애플 Bonjour 프로토콜을 위한 ECDH 기반 인증 프로토콜

(ECDH based authentication protocol for the Apple Bonjour protocol)

권순홍*, 이종혁**

(Soonhong Kwon, Jong-Hyouk Lee)

요 약

애플사는 장비 간 간편한 파일 송수신을 위해 자체적인 프로토콜인 Bonjour 프로토콜을 제공한다. 대표 서비스로는 Airdrop이 있으며, 애플사의 데스크탑, 노트북, 스마트폰 간에 간편한 데이터 송수신 프로토콜로 널리 사용되고 있다. 하지만, 2016년 해킹 보안 컨퍼런스인 Black Hat에서 Bonjour 프로토콜 취약점을 통해 중간자 공격이 가능함을 보였다. 본 논문은 장비 간 안전한 파일 송수신을 위해 Bonjour 프로토콜의 알려진 취약점을 설명하고, Bonjour 프로토콜의 안전성을 높이기 위해 ECDH(Elliptic Curve Diffie-Hellman) 기반 인증 프로토콜을 제안한다. 제안된 프로토콜을 상세한 동작 프로시저와 함께 설명하며, 중간자 공격과 신분위장 공격 가능성을 줄일 수 있음을 증명한다.

■ **중심어** : Bonjour 프로토콜 ; Airdrop ; ECDH 프로토콜 ; 중간자 공격 ; 신분위장 공격

Abstract

Apple provides its own protocol, the Bonjour protocol, for convenient file transmission and reception between device. Airdrop is a Bonjour-based, representative service that is widely used as a simple data transmission/reception protocol for Apple's desktops, laptops and smartphones. However, it was demonstrated in Black Hat, a hacking security conference in 2016, that it is possible to commence a Man-in-the-Middle attack by exploiting the Bonjour protocol's weak points. In this paper, we explain the Bonjour protocol's such vulnerability for secure file transmission/reception between devices and propose an ECDH (Elliptic Curve Diffie-Hellman) based authentication protocol to enhance the protocol's security. The proposed protocol is described along with detailed operational procedures, demonstrating that it is possible to reduce the possibility of Man-in-the-Middle attack and its masquerade variant.

■ **keywords** : Bonjour protocol ; Airdrop ; ECDH protocol ; Man-in-the-Middle attack ; Masquerade attack

Ⅰ. 서 론

오늘날 애플사 장비의 사용량이 증가하고 있다. 애플사는 Bonjour 프로토콜을 활용한 서비스들을 연구/개발하고 있다. Bonjour 프로토콜은 애플사에서 자체적으로 개발한 Zeroconf를 구현한 프로토콜로써 기존의 프로토콜과는 다르게 DNS(Domain Name System) 서버, DHCP(Dynamic Host Configuration Protocol) 서버의 도움 없이 IP 주소 할당, 도메인 이름을 할당할 수 있는 기술이다[1, 2]. 이 기술을 활용한 대표 서비스로는 Airdrop이 있다. Airdrop을 사용함으로써 장비는 별도의 구성없이 자신의 IP 주소, 도메인 이름과 서비스를 자체적으로 설정한 뒤 장비와 연결하여 데이터(e.g., 파일, 사진, 동영상)를 간편하게 주고받을 수 있다.

2016년 해킹 보안 컨퍼런스인 Black Hat에서 Bonjour 프로토콜의 취약점을 이용하여 중간자 공격이 가능함을 보였다[4]. 또한, Airdrop 사용 시, iCloud 계정이 비로그인 되어 있는 경우에 자가 서명 인증서(Self-Signed Certificate)를 이용하여 인증을 수행하는 것을 패킷 분석을 통해 확인하였다[5]. 검증되지 않은 자가 서명 인증서를 사용하는 경우 중간자 공격 및 재전송 공격에 노출될 수 있다. 본 논문에서는 이러한 취약점을 이용한 공격의 가능성을 줄이기 위해 ECDH(Elliptic Curve Diffie-Hellman) 기반 인증 프로토콜을 적용한 Bonjour 프로토콜을 제안한다.

본 논문의 2장에서는 Bonjour 프로토콜과 동작 과정에 대해 설명하고, Bonjour 프로토콜의 대표 서비스인 Airdrop 서비스 패킷 분석 내용 및 알려진 취약점에 대해 설명하며, ECDH 프로토콜에 대해 설명한다. 3장에서는 ECDH 기반

* 학생회원, 상명대학교 프로토콜공학연구실
** 정회원, 상명대학교 프로토콜공학연구실

접수일자 : 2019년 07월 22일
수정일자 : 1차 2019년 08월 19일
게재확정일 : 2019년 08월 19일
교신저자 : 이종혁, e-mail : jonghyouk@pel.smuc.ac.kr

인증 프로토콜을 적용한 Bonjour 프로토콜에 대해 설명한다. 4장에서는 보안 분석을 통해 제안하는 프로토콜의 장점 및 단점에 대해 설명하며, 5장에서 본 논문의 결론을 맺는다.

Ⅱ. 관련 연구

1. Bonjour 프로토콜

Bonjour 프로토콜은 애플사에서 Zeroconf를 구현한 자체적인 프로토콜이다. Zeroconf는 Zero configuration Networking으로 DNS 서버, DHCP 서버의 구성없이 device-to-device 연결이나 무선 환경에서 수동 설정 없이 자동으로 네트워크를 구성할 수 있는 기술이다. 애플사는 Bonjour 프로토콜을 사용하여 Airdrop, Airplay, Airprint와 같은 서비스를 제공한다. Bonjour 프로토콜은 mDNS나 DNS-SD 프로토콜을 기반으로 동작한다. mDNS는 Multicast DNS로 IP 주소와 도메인 이름을 매핑하기 위해 사용되는 프로토콜이다. mDNS를 통해 로컬 네트워크에 참여한 장비를 자동으로 찾을 수 있다. DNS-SD는 DNS-Service Discovery로 서비스를 탐색/설정하기 위해 사용되는 프로토콜이다. mDNS와 DNS-SD 프로토콜에서 요청 메시지는 멀티캐스팅되고, 요청에 대한 응답 메시지는 유니캐스팅 된다는 특징이 있다[1, 2].

2. 동작 과정

Bonjour 프로토콜의 동작 과정은 서비스를 제공하는 장비의 네트워크 및 서비스 설정을 위한 과정인 Publication, 서비스를 사용하려고 하는 장비에 의한 특정 서비스를 제공하는 장비를 찾는 과정인 Discovery, 서비스를 제공하는 장비와 서비스를 사용하려고 하는 장비의 연결을 위해 서비스 인스턴스 이름을 기반으로 IP 주소, 포트 번호를 매핑하는 과정인 Resolution 과정 세 단계로 나누어진다. 본 절에서는 Bonjour 프로토콜의 세 단계에 대해 설명한다.

가. Publication

Publication은 서비스를 제공하는 장비의 설정 과정으로써 네트워크에서 특정 서비스를 제공하는 장비는 자신의 서비스 정보를 네트워크에 참여하는 모든 장비에게 멀티캐스팅한다. 이 과정에서 서비스를 제공하는 장비는 자신의 DNS record 정보(IP 주소, 도메인 이름, 포트 번호)로 사용하고자 하는 정보를 요청 메시지로 하여 다른 장비들에게 멀티캐스팅한다. 해당 요청 메시지를 수신한 장비들은 요청 메시지의 DNS record와 자신의 정보를 비교한다. 만약, 동일한 항목(IP 주소, 도메인 이름, 포트 번호)이 없다면 수신 장비들은 요청에 대한 응답을 보내지 않는다. 하지만, 송신 장비의 DNS record와 동일한 항목이 있다면 요청한 DNS record 정보와 동일한 항목을 가진 장비의 정보를 서비스를 제공하고자 하는 장비에 유니캐스팅하여 해당 DNS record로 장비를 설정하지 못함을 알린다. 응답 메시지를 요청 메시지의 TTL(Time-To-Live) 시간동안 받지 않은 송신 장비는 요청 메시지에 등록된 DNS record를 SRV(Service record)에 등록하며, mDNS DB에 저장된다[3]. 아래 그림 1은 Publication의 동작 과정을 나타낸다.

1) 서비스를 제공하고자 하는 특정 장비 Dev_B는 로컬 네트워크에 참여한 장비들 Dev_T에 IPv6 주소인 “fe:80::1234” 사용 가능 여부 요청 메시지 멀티캐스팅
2) 요청 메시지의 TTL 시간 동안 응답 메시지를 수신하지 않을 경우 Dev_B는 IPv6 주소 “fe:80::1234”를 자가 설정
3) Dev_B는 도메인 이름을 설정하기 위해 도메인 이름 “exa-printer.local”에 대해 사용 가능 여부 요청 메시지를 Dev_T에 멀티캐스팅
4) 요청 메시지의 TTL 시간 동안 응답 메시지를 수신하지 않을 경우 Dev_B는 도메인 이름 “exa-printer.local.”를 자가 설정
5) Dev_B는 포트 번호를 설정하기 위해 포트 번호 “1010” 사용 가능 여부 요청 메시지를 Dev_T에 멀티캐스팅
6) 요청 메시지의 TTL 시간 동안 응답 메시지를 수신하지 않을 경우 Dev_B는 포트 번호 “1010”을 자신의 포트 번호로 자가 설정
7) 2), 4), 5)에서 자가 설정한 정보(IP 주소, 도메인 이름, 포트 번호)를 SRV 레코드에 등록

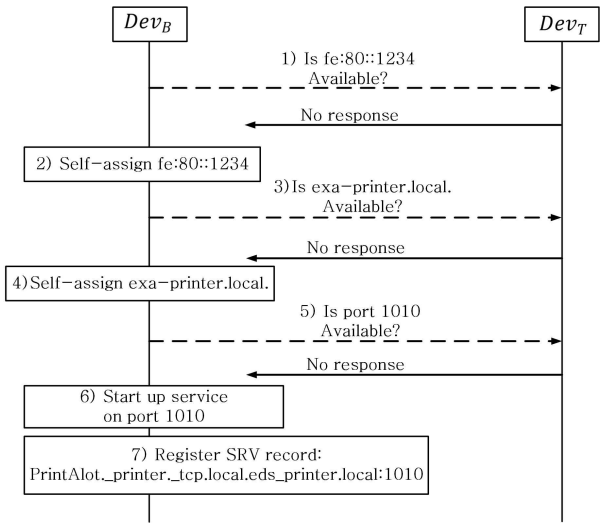

그림 1. Publication 동작 과정

나. Discovery

Discovery는 Publication에서 등록된 정보를 기반으로 특정 서비스를 찾는 과정이다. Publication 과정과 같이 mDNS responder(DNS resolve 요청에 대한 응답)와 함께 동작한다. 서비스를 사용하려고 하는 장비는 특정 서비스를 제공하는 장비를 네트워크에서 찾기 위해 PTR(Pointer record) 레코드에 대해 검색하는 것으로 Discovery 과정을 시작한다. 만약, 프린터 서비스를 제공하는 기기를 찾는 경우 "_printer._tcp"와 같이 서비스 이름을 검색한다. 검색 할 경우 로컬 네트워크에 요청 메시지를 멀티캐스팅하고, 해당 서비스 타입을 가진 서비스를 제공하는 장비가 있다면 해당 요청 메시지를 보낸 장비에게 자신의 정보를 유니캐스팅한다[3]. 아래 그림 2는 Discovery 동작 과정을 나타낸다.

1) 서비스를 사용하고자 하는 장비 Dev_A는 "_printer._tcp" 타입의 서비스 이름을 가진 장비 여부에 대해 요청 메시지로 로컬 네트워크의 장비들 Dev_T에 멀티캐스팅
2) 로컬 네트워크에 참여한 장비들 Dev_T중 해당 서비스 타입을 가진 장비 Dev_B는 자신의 인스턴스 이름인 "PrintAlot._printer._tcp.local."을 응답 메시지로 Dev_A에 유니캐스팅

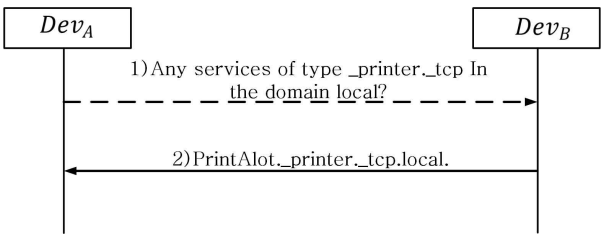

그림 2. Discovery 동작 과정

다. Resolution

Resolution 과정에서는 서비스를 사용하려고 하는 장비가 서비스를 제공하는 장비의 도메인 이름과 IP 주소, 포트 번호를 확인하는 과정이다. 서비스를 사용하려고 하는 장비는 인스턴스 이름인 "PrintAlot._printer._tcp.local"을 통해 도메인 이름과 포트 번호를 요청한다. 요청에 대한 응답 메시지를 통해 확인한 도메인 이름인 "exa-printer.local"을 이용하여 IP 주소를 요청하고 서비스를 제공하는 장치로부터 IP 주소를 수신하여 확인하는 과정이다. 이 과정 또는 mDNS responder와 함께 동작한다. Resolution 과정에서는 상황에 따라 쓰이는 레코드 타입이 다르다. 인스턴스 이름을 통해 도메인 이름과 포트 번호를 요청할 때는 SRV 레코드 타입으로 요청하고, 도메인 이름을 통해 IP 주소를 요청할 때는 TXT 레코드 타입으로 요청하고 응답한다.

특정 서비스를 사용하고자 하는 장비는 인스턴스 이름을 기반으로 도메인 이름과 포트 번호를 요청하는 메시지를 멀티캐스팅한다. 해당 메시지를 수신한 다른 장비들은 인스턴스 이름에 대해 자신의 인스턴스 이름과 비교한다. 동일한 인스턴스 이름을 가진 장비는 자신의 도메인 이름과 포트 번호를 서비스를 사용하고자 하는 장비에게 유니캐스팅한다. 특정 서비스를 사용하고자 하는 장비는 자신이 사용하고자 하는 서비스를 제공하는 장비의 도메인 이름과 포트 번호를 수신하게 되고, 수신한 정보인 도메인 이름을 통해 서비스를 제공하는 장비에게 IP 주소를 요청하는 메시지를 멀티캐스팅한다. 요청 메시지를 수신한 장비들은 요청 메시지에 해당하는 도메인 이름과 자신의 도메인 이름을 비교하며, 동일한 경우 IP 주소를 요청 메시지를 송신한 장비에게 유니캐스팅한다. 이로써 IP 주소나 호스트 이름이 변경되더라도 같은 로컬 네트워크에 참여하였을 경우 해당 서비스를 제공하는 장비를 찾는 경우가 발생하지 않는다[3]. 아래 그림 3은 Resolution 동작 과정을 나타낸다.

1) 서비스를 사용하고자 하는 장비 Dev_A는 Discovery 과정을 통해 수신한 인스턴스 이름을 사용하여 도메인 이름과 포트 번호를 요청하는 메시지를 Dev_T에 멀티캐스팅
2) 요청 메시지를 수신한 장비들 Dev_T중 해당 인스턴스 이름과 서비스 타입을 가진 장비 Dev_B는 자신의 도메인 이름과 포트 번호를 응답 메시지로 장비 Dev_A에 유니캐스팅
3) 요청 메시지에 대한 응답 메시지를 수신한 장비 Dev_A는 응답 메시지를 통해 확인한 해당 도메인 이름을 가진 장비의 IPv6를 요청하는 메시지 멀티캐스팅
4) 요청 메시지를 수신한 장비들 Dev_T중 해당 도메인 이름을 가진 장비 Dev_B는 자신의 IPv6 주소를 응답 메시지로 유니캐스팅

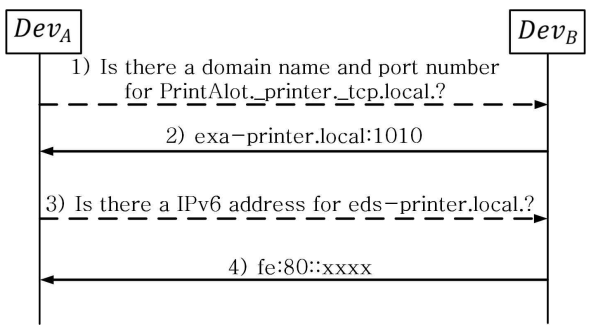

그림 3. Resolution 동작 과정

3. ECDH 프로토콜

ECDH 프로토콜은 Elliptic Curve Diffie-Hellman의 약어로써, 이산 대수 문제를 기반으로 하는 Diffie-Hellman과 다르게 타원 곡선 이산 대수 문제를 기반으로 한다. 이는 Diffie-Hellman 키 교환 방식에서 발생할 수 있는 중간자 공격을 막기 위해 고안된 프로토콜이다. ECDH는 타원 곡선 개인 값과 공개키를 통해 공유키를 수립하는 키 합의 프로토콜이다[6]. 다음 그림 4는 ECDH 프로토콜의 동작 과정이다.

1) Dev_A와 Dev_B는 자신의 타원 곡선 개인 값과 공개키 생성

- Elliptic Curve Private Value:

$2 < d_A, d_B < n-1$ (1)

- Elliptic Curve Public Key:

$H_A = d_A \cdot G$ (Domain parameter) (2)

$H_B = d_B \cdot G$ (3)

2) Dev_A와 Dev_B는 자신의 타원 곡선 공개키 교환

3) Dev_A는 Dev_B의 타원 곡선 공개키로, Dev_B는 Dev_A의 공개키를 통해 공유키 생성

- Shared Key:

$K_A = d_A \cdot H_B, K_B = d_B \cdot H_A$ (4)

4) 각 통신 주체가 생성한 공유키 교환

5) 각 통신 주체는 수신 받은 공유키와 자신의 공유키 비교

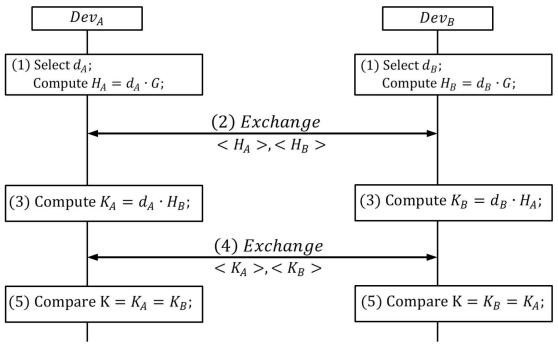

그림 4. ECDH 키 합의 동작 과정

4. Airdrop 패킷 분석 및 알려진 취약점

Bonjour 프로토콜 기반 Airdrop 서비스의 동작 과정은 Everyone 모드와 Contact Only 모드로 나눌 수 있다. Everyone 모드의 경우 연락처 목록에 있는 장비가 아닌 경우에도 장비 간 연결을 수립하지만, Contact Only 모드의 경우 iCloud 인증서와 자신의 연락처 목록을 비교하여 일치하는 항목이 있는 경우에만 장비 간 연결을 수립한다. Contact Only 모드의 경우 iCloud와 연결된 메일 주소와 전화번호를 기반으로 신원 정보 해시를 생성한다. 생성한 신원 정보 해시는 멀티캐스팅되며, 이를 수신한 장비는 자신의 연락처 목록과 비교를 통해 장비를 인증하게 된다. 송신 장비는 수신 장비의 IP 주소로 TCP 연결을 수립하기 위한 TLS Handshake 과정에서 각 장비의 iCloud 인증서를 교환하여 자신의 연락처를 비교하고, 자신의 연락처와 일치하는 항목이 있으면 Handshake를 진행하지만, 일치하지 않는 경우 Handshake 과정을 중단한다. TLS 연결이 수립되었다면, 송신 장비는 수신 장비에게 공유하고자 하는 데이터를 전송할 수 있다. 반면, Everyone 모드에서는 Contact Only 모드와는 다르게 자신의 연락처에 있는 장비가 아니더라도 자가 서명 인증서를 통해 TLS 연결을 수립한다. Airdrop의 패킷 분석을 통해 송신 장비로 iPhone(iOS 11.3.1)를 사용하였고, 수신 장비로는 MacBook Pro(macOS High Sierra 10.13.4)를 사용하였다. Contact Only 모드의 경우 송신 장비와 수신 장비 간 iCloud 인증서가 송수신되는 것을 확인할 수 있었으며, Everyone 모드의 경우 수신 장비만 송신 장비에게 자가 서명 인증서를 보내는 것을 확인하였다[5].

III. 제안하는 기법

Bonjour 프로토콜의 취약점을 이용하여 발생하는 중간자 공격과 신분위장 공격의 가능성을 줄이기 위해 본 장에서는 Bonjour 프로토콜을 위한 ECDH 기반 인증 프로토콜을 제안한다.

1. 가정 사항 및 사용되는 표기법

본 절에서는 제안하는 프로토콜의 가정 사항 및 사용되는 표기법에 대해 설명하며, 다음은 ECDH 기반 인증을 Bonjour 프로토콜에 적용하기 위한 가정 사항에 대해 설명한다.

1) 서비스를 제공하는 장비는 Publication 과정에서 IP 주소, 도메인 주소, 포트 번호와 같은 설정 정보를 해시하고, 개인키를 통해 암호화하여 로컬 네트워크에 멀티캐스팅

2) 서비스를 사용하고자 하는 장비는 Discovery 과정 중 Publication에서 멀티캐스팅한 설정 정보를 수신하여 서비스를 제공하는 장비의 공개키를 통해 검증

3) 통신을 하고자 하는 각 통신 주체는 서로의 공개키를 알고 있어야함

4) 각 통신 주체는 타원 곡선 개인 값을 선택하기 전에 서로 동일한 타원 곡선 $E(F_q)$, 생성원 $G \in E(F_q)$, 생성원의 위수 n을 미리 알고 있다고 가정

5) Resolution 과정 전에 ECDH 기반 키 합의가 안전하게 이루어졌다고 가정

6) 각 통신 주체는 통신할 때 요청 메시지는 멀티캐스팅하고, 요청에 대한 응답 메시지는 유니캐스팅

아래 표 1은 제안하는 프로토콜에서 사용되는 표기법을 나타낸다.

표 1. 제안하는 프로토콜에서의 표기법

Notation	Definition
Dev_T	Total Device in the Local Network
Dev_A	Device A (Service User)
Dev_B	Device B (Service Provider)
Pri_x	$Dev_x's$ Private Key
Pub_x	$Dev_x's$ Public Key
d_x	$Dev_x's$ Elliptic Curve Private Value
H_x	$Dev_x's$ Elliptic Curve Public Key
S_x	$Dev_x's$ Signified Elliptic Curve Public Key
K_x	$Dev_x's$ Shared Key
G	Domain Parameter
$E_{Pri_x}(i)$	Encryption is performed on input value i using $Dev_x's$ Private Key
$D_{Pub_x}(i)$	Decryption is performed on input value i using $Dev_x's$ Public Key
$E_K(i)$	Encryption is performed on input value i using Shared Key K
$D_K(i)$	Decryption is performed on input value i using Shared Key K
Req_s	SRV record request message
Res_s	SRV record response message
Req_t	TXT record request message
Res_t	TXT record response message
N_s	Encrypted SRV record type request message
C_s	Encrypted SRV record type response message
N_t	Encrypted TXT record type request message
C_t	Encrypted TXT record type response message

2. 시나리오

본 절에서는 제안하는 기법의 시나리오에 대해 설명한다. 본 과정의 통신 주체는 크게 Dev_A와 Dev_B이다. Dev_A와 Dev_B가 통신할 때, 다음과 같은 동작을 통해 통신 주체 간에 공유키를 수립하고, 임의의 난수와 공유키를 이용하여 Bonjour 프로토콜의 Resolution 과정을 수행한다.

1) Dev_A와 Dev_B는 타원 곡선 개인 값 d_A와 d_B를 선택하고, 자신의 개인 값과 G(Domain Parameter)를 사용하여 연산을 통해 타원 곡선 공개키 H_A와 H_B를 생성
2) 각 통신 주체는 자신의 개인키를 이용하여 타원 곡선 공개키 서명
3) 서명된 타원 곡선 공개키를 서로 교환
4) 상대 장비의 공개키를 통한 검증을 수행하여 타원 곡선 공개키를 얻고, 자신의 타원 곡선 개인 값과 상대 장비의 타원 공개키를 이용하여 공유키인 K_A와 K_B를 생성
5) 공유키를 서로 교환
6) K_A와 K_B를 비교하여 값이 같으면 Bonjour 프로토콜의 Resolution 과정을 수행하고, 공유키가 다를 경우 세션을 수립하지 않음

3. 동작 과정

다음의 동작 과정은 Publication, Discovery 과정을 거친 후 Resolution 과정 수행 전에 안전한 세션을 수립하기 위한 비대칭키 기반 ECDH 키 합의 과정을 수행한 후, Resolution 과정을 수행하는 과정이다. Dev_A와 Dev_B는 각자의 공유키를 만들어 비교하는 과정을 수행하고, 공유키가 같다면, Resolution 과정을 수행하지만, 공유키가 다를 경우 장비 간 세션을 수립하지 않는다. 이러한 동작 과정을 위해 Resolution 과정을 비대칭키 기반 ECDH 키 합의 과정, SRV 레코드 메시지 인증과정, TXT 레코드 메시지 인증 과정 세 단계로 나누어 설명한다. SRV 레코드는 DNS에서 도메인 이름과 포트 번호를 저장하기 위해 사용되는 레코드이다. SRV 레코드를 사용하여 서비스를 사용하고자 하는 장비는 Discovery 과정에서 얻은 인스턴스 이름을 통해 도메인 이름과 포트 번호를 수신한다. TXT 레코드는 주어진 호스트에 대해 IPv6 주소를 알려주는 레코드이다. TXT 레코드를 사용하여 서비스를 사용하고자 하는 장비는 SRV 레코드 메시지를 통해 얻은 도메인 이름을 통해 IPv6 주소 값을 수신한다. SRV 레코드와 TXT 레코드 메시지 송수신 과정에서 별도의 메시지 인증 방법이 존재하지 않아 발생하는 취약점을 간단한 메시지 인증 방법인 Challenge-response를 통해 보안한다[7]. 다음 그림 5는 비대칭키 기반 ECDH 키 합의 과정, SRV 레코드 메시지 인증 과정, TXT 레코드 메시지 인증 과정을 상세히 나타낸다.

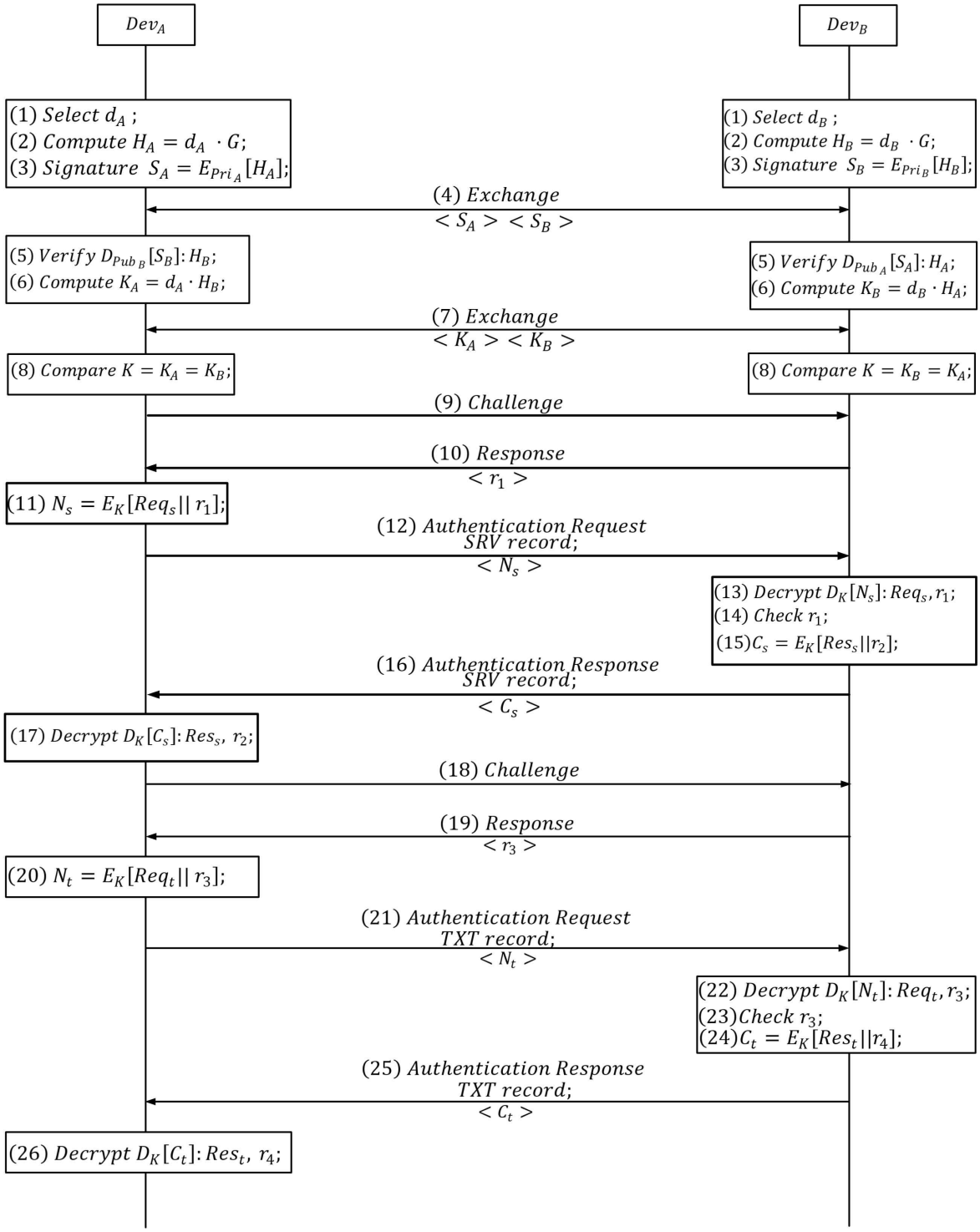

그림 5. ECDH 기반 인증 프로토콜을 적용한 Bonjour 프로토콜

가. 비대칭키 기반 ECDH 키 합의 과정

(1) 각 통신 주체인 Dev_A와 Dev_B는 자신의 개인 값 d_A와 d_B를 선정

(2) (1)에서 선정한 개인 값 d_A, d_B와 각 통신 주체가 이전에 합의했던 Domain Parameter G 를 이용해 타원 곡선 공개키인 $H_A = d_A \cdot G$와 $H_B = d_B \cdot G$ 를 생성

(3) (2)에서 생성한 타원 곡선 공개키 H_A와 H_B를 자신의 개인키 Pri_A와 Pri_B를 이용하여 서명한 서명 값 S_A와 S_B 생성

(4) 각 통신 주체는 S_A와 S_B를 교환

(5) 각 통신 주체는 상대방의 공개키 Pub_A와 Pub_B를 이용하여 서명 값을 검증하고 상대방의 타원 곡선 공개키를 얻음

(6) Dev_A와 Dev_B는 자신의 개인 값과 상대방의 타원 곡선 공개키를 이용하여 공유키인 $K_A = d_A \cdot H_B$와 $K_B = d_B \cdot H_A$를 생성

(7) Dev_A와 Dev_B는 (6)에서 생성된 공유키 K_A와 K_B 교환

(8) Dev_A와 Dev_B는 (6)에서 생성한 자신의 공유키와 상대의 공유키 비교

상대방의 공유키가 자신의 공유키와 같지 않다면, 상대방과의 세션을 수립하지 않고, 자신의 공유키와 상대방의 공유키가 같은 경우 세션을 수립하고 다음 Resolution 과정을 수행한다.

나. SRV 레코드 메시지 인증 과정

(9) Dev_A는 Dev_B에게 $Nonce$값을 요청

(10) Dev_B는 $Nonce$값을 r_1으로 Dev_A에 응답

(11) Dev_A는 (10)에서 받은 r_1과 SRV 레코드 타입의 요청 메시지 Req_s를 공유키 K로 암호화하여 암호화된 메시지인 $N_s = E_k[Req_s \| r_1]$를 생성

(12) Dev_A는 (11)에서 생성한 N_s를 Dev_B에 송신

(13) Dev_B는 공유키 K로 암호화된 요청 메시지인 N_s를 복호화하여 요청 메시지인 Req_s와 난수 r_1확인

(14) Dev_B는 복호화하여 얻은 난수 값 r_1이 (10)에서 자신의 보낸 난수 값 r_1과 일치하는지 확인 후 일치하지 않는다면, 다음 과정을 수행하지 않고, 일치한다면 다음 과정을 수행

(15) Dev_B는 Dev_A의 요청 메시지에 대한 응답 메시지 Res_s와 난수 값 $r_2 = r_1 + 1$를 공유키 K로 암호화하여 암호화된 응답 메시지인 $C_s = E_K[Res_s \| r_2]$ 생성

(16) Dev_B는 (15)에서 생성한 메시지 C_s를 Dev_A에 송신

(17) Dev_A는 공유키 K로 암호화된 응답 메시지인 C_s를 복호화하여 응답 메시지인 Res_s와 난수 값 r_2 확인

Dev_A는 이 과정을 통해 자신이 요청했던 서비스를 제공 가능한 장비 Dev_B의 도메인 이름과 포트 번호를 수신하여 확인하였다. SRV 레코드 메시지 인증 과정에서 얻은 도메인 이름을 통해 TXT 레코드 인증 과정에서 Dev_B의 IP 주소를 수신하고 확인한다.

다. TXT 레코드 메시지 인증 과정

(18) Dev_A는 Dev_B에게 $Nonce$값 요청

(19) Dev_B는 $Nonce$값 r_3로 Dev_A에 응답

(20) Dev_A는 (19)에서 받은 r_3와 TXT 레코드 타입의 요청 메시지 Req_t를 공유키 K로 암호화하여 암호화된 요청 메시지인 $N_t = E_K[Req_t \| r_3]$ 생성

(21) Dev_A는 (20)에서 생성한 메시지 N_t를 Dev_B에게 송신

(22) Dev_B는 공유키 K로 암호화된 요청 메시지인 N_t를 복호화하여 요청 메시지인 Req_t와 난수 값인 r_3를 얻음

(23) Dev_B는 복호화하여 얻은 난수 값 r_3가 (19)에서 자신이 보낸 난수 값 r_3와 일치하는지 확인하고, 만약 일치하지 않는다면, 세션을 중단하고, 일치한다면 다음 과정을 수행

(24) Dev_B는 Dev_A의 요청 메시지에 대한 응답 메시지 Res_t와 난수 값 $r_4 = r_3 + 1$를 공유키 K로 암호화하여 암호화된 응답 메시지인 C_t를 생성

(25) Dev_B는 (24)에서 생성한 메시지 C_t를 Dev_A에 송신

(26) Dev_A는 공유키 K로 암호화된 응답 메시지인 C_t를 복호화하여 응답 메시지인 Res_t와 난수 값 r_4를 확인

TXT 레코드 인증 과정을 통해 Dev_A는 서비스를 제공하는 장비인 Dev_B의 IPv6 주소 값을 수신하고 확인하였으며, Dev_A는 특정 서비스를 제공하는 장비의 설정 정보(도메인 이름, 포트 번호, IP 주소)를 확인하였다. 따라서 장비의 IP 주소나 호스트 이름이 변경되더라도 같은 로컬 네트워크에 참여할 경우 해당 서비스를 제공하는 장비를 재탐색하지 않고 자동으로 연결을 수립할 수 있다.

Ⅳ. 보안 분석

Bonjour 프로토콜의 Resolution 과정에서 별도의 메시지 인증 방법이 없다는 취약점을 통해 중간자 공격이 가능함을 확인하였고, Bonjour 프로토콜을 활용한 서비스인 Airdrop Everyone 모드에서 자가 서명 인증서를 사용하여 별도의 사용자 인증을 하지 않는 취약점으로 인해 신분 위장 공격이 가능함을 확인하였다.

이를 보안하기 위해 제안하는 기법인 ECDH 기반 인증 프로토콜을 적용한 Bonjour 프로토콜을 통해 Bonjour 프로토콜의 동작 과정에서 발생할 수 있는 취약점과 Bonjour 프로토콜을 활용한 서비스인 Airdrop Everyone 모드에서의 취약점으로 인해 발생할 수 있는 공격의 가능성을 줄였다.

Resolution 과정 전에 비대칭키 기반 ECDH 키 합의 방식을 적용하여 서명/검증을 수행함으로써 Airdrop 서비스의 Everyone 모드에서 별도의 사용자 인증이 존재하지 않아 발생하는 신분 위장 공격의 가능성을 줄일 수 있음을 증명하였다. 또한, 비대칭키 기반 ECDH 키 합의 과정에서 생성

한 공유키와 간단한 메시지 인증 기법인 Challenge-response를 Resolution 과정에 적용함으로써 별도의 메시지 인증 방법이 존재하지 않아 발생하였던 중간자 공격의 가능성을 줄일 수 있음을 증명하였다. 하지만 제안하는 기법인 ECDH 기반 인증 프로토콜의 경우 기존의 Diffie-Hellman 키 교환 방식을 사용하였기 때문에 중간자 공격에 대응하기에는 한계점이 존재한다. 아래 표 2는 기존 Bonjour 프로토콜과 제안하는 프로토콜의 차이에 대해 표로 나타내었다.

표 2. Bonjour 프로토콜과 제안하는 프로토콜의 차이

	Bonjour protocol	Proposed protocol
MITM attack	O	△
Masquerade attack	O	X
Key agreement	X	O
Message authentication	X	O
User authentication	X	O

V. 결 론

애플사는 장비 간 간편한 장비 탐색을 위해 자체적인 프로토콜인 Bonjour 프로토콜을 개발하였고, Bonjour 프로토콜을 활용한 서비스로는 Airdrop이 있다. Black Hat에서 Bonjour 프로토콜의 중간자 공격 취약점을 발표하였으며[4], 패킷 분석을 통해 Airdrop에서 Everyone 모드의 경우 별도의 사용자 인증이 존재하지 않아 신분 위장 공격에 취약하다는 것을 확인하였다[5]. 이러한 취약점을 보안하기 위해 본 논문에서는 비대칭키 기반 ECDH 인증 프로토콜을 적용함으로써 안전한 세션을 수립하였다. 이를 통해 신분 위장 공격을 방지하며, Resolution 과정에서 비대칭키 기반 ECDH 키 합의 과정을 통해 생성한 공유키와 간단한 메시지 인증 방법인 Challenge-response를 통해 메시지 인증을 수행함으로써 중간자 공격의 가능성을 줄일 수 있음을 증명하였다[7]. 하지만 제안하는 기법인 ECDH 기반 인증 프로토콜을 적용한 Bonjour 프로토콜의 경우 기존의 Diffie-Hellman 키 합의 방식을 사용하였기 때문에 중간자 공격에 취약한 상황이다. 추후 모바일 기기의 가상화 기반 도메인 분리 보안 플랫폼을 통해 취약점으로부터 발생할 수 있는 문제를 보안하고자 하며[8], 네트워크 보안 로그를 시각화함으로써 비정상적인 행위를 확인하고자 한다[9].

REFERENCES

[1] F. Siddiqui, S. Zeadally, T. Kacem, and S. Fowler, "Zero Configuration Networking: Implementation, performance, and security," *Computers & electrical engineering*, vol. 38, no. 5, pp. 1129-1145, 2012.

[2] iOS Security. https://www.apple.com/business/docs/site/iOS_Security_Guide.pdf (accessed Jun., 2019).

[3] Bonjour Operations. https://developer.apple.com b/library/archive/documentation/Cocoa/Conceptual/NetServices/Articles/NetServicesArchitecture.html (accessed June, 2019).

[4] X. Bai, et al., "Staying Secure and Unprepared: Understanding and Mitigating the Security Risks of Apple ZeroConf," *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 655-674, San Jose, USA, 2016.

[5] Y.J. Song, et al., "Analysis of Airdrop Packets and Known Vulnerabilities," *Proc. of Symposium of the Korean Institute of communications and Information Sciences*, pp. 987-988, June, 2018.

[6] S.H. Kim, "Comparison and analysis on efficiency of scalar multiplication for Elliptic Curve Cryptosystem", *M. S. dissertation, Korea Maritime and Ocean University graduate school*, Busan, 2003.

[7] S.h. Kwon, et al., "Simple Challenge-Response Authentication for Apple's Bonjour Protocol," *Proc. of Symposium of the Korean Institute of communications and Information Sciences*, pp. 989-990, June, 2018.

[8] J.N. Kim, "Implementation of Virtualization-based Domain Separation Security Platform for Smart Devices," *Smart Media Journal*, vol. 5, no. 4, pp. 116-123, 2016.

[9] W.J. Joe, H.J. Shin, and H.S. Kim, "A log visualization method for network security monitoring," *Smart Media Journal*, vol. 7, no. 4, pp. 70-78, 2018.

저 자 소 개

권순홍(준회원)

2016년 3월~현재 상명대학교 컴퓨터공학과 재학

<주관심분야 : 네트워크 보안, 시스템 보안>

이종혁(정회원)

2010년 2월 성균대학교 공학박사
2009년 6월~2012년 2월 프랑스 INRIA 연구원
2012년 3월~2013년 8월 프랑스 그랑제꼴 TELECOM Bretagne 조교수
2013년 9월~현재 상명대학교 소프트웨어학과 부교수

<주관심분야 : 프로토콜 엔지니어링 및 정보보호>