되었다.

4. 플랫폼 독립적인 객체기반 키 분배 서버의 구현

본 논문에서는 자바를 통해 키 분배 서버를 새롭게 구현하였는데, 구현상의 특징으로는 키 분배 서버의 각 구성 요소가 그 기능 및 특성에 의해 모듈화된 클래스로 구현되었다는 것과 클래스간, 즉 키 분배 서버 각 구성 요소의 통신은 일정한 포맷을 가진 메시지 기반으로 이루어진다는 것이다. 아래에서는 통합 정보 보호 서비스 시스템 내에서 일어나는 키 분배 과정을 통해서 구현된 키 분배 서버를 살펴보겠다.

기본 단계 :
클라이언트는 통합 정보보호 시스템의 인증 서버에 정당한 사용자 확인을 요청한다. 그러면 인증 서버는 사용자 인증 후, 사용자의 권한 속성 티켓을 발부하여, 권한 속성 서버에 접근을 허락하며, 권한 속성 서버는 적절한 절차 후에 키 분배 서버에서 세션키 생성 정보를 포함한 응용 서버에 접근하여 서비스를 받을 수 있는 티켓을 얻도록 키 분배 서버 티켓을 발부한다.

첫 번째 단계 :
KDS 클래스는 클라이언트의 세션키 요청(서비스 티켓 요청)을 받아들인다. 즉, KDSHandler 클래스에서 클라이언트의 SAMtoKDS_SERVICEINFO_REQUEST 메시지를 수신한다.

두 번째 단계 :
KDS Handler 클래스에서 SAMtoKDS_SERVICEINFO_REQUEST 메시지에 대해 실제적인 수행이 발생하는 KDS Process 클래스의 Parse_SAMtoKDS_Packet() 메소드를 호출한다.

세 번째 단계 :
KDS Process 클래스의 Parse_SAMtoKDS_Packet() 메소드에서, 수신한 메시지에 대한 KDS 티켓을 복호화하여 알아낸 정보를 이용하여 소스 ID를 검증한다. 여기서 복호화는 MySealedObject 클래스와 Cipher 클래스를 사용한다.

네 번째 단계 :
소스 ID가 검증된 다음에는 기본적인 서비스 티켓 생성을 위하여, target ID, user ID, 클라이언트 하부 메카니즘과 응용 서버 측의 권한 속성 검증 서버의 세션키, 권한 속성 인증서 등의 정보를 SMIB에서 얻어 온, 두 도메인 간 키 분배 서버의 세션키로 암호화한다.
암호화는 MySealedObject라는 클래스를 사용해 DES ECB 모드를 사용한다.
위에서 암호화된 정보와 양단간 키 분배 서버의 세션키를 응용 서버 측의 공개키를 가지고 암호화하고 다시 클라이언트 측의 키 분배 서버의 비밀키로 서명한 후, 세션키 생성 정보와 함께 클라이언트의 하부 메카니즘에 KDSAtoSAMA_SERVICEINFO_RESPONSE 메시지를 통하여 넘겨준다.

다섯 번째 단계 : 클라이언트의 하부 메카니즘은 응용 서버측의 하부 메카니즘에게 GSS-API를 사용해 토큰 형태(ClienttoServer_GSS_TOKEN)로 서비스 요청(세션 키 생성 정보 포함)정보를 보낸다.

[여기까지는 클라이언트 측의 키 분배 서버에서 일어나는 과정임]

여섯 번째 단계 : 응용 서버 측의 하부 메카니즘은 클라이언트 쪽에서 받은 토큰을 권한 속성 검증 서버에게 보내고 (SAMtoPVF_GSS_TOKEN), 권한 속성 검증 서버는 토큰의 정보 중, 클라이언트 쪽의 키 분배 서버의 비밀키로 서명된 것을 증명하기 위해서 응용 서버 측의 키 분배 서버에게 보내(PVFtoKDS_ServiceTicket) 증명을 요청한다..

일곱 번째 단계 : 키 분배 서버에서는 클라이언트 측 키 분배 서버의 비밀키로 서명 된 것을 검증하고, 응용 서버 측의 공개키로 암호화된 양단간 키 분배 서버의 세션 키를 복호화 한다. 복호화 된 세션 키를 이용해 티켓 내에 들어있는 사용자 식별자와 응용 서버의 식별자가 사용자 권한 속성 검증기가 처리할 수 있는 응용 서버인가를 검증한 후 사용자 권한 속성 검증 서버에게 사용자의 권한 속성을 검증하도록 요청한다.(KDStoPVF_RESPONSE 메시지를 보낸다.)

여덟 번째 단계 : 사용자 권한 속성 서버는 사용자가 응용 서버에 서비스 제공에 정당한 권한을 가진 사용자인지를 검증한 후, 세션 키 생성에 필요한 정보를 응용 서버의 하부 메카니즘에게 보낸다.(PVFtoSAM_RESPONSE 메시지를 보낸다.)

마지막 단계 : 응용 서버의 하부 메카니즘은 사용자 권한 속성 검증 서버로부터 받은 정보를 가지고 기밀성, 무결성이 보장되는 세션 키를 도출한다.

5. 결론

현재의 인터넷 환경에서 무엇보다도 보안은 중요한 요소로 자리잡고 있다. 또한 서로 상이한 여러 네트웍과 시스템 사이에서의 이식성도 빼놓을 수 없는 문제이다. 따라서 본 논문에서 설계, 구현된 자바 기반의 키 분배 서버는 플랫폼 독립적으로 구현되어 어떠한 환경에서도 사용이 가능하고, 객체 기반의 클래스로 설계되어 확장 및 유지 보수가 매우 용이하므로 현재 폭발적으로 증가하는 전자 상거래, 사용자 인증, 전자 공증 그리고 신용 인증 등 여러 가지 정보보호 서비스에 쉽게 응용 및 적용이 가능하다. 향후 연구되어야 할 과제로는 키 분배 서버와 정보보호 서비스 시스템간의 간편한 연계를 위한 인터페이스 부분의 개발과 키 분배 서버 및 정보보호 서비스 시스템 자체의 효율적인 관리를 위한 기법이 필요하다고 생각된다.

6. 참고문헌

[1] 김동규 외, "최적의 공개 키 요소를 사용한 세션키 분배 서버 및 프로토콜의 설계", 한국정보과학회지, 1997.12
[2] 김동규 외, 분산 통신망환경 통합 정보보호 소프트웨어 기술, 1차년도 보고서, 1997.1
[3] 김동규 외, 분산 통신망환경 통합 정보보호 소프트웨어 기술, 2차년도 보고서, 1998.1
[4] 김동규 외, 분산 통신망환경 통합 정보보호 소프트웨어 기술, 3차년도 보고서, 1999.1
[5] Jonathan Knudsen, "Java Cryptography", O'Relly, 1999
[6] Scott Oaks, "Java Security", O'REILLY, 1999
[7] William Stallings, "Network Security Essentials", Prentice Hall, 2000