情報保護學會論文誌
第19卷 第1號, 2009. 2

피싱 및 파밍 방지를 위한 인지 기반의 접근 방법*

김 주 현,[1†] 맹 영 재,[1] 양 대 헌,[1‡] 이 경 희[2]

[1]인하대학교 정보통신대학원, [2]수원대학교

Cognitive Approach to Anti-Phishing and Anti-Pharming*

JuHyun Kim,[1†] YoungJae Maeng,[1] DaeHun Nyang,[1‡] KyungHee Lee[2]

[1]Graduate School of IT&T, INHA University, [2]The University of Suwon

요 약

현재 피싱 공격을 방지하는 여러 가지 기법들이 연구 되고 있다. 이들 중에서 작업관리창이나 브라우저의 주소창에서 피싱 사이트를 구별해 주는 프로그램들이 있으나 이런 프로그램들은 사이트의 도메인이나 IP주소로만 피싱 사이트를 판단한다. 이 접근 방법으로는 DNS 파밍 공격 등은 방어할 수 있지만 숨겨진 공격(Hidden Attack) 등의 HTML 코드를 변경하는 기법에는 취약하다. 이 논문에서는 프로그램이 IP 나 사이트의 도메인을 분석하여 피싱 및 파밍 여부를 판단하는 기존의 방식이 아닌 플러그인과 서버 사이에서 HTML 코드의 변경 유무를 파악하고 피싱 여부를 팝업이나 플래시 등으로 위조하기 어려운 시스템 트레이와 풍선도움말을 사용하여 접속 사이트와 시스템 트레이의 그림을 사용자가 비교함으로서 직접 피싱 및 파밍 여부를 쉽게 결정할 수 있도록 하는 이미지 비교를 통한 인지 기반의 접근 방법을 제시한다.

ABSTRACT

Recently, lots of anti-phishing schemes have been developed. Several products identify phishing sites and show the results on the address bar of the internet browser, but they determine only by domain names or IP addresses. Although this kind of method is effective against recent DNS pharming attacks, there is still a possibility that hidden attacks which modifies HTML codes could incapacitate those anti-phishing programs.

In this paper, the cognitive approach which compares images to decide phishing or pharming is presented, using system tray and balloon tips that are hard to fake with pop-ups or flash in order for users to compare pictures from connecting sites and system tray. It differs from an old method that a program analyzes IP or domains to judge if it is phishing or pharming, but observes if there were HTML code changing between plug-ins and a server.

Keywords : Phishing, Pharming, Hidden Attack, Graphical substitution

Ⅰ. 서 론

피싱(Phishing)은 개인정보를 뜻하는 Private data와 낚시의 의미를 갖는 Fishing 의 합성어이다. 피싱은 전자우편 또는 메신저를 사용해서 신뢰할 수 있는 사람 또는 기업이 보낸 메시지인 것처럼 가장함으로써, 사용

접수일(2008년 6월 24일), 수정일(1차: 2008년 8월 24일, 2차: 9월 18일), 게재확정일(2008년 10월 30일)
* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었습니다. (IITA-2006-C1090-0630-0028)
† 주저자, ibmhdd@seclab.inha.ac.kr
‡ 교신저자, nyang@inha.ac.kr

자의 비밀번호 및 신용카드 정보와 같이 기밀을 요하는 정보를 부정하게 얻으려는 사회공학의 한 종류이다. 피싱 사고에 대한 신고가 늘어남에 따라, 피싱을 막으려는 연구들이 진행되고 있다[1.2].

이와 같은 피싱 공격은 사용자가 주의를 기울임으로써 통해 어느 정도 방어할 수 있다. 하지만 스크립트를 이용하여 브라우저의 주소창과 상태표시줄을 팝업창으로 정교하게 덮거나 브라우저 팝업 기능중 주소창 및 도구모음을 지울 수 있는 기능을 활용하여 브라우저 맨 위에 도구모음창과 동일한 그림을 보여주는 이미지 교체(Graphical Substitution) 기법 등은 방어가 어렵다[14]. 또한 DNS 파밍(DNS Pharming)과 같이 합법적으로 소유하고 있던 사용자의 도메인을 탈취하거나 DNS 이름을 속여서 사용자들이 진짜 사이트로 오인하도록 유도하는 방법을 이용하면 해당 사이트가 공식적으로 운영하고 있던 도메인 자체를 중간에 탈취하기 때문에 사용자는 정당한 URL을 입력해도 위장된 사이트로 이동하게 된다[11]. 이러한 파밍 공격의 경우 사용자가 공격을 당하고 있다는 사실을 알기 어렵기 때문에 피해를 당하기 쉽다[3].

본 논문에서는 이러한 피싱 공격 및 파밍 공격에 대한 방지책으로, 프로그램이 피싱 및 파밍 여부를 판단하는 것이 아니라 사용자가 접속 사이트와 시스템 트레이의 그림을 비교함으로서 직접 피싱 및 파밍 여부를 쉽게 결정할 수 있도록 하는 인지 기반의 접근 방법을 사용한다. 이와 함께 정상적인 서버와 사용자 사이에 전송된 HTML 코드의 변경되었는지를 HTML 코드의 해쉬 결과를 비교함으로서 판단하여 숨겨진 공격과 DNS 스푸핑 및 DNS 파밍 공격을 방지하는 방법을 제안하고자 한다. 2장에서는 피싱과 관련된 배경 이론과 현황 및 현재 국내외에 상용화된 피싱 방지 프로그램의 성능을 확인과 함께 본 연구를 위해 사용된 기본 기술, 3장에서는 본 연구의 프로토콜 구성, 4장에서는 여러 가지 공격에 대한 안전성에 대해서 살펴보겠다.

Ⅱ. 배경 연구

2.1 피싱 공격

피싱의 공격 방법에 따라 E-mail 피싱 공격, 파밍 공격, 숨겨진 공격 등으로 구분 된다.

2.1.1 E-Mail 피싱 공격

피싱은 1996년 American Online(AOL)의 메신저를 사용하는 해커들이 일반 사용자에게 조작된 전자우편을 보내는 해킹기법에서 유래되었다[1]. 이 해커들은 자신이 보낸 전자우편을 AOL 이 발송된 것처럼 속이고 사용자들은 전자우편을 보고 사용자들의 계정정보를 빼냈다.

이러한 피싱 공격을 막지 못하는 이유 중 하나는 웹브라우저나 메일 관리 프로그램의 주소창과 링크 연결에 있어서 취약점들을 가지고 있기 때문이다[7]. 피싱 공격을 당할 때 웹브라우저의 주소창의 URL과 링크가 스푸핑 되어 실제 기관과 매우 유사한 주소를 보여주기 때문에 메일을 받은 사용자가 웹사이트의 진위 여부를 바로 판단하기는 어렵다. 또한 사용자가 URL 정보가 맵핑되어 있는 이미지위로 마우스를 올려놓았을 때 웹브라우저 하단에 보여주는 링크 정보도 악의적인 링크가 아닌 정상적인 링크라고 보여주기 때문에 사용자들이 쉽게 속을 수 있다[8,9].

2.1.2 URL 스푸핑 공격 및 IP주소 접속 공격

URL 주소 스푸핑은 인터넷 브라우저에 표기되어 있는 주소를 속여 특정 사이트를 정상적인 사이트처럼 방문하여 이용할 수 있도록 하는 변조공격의 일종이다. 기존에는 브라우저에서 %01과 %00 과 같은 문자를 인식하지 못하는 부분을 이용했으며 정상 사이트와 유사한 URL을 이용하거나 웹브라우저를 이용하여 로그인 하는 방식을 이용하기도 했다. 예를 들어 HTTP://mybank.com:ebanking@evilsite.com/phishing/ 의 경우 evilsite.com 에 접속 하는 주소이지만 사용자는 mybank.com 에 접속하는 것으로 착각할 수 있다[4].

URL의 도메인 이름 대신에 IP주소를 사용하여 도메인 이름을 혼란스럽게 하여 콘텐츠 필터링을 피하거나 목적지를 속이기 위한 방식으로 IP주소를 Hex, Oct, 십진수 등 다양하게 나타낸다.

2.1.3 파밍 공격

URL을 속이는 기법이 아닌 DNS 또는 프록시 서버의 주소를 직간접적으로 변조하는 파밍 공격은 사용자 측에서 보면 피싱 보다 더욱 쉽게 속을 수 있다. 파밍으

로 이용될 수 있는 방법은 DNS 주소의 변조, 클라이언트 호스트 파일 변경, 클라이언트 DNS 서버설정 주소 변경, 등록된 도메인의 정보 변경, 프록시 서버 이용, DNS Cache Poisoning 등의 방법이 있다[5,6]. 기존의 피싱 공격은 유사한 이름의 도메인 주소를 이용하거나 정상적인 사이트를 통한 리다이렉트, 정교한 위조 페이지 등을 이용하여 위조된 피싱 사이트로 유도하였으나 사용자가 주의 깊게 살펴보면 피싱 사이트를 인지할 수 있는 부분도 있었다. 하지만 이렇게 DNS 주소를 변경시키게 되면 사용자의 판단은 더욱 어려워지고 믿고 접속할 가능성이 높아진다.

2.1.4 숨겨진 공격(Hidden Attack)

숨겨진 공격에 가장 일반적으로 사용되는 공격은 숨겨진 프레임 공격과 Overriding Page Content 및 이미지 교체 방법이 있다[2,14].

숨겨진 프레임 공격 방법은 숨김 공격에 가장 많이 이용되는 프레임을 기반으로 하는 공격이다. 그림 1은 두 개의 프레임이 정의되어 있는 것을 보여준다. 첫 번째 프레임은 정상적인 URL 정보가 포함되어 있고, 두 번째 프레임은 숨겨진 프레임으로 피싱 페이지를 참조하도록 하고 있다. 숨겨진 프레임은 피싱 컨텐츠로 링크를 포함 한다. 이러한 공격(Ajax를 이용한 스크립트 캡쳐 등)을 통해 개인정보가 누출될 수 있다[14].

Overriding Page Content 방법은 가장 많이 사용되는 거짓된 컨텐츠를 삽입하는 방식으로 실제 웹 페이지 화면에 숨겨진 웹 페이지를 삽입시키는 방법이다[2]. DHTML 함수(DIV)를 이용하여 페이지에 위장된 페이지를 삽입하는 공격이 가장 많이 사용된다. 이 방법은 공격자가 실제 웹 페이지 상단 위에 공격코드를 포함한 하나의 완전한 페이지를 생성하도록 할 수 있다. 그리고 간단한 HTML embedded commands를 사용하면, 공격자가 사용자의 세션을 가로챌 수 있다.

```
<frameset rows="100%,*"
framespacing="0">
<frame name="real"
src="http://mybank.com/">
<frame name="hiddenContent"
src=http://evilsite.com/bad.htm>
</frameset>
```

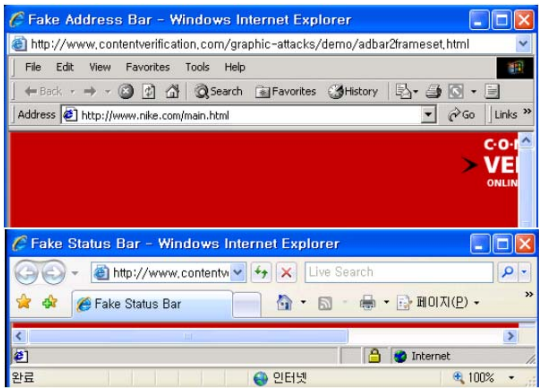
[그림 1] frame 기반의 공격 코드 예시

이미지 교체 방법은 브라우저의 주소창 및 도구모음을 없애고 동일한 모양의 이미지를 표시함으로서 올바른 URL을 보여주거나 브라우저 좌측 하단(Padlock Zone)의 HTTP 통신을 HTTPS 암호화 통신처럼 보이도록 이미지를 교체하여 사용자를 속이기도 한다[14]. 이 기법은 인터넷 익스플로러 7 부터는 팝업창에 강제적으로 URL을 표시하는 기능에 의하여 인식이 가능하지만 인터넷 익스플로러 7 이하의 버전을 사용하는 사용자는 피싱 공격을 당하기 쉽게 된다. 그림 2는 그래픽 교체 방법을 보여준다.

2.2 국내외 피싱 방지 프로그램

현재 피싱 공격을 방지하기 위한 많은 피싱 방지 프로그램들이 있다. 피싱 사이트를 신고 받거나 조사하여 블랙리스트를 만드는 방법과 접속하는 사이트와 SSL 통신을 이용하여 서버를 확인 하는 방법, 피싱프로그램을 사용하는 사용자들의 의견으로 피싱사이트를 구별하는 사회공학적인 방법, 사이트의 생성 정보를 이용하는 방법, 접속한 사이트의 실제 이름을 크게 표시해주는 방법 등이 있다. 국내외에서 사용되는 피싱 방지 프로그램은 다음과 같다.

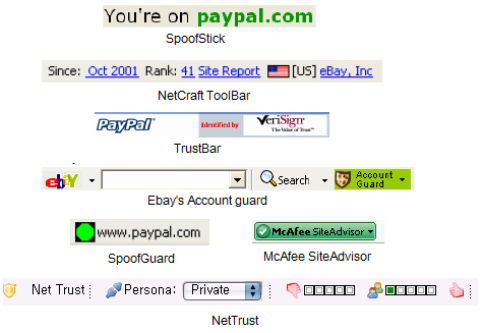
- **브라우저 내장 피싱 필터**: 마이크로소프트 인터넷 익스플로러 7 과 파이어폭스 3 과 같은 최근에 나온 브라우저에는 기본적인 피싱 방지 필터가 있다.브라우저 내장 피싱 방지 필터는 브라우저 제작



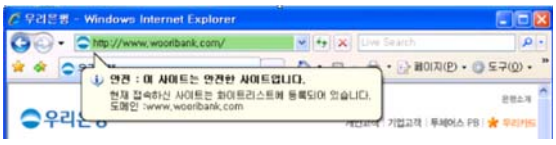
[그림 2] 도구모음 및 Padlock Zone 변경

회사에서 피싱 사이트를 찾고 사용자들도 피싱사이트를 신고하여 블랙리스트를 만들어서 피싱을 방지하는 방법이다. 블랙리스트만 사용하기 때문에 DNS 파밍이나 숨겨진 공격에는 취약하다.

- **SpoofStick**: 접속한 사이트의 메인 도메인 이름을 크게 출력한다. www.mybank.phishing. co.kr을 접속하게 되면 툴바에 phishing.com 에 접속했음을 표시해준다. 하지만 스푸핑된 사이트에 대한 경고가 아닌 단순하게 사이트의 도메인 이름을 크게 보여주기 때문에 I 와 l처럼 구별하기 어려운 글자를 사용하는 피싱사이트에 노출되기 쉽다[10,17].
- **NetCraft Toolbar**: 사용자가 접속한 서버의 등록일자 및 호스팅중인 나라와 툴바 사용자들 사이에서 접속하고 있는 사이트의 접속량을 판단하여 피싱을 구별하는 툴바 형식의 프로그램이다[16]. 일반적인 피싱 사이트의 경우 생성 시기가 짧은 점을 이용할 수 있으나 브라우저의 신뢰 할 수 있는 사이트 목록에 저장된 사이트가 DNS 파밍 공격을 당한 경우에는 프로그램에서 신뢰할 수 있는 사이트라고 보여주는 문제점이 있다[10].
- **TrustBar**: 접속하고자 하는 사이트에 SSL을 이용하여 접속한 후 사이트의 로고와 인증서의 서명 유무를 툴바에 보여준다. 대부분의 인증된 사이트는 SSL을 이용하여 사용자의 주요 정보를 전달하므로 SSL을 사용하지만 피싱 사이트는 SSL을 사용하지 않는 점을 이용한 프로그램이다[19]. 하지만 피싱 사이트가 아닌 사이트에서 SSL을 사용하지 않는다면 사용자가 피싱사이트로 오인하는 단점이 있다[10].
- **eBay's Account Guard**: 접속한 사이트가 eBay 이거나 PayPal 사이트일 경우에는 녹색 아이콘을 보여주고 피싱사이트나 블랙리스트에 등록된 사이트일 경우에는 붉은 아이콘을 보여주는 프로그램이다[10,15].
- **McAfee SiteAdviser**: 사용자가 접속하고자 하는 사이트에 하여 McAfee에서 사전에 사이트를 분석한 정보를 기반으로 사용자에게 안전한 사이트임을 툴바에 3가지의 색으로 표시해주는 프로그램이다[18]. 새로 생긴 사이트나 분석 정보가 없는 사이트에 대해서는 분석이 안된 사이트로 표시하여 피싱 경고에 대한 신뢰성이 부족하다.
- **Ljean Camp NetTrust**: 인증서로 인증된 가명을 기반으로 사이트에 접속한 사용자들의 의견을 통계로 피싱사이트를 판단하는 사회공학 기법을 이용한 툴바 형식의 프로그램이다[20]. 프로그램을 사용하는 사용자들이 많을수록 피싱사이트를 찾아내기 수월해 지지만 악의적인 사용자가 피싱 사이트에 긍정적인 의견을 많이 넣는 다면 사용자들이 피싱사이트를 정상적인 사이트로 오인하는 경우가 생긴다.
- **SoftForum ClientPhishingPro**: 이미 신고된 사이트 리스트를 바탕으로 피싱 사이트를 검출하고 사용자에게 접근금지 경고를 한다. 또한 블랙리스트 업데이트 기능을 제공하여 실시간 대응을 지원하고 있다. 이 프로그램의 경우 DNS 파밍을 방지하기 위하여 호스트 파일 변경 및 사용자의 DNS 서버 IP를 변경 여부를 확인하여 사용자에게 경고하며 사용자의 브라우저 주소창에 프로그램이 상주하여 사용자가 URL을 입력하면 화이트리스트에 있는 URL 과 IP를 비교하여 파밍 공격을 예방하고 사용자에게는 주소창의 색을 변화시키면서 안전한 사이트임을 그림 4와 같이 표시한다[21].
- **Softrun NoPhishing**: URL 과 IP를 확인하여 피싱 사이트를 검출한다. 이 프로그램은 주소창에 구분을 해주는 방식이 아닌 팝업창을 띄워주는 방식으로 피싱 사이트를 사용자에게 경고한다. 그림 5는 프로그램이 DNS 파밍을 차단할 때 보여주는 팝업



[그림 3] 툴바 기반의 국외 피싱 방지 프로그램



[그림 4] 소프트포럼 ClientPhishing Pro

[그림 5] 소프트런 NoPhishing 프로그램의 DNS 파밍 방지경고

창이다. 이 프로그램은 스푸핑 차단을 위해 프로그램내에 저장된 스푸핑 의심사이트를 검색한다[22]. 하지만 프로그램내에 없는 스푸핑된 도메인은 검색하지 못한다는 단점이 있다.

툴바 형식의 프로그램의 경우 이미지 교체 방법에 취약하다. 또한 브라우저 전체에 비하여 작은 부분을 차지하는 툴바의 아이콘에 사용자들이 주의를 기울이지 않으며 정상적인 사이트에 대한 정보가 없다면 이를 피싱 사이트로 오인하기도 하는 부분 때문에 툴바를 지우는 확률이 크다[10]. 또한 새로 만들어진 피싱사이트에 대한 정보가 없다면 사용자는 피싱 피해에 노출며 기존에 나온 피싱 방지 프로그램들의 가장 큰 문제점은 HTML 코드 내에 있는 공격코드에 대한 분석이 부족하다[13]. 최근 HTML 기술이 고도화 됨에 따라 숨겨진 공격을 통하여 사용자의 정보를 가로챌 수 있다. 본 논문에서는 이러한 이유로 프로그램 스스로 피싱 여부를 판단하여 사용자에게 전달하는 방식이 아닌 이미지 비교를 통한 인지력을 기반으로 피싱 사이트를 판단하고 숨겨진 공격을 방지하기 위하여 HTML 코드의 변경유무를 분석한다.

Ⅲ. 브라우저 플러그인과 풍선도움말 기능을 활용한 피싱 방지 기법

이 단원에서는 본 논문이 제시하는 웹브라우저상의 이미지와 풍선도움말에 있는 이미지를 비교하여 사용자 스스로 피싱 사이트를 구분할 수 있도록 하는 사용자 인지 기반의 기법을 소개한다. 이미지를 비교하는 방법을 사용하는 이유는 현재 나와 있는 피싱 방지 프로그램의 경우 팝업과 메시지 박스 방식이나 주소창에 피싱 사이트와 파밍을 경고하는 방식을 사용하고 있지만 이는 실질적으로 프로그램 자체는 피싱 사이트임을 판단하지만 페이지 자체가 정상 사이트와 완전히 동일하다면 사용자는 피싱 사이트를 보고 있지만 프로그램이 오작동을 한다고 생각하게 된다[10]. 또한, 피싱 사이트에서 정상 사이트를 팝업으로 뛰운 후 정상 사이트를 다시 피싱 사이트로 덮는 숨겨진 공격을 이용하면 피싱 방지 프로그램을 속일 수 있다. 파밍의 경우 URL 이 동일하기 때문에 이와 같은 사용자로 하여금 혼란을 주기 쉽다[4]. 또한 주소창에 표시하는 방식의 경우 이미지 교체 방법과 같이 주소창 및 도구관리 창을 위조하는 방법의 경우 사용자가 구별하기 쉽지 않게 된다[14]. 이러한 공격을 방지하기 위하여 팝업방식의 경고창이 아닌 작업 표시줄의 아이콘과 풍선도움말 기능을 이용하여 자바스크립트를 통한 팝업창 위조를 방지하였으며 풍선도움말 뿐만 아니라 브라우저 자체에도 올바른 이미지 또는 피싱 경고 이미지를 보여주어 사용자로 하여금 피싱 사이트에 접속했는지를 인지할 수 있도록 하였다. 또한 HTML 코드의 해시 결과 비교를 통하여 HTML 코드가 변경되었는지를 감시하여 숨겨진 공격과 DNS 스푸핑 및 DNS 파밍 공격을 방지하게 된다.

또한 통신 과정에서 서버에 과부하 우려가 있으므로 해시와 대칭키를 이용하여 데이터를 암호화 하여 부하를 방지 한다. 설치 이후 플러그인은 URL을 통하여 서버에 접속하면 DNS 파밍에 취약하므로 플러그인에 저장된 서버의 IP 로 접속을 하여 통신을 한다. 만약 플러그인이 접속하고자 하는 서버의 IP 가 변경 되었다면 서버측은 HTML 코드내에서 플러그인의 함수를 호출하거나 플러그인의 변수에 특정한 값을 넣어줄 수 있는 플러그인 쓰레드 기능을 이용하여 플러그인에 안전하게 IP를 전달해야 한다. 그리고 본 프로그램의 처음 설치는 안전하다고 가정한다. 이 가정이 전제되지 않으면 인터넷을 통해 배포되는 어떠한 프로그램도 안전할 수 없기 때문이다. 안전한 배포를 위하여 공인인증서를 통해 서명을 확인하는 방법을 적용하였다.

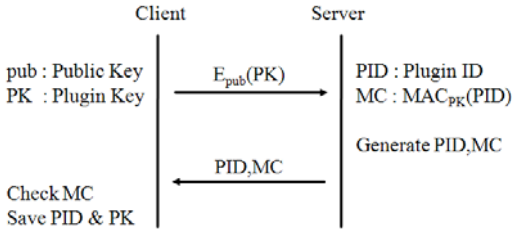
3.1 PluginID와 PluginKey 의 교환

인터넷 환경에서 배포되는 플러그인은 신뢰성을 보장하는 공인인증서를 통하여 코드가 서명이 되어

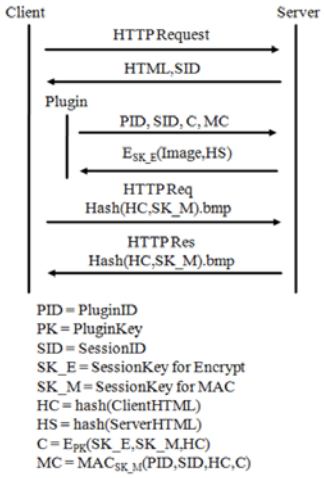
있다. 사용자가 플러그인을 설치하지 않은 상태에서 플러그인을 요청하게 되면 서버에게 codebase 등에 있는 플러그인의 설치파일을 다운로드 받게 되고 공인인증서를 통한 서명을 확인시켜 주면서 사용자에게 설치를 묻게 된다. 하지만 처음으로 플러그인을 설치하는 과정부터 피싱 및 파밍 공격을 받게 된다면 사용자와 올바른 서버 모두에게 위험하게 된다. 이러한 이유로 플러그인의 첫 설치 과정에서는 사용자의 공인인증서 등을 통하여 서버가 사용자를 인증하는 동시에 사용자도 서버를 인증할 수 있는 과정이 필요하다. 첫 실행일 경우에는 다음을 수행한다.

1. 플러그인은 자신이 사용할 PluginKey를 생성하고 공인인증서의 공개키로 암호화 하여 서버에 전송한다.
2. 서버는 PluginKey를 복호화 하고 플러그인을 위한 PluginID를 생성하여 데이터베이스에(PluginID, PluginKey)쌍을 저장 하고 플러그인 모듈에게 PluginID와 PluginID의 MAC를 전송한다.
3. 사용자는 MAC을 확인하여 PluginID 가 Active Attack 등에 의해 변경되었는지를 확인하고 레지스트리를 이용하여 PluginID와 Plugin Key를 저장하고 필요시에 사용한다.

이 과정은 플러그인이 처음 설치될 때와 서버 측에서 PluginKey의 갱신을 요청할 때 수행한다. [그림 6]은 PluginID와 PluginKey의 교환 과정을 보여준다.



[그림 6] PluginID 와 PluginKey 교환 과정



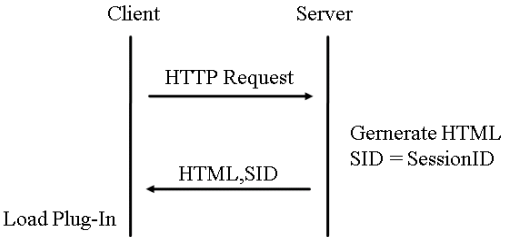
[그림 7] 서버와 사용자 사이의 통신

3.2 사용자와 서버의 통신 과정

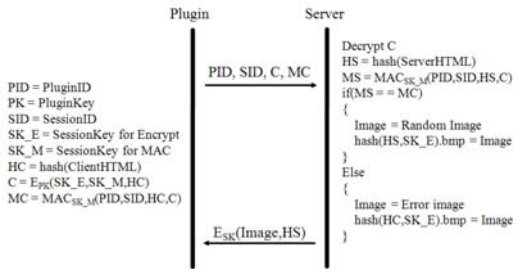
플러그인의 첫 설치 이후 사용자가 서버에 접속을 하면 그림 7의 프로토콜을 수행 한다. 사용자는 로그인 페이지를 포함하는 HTML 코드를 서버로부터 요청하고 서버는 SessionID와 HTML 코드를 사용자에게 전송한다. HTML 코드와 SessionID를 받으면 사용자의 브라우저는 플러그인을 호출한다. 플러그인은 풍선도움말에 보일 이미지를 서버로부터 안전하게 전달 받고 이를 시스템 트레이의 풍선도움말에 보여주고, HTML 코드는 플러그인에 있는 함수를 호출하여 브라우저에 보일 이미지의 이름을 받아 서버에 요청한 후 브라우저의 로그인 박스 옆에 표시한다.

3.2.1 웹페이지 요청

사용자가 접속하고자 하는 서버측에 HTML 코드를 요청하면 서버는 사용자에게 제공할 SessionID와 HTML 코드를 생성한다. 생성된 SessionID와 HTML 코드의 해시 값을 서버의 데이터베이스에 저장하고 SessionID와 HTML 코드를 사용자에게 전송한다. SessionID는 쿠키 혹은 HTTP의 GET 방식을 통하여 전송한다. HTML 코드의 해시 값은 사용자가 전송받은



[그림 8] HTML 코드 요청 요청 과정



[그림 9] 플러그인과 서버의 통신 과정

HTML 코드의 무결성을 보장하기 위해서 사용한다. [그림 8]은 HTML 코드와 세션아이디의 교환 과정을 보여준다.

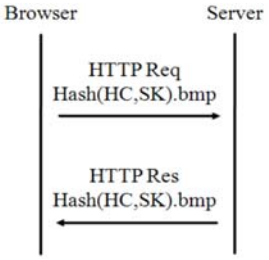
3.2.2 플러그인 호출 및 데이터 전송

브라우저는 서버로부터 HTML코드를 받으면 코드 내에 있는 내용을 순서대로 실행하게 된다. 코드를 실행하면서 <OBJECT>나 <EMBED> 태그를 통하여 플러그인을 호출한다. 플러그인을 수행하는 과정은 [그림 9]에서 볼 수 있다.

1. 브라우저 컨테이너를 통하여 플러그인은 브라우저가 받은 HTML 코드를 플러그인에 불러온다.
2. 서버측과 한 세션에서만 사용할 세션키인 SK_E와 MAC을 위한 SK_M을 생성한다.
3. 세션키 SK_E와 MAC을 위한 SK_M 및 HTML 코드의 해시 결과인 HC를 PluginKey를 이용하여 암호화하여 C를 생성한다.
4. 서버와의 메시지 인증을 위하여 SID와 PluginID, HC, C를 세션키를 이용한 메시지 인증 코드인 MC를 생성한다.
5. PluginID, SID, C, MC를 서버로 보낸다.

3.2.3 이미지 전달

서버는 PID를 인덱스로 데이터베이스에서 플러그인 키 PK를 찾아 C를 복호화 한다. 복호화의 결과로 두 개의 세션키를 얻어내고, SID를 인덱스로 해서 데이터베이스에서 사용자에게 보냈던 HTML 페이지의 해시 값을 얻어낸다. 서버는 브라우저로부터 전송받은 메시지 인증 코드를 자신이 가지고 있는 데이터를 이용하여 만든 메시지 인증 코드와 비교하여 사용자가 서버에서 전



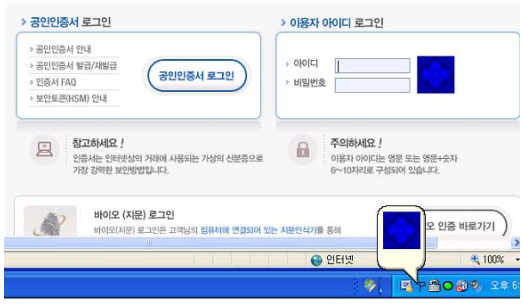
[그림 10] 브라우저에 보이는 이미지 요청 과정

달한 모든 데이터를 안전하게 받았는지 확인한다. 만일 이 과정에서 메시지 인증 코드가 일치하지 않으면 사용자는 공격을 받은 것이므로 사용자의 플러그인에게 경고메시지를 보여주는 이미지를 전송한다. 메시지 인증 코드가 일k치하면 임의의 이미지를 선택하고 이 파일을 HTML의 해시 값인 HC와 세션키인 SK_E를 해시한 데이터로 파일명을 변경한다. 이는 후에 브라우저의 HTTP Request 요구에 대한 응답으로 보낼 이미지를 설정하는 과정이다. 파일명을 세션키와 HTML 코드의 해시 값으로 변경하는 이유는 HTTP Request를 통하여 브라우저가 요청하는 파일의 이름을 공격자가 모르게 함과 동시에 사용자와 서버에서 한 세션에 사용할 수 있는 유일한 파일명을 만들기 위해서이다. 서버에서 사용자에게 보낼 이미지를 선택한 다음 이미지의 이름을 변경했으면 그 이미지와 서버에서 브라우저로 보낸 HTML 코드의 해시값을 서버와 플러그인이 공유하는 세션키로 암호화 하여 플러그인에게 전송한다.

플러그인은 세션키를 이용하여 이미지와 HTML 코드를 해시한 데이터를 복호화 한다. 우선 플러그인이 가지고 있는 HTML 해시 값과 서버로부터 전송받은 HTML 해시 값을 비교하여 이상이 없으면 서버로부터 받은 이미지를 풍선 도움말에 출력하고 해시 결과가 일치하지 않는다면 플러그인은 오류메시지를 풍선도움말에 출력한다.

3.2.4 웹 브라우저에 이미지 출력

플러그인을 <OBJECT>나 <EMBED>태그로 호출하기 위해서는 ID를 부여한다. 부여된 ID를 통하여 HTML 코드 내에서는 플러그인을 ID 로 인식하여 상호 통신을 하게 된다. 플러그인은 IDL(interface definition language)에서 지정한 함수를 자바스크립트를 통하여 호출 할 수 있게 된다. 자바스크립트를 통해

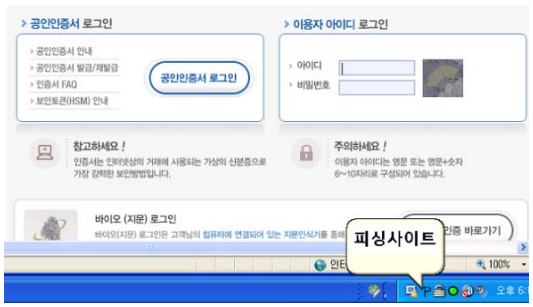


[그림 11] 풍선도움말과 브라우저 이미지 비교

함수를 호출하면 그 함수는 플러그인이 가지고 있는 HTML 코드와 세션키를 해시한 결과를 반환하게 된다. <IMG SRC = 'ObjectID.ImageName()'.bmp>와 같은 DOM (Document Object Model)방법으로 플러그인의 함수를 실행한다. 이 함수는 자신을 호출한 HTML 문서 전체와 세션키의 해시 값을 반환한다. 반환된 결과를 이미지의 이름으로 해서 HTTP request를 통하여 요청하면, 서버에는 플러그인에 전달한 이미지의 이름을 미리 HTML 코드와 세션키의 해시된 결과로 변환해 두었기 때문에 이미지를 보여줄 수 있게 된다. 피싱 사이트에서는 풍선 도움말에서 보여지는 그림과 똑같은 그림을 보여 줄 수 없으므로, 사용자가 쉽게 피싱/파밍 여부를 판단할 수 있게 된다. [그림 10]은 브라우저에 보일 이미지를 교환하는 과정을 보여준다.

3.3 사용자의 이미지 비교

사용자는 [그림 11]같이 플러그인이 제공하는 작업표시줄에 있는 아이콘의 풍선도움말과 웹브라우저에 있는 이미지를 비교하여 피싱 사이트를 구분할 수 있게 된다. 피싱 사이트일 경우에는 [그림 12] 또는 [그림 13]같이 피싱 경고 메시지가 보이거나 브라우저와 풍선도움말에



[그림 12] 코드를 변경한 피싱사이트

[그림 13] 팝업 방식을 이용한 피싱사이트

보이는 이미지가 다르다. 작업표시줄의 풍선도움말 기능을 사용하면 피싱 사이트의 악성 자바 코드 등이 이를 위조할 수 없게 된다. 또한, 아이콘 위에 표시되는 풍선도움말의 경우 아이콘의 위치를 가리키는 모양이기 때문에 사용자는 위조된 풍선도움말이 아님을 알 수 있게 된다.

Ⅳ. 안전성 분석

브라우저의 도구 모음 등에 이미지를 표현하게 된다면 팝업창을 위조하여 표시가 가능하기 때문에 팝업창과 같은 기술로 구현하기 어려운 작업표시줄의 아이콘과 아이콘을 가리키는 풍선도움말을 사용함으로서 사용자가 이미지를 비교하는데 있어서 불편함이 없으며 팝업창 등으로 위조가 되지 않아 안전하다. 최근에 나온 브라우저의 경우에는 팝업 방지 기능을 가지고 있지만 플래쉬 기반이나 Ajax 기반의 팝업창은 브라우저 팝업 방지기능으로 해결되지 않으므로 팝업창을 이용한 위조를 방지해야 한다. 또한, 플러그인이 보여주는 이미지를 암호화 하여 전송함으로서 공격자로부터 이미지 유출을 방지한다. 그리고 브라우저 컨테이너 방법을 통하여 HTML 코드를 플러그인이 직접 받아냄으로서 공격자가 자바스크립트 등을 이용하여 올바른 코드를 플러그인의 특정 변수에 저장하여 플러그인에 전송하는 방법을 사용하지 못하게 하였다. 또한 플러그인이 직접 HTML 코드를 받아내고 서버와 HTML 코드의 해시 결과를 비교함으로서 HTML 코드의 무결성을 보장한다.

4.1 DNS 파밍에 대한 방지

파밍에 의하여 피싱 사이트에 사용자가 접속하게 되

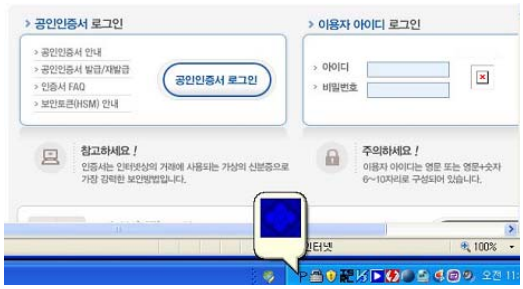
면 URL이 동일하므로 사용자는 피싱 사이트라고 의심하지 않게 된다. 본 논문에서 제시한 기법은 플러그인을 사용하기 때문에 플러그인의 고유한 CLSID를 요청하는 HTML 코드내의 명령어가 있어야 플러그인을 호출할 수 있다. CLSID가 다른 프로그램을 호출하게 되면 플러그인이 사용자에게 설치되어 있지 않기 때문에 설치를 물어보게 된다. 플러그인의 설치를 물어볼 때 플러그인은 공인인증서에 의하여 서명이 되어 있으므로 그 서명인을 구별한다면 잘못된 플러그인의 설치를 피할 수 있다. 공격자는 이와 같은 방법을 피하기 위하여 정상적인 플러그인을 호출해야 한다. 플러그인이 호출되면 플러그인은 키 교환 및 이미지 교환을 위하여 플러그인 내에 저장해 놓은 정상 사이트의 IP 에 접속을 하게 된다. 서버에서는 플러그인의 접속 요청을 확인하여 사용자의 IP를 확인하고 플러그인을 호출하기 전에 HTML 코드를 사용자에게 전달했는지를 파악할 수 있다. 이 과정에서 서버는 사용자에게 HTML 코드를 전달하지 않았음을 파악하고 사용자에게 피싱을 알리는 이미지를 전송하게 된다. 또한 피싱 사이트가 파밍을 이용하여 플러그인에 정상적인 그림을 출력했다 하더라고 브라우저에서 HTTP Request로 요청하는 그림의 주소는 절대주소가 아닌 상대주소 이므로 이미지를 표시할 수 없게 된다. 이미지를 표시하기 위하여 HTML 코드를 수정하게 되면 해시 결과가 일치하지 않으므로 플러그인은 피싱 경고 이미지를 전달받게 된다. [그림 14] DNS 파밍에 의한 공격을 보여준다. 로그인박스 옆에 보이는 그림은 이미지의 이름을 해시된 결과를 상대주소로 하여 요청하기 때문에 공격자의 서버에는 이미지의 파일이 없다. 그러므로 이미지는 보이지않고 플러그인을 호출했다 하더라도 HTML 코드의 해시 결과가 다르다면 피싱을 경고하고 HTML 코드의 해시 결과가 같다면 플러그인은 정상적인 그림을 보여주지만 코드의 수정이 없다면 공격이 이루어지지 않고 사용자는 웹브라우저의 그림이 이상함을 보고 피싱사이트에 접속했음을 확인할 수 있다.

4.2 숨겨진 공격 방지

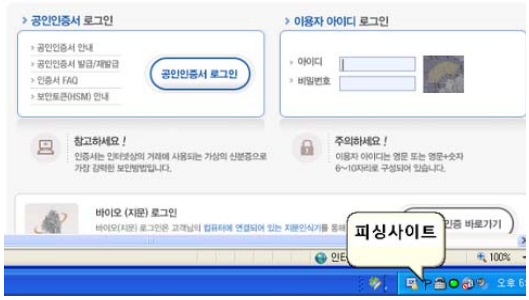
본 논문에서 제시한 기법은 HTML 코드의 해시를 통하여 HTML 코드의 무결성을 보장한다. Overriding Page Content기법과 같이 정상적인 서버에서 생성한 HTML 코드에 공격자가 정보를 습득하기 위한 코드를 추가해야 하므로 코드의 변경이 필요하게 된다. 변경된 코드가 사용자의 브라우저에 전송되고 플러그인이 호출되면 플러그인은 변경된 HTML 코드를 해시한 결과를 서버에 전송하고 서버는 사용자로부터 받은 HTML 코드의 해시 결과와 서버에 저장된 HTML 코드의 해시 결과를 비교하여 사용자에게 보냈던 HTML 코드가 변경됨을 인식하고 사용자에게 경고이미지를 전송하게 된다. 이 과정에서 코드 변경을 이용하는 숨겨진 공격을 방지 할 수 있게 된다. 또한, 변경된 HTML로 만들어진 이미지 파일이름으로는 웹 브라우저에 정상적인 이미지를 표시할 수 없게 된다. [그림 15]는 금융사이트의 로그인 이미지 부분을 임의적으로 바꾼 후 플러그인을 호출했을때의 모습이다. 웹페이지에는 일반적인 그림이 나타나지만 플러그인은 HTML 코드의 해쉬 결과가 다름을 확인하고 피싱사이트 경고를 보여준다.

4.3 상용프로그램과의 비교

본 논문에서 사용하는 플러그인은 HTML 코드의 변경을 분석하여 숨겨진 공격을 방지하는데 주된 목적이 있다. 서버와 클라이언트 사이에 전달된 HTML 코드를 해시 결과를 통하여 비교함으로서 숨겨진 공격을 파악



[그림 14] DNS 파밍에 의한 브라우저 및 플러그인 그림



[그림 15] 코드를 변경한 피싱사이트

[표 1] 피싱 방지 프로그램 성능 분석

	특징	URL 스푸핑	아이피 기반 접속	DNS 파밍	Hidden Attack	비고
브라우저 필터	블랙리스트	○	X	X	X	기본적 기능 제공
SpoofStick	URL 확대	△	X	X	X	주소 표시에 중점
NetCraft ToolBar	서버 정보 확인	○	○	○	X	호스팅 정보 기반
TrustBar	SSL 접속 및 인증	○	X	○	X	SSL 미사용 서버 사용 불가
Account Guard	IP 확인	○	○	○	X	Ebay/Paypal 전용
SiteAdviser	블랙리스트	○	○	○	X	사이트 분석 정보 기반
NetTrust	사회공학	○	X	X	X	악의적인 사용자 대처 능력 부족
ClientPhishingPro	블랙리스트	○	○	○	X	주소표시줄 풍선도움말 사용
Nophishing	블랙리스트	○	○	○	X	팝업 기반의 경고
제안 기법	사용자 인지기반	○	○	○	○	HTML 코드의 변경 유무 확인

할 수 있다. 현재의 상용 프로그램들은 도메인 네임과 IP 의 비교를 통하여 피싱과 파밍을 방지하고 사용자나 기업에서 피싱/파밍 사이트를 찾아내고 블랙리스트에 저장하는 방식을 사용하지만 본 플러그인은 웹브라우저에서 요청하는 이미지의 유무와 사이트와 사용자의 HTML 코드 비교를 통하여 DNS 파밍 및 스푸핑을 구별할 수 있다. 또한 툴바 형식의 프로그램의 경우 이미지 치환 공격에 취약하므로 이미지 치환으로 위조하기 어려운 작업관리줄의 트래이 아이콘과 풍선도움말을 사용하였다. 하지만 본 플러그인은 플러그인을 호출하는 서버와 직접적인 통신을 하는 기능이 주된 기능이므로 블랙리스트에 의한 피싱 구별 방법은 사용하지 않았다. 표 1는 2.2장의 상용 피싱 방지 프로그램과 본 플러그인의 공격 방지를 비교한 표이다[12].

Ⅴ. 결 론

피싱 공격과 파밍 공격 및 숨겨진 공격 등의 공격은 사용자가 스스로 구별하기에는 어려움이 있다. 또한 숨겨진 공격의 경우 일상적인 사용이 가능할 정도로 사이트의 변조가 가능하므로 HTML 코드의 무결성을 보장하는 것이 중요하다. 현재 상용 피싱 방지 솔루션들은 DNS 스푸핑 및 DNS 파밍 공격 등을 방어 할 수 있지만 Overriding Page Content 과 같은 코드의 변경을 통한 사용자 정보를 가로채는 기법을 방지 하는 데는 부족함이 있다.

본 논문에서 제안한 피싱 방지 플러그인 모델은 사용자로 하여금 피싱에 대한 인식을 할 수 있게 함과 동시에 HTML 코드의 변경을 감시함으로서 숨겨진 공격을 방지하고 이와 동시에 DNS 파밍 및 DNS 스푸핑 공격을 방지할 수 있다. 현재 ActiveX 나 XPCOM 과 같은 플러그인을 초반에 설치할 경우 사용자가 플러그인이 공격프로그램인지 신뢰할 수 있는 프로그램인지에 대한 판단을 하기가 쉽지 않다. 현재 인터넷을 기반으로 설치되고 있는 모든 플러그인은 프로그램 제작사의 공개키로 서명이 되어있고, 이를 기반으로 사용자가 신뢰할 수 있게 하는 것이 일반적이다. 이러한 부분은 사용자가 처음부터 피싱/파밍 공격에 노출 된다면 플러그인의 첫 설치부터 문제가 된다. 플러그인을 설치하는데 있어서 사용자의 공인인증서를 이용하는 방법 등의 사용자와 서버 양측에 신뢰를 줄 수 있는 장치가 필요하다. 또한 서버의 입장에서 플러그인을 전달받는 사용자가 공격자인지 선의의 사용자를 판단할 수 있는 도구가 마련되어야 한다. 플러그인을 처음 설치하면 Plugin ID 를 서버로부터 받으므로 서버가 사용자와 공격자를 구별할 방법이 강구된다.

향후 연구로는 본 논문에서 사용한 인지기반의 피싱 방지 프로그램을 모바일 인터넷과 같은 제한된 시스템에서 적용 시키는 방법과 ActiveX 와 XPCOM 기반이 아닌 SSL 기반으로 확대하여 미래에 플러그인을 지원하지 않을 브라우저에 대해서도 확대 적용 하고자 한다.

참고문헌

[1] Phishing Activity Trends, http://www. antiphishing. org/reports/apwg_report_may_ 2007.pdf.

[2] G. Ollmann, The Phishing Guide, NGS Software Ltd., Sep. 2004.

[3] P.P. Swire, “Report from the National Consumers League Anti-Phishing Retreat,” National Consumers Leage, Mar. 2006.

[4] G. Tally and R. Thomas, “Anti-Phishing: Best Practices for Institutions and Consumers,” Anti-Phishng Working Group, pp. 8-20, Nov. 2004

[5] J. Stewart, “DNS Cache Poisoning - The Next Generation,” LURHQ, pp. 1-13, Jan. 2003.

[6] D. allan, “Identity Theft, Phishing and Pharming: Accountability & esponsibilities,” OWASP AppSec, pp. 23-27, Oct. 2005.

[7] G. Ollmann, Security Best Practice: Host Naming and URL Conventions, NGS Software Ltd, pp. 5-6, Jan. 2005.

[8] L. Fette, “Learning to Detect Phishing Emails,” Proceedings of the 16th International conference on World Wide Web, pp. 649-656, May 2007.

[9] P. Kumaraguru, “Protecting People from Phishing: The Design and Evaluation of an Embedded Training Email System,” SIGCHI conference on Human factors in computing systems, pp. 905-914, May 2007.

[10] M. Wu, “Do Security Toolbars Actually Prevent Phishing Attack?,” SIGCHI conference on Human Factors in computing systems, pp. 601-610, Apr. 2006.

[11] C. Karlof, “Dynamic pharming attacks and locked same-origin pol icies for web browsers,” Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 58-71, Oct. 2007.

[12] R. Dhamija, “Phish and HIPs: Human Interactive Proofs to Detect Phishing Attacks,” Human Interactiv e Proofs, LNCS 3517, pp. 131-139, 2005.

[13] Y. Zhang, S. Egelman, L. Cranor, and J. Hong, “Phinding Phish: Evaluating Anti-Phishing Tools,” Proceedings of the 14th Annual Network and Distributed System Security Symposium, Mar. 2007.

[14] Hidden Frame & Graphical Substitutio n, http://www.contentverification.com/ attacks.html

[15] eBay's Account guard, http://pages.ebay.com/ help/confidence/account-guard.html.

[16] Netcraft Toolbar, http://toolbar.net craft.com

[17] spoofstick, http://spoofstick.com.

[18] McAfee Siteadviser, http://www.siteadviser.com

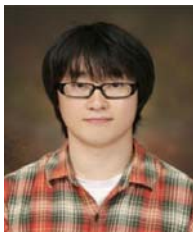
[19] TrustBar, http://www.cs.biu.ac.il/~herzbea/TrustBar/.

[20] NetTrust, http://www.ljean.com/NetTrust/.

[21] ClientPhishingPro, http://www.softforum.co.kr.

[22] NoPhishing, http://www.softrun.com.

<著者紹介>



김 주 현 (JuHyun Kim) 학생회원
2007년 2월: 인하대학교 정보통신공학과 졸업
2007년 3월~현재: 인하대학교 정보통신대학원 석사 과정
<관심분야> 네트워크 보안, 인터넷 보안

맹 영 재 (YoungJae Maeng) 학생회원
2006년 8월: 인하대학교 컴퓨터 공학과 졸업
2008년 8월: 인하대학교 정보통신대학원 석사
2008년 9월~현재: 인하대학교 정보공학과 박사 과정
<관심분야> 인터넷 보안, 네트워크 보안, 웹 인증 보안

양 대 헌 (DaeHun Nyang) 종신회원
1994년 2월: 한국과학기술원 과학기술 대학 전기 및 전자 공학과 졸업
1996년 2월: 연세대학교 컴퓨터 과학과 석사
2000년 8월: 연세대학교 컴퓨터 과학과 박사
2000년 9월~2003년 2월: 한국전자통신연구원 정보보호연구본부 선임연구원
2003년 2월~현재: 인하대학교 정보통신대학원 부교수
<관심분야> 암호이론, 암호프로토콜, 인증프로토콜, 무선 인터넷 보안

이 경 희 (KyungHee Lee) 정회원
1989년: 서울대학교 식품영양학과 학사
1993년: 연세대학교 전산과학과 학사
1998년: 연세대학교 컴퓨터과학과 석사
2004년: 연세대학교 컴퓨터과학과 박사
1993년 1월~1996년 5월: LG소프트(주) 연구원
2000년 12월~2005년 2월: 한국전자통신연구원 선임연구원
2005년 3월~현재: 수원대학교 조교수
<관심분야> 영상처리, 컴퓨터비젼, 인공지능, 패턴인식, 생체인식, 얼굴인식, 다중생체인식