논문 20-45-06-09 The Journal of Korean Institute of Communications and Information Sciences '20-06 Vol.45 No.06
https://doi.org/10.7840/kics.2020.45.6.1004

고유한 코드 함수 실행 결과를 이용한 스마트폰의 적법 소지 인증

박 준 철◆°

Authenticating Legal Possession of a Smartphone Using the Execution Result of Its Unique Code Function

Jun-Cheol Park◆°

요 약

추가 인증 또는 패스워드 재설정 등의 상황에서 사용되는 SMS 기반 인증 및 이를 개선한 방식들은 인증 서버의 저장 내용이 공격자에 유출되는 경우, 사용자 측의 보안상 잘못이 없더라도 공격자에 의한 사용자 위장이 가능하게 만든다. 이에 사용자의 스마트폰이나 통신 과정을 통한 정보 유출 공격 뿐 아니라, 서버 자체에 대한 공격에도 안전성을 보장할 수 있는 인증 기법이 필요하다. 본 논문에서는 스마트폰별로 고유한 코드 함수의 실행 결과를 이용하여, 사용자가 스마트폰을 적법하게 소지하고 있음을 강력히 입증하는 기법을 제시한다. 제안 기법은 스마트폰을 활용하여 기존 SMS 기반 인증을 대체하는 훨씬 더 안전한 인증이 가능하게 한다. 이 기법은 코드 실행 결과를 검증하는 별도의 서버를 두어, 어떤 서버든 단독 서버의 정보만으로는 사용자 인증을 받기에 불충분하도록 정교하게 서버 간의 정보 및 역할을 분리함으로써, 강한 보안성을 제공한다. 사용자는 모바일 환경에서의 앱 설치, 구동 및 개인 정보 입력 등의 익숙한 과정을 통해 SMS 인증코드와 비교하여 제안 기법이 제공하는 향상된 보안성을 누릴 수 있다.

Key Words : Authentication, Code Segment, Smartphone Possession, Server Compromised, Security

ABSTRACT

SMS-based authentication and improved methods for supplementary authentication or resetting forgotten passwords enable user impersonation by the attacker if the contents of the authentication server are leaked to the attacker, even if there is no user error in security. Hence an authentication method is required to ensure security not only for information leakage through a user's smartphone or communication process but also for attacks on the server itself. We propose a scheme strongly demonstrates that a user possesses a smartphone legally using the execution result of a unique code function in the phone. The scheme enables much more secure authentication replacing the existing SMS-based authentication by using a smartphone. It assumes a separate server that verifies the results of code execution to provide strong security by separating the information and roles between servers so that any server's information alone is not sufficient for authenticating users. Users can enjoy the improved security provided by the proposed scheme compared to the SMS authentication code through familiar processes such as installing the application in the mobile environment, running, and entering personal information.

◆° First and Corresponding Author : Dept. of Computer Engineering, Hongik University, jcpark@hongik.ac.kr, 정교수, 종신회원
논문번호 : 202001-004-B-RN, Received January 8, 2020; Revised March 16, 2020; Accepted March 27, 2020

Ⅰ. 서 론

사용자 인증을 위하여 전통적인 패스워드 기반으로부터 지문, 홍채 등의 생체인식 기법까지 다양한 기술들이 개발되어 활용되고 있는데, 최근에는 한 가지 인증 방식만을 채택하지 않고 추가의 인증 수단을 더해서 인증의 보안성을 향상시키려는 시도, 즉 두 가지 수단 인증(two-factor authentication, 이하 2FA)[1-5]이 널리 사용되고 있다. 추가 인증 수단으로는 휴대폰의 문자 메시지(Short Message Service, 이하 SMS)를 통하여 인증코드를 보내는 방식이 가장 보편적이다. 비단 추가적 인증의 목적 뿐 아니라, SMS 기반 인증은 패스워드를 분실한 경우 같이 주된 인증 수단이 사용불능일 때, 이를 복원하기 위한 긴급 상황의 인증 수단으로도 널리 활용되고 있다. 이에 SMS 기반 인증의 안전성을 높이는 여러 개선 방안들이 제시된 바 있다. 하지만 SMS 기반 인증이나 이의 개선 방안으로 제시된 것들은 인증 받으려는 사용자가 인증 서버에 제시하는 내용이 서버의 저장 내용을 통하여 검증이 가능하다는 공통점을 가진다. 이는 곧 서버에 저장된 정보가 공격자에게 유출된다면, 취득한 정보로부터 공격자가 서버에 등록된 사용자로 위장(impersonation)할 수 있음을 의미한다. 서버를 유지 관리하는 대다수 주체는 보안 의식이 높고 서버 보호를 위해 다양한 조치들을 강구하고 있지만, 그럼에도 불구하고 잘 알려진 회사나 기관의 서버들을 포함하여 여러 사이트들의 서버가 해킹 당하여 저장 내용이 공격자 손에 들어갔다는 소식은 흔하게 들려오고 있다. 따라서 사용자 측면의 보호를 제공함은 물론, 서버 측 저장 내용의 유출이 발생했을 때도 보안성이 여전히 담보되는 인증 방식이 반드시 필요하다.

2FA를 위한 SMS 기반 방식을 포함하여 기존의 챌린지-응답 형태의 인증 방식들은 응답이 챌린지와 동일한 경우와 챌린지를 비밀 값 등을 이용, 변형 후 응답으로 제공하는 경우로 구분될 수 있지만, 인증 서버가 저장하고 있는 내용만으로 응답의 올바름을 검증한다는 점에서는 동일하다. 본 논문은 스마트폰에 폰별로 고유한 코드 함수를 내장시키고, 이의 실행 결과를 활용하여 스마트폰의 적법한 소지를 강력하게 인증하는 기법을 제안한다. 이 기법의 채택 시 서버의 저장 정보가 공격자에 유출되어도 공격자에 의한 사용자 위장이나 사용자 비밀 정보의 탈취를 막을 수 있어, 사용자의 잘못 없이 사용자에게 피해가 발생하는 것을 방지할 수 있다.

본 논문은 다음과 같이 구성된다. 2장에서 2FA를 위한 SMS 기반 인증과 이를 개선한 방안들을 살펴보고, 이러한 방식들을 보안성 측면을 중심으로 제안 기법과 비교해 살펴본다. 3장에서는 스마트폰에 일종의 함수인 코드 함수를 포함하는 인증 앱을 설치하고 이를 통해 스마트폰의 적법 소지를 확증하는 기법을 상세히 설명한다. 이어서 4장에서는 가능한 공격 상황별로 제안 기법의 상세한 보안성 분석을 제시하며, 마지막 5장에서 결론을 맺는다.

Ⅱ. 배경 및 관련연구

2.1 2FA에서 SMS 기반 인증의 보안 취약점

악성코드를 통해 기밀성 및 프라이버시를 위협하는 공격이 계속 증가하고 있어[1], 전통적인 ID/패스 워드 기반 인증은 온라인 뱅킹 등의 주요 응용에는 충분한 보안성을 제공하지 못한다. 이 때문에 SMS 인증코드(보통 6자리 숫자)를 추가한 2FA가 많이 사용되는데, 이는 대개 사용자가 어떤 웹 사이트에 ID 및 패스워드를 입력한 후, 등록된 자신의 휴대폰 번호로 수신한 SMS 일회용 인증코드를 해당 사이트에 사용자가 직접 입력하는 식으로 진행된다. 사이트의 인증 서버는 입력받은 숫자가 전송된 숫자와 일치하는지 확인한 후, 일치하면 상대방을 등록된 사용자로 인증하게 된다. 그 외에도 이와 같은 SMS 인증코드는 패스워드를 분실하거나 한 경우 패스워드 재설정을 위해서 널리 사용된다. 따라서 이러한 유형의 2FA가 보안상 취약하다면, 공격자가 피해자로 위장하여 중요한 트랜잭션을 실행시킨다거나, 또는 심지어 피해자의 패스워드를 공격자가 원하는 값으로 바꿔버릴 수도 있게 된다.

SMS 인증코드는 공격자가 휴대폰의 소유자 몰래 인증 상대방에게 제시할 때 문제가 되는데, 이러한 공격의 방식은 세 가지 유형으로 구분된다. (1) 스마트폰에 설치된 악성코드를 통하여 수신한 SMS 문자 내용이 공격자에게 유출될 수 있는데, 실제로 안드로이드 계열 스마트폰에서 여러 부류의 악성코드들[6-8]이 SMS 탈취에 사용되는 것으로 보고되었다. 안드로이드 버전 4.4 이후의 폰의 악성코드는 브로드캐스트 수신자 등록을 통하여 SMS 텍스트를 내부적으로 수신할 수 있으며, SEND_SMS 권한이 있는 경우 수신한 SMS 인증코드를 다른 SMS를 통해 전송할 수 있고, 또는 INTERNET 권한이 있는 경우 HTTP 등을 사용해 네트워크 인터페이스를 통해 외부로 SMS 인증코드를 보낼 수 있다. 애초, 스마트폰에 악성코드가 설치되는 주된 경로로는 SMS나 이메일 피싱(phishing), 스팸(spam) 등[9,10]으로 유인하여 공격자가 설정한 가

짜 사이트나 링크를 사용자가 클릭하는 것을 들 수 있다. 그밖에 Cross Platform Infection 공격[1]은 감염된 PC가 공격 대상인 스마트폰과 같은 LAN 상에 위치할 때, 이 PC가 폰에게 자신을 이 LAN의 외부와 통로 역할을 하는 게이트웨이(gateway)라고 속임으로써 시작된다. 폰이 이 PC를 게이트웨이로 보는 랜 구성을 받아들이고 IP 주소를 요구할 때, 이 PC는 적절한 IP 주소와 함께 자신에게 할당된 IP 주소를 게이트웨이 IP 주소인 것처럼 폰에게 전달한다. 이후 폰의 외부 트래픽은 모두 이 PC를 거치게 되므로, 폰의 모바일 브라우저가 외부 사이트에 접근할 때 공격자는 이 PC를 통해 가짜 웹 사이트가 폰을 상대하도록 만들 수 있다. 결국 폰은 공격자가 준비한 사이트에 접근하게 되며, 이를 통해 공격자는 스마트폰에 원하는 악성코드를 심을 기회를 얻게 된다. 예외적으로, 자녀 등의 스마트폰 사용을 감시하고자 부모가 자녀의 폰에 NEXSPY[11] 같은 상용 앱을 설치하여 모든 SMS 메시지 내용을 확인할 수 있는데, 이런 스파이웨어를 공격자가 피해자 폰에 몰래 설치하여 SMS를 빼내는 것도 가능하다. (2) 사회공학(social engineering)에 의한 기법으로 SMS 내용을 공격자가 획득[2,12]할 수 있는데, 예를 들면, 공격자가 피해자의 이메일 주소 및 폰 번호를 알고 피해자에게 SMS 메시지를 보내는데, 메시지 내용은, '나는 당신의 이메일 주소 관리자인데 당신 메일 계정에 공격으로 의심되는 로그온 시도가 발견되었다. 이 시도로부터 계정을 안전하게 지키기 위해 당신의 등록된 폰 번호로 소유자 확인용 코드를 보낼 것이니 이 코드를 받으면 코드 그대로를 입력해 SMS 답장을 달라'와 같은 식이다. 공격자는 이미 서비스 제공자에게 해당 피해자의 계정 복귀를 위한 코드를 요청한 상태로, 이 피해자가 수신하는 코드를 공격자에게 SMS로 알리면, 공격자는 이 코드 내용을 사이트에 그대로 입력함으로써 피해자의 계정을 소유할 수 있게 된다. 피해자는 수신한 코드를 사이트에 입력하지 않고 SMS 답장으로 보내는 것이 이상함을 알아채지 못했다면 그대로 피해를 볼 것이다. (3) 최근 심각한 문제가 되는 공격은 SIM Swapping이라 불리는 기법[13,14]인데, 피해자의 폰 번호를 현재 사용 중인 폰에서 공격자가 제어하는 폰으로 이동시키는 것을 말한다. 그 결과 공격자의 폰(대부분 경우 대포 폰)에서 피해자 번호로 전송된 SMS 수신이 가능해지기 때문에, SMS 인증코드는 피해자 폰에 악성코드 유입을 허용한다든지 하는 피해자 잘못 없이도 공격자 손에 들어간다. SIM Swapping을 실현하는 전형적인 방법은, 통신사 고객센터에 공격자가 전화하여 자신을 피해자라 주장하며 USIM 카드를 분실했으니 새로운 USIM으로 자신의 정보를 이전하고 싶다 요청하는 것이다. 이를 위해 물론 피해자의 개인 정보를 공격자가 파악하는 것이 필요하고 사회공학 기반의 방법 등이 사용될 수 있다. SIM Swapping은 성공하면, 공격자의 대포 폰으로 피해자에게 오는 모든 SMS를 받을 수 있기 때문에, 공격자는 피해자 계정 패스워드 변경 등을 성공적으로 이뤄낼 수 있다. 이런 공격에 대해 무지한 고객센터 직원은 별 의심 없이 SIM 변경을 허가할 수 있으며, 그렇지 않은 경우라도 직원을 매수하거나, 또는 직원 스스로가 공격자가 된다면 SIM 교체를 진행하는 것이 가능하다. 실제 수백만 불 가치의 가상화폐가 이 방식으로 도난당했으며[14], 트위터(Twitter)의 CEO도 SIM Swapping 공격을 당해 자신이 보내지도 않은 트윗으로 곤욕을 치른 사례[13]가 알려졌다.

상기의 취약점 및 공격 유형들이 보여주듯 SMS 메시지를 활용한 2FA는 안전하지 않으며, 그 결과, 미국 NIST는 정부 제공 서비스의 SMS 기반 인증 사용 금지를 권고하였고[15], 국내의 경우 또한 은행들은 SMS 기반 인증을 사용하지 않게 되었다[16].

2.2 2FA용 개선된 추가 인증 기법들의 한계

보안성 향상을 위한 세 가지 유형의 접근 방식을 소개한다. (1) 우선, SMS 메시지 자체의 안전도를 높이려는 시도를 들 수 있다. 이동통신망 상에서 SMS 메시지의 기밀성(confidentiality) 보장을 위해 Blowfish 대칭키 알고리즘으로 메시지를 암호화 후 전송, 수신지에서 복호화하는 방안이 제시[3]되었는데, 송수신자간의 분배된 암/복호화 키의 외부 접근을 막는 방법은 고려되지 않았다. 유사 연구로, 암호화 기법으로 메시지 기밀성 보장 뿐 아니라 송수신자 인증도 제공하려는 방법[17]은 관련되는 폰들의 신원 확인을 제공하는 별도 서버를 두어, 이 서버가 키 관련 모든 정보를 저장하게 하고 있다. 따라서 서버의 저장 정보가 노출되면 암호화가 깨져버리는 취약점을 가진다. 다른 시도로, 폰에 도착한 SMS 메시지가 임의로 유출되는 것을 방지하는 기법[18]을 들 수 있다. 안드로이드 폰에 메시징 관련 부분을 수정하여, SMS 메시지가 도착하면 바로 메시지에 오염(taint) 택(tag)을 부착하고, 이 메시지가 폰을 떠나는 순간까지 임의로 택이 제거되지 못하도록 제어한다. 그럼 메시지가 다른 SMS 메시지나 HTTP 등을 통해 폰에서 전송되려 할 때, 이 택에 기술된 보안 정책이 강제 실현되게 함으로써 메시지가 불법 유출되는 것을 막는 것이다. 제안 기법은 SMS 사용을 인증 앱을 통한 방식으로 대체하

려는 시도이기에 이들 방식과는 접근 방향부터 차이가 있다. (2) Authy나 Google Authenticator(Google OTP) 등은 SMS 인증코드 대신 앱을 통해 인증코드를 주고받으려는 것으로, 그 통신 과정에서 SIM을 개입시키지 않음으로 SIM Swapping 공격으로부터 자유롭다. 제안 기법 역시 이들과 같이 앱을 통하여 SMS 인증코드의 대체를 목적으로 하지만, 수신 내용을 그대로 응답으로 전달하지 않고 스마트폰 내부의 비밀 값과 서버의 챌린지에 따라 응답이 예측 불가능하게 달라진다는 점이 다르다. (3) 응답으로 보낼 인증코드가 수신한 값과 다르게 하여, 단순 유출에 따른 도용을 무력하게 만드는 방식들[19,5,4]이 발표되었다. 이들은 스마트폰 내부의 값이나 GPS 위치 등을 응답에 포함시키거나 사용자가 수신된 링크를 클릭하게 하는 등, 응답을 서버가 보낸 챌린지와 다르게 만드는 시도를 한다. Kwon 등의 방식[19]의 경우, 제안 기법은 스마트폰 USIM에 저장된 비밀 값을 응답 생성에 사용하는 아이디어를 이 방식으로부터 채용했지만, 서버로부터의 챌린지와 비밀 값 및 위치 정보만으로 응답을 생성하므로, 서버 저장 내용으로부터 인증에 필요한 모든 정보를 얻을 수 있다는 한계를 가진다. 반면 제안 기법은 서버 측 저장 정보의 공격에 대비한 보안성이 강화되었다. Abdurrahman 등의 방식[5]는 스마트폰이 서버와의 사전 공유 값(pre-shared value), GPS 위치 및 현재 타임스탬프를 사용해 응답을 만들고 서버에 보내 인증을 요청하면서, GPS 위치와 타임스탬프를 별도 GPS 서버에 보낸다. 인증 서버는 GPS 서버에게 사용자의 관련 정보를 받아서, 스마트폰으로부터의 응답을 검증하는 과정을 거친다. 인증 서버 외에 별도의 서버를 둔다는 점에서 제안 기법과 공통점을 가지나, GPS 위치 및 타임스탬프는 사실상 비밀 값이 아니라는 한계를 가진다. 즉 공격자가 피해자 스마트폰 취득 시 적법한 GPS 위치 및 타임스탬프 값을 스스로 만들어 낼 수 있다. 따라서 사전 공유 값이 유일하게 공격으로부터의 방어 수단이라 볼 수 있는데, 사전 공유 값조차 스마트폰 내에서 안전하게 보호하는 수단을 제공하지 못한다. Lee 등의 방식[4]는 사용자가 웹 사이트에서 인증 받은 후 2차 인증을 받기 위해, 서버가 모바일 메신저로 보낸 인증 URI 링크를 클릭함으로써 인증을 완료하게 한다. 링크가 챌린지 역할을 하고 링크 클릭 결과가 응답 역할을 하는 셈인데, 전송 중 링크를 보호하려면 암호화 등 부담이 요구된다. 또한 서버는 응답을 검증하기 위해 보낸 링크를 유지해야 하므로 서버의 저장 정보가 공격자에 유출된다면 공격자에 의한 사용자 위장이 가능하다는 취약점을 가진다. 제안 기법은 USIM 내의 비밀 값 뿐 아니라 스마트폰의 코드 함수 역시 외부로부터 보호하며, 서버의 저장 내용의 유출에도 대비하여 저장 정보 및 전송 내용을 정교하게 설계하였다는 점에서 기존 연구들과 차별된다.

비교 정리하면, 널리 사용되는 SMS 인증코드 기반의 2FA나 이를 개선한 2FA 방식들은 서버의 저장 정보 노출 시 대비가 없다는 것 등의 보안 취약점이 존재하나, 제안 기법은 이러한 알려진 공격에 대한 보안 취약점이 존재하지 않도록 설계된 인증 방식으로 2FA의 기존 SMS 인증코드 및 다른 수단을 안전하게 대체할 수 있다. 이를 위해 제안 기법이 의존하는 스마트폰의 적법 소지 인증은, 인증 서버 입장에서 상대방이 등록된 스마트폰의 적법한 소유자로서 해당 스마트폰을 현재 소지하고 있음을 확인하는 것으로 상대방이 단순히 SMS 문자를 정확히 수신했다는 확인보다 훨씬 강력하게 상대방이 스마트폰 적법 소유자일 수밖에 없음을 증빙한다.

Ⅲ. 코드 세그먼트 실행 결과를 통한 인증

3G 단말기부터 적용되는 심(SIM) 카드를 내장한 스마트폰 소유자를 대상으로, 스마트폰에 탑재된 특정 코드 함수의 실행 결과를 통하여 스마트폰의 실제 소지 및 소유권을 확인하게 하는 인증 기법을 제안한다. 2018년 8월 기준 95%인 한국의 스마트폰 보급률 및 주요국의 보급률 상승 추세를 볼 때, 스마트폰 소유자로 국한되는 제안 기법의 적용 범위에 큰 제한이 있다고 보기는 어렵다. 제안 기법은 코드 함수 실행 결과가 옳은지 확인하는 코드 함수 관리 서버(Code Functions Management Server, 이하 *CMS*라 함)와 스마트폰 및 *CMS*로부터의 정보를 취합하여 실제 인증 여부를 판단하는 인증 서버(Authentication Server, 이하 *AuthS*라 함)가 소지자의 입력을 받은 스마트폰과 연계하여 동작하는 형태를 따른다. 여기서 *CMS*는 한 번의 스마트폰 앱 다운로드 및 등록을 거치면 다수의 *AuthS*(스마트폰 이용한 인증을 요구하는 사이트)들에게 해당 스마트폰의 코드 함수 실행 결과를 활용한 사용자 인증 서비스를 제공할 수 있다. *CMS* 서버는 다수의 인증 서버들에게 동일한 서비스를 제공하기에, 인증 서버별 또는 사이트별로 별개의 *CMS* 서버를 둘 필요가 없다. 따라서 제안 기법을 채택하는 사이트들의 수가 증가하더라도 *CMS* 서버의 수가 선형적으로 증가할 필요가 없으며, *CMS* 서버는 부하 분산 및 서비스 가용성을 고려하여 적절한 수준에서

다중화 배치될 수 있다. 이하 사용하는 표기법은 다음과 같다. $h(M)$은 SHA-256 해시 함수를 메시지 M에 적용한 결과이며, $E(S,K)$(또는 $D(S,K)$)는 256-bit 키 K를 사용하여 S를 AES-256 대칭키 암호화(또는 복호화)한 결과이고, ‖은 연접(concatenation) 연산자를 나타낸다.

3.1 인증 어플리케이션 다운로드 및 설치과정

사용자는 Google Play 같은 온라인 앱 스토어에서 제안 기법을 구현한 인증 앱(이하 앱Q)을 다운로드받은 후 그림 1과 같은 설치 과정을 진행한다. 사용자의 스마트폰에서 앱Q를 첫 실행 시키면 앱Q는 *CMS*에 접근하여 앱Q에서 관리할 코드 함수(입력을 받아 결과를 반환하는 일종의 함수)를 받아 설치한다. 코드 함수 C는 스마트폰마다 서로 다르게, 비밀리에 설치되기 때문에, USIM에 저장된 비밀 값에그림 1. 앱Q의 설치 파트 1: 랜덤으로 선택된 코드 함수 C를 포함 연계된 입력이 주어질 때 그 실행 결과는 사실상 스마트폰의 적법한 소지를 강력하게 인증하는 징표가 된다. *CMS*가 스마트폰마다 서로 다른 코드 함수를 부여하는 방식은 3.2절에서 상세히 설명한다. 코드 함수 C를 수신한 스마트폰은 C가 정상적으로 설치되었는지를 *CMS*에 확인시키기 위해 *CMS*가 제시한 입력 값 IN에 대하여 C를 실행한 결과 R을 해시 함수 결과 $h(C)$와 함께 *CMS*에 전송한다. 아울러 사용자는 긴급 상황에서 제안 방식에 따른 인증 시도를 막기 위해, 등록 정보의 무효화 요청의 목적으로만 사용하려 만든 이메일 주소도 *CMS*에 보낸다. 등록 정보 무효화 요청의 세부 내용은 3.4절에서 제시한다.

스마트폰에 앱Q가 설치된 이후, 제안 인증 서비스를 지원하는 웹사이트의 인증 서버를, 스마트폰의 USIM 내에 최초 등록하는 과정이 그림 2와 같이 진행된다. 이 인증 서버(그림 2의 서버 X_i)는 이미 해당 사용자 및 사용자 소유 스마트폰의 정보를 포함한 계

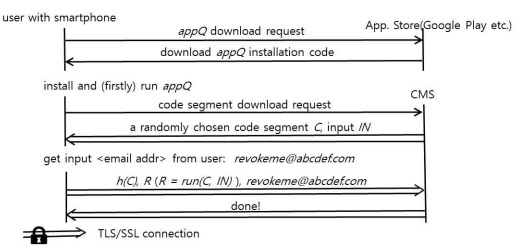

그림 1. 앱Q의 설치 파트 1: 랜덤으로 선택된 코드 함수 C를 포함

Fig. 1. Installation of appQ Part 1: embedding a randomly chosen code function C

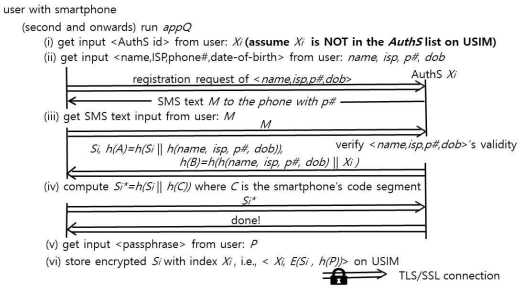

그림 2. 앱Q의 설치 파트 2: 인증 서버 X_i에 개인 정보 제공 및 비밀 값 S_i 공유

Fig. 2. Installation of appQ Part 2: Giving AuthS X_i Personal Info. and Sharing a Secret S_i

정을 가지고 있다고 가정한다. 사용자는 폰 번호를 포함한 개인 정보를 보내고, 서버 X_i는 회신된 SMS 메시지 값의 일치를 통해 해당 스마트폰을 현재 소지하고 있는 자가 소유주임을 확인한다. 이후 서버는 이 스마트폰과 공유할 비밀 값 S_i를 정한 후, 수신된 개인 정보로부터 각각 $h(A) = h(S_i \| h(name,isp,p\#,dob))$, $h(B) = h(h(name,isp,p\#,dob) \| X_i)$를 계산 후 저장한다. $h(A)$는 추후 상대방에서 이 값을 알고 있음을 입증하면 입증한 자가 스마트폰의 적법 소지를 인증 받는데 활용되는 값으로, USIM에 접근하여 복호화를 진행할 수 있었음을 보여준다. 한편 $h(B)$는 고정 값으로, 수신자인 서버 X_i가 $h(B)$에 매치되는 $h(A)$ 및 이에 연계된 사용자 정보를 추출할 수 있도록 한다. 스마트폰은 서버 X_i로부터 받은 비밀 값 S_i를 자신의 코드 함수 C에 대한 $h(C)$와 연접하여 해시 함수 적용한 $S_i^* = h(S_i \| h(C))$를 서버 X_i에게 보낸다. 향후 S_i^* 값은 서버 X_i의 챌린지(challenge) 값과 연접되어 해시 함수 적용 후 스마트폰의 코드 함수 실행 시의 입력 값으로 사용된다. 그리하여 서버 X_i에 도착한 인증 요청이, 이 스마트폰의 등록된 코드 함수 C에 의해서 적절한 입력에 대하여 계산된 값인지 검증하는데 사용된다. 이제 사용자는 비밀 값 S_i를, 입력한 패스프레이즈(passphrase)의 해시 값을 키로 사용하여 암호화시킨 후, 스마트폰의 USIM에 상대 인증 서버 ID X_i(사이트 URI)와 함께 저장한다.

3.2 서로 다른 코드 함수들 생성과 관리

3.2.1 템플릿(template)을 활용한 서로 다른 코드 함수들의 생성

코드 함수를 탑재한 앱을 설치한 스마트폰이, 매번 새로운 값을 입력으로 하여 이 코드 함수를 실행시킨 결과를 올바르게 제시할 수 있는지가 스마트폰의 주인에 의한 적법한 소지를 검증하는데 사용된다. 따라서 스마트폰마다 코드 함수가 달라야 하고, 탑재된 코드 함수는 오직 랜덤 생성 및 설치에 관여한 *CMS* 만이 그 내용을 알아야 한다. 코드 함수는 하나의 입력과 하나의 출력(반환 값)을 가지는 일종의 함수라고 볼 수 있다. 다수의 스마트폰에 서로 다른 코드 함수들을 제공하기 위해 *CMS*는 그림 3의 사례와 같은 형태의 함수 생성 템플릿을 활용한다.

그림 3의 템플릿에서 대괄호({})로 묶인 부분은 각각 그 중 하나만을 선택하여 전체 코드 함수를 완성한다. 전체 대괄호의 수를 n이라 하고, 각 대괄호 안의 선택지의 수를 k(그림 3은 $k=4$인 경우)라 하면, 전체 가능한 코드 함수의 수는 $k \times k \times \cdots \times k = k^n$ 이 되는데, $k=4$이고 $n=15$라 한다면, $k^n = 4^{15} = 2^{30} > 10^9$이 되므로, 10억대 이상의 스마트폰들을 수용할 수 있게 된다. 실제 코드 함수를 배정할 때, *CMS*는 그림 3에 표시된 것처럼, 배정 시마다 랜덤으로 만들어진 데드 코드(dead code)들을 중간 중간에 포함시킨다. 데드 코드는 실행이 되지 않는 코드(예를 들면, if 1 < 0 then print('hello'))이다. 이런 랜덤하게 만들어진 코드들을 임의로 여러 위치에 삽입함으로써, 코드의 실행에는 전혀 영향을 미치지 않으면서 코드 함수 전체의 해시 값을 구할 때 이런 데드 코드들이 반영되어 예측할 수 없는 해시 결과 값을 가지도록 한다. 따라서 설사 *CMS*가 사용하는 템플릿의 모든 대괄호의 선택지들을 전부 아는 누군가가 있다 하더라도, 코드 함수의 해시 값만으로는 그 함수가 각 대괄호의 어떤 선택지를 포함하고 있는지 모르게, 즉 코드 함수의 실행 부분을 복원하지 못하게 하는 것이 가능하다.

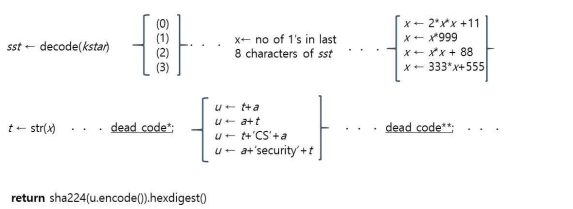

그림 3. 사례: 하나의 템플릿으로부터 서로 다른 코드 함수들을 생성 ($k=4$인 경우)
Fig. 3. An Example: Generating Different Code Functions from a Template (case of $k=4$)

3.2.2 생성된 다수의 코드 함수들 관리

*CMS*는 이미 다른 스마트폰에 배정된 코드 함수의 재배정을 피하기 위해 그림 4의 (b)와 같은 '가용 코드 함수 테이블'을 유지한다. 매 선택지의 실제 선택 내용은 $\log_2 k$(단, k는 2^p 형태라 가정) 비트로 표현되는데, 예를 들어, 첫 번째 대괄호에서 선택지 (1)을, 두 번째 대괄호에서 선택지 (3)을 선택한다면, 0111의 4비트로 이 선택을 표현할 수 있다. 이와 같이 하여 모든 대괄호의 선택 내용을 표현한 비트 열을 '코드 함수 시그니처(signature)'라 부른다. 코드 함수 시그니처에 추가하여 사용 여부를 나타내는 'used bit' 필드가 사용된다. 새로운 스마트폰이 등록될 때, *CMS*는 가용 코드 함수 테이블의 'used bit'가 0인 임의의 엔트리를 찾아, 이를 위에서 설명한 것과 같이 임의의 랜덤 데드 코드(들)를 포함시킨 후 배정하고 'used bit'를 1로 바꾼다. 이어서, 배정된 코드 함수는 그림 4(a)와 같은 '배정된 코드 함수 테이블'에 저장하는데, 이 때 검색 키로 사용될 $h(C)$를, 사용된 코드 함수 C의 시그니처, 인증 취소용 이메일 및 엔트리가 유효한지 확인하는 'invalid flag' 내용과 함께 저장한다.

(a)

h(C)	Csig: code segment signature	email addr. (for revocation)	invalid flag
...	.....	.....	N
wx...yz	b(0)b(1) ... b(i) ... ... b(m)	revokeme@abcdef.com	N
...	.....	.....	N

(b)

Csig: code segment signature	used bit
.....	0
b(0)b(1) ... b(i) ... ... b(m)	1
⋮	⋮
.....	0

그림 4. *CMS*의 테이블 (a) 배정된 코드 세그먼트 테이블, (b) 가용 코드 세그먼트 테이블
Fig. 4. Tables in *CMS* (a) Assigned Code Segments Table, (b) Available Code Segments Table

3.3 코드 함수 실행 결과를 이용한 스마트폰의 적법 소지 인증

사용자가 앱Q를 실행시키고 상대방 인증 서버 및 사용자 이름을 포함한 개인 정보를 입력할 때, 이 서버가 이미 USIM 내에 등록된 경우라면, 앱Q는 사용자의 목적이 인증 받는 것이라고 판단하고 해당 서버에 인증용 챌린지를 요청한다. 지정한 서버가 X_i라 할 때, 이 서버는 자신의 ID X_i에 현재 시각 타임스탬프를 연접하여 이를 챌린지 K로 제공한다. 이때 챌린지는 서버의 바뀌는 현재 시각에 따라 오직 한 번만 사용됨이 보장되므로 이전 인증 세션 내용의 재사용은

근원적으로 불가능하다. 또한 서버 X_i는 챌린지 K와 함께 mid를 앱Q에 전달하는데, 이 mid는 추후 서버 X_i가 수신된 메시지가 자신이 보낸 어떤 챌린지에 대한 응답인지를 확인하는데 사용하며, 각 서버는 일련번호 등 중복되지 않는 값을 매번 mid로 사용한다. 서버 X_i는 수신한 $h(B)$와 mid를 연계시킬 수 있기 때문에, 추후 mid를 이용하여 $h(B)$를 구분하고, $h(B)$를 통하여 $h(A)$ 및 특정 사용자 정보를 추출할 수 있다.

이제 서버 X_i 및 CMS에 제시할 인증 메시지들을 생성하는데, 인증의 기본 아이디어는 스마트폰을 통해 제시되는 인증 값이 둘로 나뉘어 X_i와 CMS에 각각 전달되고(그림 5에서 ②* 및 ②**), X_i에 전달된 메시지 ②*는 CMS가 X_i에 추후 전달하는 메시지(그림 5에서 ③)의 내용을 이용해야만 검증하는 것이 가능하다. 이때 CMS는 메시지 ③을 보내기 위해, 도착 메시지 ②**의 내용을, 저장하고 있는 '배정된 코드 함수 테이블'(그림 4(a))에서 추출한 코드 함수의 실행을 통해 검증해야 한다. 따라서 결과적으로, 서버 X_i는 스마트폰에 포함된 코드 함수의 실제 내용을 전혀 모르면서도 인증 요청 메시지 ②* 및 ②**을 전송한 상대방이, 비밀 값 S_i을 알고 있었으며 CMS에 등록된 올바른 코드 함수를 실행할 수 있었음을 확인할 수 있다. 그 과정에서 CMS는 자신에게 메시지를 보낸 스마트폰 및 소유자에 대하여 정보 무효화를 위해 등록한 이메일 주소 외에 그 어떤 정보(이름, 가입된 ISP, 휴대폰 번호, 생년월일 등)도 알지 못한다.

인증을 위해 진행되는 과정 및 각각의 의미는 다음과 같다. 서버 X_i는 메시지 ②* $mid, h^2(R), E(h(A),h(h(R)\|h(A)))$를 수신하고, mid를 이용하여 이 메시지가 $h(B)$(그림 6(a) '챌린지 테이블' 통하여) 및 $h(B)$에 연계되는 $h(A)$(그림 6(b) '사용자 정보 테이블' 통하여)의 사용자에 의한 인증 시도임을 확인할 수 있으나, 추가적인 해석은 아직 불가능하다. 메시지 ②**는 $mid, X_i, h(C), K^*, E(h(C),h(R))$로서 이 중 $h(C)$는 CMS가 유지하는 '배정된 코드 함수 테이블'에서 해당 코드 함수를 찾는데 필요한 검색 키 역할을 한다. 해당 코드 함수 C를 찾으면 이제 K^*를 입력으로 C를 실행시켜 결과 $\hat{R}$을 얻는다고(즉, $\hat{R} = run(C,K^*)$) 하자. 그럼 이제, 이 $\hat{R}$에 해시 함수 적용한 $h(\hat{R})$을 이용하여 도착 메시지의 $E(h(C),h(R))$ 부분을 복호화한다. 즉 $D(E(h(C),h(R)),h(\hat{R}))$를 시도하고, 그 결과가 $h(C)$와 일치하는지 확인한다. 일치한다면, 메시지 ②**를 생성한 상대방이 정확한 R(입력 K^*로 코드 함수 C 실행 결과)을 구할 수 있었음을 인정한다. CMS는 ②**에 명시된 서버 X_i에게, K^* 및 $h(C)$, 코드 실행 결과 $h(\hat{R})$과 함께 ②**로부터의 mid를 추가해 메시지 ③을 보낸다.

이제 서버 X_i가 메시지 ②*와 ③을 가지고 스마트폰 소유자를 인증할 수 있는데, 이 과정에서 필요한 테이블 구조는 그림 6의 (a), (b)와 같다. '챌린지 테이블'(그림 6(a))은 메시지 ①의 mid와 $h(B)$ 및 챌린지 K를 연결시켜, 추후 mid를 수신할 때 이 mid가 어떤 사용자에게, 어떤 챌린지와 함께 전송되었는지 알게 한다. '사용자 정보 테이블'(그림 6(b))은 서버 X_i에 스마트폰 소유주가 정보를 등록(그림 2 참조)할 때 생성한 해당 소유자의 $h(B)$, $h(A)$, S_i 및 해당 스마트폰이 S_i와 코드 함수 C의 해시 값으로부터 계산한 S_i^*를 기존의 사용자 인증 정보 테이블과 연계시킨 것이다. 즉 서버 X_i는 기존 ID와 패스워드 기반의 인증에 필요한 사용자 튜플들을 테이블 형태로 유지하면서, 이런 튜플들 각각을 제안 기법을 위해 추가된 값들과 연계시켜 확장된 '사용자 정보 테이블'로 유지

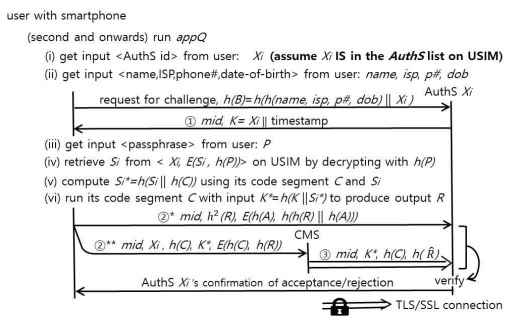

그림 5. 코드 함수 실행을 통한 스마트폰 적법 소지 인증
Fig. 5. Executing its Code Function to Authenticate Legal Possession of a Smartphone

(a)

mid (msg id)	h(B): hash of B	K (challenge)
. . .	. . .	. . .
mid	*ab...cd*	*K*
. . .	. . .	. . .

(b)

h(B): hash of B	h(A): hash of A	S(secret)	S* h(S‖h(C))	id	h(pwd)	other info.
. . .	. . .	. . .	. . .	. . . . .	. . .	. . .
ab...cd	*ef...gh*	*S_i*	*S*_i*	*id_i*	*pq...rs*	*info_i*
. . .	. . .	. . .	. . .	. . . . .	. . .	. . .

auth. credentials for proposed scheme: differ on each server
traditional user auth. credentials: (might be) differ on each server

그림 6. X_i의 테이블 (a) 챌린지 테이블, (b) 사용자 정보 테이블
Fig. 6. Tables in X_i (a) Challenges Table, (b) User Information Table

및 관리한다. 여기서 $h(B)$는 '챌린지 테이블'과 '사용자 정보 테이블'에 공통으로 존재해, 두 테이블의 관련 튜플들을 상호 연계시키는 역할을 한다.

최종적으로 서버 X_i가 스마트폰 소지자를 인증하기 위해서, 동일한 mid 값을 가지는 메시지 ②*와 ③으로 다음 (i),(ii),(iii) 단계들을 순차적으로, 이전 단계 검증을 성공적으로 통과한 경우만 진행한다.

(i) ③의 $h(\hat{R})$에 해시 함수 적용하면 ②*의 $h^2(R)$과 같은지, 즉, $h(h(\hat{R})_{(3)}) == h^2(R)_{(2)}$ 인지 확인.

(ii) '챌린지 테이블'의 'mid(msg id)' 필드를 키로 ③의 mid 값의 매치되는 튜플 $<mid, \boxed{h(B)}, \boxed{K}>$ 검색. '사용자 정보 테이블'의 'h(B): hash of B' 필드를 키로 $\boxed{h(B)}$ 값의 매치되는 튜플 $<\boxed{h(B)}, \boxed{h(A)}, \boxed{S_i}, \boxed{S_i^*}, \ldots >$ 검색. ③의 $h(\hat{R})$ 및 $\boxed{h(A)}$ 값으로 계산한 $h(h(\hat{R})_{(3)} \| \boxed{h(A)})$를 복호화 키로 ②*의 $E(h(A), h(h(R)\|h(A)))$를 복호화(즉 $D(E(h(A), h(h(R)\|h(A))), h(h(\hat{R})_{(3)} \| \boxed{h(A)}))$) 한 결과가 $\boxed{h(A)}$ 값과 같은지 확인.

(iii) 위 (ii)의 '사용자 정보 테이블'의 $\boxed{S_i}$ 및 $\boxed{S_i^*}$ 값과 ③의 $h(C)$ 값을 이용하여 $h(\boxed{S_i} \| h(C)) == \boxed{S_i^*}$인지 확인 후, '챌린지 테이블'의 $\boxed{K}$ 값 및 $\boxed{S_i^*}$ 값을 연접하여 해시 함수 적용한 결과가 ③의 K^* 값과 같은지 확인. 즉, $h(\boxed{K} \| \boxed{S_i^*}) \equiv K^*_{(3)}$확인.

검증 과정 (i)은 스마트폰이 코드 함수 C를 입력 K^*에 대해 실행하여 결과 R을 얻었음을 의미하고, (ii)는 추가적으로 스마트폰이 USIM에 안전하게 저장된 비밀 값 S_i 및 이름 등의 소유자 정보를 이용하여 $h(A)$도 생성할 수 있었음을 의미하며, (iii)은 서버 X_i에 등록 시 저장된 S_i^* 계산에 사용된 $h(C)$의 입력에 해당하는 코드 함수 C가 실행되었고, 실행 시 입력 값은 서버 X_i의 챌린지 K와 비밀 값 S_i 및 코드 함수 C로부터 도출된 $h(C)$을 통해 만들어진 $h(K\|h(S_i\|h(C)))$ 이었음을 각각 의미한다.

3.4 *CMS*에 등록된 코드 함수 정보 무효화

사용자는 스마트폰의 분실 또는 도난을 인지하거나 혹은 스마트폰 내의 비밀 값이나 코드 함수의 외부 유출 가능성이 있다고 판단될 때, 즉시 *CMS*에 자신의 코드 함수를 활용한 인증 요청을 거부해 달라는 무효화 절차를 시작해야 한다. 이 절차는 *CMS*에 접근하여 제공되는 정보 무효화 요청 웹페이지를 통해, 등록 과정에서 *CMS*에 알렸던 이메일 주소를 입력하는 것으로 시작된다. *CMS*는 '배정된 코드 함수 테이블'(그림 4(a))의 'email addr.(for revocation)' 필드를 키로 제시된 이메일 주소(예: $revokeme@abcdef.com$ 이라 함)의 매치되는 튜플 $<wx \ldots yz, b_0b_1 \ldots b_m, revokeme@abcdef.com, N>$을 검색한다. 그리고 검색된 이메일 주소로 랜덤 주소의 링크를 포함하는 내용의 이메일을 발송한다. 사용자는 $revokeme@abcdef.com$의 수신 메일서버에 로그온 해 도착한 메일을 열고, 내용에 포함된 링크를 클릭함으로써 이 이메일 주소에 접근할 수 있음을 *CMS*에 입증한다. 링크 클릭을 확인한 후, *CMS*는 $revokeme@abcdef.com$을 포함하는 튜플의 마지막 필드의 값 N을 Y로 바꾼다. 이후 *CMS*에 도착하는 해당 튜플의 코드 함수 관련 요청은 모두 무시된다. 결국 어떤 인증 서버로의 인증 요청도 문제의 코드 함수를 사용해 전송된 것이면 *CMS*의 검증 과정을 통과하지 못하게 되어 인증 서버의 인증을 취득할 수 없게 된다.

Ⅳ. 보안성 분석

제안 기법의 보안성을 분석하는데, 특히 어느 서버이든 서버의 저장 내용을 공격자가 탈취하더라도 공격자가 사용자로 위장할 수 없으며, 또한 이전 인증 세션 내용을 저장한 후 *CMS* 또는 인증 서버 X_i의 저장 내용을 획득하더라도 공격에 필요한 다른 서버의 저장 내용을 얻을 수 없음을 보인다.

4.1 공격자가 스마트폰에 접근 불가인 경우의 공격 대응

스마트폰 적법 소지 인증을 받으려면 인증 서버의 현재 시각을 반영한 '새로운' 챌린지에 대하여 그림 5의 메시지 ②* 및 ②**를(즉 ②**가 *CMS*의 검증을 통과한 후, *CMS*가 서버 X_i에 보낼 메시지 ③이 메시지 ②*와 결합하여 서버 X_i의 검증을 통과하도록) 생성할 수 있어야 한다. 그런데 스마트폰에 접근할 수 없는 공격자가 이전 인증 세션에서 공격대상자의 '변하지 않는 값' $h(B)$ 및 $h(C)$를 획득했다 하더라도, 코드 함수 C를 몰라 새로운 챌린지에 따라 변하는 R 및 $h(R)$ 값을 구할 수 없다. 또한 USIM에 저장된 비밀 값 S_i를 모르기에 새롭게 계산해야 하는 K^* 역시 올바로 생성할 수 없다. 결과적으로 공격자가 메시지

②* 및 ②**의 대칭키 암호화 부분 및 $h^2(R)$, 그리고 K^*에 대하여, 서버 X_i를 속일 수 있는 올바른 값을 계산하는 것은 불가능하다.

4.2 공격자가 스마트폰에 접근 가능인 경우의 공격 대응

공격자가 공격대상자의 스마트폰에 접근(훔치거나 분실물 취득)할 수 있다면, 공격자는 우선 스마트폰에 저장된 서버별 비밀 값 S_i들 및 코드 함수 C를 얻으려 시도할 것이다. 이 중 S_i들은 USIM에 암호화된 후 저장되므로 이들을 얻기 위해서는 복호화 키를 만드는데 사용되는 공격대상자의 패스프레이즈 P를 알아야 한다. P는 사용 후 따로 어딘가에 저장되지 않기에, 공격자는 추측에 의한 방식에 의존하거나 또는 아예 다른 경로를 통해 P 또는 직접 S_i를 알아낼 수밖에 없다. P도 S_i도 스마트폰 외부에 평문으로 전송되거나 공개되지 않기에 공격자가 이들 값을 알아내는 시도는 설사 성공하더라도 시간이 많이 걸릴 수밖에 없다. 만약 공격자가 공격대상자의 스마트폰에 미리 악성코드를 심어 두었는데도 공격대상자가 이를 모르고 패스프레이즈를 입력했다면, 공격자가 패스프레이즈 P를 획득할 수 있다. 이 상황에서 공격자가 공격대상자 스마트폰을 취득한 경우, 제안 기법은 두 가지의 대응책을 가진다. 첫째는 가입된 통신사에 스마트폰 분실을 신고하여 폰의 USIM 사용을 정지시킴으로써 USIM 내에 저장된 비밀 값 S_i들에 대한 접근을 막는 것이며, 둘째는 *CMS*에 등록 정보 무효화 요청을 하여 등록된 이메일 주소로 전송된 링크를 클릭함으로써 해당 스마트폰에 탑재된 코드 함수를 무효화시키는 것이다. 둘 중 어떤 것이든 제안 기법에 따른 인증 시도를 막을 수 있으므로, 상황에 따라 더 신속히 처리될 수 있는 방법을 택하도록 한다.

4.3 서버 저장 정보 탈취 공격 대응

인증 서버 X_i나 *CMS* 서버의 저장 내용이 공격자에게 유출된 경우에도, 제안 기법은 소유하는 정보를 서버 각각에게 정교하게 분리시킴으로써 공격자가 등록된 사용자로 위장해 인증을 성공시킬 수 없음을 보장한다. 첫째, 서버 X_i의 테이블들이 공격자 손에 들어간 경우, 공격자는 공격대상자의 비밀 값 S_i 및 메시지 ②*의 생성을 위해 필요한 $h(A)$를 얻을 수 있다. 하지만 서버 X_i에는 코드 함수를 저장하지 않기에 코드 함수 실행 결과인 R을 얻어낼 수 없어서, 메시지 ②*의 $h^2(R)$ 및 ②*와 ②**의 $h(R)$을 필요로 하는 부분을 만들어낼 수 없다. 둘째, *CMS*의 테이블들이 공격자 손에 들어간 경우, 공격자는 오직 어떤 코드 함수가 배분되어 사용 중인지를 해당자의 정보 무효화용 이메일 주소와 함께 알 수 있을 뿐, 공격대상자의 다른 어떤 정보도 알 수 없다. 따라서 개인 정보의 조합이 포함된 A를 얻을 수 없어 메시지 ②*의 $h(A)$를 사용하는 부분을 계산할 수 없다. 또한 코드 함수에 제시될 입력 값은 비밀 값 S_i 없이는 구할 수 없기에, 공격자는 코드 함수 실행 결과인 R을 요구하는 $h(R)$ 관련 값들, 즉 메시지 ②*의 $h^2(R)$과 ②**의 K^* 및 $E(h(C),h(R))$ 역시 정확히 만들어낼 수 없다.

4.4 이전 세션 도청 후 서버 정보 탈취 대응

공격자가 이전 인증 세션의 메시지 ①, ②* 및 ②**를 도청 등의 방법으로 얻은 후, *CMS*의 저장 정보를 탈취하는데 성공했더라도, 공격자는 서버 X_i에 대한 인증 요청 시 사용될 정보 중 X_i에 저장된 공격대상자의 스마트폰 비밀 값 S_i나 $h(A)$ 값을 얻을 수 없다. 그 이유는 ②*의 $E(h(A),h(h(R)\|\|h(A)))$를 복호화하여 $h(A)$를 얻으려면, 키 계산을 위해 $h(A)$ 값이 필요한데, *CMS* 저장 정보나 이전 세션 메시지들로부터는 $h(A)$를 획득할 수 없기 때문이다. 이제 이전 인증 세션의 메시지 ①, ②* 및 ②**를 얻고, 서버 X_i의 저장 정보를 탈취한 공격자가 공격대상자인 것처럼 서버 X_i를 속일 수 있을지 살펴본다. 공격자는 서버 X_i의 사용자 정보 테이블로부터 S_i^* 값을, 챌린지 테이블로부터 챌린지 K를 취해서 $K^*=h(K\|S_i^*)$를 계산할 수 있다. 하지만 메시지 ②**의 $E(h(C),h(R))$은 구할 수가 없는데, 그 이유는 매번 새로운 챌린지 K 값에 따라 변하는 입력 K^*으로 코드 함수 C를 실행한 결과의 R은 이전 세션과 다른 새로운 값을 가지게 되며, 서버 X_i가 보유한 정보에는 코드 함수 C가 포함되지 않기에 실제 C를 입력 K^*에 대하여 실행시켜 볼 수도 없기 때문이다. 따라서 설령 공격자가 이전 세션에서 *CMS*로부터 받은 메시지 ③의 $h(\hat{R})$을 알 수 있었더라도, 새로운 세션에서는 다른 $h(R)$ 값이 요구되므로 소용이 없다. 결과적으로 공격자는 *CMS*에 보낼 올바른 메시지를 구성할 방법을 가지지 못한다.

4.5 공격자가 서버 자체로 위장하는 공격 대응

공격자가 *CMS* 위치에서 *CMS* 역할을 하려고 할

때, 메시지 ①, ②* 및 ②**와 연계된 올바른 메시지 ③을 보내야만 서버 X_i를 속일 수 있다. 하지만 이러한 메시지 ③을 구성하려면 $h(R)$ 값을 정확히 알아야 하는데, 메시지 ①, ②* 및 ②**로부터는 $h(R)$을 구할 방법이 없다. 해시 함수의 단방향성으로 $h^2(R)$로부터 계산할 수 없고, 또한 AES 암호화한 결과 $E(h(C),h(R))$로부터는 암호화 대상인 $h(C)$를 알더라도 암호화에 사용된 키 $h(R)$ 값을 구하는 것 역시 불가능하기 때문이다.

다음으로 공격자가 서버 X_i 위치에서 X_i 역할을 하려고 한다고 가정하자. 공격자는 공격대상자 스마트폰에 메시지 ①을 보낼 수 있는데, 그 결과 메시지 ②*를 수신하고, 만약 *CMS*까지 속일 수 있다면 *CMS*로부터 메시지 ③을 수신할 것이다. 하지만 공격자가 이런 전 과정의 메시지들 ①, ②*, ②**(*CMS*로 향하는 것을 가로챈다면) 및 ③을 통하여 알 수 있는 공격대상자 관련 정보는 $h(C)$와 $h(R)$ 밖에 없는데, 다음 번 세션의 $h(R)$은 이렇게 얻은 $h(R)$와 다를 것이며 전혀 관련성이 없기에 공격자에게 도움이 되지 않는다. 그리고 해시 함수의 단방향성으로 $h(C)$로부터 C를 구할 수 없기에, 공격자는 임의의 입력 값에 대한 실행 결과를 내는데 필요한 공격대상자 스마트폰의 코드 함수 역시 알 방법이 없다. 따라서 이 상황의 공격자는 공격대상자로 위장하는 등의 공격을 가하는데 필요한 유용한 정보를 갖지 못한다.

4.6 가용성 저하를 목적으로 하는 서비스 거부 공격 대응

인증 서버 외에 추가된 *CMS*의 제공 서비스가 중단 되면 제안 기법에 따른 인증은 중단되기 때문에 *CMS* 서버 가용성 보장은 매우 중요하다. *CMS*의 역할은 등록/무효화 및 인증정보 제공으로 나뉘는데, 등록/무효화는 일회성으로 데이터베이스 테이블에 쓰기를 요구하는 반면 인증정보 제공은 빈번하게 발생하는데 데이터베이스 테이블의 읽기만을 요구한다. 이에 MySQL와 같은 데이터베이스관리시스템의 듀얼(dual) 복제(replication) 기능을 활용하여 서버의 양방향 이중화를 도입하는데, 이는 두 서버 각각이 등록/무효화 및 인증정보 제공의 서비스를 제공하면서 어느 쪽이든 데이터 변경이 일어나면 이것이 상대편에 복제되도록 구성하는 것이다. 그 결과 한 서버가 다운되더라도 다른 서버를 통해 서비스 제공이 가능하며, 다운된 서버가 재가동되면 다운된 시간 동안의 갱신 데이터가 살아난 서버에 복제되기에 다시 이중화 구성으로 동작한다. 따라서 이런 구성 및 배치를 통해 *CMS* 서버에의 부하 분산 은 물론 이중화된 서버 모두가 다운되지 않는 한 서비스 가용성을 제공하는 효과를 기대할 수 있다.

V. 결 론

SMS 기반 인증 및 이를 개선한 형태의 인증 방식들은 기본적으로 챌린지-응답의 형태를 따르며, 인증 담당 서버가 저장하고 있는 정보를 통해 자신이 보낸 챌린지에 대한 응답을 검증할 수 있다는 공통점을 지닌다. 이는 공격자가 서버의 저장 내용에 접근할 수 있다면 공격자가 저장 내용을 기반으로 서버를 속일 수 있는 응답을 만들어냄으로써 등록된 사용자로 위장하는 것이 가능함을 의미한다.

스마트폰의 적법한 소지를 안전하게 증명하려는 제안 기법은 인증 서버 외에 추가로 별도 서버인 *CMS*를 두어 각 서버의 역할 및 저장 정보를 정교하게 분리하고, 상호 협력을 통하여 최종적으로 인증 서버가 인증 여부의 판단을 내리도록 설계하였다. 그 결과 인증 서버와 *CMS*의 저장 내용을 공격자가 모두 얻어내지 않는 한, 하나의 서버 내용의 유출만으로는 공격자가 사용자 위장 공격을 성공시키거나 등록된 사용자의 개인 정보를 획득하는 것이 불가능함을 보였다. 긴급 상황에는 USIM 락(lock) 또는 *CMS*에 정보 무효화 요청 중 어떤 것이든 먼저 처리되는 순간부터 공격자의 인증 시도는 전부 실패로 돌아가므로, 스마트폰의 분실 또는 도난에도 제안 기법은 대비되어 있다. 악성코드를 통해 스마트폰의 비밀 값이나 코드 함수 내용을 파악하는 것 역시, 앱 간 분리로 악성코드가 인증 앱 자원에 접근하는 것을 커널 수준에서 막기 때문에 극히 어렵다. 사용자에게는 인증 시 앱을 구동시키고 몇 개 값들을 기억해서 입력하는 것만을 요구하기에, 모바일 환경에서 앱 실행에 친숙한 대다수 사용자들에게 제안 기법에 적응하는 부담은 감내할 수 있을 정도라 볼 수 있다.

향후 의미 있는 개선 방향으로 제안 기법의 보안 수준을 유지하면서도 사용자 편의성을 향상시키는 방식의 도입을 들 수 있는데, 특히 지문이나 홍채 인식 등 스마트폰에 이미 적용되어 있는 생체인식 기술을 활용해 인증 과정에서 사용자가 직접 입력하는 양을 줄이는 방식의 실현 가능성에 대한 연구가 필요하리라 판단된다.

References

[1] A. Dmitrienko, C. Liebchen, C. Rossow, and A. R. Sadeghi, "Security analysis of mobile two-factor authentication schemes," *Intel Technol. J.*, vol. 18, no. 4, pp. 138-161, 2014.

[2] R. Grimes, "The many ways to hack 2FA," *Netw. Secur.*, vol. 2019, no. 9, pp. 8-13, Sep. 2019.

[3] A. R. L. Reyes, E. D. Festijo, and R. P. Medina, "Enhanced multi-factor out-of-band authentication en route to securing SMS-based OTP," *Int. J. Eng. and Technol. Innovation*, vol. 9, no. 2, pp. 145-154, 2019.

[4] S. H. Lee, H. Kim, and D. H. Lee, "Two-factor authentication scheme based on mobile messenger with improved usability," *J. Secur. Eng.*, vol. 10, no. 5, pp. 549-566, Oct. 2013.

[5] U. A. Abdurrahman, M. Kaiiali, and J. Muhammad, "A new mobile-based multi-factor authentication scheme using pre-shared number, GPS location and time stamp," in *Proc. 2013 ICECCO*, pp. 293-296, Nov. 2013.

[6] FireEye Blogs, *We Steal SMS: An insight into Android.KorBanker Operations*, Sep. 3, 2014, Retrieved Dec. 16, 2019, from https://www.fireeye.com/blog/threat--research/2014/09/we-steal-sms-an-insight-into-android-korbanker-operations.html

[7] Mazar Bot, Retrieved Dec. 16, 2019, from https://www.tripwire.com/state-of-security/featured/mazarbot-android-malware-distributed-via-sms-spoofing-campaign/

[8] BankBot, Retrieved Dec. 16, 2019, from https://github.com/bemre/bankbot-mazain

[9] S. Mishra and D. Soni, "SMS phishing and mitigation approaches," in *Proc. 12th Int. Conf. Contemporary Computing* (IC3), Noida, India, Aug. 2019.

[10] S. S. Alqahtani and D. Alghazzawi, "A survey of emerging techniques in detecting SMS spam," *Trans. Machine Learning and Artificial Intell.*, vol. 7, no. 5, pp. 24-35, Oct. 2019.

[11] NEXSPY.com, *How to Read Someone's Text Messages Without Having Their Phone?*, Retrieved Dec. 30, 2019, from https://nexspy.com/read-someones-text-messages/

[12] H. Siadati, T. Nguyen, P. Gupta, M. Jakobsson, and N. Memon, "Mind your SMSes: Mitigating social engineering in second-factor authentication," *Comput. & Secur.*, vol. 65, pp. 14-28, Mar. 2017.

[13] N. Popper, *Hackers Hit Twitter CEO Jack Dorsey in a 'SIM Swap.' You're at Risk, Too*, The New York Times, Sept. 5, 2019, https://nytimes.com/2019/09/05/technology/sim-swap-jack-dorsey-hack.html

[14] B. Krebs, "Busting SIM swappers and SIM swap myths," *Krebs on Security*, vol. 7, Nov. 2018, from https://krebsonsecurity.com/2018/11/busting-sim-swappers-and- sim-swap-myths/

[15] P. A. Grassi, J. L. Fenton, E. M. Newton, R. A. Perlner, A. R. Regenscheid, W. E. Burr, and J. P. Richer, *NIST Special Publication 800-63B Digital Identity Guidelines Authentication and Lifecycle Management*, Retrieved Dec. 16, from https://doi.org/10.6028/NIST.SP.800-63b

[16] D. J. Seo and T. S. Kim, "Influence of personal information security vulnerabilities and perceived usefulness on bank customers' willingness to stay," *J. KICS*, vol. 40, no. 8, pp. 1577-1587, Aug. 2015.

[17] P. Vijayakumar, S. M. Ganesh, L. J. Deborah, and B. S. Rawal, "A new SmartSMS protocol for secure SMS communication in m-health environment," *Comput. and Electrical Eng.*, vol. 65, pp. 265-281, 2018.

[18] J. Li, Y. Ye, Y. Zhou, and J. Ma, "CodeTracker: A lightweight approach to track and protect authorization codes in SMS messages," *IEEE Access*, vol. 6, Mar. 2018.

[19] S. J. Kwon and J. C. Park, "Smartphone ownership and location checking scheme for fixing the vulnerabilities of SMS-based authentication," *J. KICS*, vol. 42, no. 2, pp. 1-9, Feb. 2017.

박 준 철 (Jun-Cheol Park)

1986년 2월 : 서울대학교 계산통계학과
1988년 2월 : KAIST 전산학과 석사
1998년 12월 : U of Maryland, College Park, 전산학 박사
2001년 9월~현재 : 홍익대학교 컴퓨터공학과 교수

<관심분야> 네트워크 보안, 소프트웨어 보안
[ORCID:0000-0001-8438-8185]