情報保護學會論文誌
第 22 卷 第 5 號, 2012. 10

POS시스템 내 소프트웨어 무결성 검증 방안*

조 성 아,† 김 성 훈, 이 동 훈‡
고려대학교 정보보호대학원

Software integrity verification method in POS system*

Sung-A Cho,† Sung Hoon Kim, Dong Hoon Lee‡
Graduate School of Information Security, Korea University

요 약

POS단말기란 카드결제 가맹점의 판매정보를 실시간으로 관리하는 시스템으로, 카드결제 기능이 함께 탑재되어 있어 판매 및 고객관리에 편의성을 제공해준다. 이러한 장점으로 인해 많은 가맹점들이 POS단말기를 설치하여 사용하고 있지만 내부에 저장된 매출정보, 카드유효기간, 비밀번호, 카드 검증 값 등과 같은 결제정보가 외부의 해킹이나 내부자의 부정으로 인해 카드회원 신용정보 유출 및 위조카드 등과 같은 사고의 원인이 되고 있어 해결책이 시급한 시점이다.

본 논문에서는 POS단말기의 해킹 및 위·변조로 인해 발생하는 개인정보 유출과 관련된 사고를 사전에 방지하기 위하여 화이트 리스트 기반의 POS 시스템 내 소프트웨어 무결성 검증 기법을 제안한다. 제안된 방식은 소프트웨어의 무결성을 제공하여 현재 암호화와 보안 솔루션에 의해 검증되어 설치된 프로그램의 변조를 방지하여 외부로부터의 위협 뿐 아니라 내부자에 의한 개인정보 유출 및 부정사용을 사전에 방지할 수 있다.

ABSTRACT

A Point-of-Sales (POS) terminal manages the selling process by a salesperson accessible interface in real time. Using a POS system makes a business and customer management much more efficient. For these reasons, many store install POS terminal and used it. But it has many problem that stealing personal information by hacking and insider corruption. Because POS system stored payment information like that sales information, card valid period, and password. In this paper, I proposed software integrity verification technique in POS system based on White list. This method can prevent accidents that personal information leak by hacking and POS system forge and falsification. This proposed method provides software integrity, so it can prevent inside and outside threats in advance.

Keywords: POS system security, Software Integrity, Authentication

I. 서 론

인터넷의 발달로 인해 등장한 웹 기반의 POS는 현재 많은 가맹점에서 쓰이고 있다. POS는 매출 관리뿐만 아니라 손익 관리, 재고 관리, 고객 관리, 사원 관리 등 다양한 기능이 컨버전스 되어 있어 사용자에게 편리성을 제공하고 있다. 현재 POS단말기는 약 40만대에 이르며 가맹점 수는 더 늘어나는 실정이다[1].

이와 같은 POS기기의 급속한 성장과 비례하여 이를 대상으로 하는 공격 또한 증가하고 있으며 이로 인

접수일(2012년 7월 3일), 수정일(1차: 2012년 8월 8일, 2차: 2012년 8월 30일), 게제확정일(2012년 9월 13일)
* 본 연구는 방위사업청과 국방과학연구소의 지원으로 수행되었습니다.
† 주저자, naissa@korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr

해 카드 위·변조 추이는 점점 늘어나고 있는 실정이다.

이러한 사고는 POS 기기의 외, 내부의 위협 때문에 발생한다. POS는 매장 관리자에게 다양한 기능을 통하여 상점을 보다 통합적으로 관리할 수 있도록 해주는 편리한 기능이 있는 반면에, 온라인을 통하여 결제가 이루어지고 그에 따른 모든 정보가 기기에서 취급되기 때문에 외, 내부의 위협에 쉽게 노출될 수밖에 없다[2].

즉, 내부적으로는 직접 단말기를 취급하는 내부자에 의한 정보의 유출 위협이 존재하고 외부적으로는 악성 코드나 바이러스의 위협, PC에서의 취약점에 대한 공격 등에 대한 취약점이 존재한다.

대부분의 악의적인 공격자들은 악성 소프트웨어를 설치하여 원격제어, 캡쳐 등의 방법을 통해 정보들을 실시간으로 빼가는 방법을 사용한다. 실제로 외국에서는 POS내에 악성코드가 설치되어 매장 이용자가 서비스를 이용하기 위해 신용카드를 건네고, 매장 운영자가 해당 카드로 결제하기 위해 POS를 동작시키는 순간에 화면을 캡쳐하여 외부로 전송하는 기능을 가진 악성코드가 발견된 사례가 있다[3].

본 논문에서는 앞서 언급한 바와 같이 POS단말기에서 발생하는 보안 사고를 사전에 방지하기 위한 방안을 연구하였다. 즉, POS단말기의 해킹 및 소프트웨어 위·변조로 인해 발생하는 개인정보 유출과 관련된 사고를 사전에 방지하기 위해 단말기의 소프트웨어의 무결성을 증명하는 바이너리 이미지를 관리하는 방안을 제시한다. 현재 POS의 개인정보 및 해킹 위험을 위해 실행된 연구에는 White list 기반의 프로그램, 종단간 암호화 통신인 E2E가 있지만 이는 POS 기기 내의 소프트웨어 자체의 변조를 방지하지는 않으며 POS내 소프트웨어의 변조를 방지하는 연구는 진행된 적이 없다. 하지만 암호화에 의한 안전한 전송과 보안 솔루션에 의해 검증되어 안전하게 설치된 프로그램이라 할지라도 설치된 이후 해당 소프트웨어의 변조에 대한 무결성을 보장해 주지 않는다면 외부 및 내부 위협들에 안전하다고 할 수 없다[4]. 제안된 방식은 White list 기반으로 소프트웨어의 무결성 검증을 통해 외부로부터의 위협 뿐 아니라 내부자에 의한 개인정보 유출 및 부정사용을 사전에 방지할 수 있다.

논문의 구성은 다음과 같다. 2장에서는 이와 관련된 기술 및 연구들을 알아보고, 3장에서는 이러한 POS 환경에서의 취약점을 정의하며, 4장에서는 본 논문에서 제안하는 무결성 보장 기법을 자세히 다룬다. 5장에서는 보안성 및 연산량을 분석하고 6장에서는 결론을 맺고 향후 연구 방향에 대해 논의할 것이다.

II. 관련연구

본 절에서는 제안하는 기법과 관련된 무결성을 보장해 주는 연구들에 대해서 알아볼 것이다. 또한 현재까지 POS시스템의 보안에 관련된 연구들에는 어떤 것들이 있는지 알아본다.

2.1 E2E

종단간(E2E)암호화는 전자금융거래에 있어 이용자 PC에서부터 금융사 전자금융거래 서버까지 이용자가 입력한 금융거래 데이터를 안전하게 보호기 위해 이루어진다. 즉, 이용자가 금융거래에 필요한 정보를 입력하는 시점부터 금융사 전자금융거래 서버로 데이터가 전송되어지기까지 안전한 암호 채널을 통해 전송이 이루어진다. 현재 연구되어지는 POS 시스템의 종단간(E2E)암호화는 전자금융거래에 있어 이용자 POS 단말기에서 VAN사 전자금융거래 서버까지 이용자가 결제한 금융거래 데이터를 안전하게 보호하는 것을 목적으로 한다. 이용자가 금융거래에 필요한 정보를 입력하는 시점부터 VAN사 금융거래 서버로 데이터가 전송되어지기까지 안전한 암호 채널을 통해 이용자 데이터를 전송하는 것이다[7].

2.2 White list 기반의 POS시스템

White list란 스팸 메일, 악성 코드를 유포하는 IP 주소, 피싱을 조장하는 허위 사이트 등을 데이터베이스(data base)로 만드는 Black List와는 반대되는 개념으로 알려진 IP 주소로 white list를 만들어 이로부터 전송된 이메일은 메일 서버가 언제나 수용하도록 하거나 은행, 각종 포털 사이트가 자발적으로 보안 업체나 단체에 white list로 등록해 웹 사이트의 안전성을 소비자에게 알려 주게 된다. 이의 대표적인 제품으로는 AhnLab TrusLine이 있다. 이 제품은 시스템의 안정적 운용에 대한 민감도가 높고, 정해진 프로그램만 사용하는 산업용 시스템에 최적화된 보안 솔루션이다. 즉, 관리자가 승인한 프로그램을 리스트에 등록하여 허용된 상태 그대로 동작하도록 운영된다. 특히 애플리케이션 제어, 비허가 실행 코드 차단, USB 등 매체 제어, IP/PORT 차단 등과 같은 기능을 갖추고 있어 허용된 프로그램만 실행 가능하게 함

으로써 악성코드의 침입은 물론 악성코드로 인한 정보 유출도 방지할 수 있다. POS 시스템에서 이러한 AhnLab TrusLine은 White list 기반 보안 솔루션으로 지정된 프로그램의 설치만 가능하게 하고 악성코드 방역 솔루션을 통해 감염을 차단하여 보안적인 측면을 강화하도록 한다[8].

III. POS 시스템 위협

이번 장에서는 POS 시스템을 위협하는 공격자의 모델을 정의하고, 이러한 공격자가 행할 수 있는 공격의 유형을 정의한다. 또한 안전한 POS시스템을 위한 보안요구사항에 대해 기술한다.

3.1 공격자 모델

공격자 모델은 내부공격자와 외부공격자 두가지로 나눌 수 있다.

3.1.1 내부공격자

공격자 모델에서 내부공격자의 범위는 POS단말기에 접근이 가능한 모든 공격자를 의미하는 것으로 가맹점 점원, POS 판매원 등을 포함한다. 내부자는 POS단말기에 접근이 가능하기 때문에 직접적으로 악의적인 활동이 가능하다. 예를 들어, 개인 정보 유출, 바이러스 설치, 시스템 변조 등이 그것이다. 실제 사례로 09년 11월 POS 판매원에 의해 시스템이 조작되어 결제 금액, 계좌의 변조로 인한 피해 사례가 있었다.

3.1.2 외부공격자

외부공격자란 POS단말기에 직접적인 접근이 불가능하지만 악의적인 목적을 가지고 간접적으로 접근을 시도하는 공격자를 말한다. 그들은 원격제어, 악성코드 설치 등을 통해 개인정보를 유출하는 방식으로 접근을 시도한다. 실제적으로 많은 공격 시도가 있었으며 대부분의 공격 기법들은 원격제어를 통한 암호화된 승인전문 정보 탈취나 악성코드 프로그램 등을 통한 승인전문 정보 실시간 탈취이다.

3.2 공격 유형

POS 시스템 공격 유형은 정보통신단체표준 TTAK.KO-12.0181 "POS시스템 보안 요구사항"을 기반으로 6가지 종류로 분류되었다[10].

키로깅이나 메모리 해킹 같은 경우 해당 공격이 유효하려면 POS 시스템 내에 악성 프로그램이 설치되어 있어야만 한다. 즉, White list 기반의 방어 솔루션이 POS내에 존재한다면 불법 프로그램이 설치되는 것을 사전에 차단할 수 있게 되므로 이러한 유형의 공격을 무력화 시킬 수 있다.

E2E 보안 솔루션의 경우 신용카드의 정보가 신용카드 단말기 또는 키로그를 통해 입력되는 순간부터 VAN서버에 저장되기까지 모든 구간이 암호화를 통해 데이터 유출을 방지한다. 즉, 탭핑이나 스키밍, 키로깅 같은 공격의 경우, 정보를 빼내가더라도 정보가 암호화 되어 있어 유효한 정보라고 할 수 없고, 재전송 공격 또한 Timestamp를 사용하여 무력화 시킬 수 있다. 마지막으로 E2E 보안 솔루션은 정보가 입력되는 순간부터 VAN서버까지 데이터가 암호화 되어 있기 때문에 POS 시스템 메모리 내에서 이루어지는 공격 또한 유효하지 않다고 할 수 있다.

하지만 소프트웨어 변조는 POS 단말기에 설치된 소프트웨어의 파일을 직접 변조시키는 공격 유형으로 프로그램의 형태뿐만 아니라 어떠한 실행 파일 형태로

〔표 1〕 POS기기 공격에 대한 각 방안의 방어 여부

공격유형	공격 내용	White list	E2E
탭핑	카드 입력기와 본체 사이의 케이블을 도청하여 정보를 빼내는 기법	×	○
스키밍	카드 투입구에 설치되어 기기에 투입된 카드 정보를 복사해 가는 공격	×	○
키로깅	단말기에 입력된 정보를 중간에서 가로채어 해당 정보를 해커에게 전송하는 공격 기법	○	○
재전송 공격	암호문이 어떤 거래를 시도하는데 사용된 것인지 안다면 암호문에 대한 재전송을 통해 거래를 성공시킬 수 있는 기법	×	○
메모리 해킹	이용자가 입력한 데이터 등이 메모리상에서 평문으로 암호화 또는 복호화 되는 구간을 포착하여 주요 금융 정보를 추출해 내는 공격 기법	○	○
소프트웨어 변조	POS프로그램이 악성코드를 통해 변조되어 주요 금융 정보 유출이 가능한 공격 기법	×	×

도 존재 가능하다. 그렇기 때문에 소프트웨어 형태로 배포되는 모든 솔루션을 무력화 시킬 수 있는 매우 위협적인 공격이라 할 수 있다. 즉, 위에서 제시된 White list 보안 솔루션 및 E2E 보안 솔루션은 이러한 공격에 취약할 수 있다. 이렇게 위협적임에도 불구하고, 아직까지 이를 방어하는 방안에 대한 연구는 없다. 본 논문에서는 이러한 소프트웨어 변조 공격에 대응할 수 있는 방어 기법을 제안하고자 한다.

3.3 POS 시스템 보안 요구 사항

3.3.1 인증

POS 기기의 경우, 사용하고자 하는 POS 기기가 정당한 사업자에 의해 만들어진 것인지, 등록된 기기가 정당한지에 대해 CA 및 VAN사에 인증하는 과정이 필요하다. 또한, POS를 사용하는 사용자가 정당한 사용자에 의해 관리/사용 되고 있는지에 대해 CA 및 VAN사에 인증을 받아야 한다. 이에 정당한 사용자가 아니라면 POS 단말기를 사용하거나 내부에 접근할 수 없도록 관리자에 대한 인증이 제공되어져야 한다. 마지막으로, POS 내에 사용되는 프로그램에 대해 인증과정이 필요하다. 이를 위해, POS 프로그램, 원격 제어프로그램 등 프로그램 이용 및 설치 시 원격지의 특정 IP 주소, 포트, 프로토콜 등만 허용하도록 POS 시스템 내에 설정하여, 이를 통해 인증이 이루어지도록 해야 한다.

3.3.2 기밀성

POS시스템 내부에 저장되어 있는 데이터와 외부로 송/수신 되는 데이터에 대한 적절한 보호조치가 필요하다. 즉, POS 단말기 내부에 저장되어 있는 데이터에 대해서 제 3자가 그 내용을 알 수 없도록 보호해야 하며, 외부로 데이터가 송/수신 될 때, 위/변조 되지 않은 적절한 데이터 보호가 이루어져야 한다.

3.3.3 무결성

무결성은 크게 데이터 측면과 소프트웨어 측면에서 살펴볼 수 있다. 먼저 데이터 측면에서 단말기 내부에 저장되어 있는 데이터와 외부로 송/수신 되는 데이터에 대해 위/변조 되지 않도록 적절한 보호가 필요하며, 소프트웨어의 경우 악성코드에 의해 해당 소프트웨어가 변조되었다면, 그 취약점으로 인해 개인정보 유출, 시스템 조작 등의 문제점을 유발시킬 수 있다. 따라서, POS 내 소프트웨어의 상태에 대해 신뢰할 수 있도록 무결성을 제공해야 한다.

〔표 2〕 보안 요구 사항과 만족 사항

보안사항	보안대상	요구사항
인증	POS기기	기기의 정당성 여부
	사용자	사용자에 대한 정당성 여부
	프로그램	POS 내 소프트웨어 대한 정당성 여부
기밀성	데이터	POS 내 저장된 데이터에 대한 보호
	송수신	송/수신 되는 데이터에 대한 보호
무결성	데이터	데이터 위/변조 방지
	소프트웨어	POS 내 소프트웨어에 대한 위/변조 방지
가용성	POS 시스템	POS 시스템의 오작동 및 고장에 대한 대비사항 마련

3.3.4 가용성

POS시스템에서 결제를 수행할 때 지연이 없도록 하여야 하며 오작동 및 기기의 고장 시 대비 사항을 마련해 놓아야 한다.

IV. 제안하는 무결성 보장 기법

IC 카드로 변경되는 정책이 진행 중인 가운데 이는 카드 내에서 연산이 행해지기 때문에 기존의 암호화 방식 E2E는 대응 사항이 되지 못하고 있고 연산이 되기 전 단계에서 변조가 되는 위협 존재하기 때문에 다른 대응방안이 필요한 실정이다. 현재 POS기기 보안에서 악의적인 공격자의 접근을 제어하기 위해 화이트 기반의 접근 솔루션은 사용되고 있지만 해당 소프트웨어의 변조에 대한 대책은 없는 실정이다. 제안방식은 POS시스템 내부의 소프트웨어의 무결성 보장 기법에 대해 제안함으로써 한층 더 안정성을 강화시켰다. 또한 현실에 본 시스템을 도입 시 대규모 기업을 대상으로 각 기업 단 CA에서 관리가 가능할 것이며 소규모 기업은 정부 및 공기관에서 관리가 가능할 수 있어 현실적으로도 도입 가능한 제안 시스템이다.

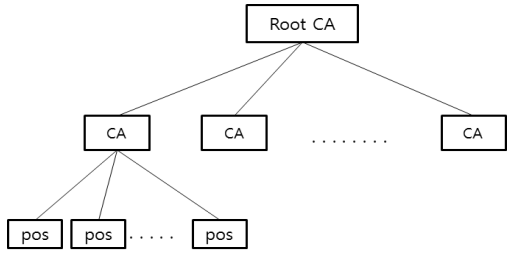
4.1 용어표기

- *Root CA* : 신뢰 기관
- *CA* : POS 소프트웨어 무결성 검증 기관
- P_i : I번째 등록된 POS 기기
- U_i : 사업자 정보
- N_{pos} : POS기기에서 생성한 난수 값
- K_i : P_i와 CA사이에 사전 공유한 키
- K_{CV} : CA와 VAN사 사이에 사전 공유한 키
- S_i : I번째 소프트웨어
- $ID(S_i)$: I번째 소프트웨어의 식별 값
- $B(S_i)$: I번째 소프트웨어 바이너리 이미지 값
- T_{P_i} : 재사용 공격 방지를 위해 P_i가 생성한 타임스탬프
- $E_K(m)$: 메시지를 키 K로 암호화한 값
- $MAC_K(m)$: I번째 Software의 바이너리 이미지 값을 키 K로 해시한 값
- m_{auten} : 인증 및 요청 메시지
- $m_{register}$: 새로운 소프트웨어 설치 요청
- SET_r : POS 내 소프트웨어들의 바이너리 이미지 값 중 랜덤하게 특정 값만을 불러 정해 놓은 r번째 값

4.2 제안하는 시스템 모델

제안하는 시스템은 White list 기반으로 정해진 프로그램만 사용할 수 있도록 하며 각 소프트웨어의 바이너리 값을 이용하여 가맹점에 설치된 POS 기기 소프트웨어의 무결성을 보장하는 시스템이다. POS 기기는 각각의 CA에 연결되어져 모니터링, 검증되며 각각의 CA는 Root CA단에서 전체적으로 통합 관리한다. 하나의 CA에 연결되어진 POS기기들은 모두 같은 기기, OS와 환경 등을 가지고 있는 것을 가정으로 한다[15]. 또한 White list 기반으로 설치 가능한 모든 software에 대한 정보는 POS기기 외에도 CA단에서 가지고 있어 설치시나 검증 시 편의성을 지닌다.

제안 시스템에서 VAN사는 CA로부터 소프트웨어의 MAC 값을 받는 과정을 가진다. 즉, 현재 VAN사에서는 거래요청을 승인해주는 역할만 하지만 제안 시스템에서는 CA에게 받은 MAC값을 소유하고 있어, 이를 통해 무결성 검증 역할도 동시에 수행한다. 각 POS기기마다 무결성 검사 주기를 설정할 수 있도록



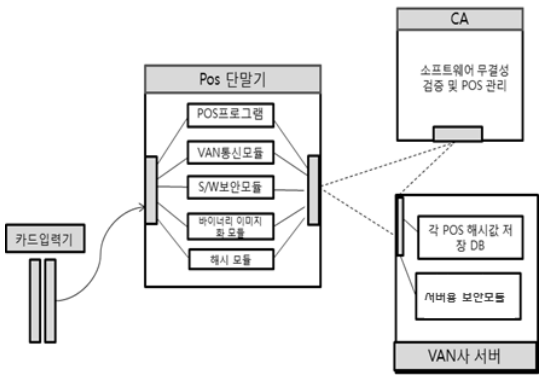
〔그림 1〕 인증 설계도

하고 주기가 상당히 짧은 경우는 성능과 소요 시간을 고려해 바이너리의 전체가 아닌 부분적인 값을 랜덤하게 사용해 검증을 하도록 한다. 부분적 값의 랜덤 사용 시 비용 및 서버 부하의 문제에서 강점을 가지게 된다.

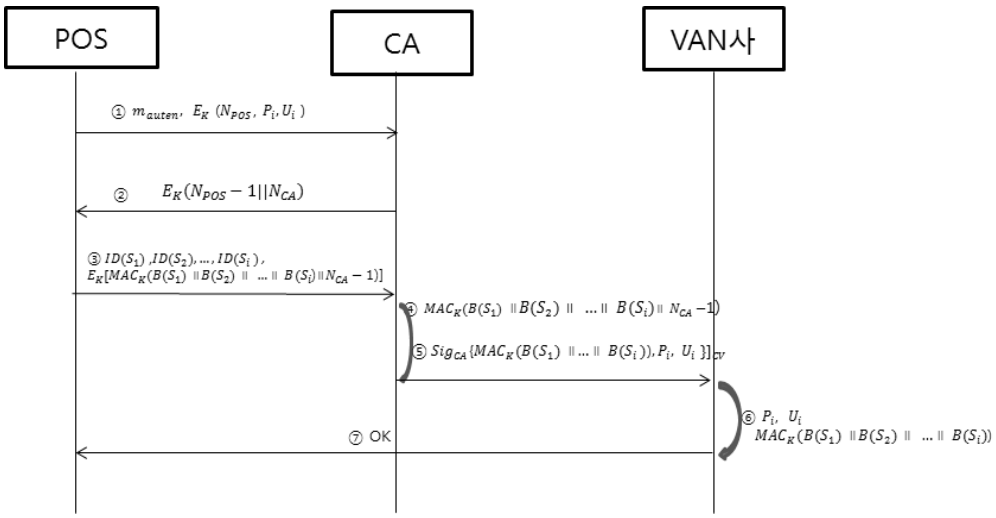
POS와 CA가 공유하는 키 즉, POS 내부에 저장되는 키는 제조공장에서 단말기의 안전한 하드웨어에 저장이 된다고 가정함으로써 안전하다고 할 수 있다. 또한 제조 과정에서 단말기의 식별번호와 신뢰성을 부여하고 이는 인증과정에서 활용하도록 한다. 한번 등록된 POS 같은 경우 그 식별번호로 이중적인 등록을 할 수가 없으므로 불법복제에 대한 부분은 해결이 가능하다.

4.3 제안 프로토콜

본 제안방식은 크게 POS 구입 시 등록 단계, 사용 시 무결성 검증 단계, 새로운 소프트웨어 설치 및 업데이트 단계로 나눌 수 있다. POS와 CA 사이에는 K_i, CA와 VAN사 사이에는 K_{CV}가 사전에 안전하게 공유되었음을 가정한다.



〔그림 2〕 제안하는 시스템 구조



〔그림 3〕 POS 등록 프로토콜

4.3.1 POS 구입 시 등록 단계

POS 등록단계는 POS기기 사용 시 처음 CA와 VAN사에 기기를 등록하는 단계로 〔그림 3〕과 같다.

1) POS → CA : m_{auten},, $E_K(N_{POS}, P_i, U_i)$
CA와 VAN사에 POS 기기 인증 및 등록을 위해 POS기기 인증 요청인 m_{auten}, POS 단에서 기기의 식별 값인 P_i와 사용자인 사업자 정보 U_i, POS에서 생성한 난수 N_{POS} 를 사전 공유한 키로 암호화 한 값을 CA로 보낸다.

2) CA → POS : $E_K(N_{POS}-1\|N_{CA})$
CA에서는 자신이 정당한 CA임을 인증하기 위해 $N_{POS}-1$ 값을 계산하고 임의의 값 N_{CA}를 생성하여 미리 공유한 키인 K_i로 암호화 하여 POS로 전송한다.

3) POS → CA : $ID(S_1)$, $ID(S_2)$,…., $ID(S_i)$, E_K〔$MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i) \| N_{CA}-1)$〕
POS는 현재 기기에 설치되어있는 소프트웨어들의 식별 값을 전송한다. 또한 POS 기기의 신뢰성을 증명하기 위해 미리 공유한 키로 각 바이너리 이미지들과 $N_{CA}-1$를 MAC한 값을 암호화하여 전송한다.

4) CA : $MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i) \| N_{CA}-1)$
CA는 Whitelist에 있는 각 소프트웨어들의 식별자 값을 통해 각각의 소프트웨어를 바이너리 이미지화 시키고 그 값들을 연결하여 2) 단계에서 보낸 임의의 값 $N_{CA}-1$과 연결하여 미리 공유한 키 K로 MAC시켜 일치여부를 확인한다.

5) CA → VAN사 : P_i, U_i, 〔$Sig_{CA}\{MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i)\}_{K_{CV}}$,
MAC_K ($\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i)$)값이 일치할 경우 CA에서는 자신의 정당성을 증명하기 위해 개인키로 서명하여 VAN사와 미리 공유한 키 K_{CV}로 암호화 한 값 〔$Sig_{CA}\{MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i)\}_{CV}$을 보낸다. 이와 동시에 MAC값과 일치하는 POS의 식별값인 P_i, 사업자 정보 U_i도 전송한다.

6) VAN사 : U_i ,P_i, $MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i))$ 등록
VAN사에서는 사전 공유한 키로 복호화 하고 CA의 공개키를 이용하여 CA의 진실성을 파악한 뒤 U_i, P_i와 $MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \mathrm{B}(S_i))$을 등록한다.

7) VAN사 → POS : OK
VAN사는 POS로 등록신청이 완료되었다는 메시지를 보낸다.

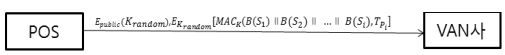
4.3.2 POS 사용 시 무결성 검증 단계

POS 기기 사용 시 무결성을 검증하는 방법으로는 두 가지의 방안을 제시한다. 이 두 가지의 방안은 자

체적인 검사 주기의 설정으로 인해 선택되어 질 수 있으며 검사 주기가 짧아질 경우 전체적인 속도, 성능적인 부분에서 문제가 될 수 있으므로 제안되었다.

- 검사 주기가 길 때 검증 과정

1) POS : $B(S_1)$, $B(S_2)$, … , $B(S_i)$
POS기기는 각 소프트웨어의 무결성을 검증하기 위해 내부의 각 Software, 즉 S_1, S_2 , …. , S_i들의 바이너리 이미지 값 $B(S_1)$, $B(S_2)$, … , $B(S_i)$을 생성한다.
2) POS : $MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i))$
각 바이너리 값을 통해 연속값을 생성하여 사전에 공유한 비밀 키를 통해 MAC 모듈로 $MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i))$ 출력한다
3) POS → VAN사 : $E_{public}(K_{random})$, $E_{K_{random}}$ 〔$MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i)$, T_{P_i}〕
POS기기는 CA와 사전 공유한 K를 VAN사의 공개키로 암호화하고 MAC값. 즉 $MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i))$와 재전송 공격을 방지하기 위한 T_{P_i}을 랜덤 키인 K_{random}로 암호화하여 결제정보와 함께 VAN사로 전송한다.
4) VAN사 : $MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i))$
POS기기로부터 온 값을 자신의 개인키를 이용해 K를 도출하고 그 K로 $MAC_K(B(S_1) \| B(S_2) \| \ldots B(S_i))$값을 확인한다. 그리고 VAN사단에 저장되어 있는 값의 일치 여부를 확인한다.
5) VAN사 → POS : OK or VAN → CA : Error
일치할 경우 무결성이 보장되는 것이므로 POS로 OK 메시지를 보내고, 일치하지 않을 시 이러한 정보를 CA로 POS 식별자와 함께 전송하여 후 조치가 취해지도록 한다.

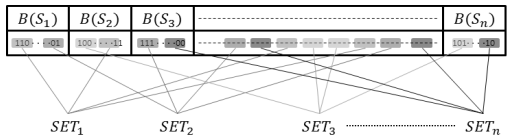


〔그림 4〕 검사 주기가 길어질 경우 POS 무결성 검증

- 검사 주기가 짧을 때 검증 과정

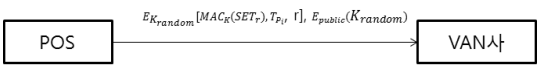
POS에 설치된 모든 Software들의 바이너리 이미지를 MAC 하는 것은 CPU load가 매우 큰 작업이기 때문에 검사 주기가 짧아질 경우 성능 면에서 문제가 발생할 수 있다. 따라서 CPU load를 줄이고, 짧은 주기마다 Software의 변조 여부를 검사하기 위해서 부분적으로만 해시함수화 하는 방안을 제시한다. 이러한 방안을 수행하기 위해서는 사전에 전체바이너리 이미지 중 부분적인 이미지 값들을 하나의 SET으로 설정해 두어야 하며 이러한 값들은 등록 당시 CA와 VAN사에 저장되어 있다는 것을 가정으로 하여 진행된다.

1) POS : SET_r = B(SET에 해당하는 바이너리 값)
POS에 설치된 Software의 무결성을 검증하기 위해 random으로 SET를 선택하고, SET에 해당하는 Software들의 특정 바이너리 값들을 불러와 이미지를 생성한다.



〔그림 5〕 검사 주기가 짧아질 경우 SET 설정

2) POS : $MAC_k(SET_r)$
선택된 SET의 연속된 바이너리 값을 사전에 공유한 비밀키를 통해 MAC 모듈로 $MAC_k(SET_r)$을 출력한다.
3) POS → VAN사 : $E_{K_{random}}[MAC_K(SET_r)$, T_{P_i}, r〕, $E_{public}(K_{random})$
MAC 모듈에 의해 출력된 값과 재전송 공격을 방지하기 위한 타임스탬프 T_{P_i}, 그리고 몇 번째 SET인지를 확인할 수 있는 r값을 랜덤 키인 K_{random}으로 암호화 하고 K_{random}을 CA의 공개키로 암호화한 값을 결제정보와 함께 VAN사로 전송한다.
4) VAN사 : $MAC_k(SET_r)$
받은 MAC값과 VAN사에 저장된 r번째 SET를 공유한 키로 해쉬 한 값이 일치하는지 검증한다.
5) VAN사 → POS : OK or VAN → CA : Error
일치할 경우 무결성이 보장되는 것이므로 POS로 OK 메시지를 보내고 ,일치하지 않을 시 이러한 정보를 CA로 POS 식별자와 함께 전송하여 후 조치가 취해지도록 한다.



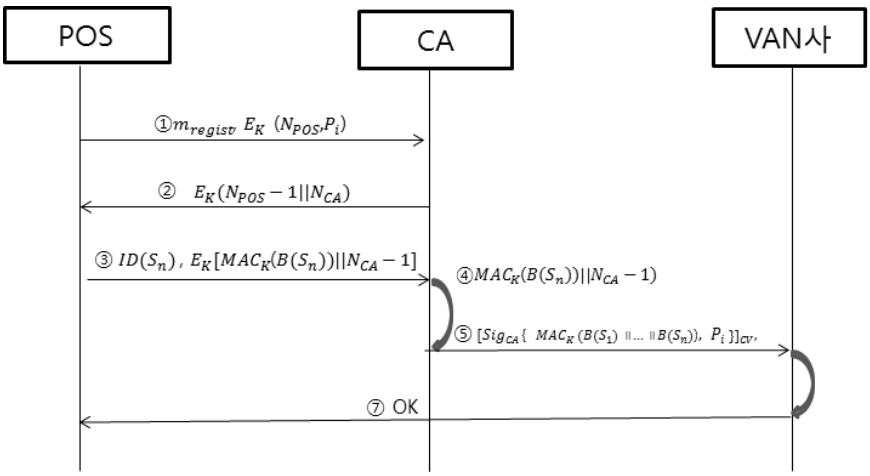
〔그림 6〕 검사 주기가 짧을 경우 POS 무결성 검증

이러한 방법은 소요시간이나 차지하는 CPU load가 적으므로 성능 면에서 우월성을 가지지만 SET에 포함되어 있지 않은 바이너리 값들은 검증이 안 된다는 단점을 지닌다. 이에 대한 대응책으로 매번 SET는 랜덤으로 다른 값이 나오게 설정할 것이다, 또한 이 방안 검증 단계에서는 새로운 소프트웨어 추가나 삭제 시 전체적인 SET을 다시 설정, 등록해야 하는 번거로움을 지닌다.

4.3.3 새로운 소프트웨어 추가, 업데이트 단계

새로운 소프트웨어를 추가하거나 업데이트할 때 소프트웨어들의 바이너리 값을 통해 무결성을 검증하기 위해서는 CA와 VAN사에 바뀐 내용들에 대한 등록이 필요하다. 이 단계는 새로운 소프트웨어를 추가하고 업데이트하는 단계로 〔그림 7〕과 같이 이루어진다.

1) POS → CA : m_{regist}, $E_K(N_{pos}, P_i)$
   POS는 새로운 소프트웨어를 등록하기 위해서, 또는 기존의 소프트웨어를 업데이트하기 위해서 CA로 새로운 소프트웨어 설치 메시지인 m_{regist}와 자신의 식별 값인 P_i, 그리고 생성한 난수 N_{pos}를 미리 공유한 키 K로 암호화하여 전송한다.
2) CA → POS : $E_K(N_{pos-1} \| N_{CA})$
   CA에서는 이에 대한 응답 값으로 POS에게 전달받은 $N_{pos}-1$과 자신이 생성한 난수 N_{CA}를 연결하여 사전 공유한 키 K로 암호화하여 전송한다.
3) POS → CA : $ID(S_n)$, $E_K〔MAC_K(\mathrm{B}(S_n))||N_{CA}-1〕$
   POS는 새롭게 설치할 소프트웨어의 식별 값과 설치할 소프트웨어들의 무결성을 검증하기 위해 이의 바이너리 이미지화 한 값의 MAC 값과 전 단계에서 받은 랜덤값 N_{CA}에서 1을 뺀 값을 연결하여 미리 공유한 키 K로 암호화여 전송한다.
4) CA : $MAC_K(\mathrm{B}(S_n)||N_{CA}-1)$
   CA는 White list에 있는 식별자 값을 통해 설치할 소프트웨어를 바이너리 이미지화시킨 값과 자신이 보낸 난수 $N_{CA}-1$과 연결하여 미리 공유한 키 K로 MAC하여 일치여부를 확인한다.
5) CA → VAN사 : $〔Sig_{CA}\{MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots \| \mathrm{B}(S_i) \| \mathrm{B}(S_n)\}_{CV}$, P_i
   $MAC_K(\mathrm{B}(S_n)||N_{CA}-1)$값이 일치할 경우 CA에서는 자신의 개인키로 서명하여 VAN사와 미리 공유한 키 K_{CV}로 암호화 한 값 $〔Sig_{CA}\{MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots \| \mathrm{B}(S_i) \| \mathrm{B}(S_n)\}_{CV}$와 POS 식별 값인 P_i를 VAN사로 보낸다.
6) VAN사 : P_i, $MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots \| \mathrm{B}(S_i) \| \mathrm{B}(S_n)$)등록
   VAN사에서는 사전 공유한 키로 복호화 하고 CA의 공개키를 이용하여 CA의 진실성을 파악



〔그림 7〕 새로운 소프트웨어 추가 프로토콜

한 뒤 P_i와 $MAC_K(\mathrm{B}(S_1) \| \mathrm{B}(S_2) \| \ldots, \| \mathrm{B}(S_i) \| \mathrm{B}(S_n)$을 등록한다.

7) VAN사 → POS : OK
VAN사는 POS로 새로운 소프트웨어 등록이 완료 되었다는 메시지를 보낸다.

V. 분석

본 제안 방식은 POS 단말기 내 소프트웨어의 무결성을 소프트웨어의 바이너리 이미지 값의 MAC 값을 이용하여 제안하였다. 즉, 앞에서 언급한 보안요구사항의 인증과 소프트웨어 무결성 측면의 문제점을 해결하고 보안성을 제공한다.

5.1 인증

〔정리 1〕 인증 프로토콜 단계 1~3에서 정당한 POS기기와 사용자만이 CA의 인증을 받을 수 있고, 정당한 CA만이 POS의 인증을 받을 수 있다.

〔증명〕 등록 과정 중 POS가 CA를 인증하기 위해서는, POS는 난수를 생성하여 사전에 공유한 키로 암호화하여 CA로 전송하여야 한다. 사전에 키를 안전하게 공유했다고 가정하였으므로 K 없이 POS와 CA를 제외한 어느 누구도 이 값을 알아낼 수 없다. 따라서 CA가 정당한 기관이라면 POS에서 보낸 난수에서 1을 뺀 값을 자신이 생성한 난수와 연결해 K로 암호화하여 보낼 수 있을 것이다. 또한 CA는 POS가 정당한 기기임을 인증하기 위해 CA단에서 생성한 난수를 보낼 경우 K로 암호화하여 보내고 이에 1을 뺀 값을 다음단계에서 받는다.

〔정리 2〕 인증 프로토콜에서 정당한 CA만이 VAN사의 인증을 받을 수 있고, 정당한 VAN사만이 서로의 인증을 받을 수 있다.

〔증명〕 등록 과정에서 CA가 VAN사에게 자신을 인증하기 위해서 VAN사로 자신의 개인키로 암호화한 MAC 값을 사전에 CA와 VAN사와 미리 공유한 CV로 암호화하여 전송한다. 이는 사전에 공유한 키가 없거나 CA만이 소유하고 있는 개인키로 암호화하기 때문에 CA가 정당한 자신임을 입증할 수 있는 수단이 된다. 또한 VAN사는 CA의 공개키로 정당한 CA임을 증명하며, 사전 공유한 키 CV로 복호화하여 자신임을 입증하고 데이터를 저장할 수 있다.

〔정리 3〕 인증 프로토콜 단계 3~4에서 POS내 정당한 프로그램만이 CA의 인증을 받을 수 있다.

〔증명〕 등록 과정에서 POS기기는 POS 내 프로그램이 정당한지에 대해 증명하기 위해 설치할 프로그램의 식별번호를 전송한다. 본 제안 시스템은 White list 기반 시스템으로써 사전에 설치 가능한 프로그램이 정해져 있음에 따라 이 프로그램이 정당한 프로그램인지에 대해 사전에 판단된다.

5.2 기밀성

〔정리 4〕 POS내 저장된 데이터 및 송, 수신되는 데이터에 대한 기밀성이 보장 되어져야 한다.

〔증명〕 POS 내의 데이터를 보호하기 위해 내부 안전한 하드웨어에 저장된 키로 암호화 하여 보관하며 데이터 송, 수신시에도 각 과정에서 사전 공유한 키 'K'로 암호화하여 전송한다.

5.3 무결성

〔정리 5〕 POS 시스템 내 설치되어 있거나 설치될 소프트웨어는 변조가 어렵다.

〔증명〕 등록 시 POS기기에서는 설치된 소프트웨어의 정당한 바이너리 값을 CA를 통해 VAN사에 등록하고 검사 중에는 실시간의 소프트웨어의 바이너리 값을 출력하여 보내기 때문에 소프트웨어의 변조가 불가능하다. 또한, 새로운 소프트웨어 설치 시에도 Whitelist기반으로 CA단에서 소프트웨어의 정당성에 대한 확인을 거쳐 등록을 하여 소프트웨어 무결성에 대해 보장한다고 할 수 있다.

5.4 연산량

〔정리 6〕 제안 방법이 실행될 때 연산량은 다음과 같다. 아래 표는 각 단계에 대한 연산량에 대한 설명이다. 등록단계는 한 기기 당 1번만 이루어지며, 검증단계는 주기 설정에 따라 하루에 여러 번 발생할 수 있다. 또한 소프트웨어 추가 단계는 새로운 소프트웨

〔표 3〕 각 단계별 연산량

구 분		연산량
제안 프로토콜	등록 단계	3s + 1p + 1h + 5f
	검증 단계 (짧은 주기)	1s + 1p + 1h + f
	검증 단계 (긴 주기)	1s + 1p + 1h + f
	소프트웨어 추가 단계	1s + 1p + 1h + 5f

어를 추가할 시에만 일어나기 때문에 많은 경우가 일어나지 않는다. 다음 표에서 s는 대칭키 연산, p는 비대칭키 연산, h는 해시연산을 의미하며 f는 각 단계에서 발생하는 플로우 수를 의미한다.

VI. 결론

POS 기기의 내부자, 외부자의 위협은 현재도 여전히 문제가 되고 있다. 이에 본 논문에서는 현재까지의 POS기기의 위협에 대해 조사하고 소프트웨어의 무결성 검증을 통해 보안성을 강화시키는 방법을 제안하였다. 또한 본 제안 방안이 인증, 기밀성, 무결성, 가용성 등 POS보안 요구사항을 만족하는 것에 대하여 검증하였다. 이는 우리나라에서 사용되는 결제단말기 CAT단말기에도 적용이 가능하다. 현재 국내에서는 CAT, POS 두 종류의 단말기가 사용되고 있으며 대부분 중소형 가맹점에서 사용하는 CAT단말기는 결제, 승인 기능만을 가지고 있지만 온라인 기반의 단말기이며 메모리가 탑재되어 있다. 제안 시스템을 적용시키기 위한 최소사양은 제시한 해시연산과 바이너리 이미지화 연산수행, 온라인 기반이라는 점에서 현재의 CAT단말기에서도 적용이 가능하다[16]. 또한 같은 이유로 현재 영국 및 유럽에서 사용되고 있는 CAB 단말기 형태에도 적용이 가능하다[17].

하지만 이러한 방법은 파일에서의 소프트웨어 단에서의 검증만을 목적으로 하기 때문에 메모리 단으로 올라온 위협은 탐지할 수 없다는 한계점을 지닌다. 따라서 향후 메모리 무결성 검증을 위한 Software attestation에 대한 연구를 할 계획이다.

참고문헌

[1] 이투데이, "금감원, MS카드 제한, 2014년 2월부터 전면시행," http://www.etoday.co.kr/news/section/newsview.php?TM=news&SM=2107&idxno=585387, 2012년 5월.

[2] 이종오, "신용카드 POS가맹점의 카드회원 신용정보 보안상 문제점 및 대응방안," 여신금융협회, 2008년 9월.

[3] 뉴스 Zum, "외국 호텔서 스파이웨어 목마," http://news.zum.com/articles/22378, 2012년 4월.

[4] Iulia Ion and Boris Dragovic, "Don't trust POS terminals! Verify in-shop payments with your phone," Proceedings of SMPU, 2010.

[5] msdn "Introduction to Code Signing", Mar. 2012.

[6] 전상훈, 허진영, 최종선, 최재영, "침입 복구 및 대응 시스템을 위한 실시간 파일 무결성 검사," 정보보호과학회논문지, 32(5,6) pp 279-287, 2005년 6월.

[7] 문연관, "POS 시스템 보안강화를 위한 종단간 암호화 적용방법에 관한 연구," 석사학위논문, 동국대학교, 2010년 6월.

[8] AhnLab "Brochure_TrusLine," 2010년 9월.

[9] 손영호, "최근 신용카드 범죄수법 동향 및 대응방안," 여신금융협회 2009년 12월

[10] 정보통신단체표준, "POS 시스템 보안 요구사항," 2011년 12월.

[11] 백금옥, 임철호, 손진곤 "키로깅 방지를 위한 가상키보드 시스템," 보안공학연구논문지, 제 7권 4호, pp 319-334, 2010년 8월.

[12] Syverson, P. "A taxonomy of replay attacks," Computer Security Foundations Workshop VII of IEEE, pp187-191, Jun 1994.

[13] 이철원, 배병철 "사이버테러 동향 및 대응방안," 한국위기관리논집, 제 5권 2호, pp119-135, 2009년 12월.

[14] Colin Boyd and Anish Mathuria, "Protocols for Authentication and Key Establishment," Springer, 2003.

[15] Etnews, "롯데, 계산은 우리 제품으로 POS 국산화 바람," http://www.etnews.com/news/computing/informatization/2576790_1475.html, 2012년 4월.

[16] 파이낸셜뉴스, "BC카드 중소형 가맹점 지원 스마트포스 출시," http://www.fnnews.com/view?ra=Sent0401m_View&corp=fnnews&arcid=201206070100057180003154&cDateYear=2012&cDateMonth=06&cDateDay=07, 2012년 6월.

[17] EMVCO homepage, http://www.emvco.com/

〈著者紹介〉

조 성 아 (Sung-A Cho) 학생회원
2011년 2월: 숙명여자대학교 통계학과 졸업
2011년 2월~현재: 고려대학교 정보보호대학원 금융보안학과 석사과정
〈관심분야〉 정보보호, 금융보안, 암호 프로토콜

김 성 훈 (Sung Hoon Kim) 학생회원
2006년 8월: 서울시립대학교 수학과 졸업
2009년 2월: 고려대학교 정보보호대학원 석사
2009년 1월~2011년 2월: (주)알티캐스트 CAS개발본부
2011년 3월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 소프트웨어 보안, 소프트웨어 난독화

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월: 고려대학교 경제학과(학사)
1987년 12월: Oklahoma University 전산학 대학원(공학석사)
1992년 5월: Oklahoma University 전산학 대학원(공학박사)
1992년 8월: 단국대학교 전자계산학과 전임강사
1993년 3월~1997년 2월: 고려대학교 전산학과 조교수
1997년 3월~2001년 2월: 고려대학교 전산학과 부교수
2001년 2월~현재: 고려대학교 정보경영공학전문대학원 교수
〈관심분야〉 암호프로토콜, 암호이론, USN 이론, 키 교환, 익명성 연구, PET 기술