후킹의 예술(Art of Hooking)

- COM 후킹을 통한 현 금융권의 포괄적 취약성 노출 -

발표자: AmesianX (amesianx@nate.com)

CodeEngn 리버스 엔지니어링 세미나

- 무엇을 논의하고자 함인가?
- 웹의 발전배경
- 리버스 엔지니어링의 대두
- ActiveX 의 설치과정
- ActiveX 문제의 시발점
- 문제해결 대안으로 발생한 또다른 문제점
- 정상적인 함수에서 발생하는 문제점
- 정상적인 함수의 취약성 판별(모니터링)
- 목차가 많아서 다 못 적었습니다.

무엇을 논의하고자 함인가?

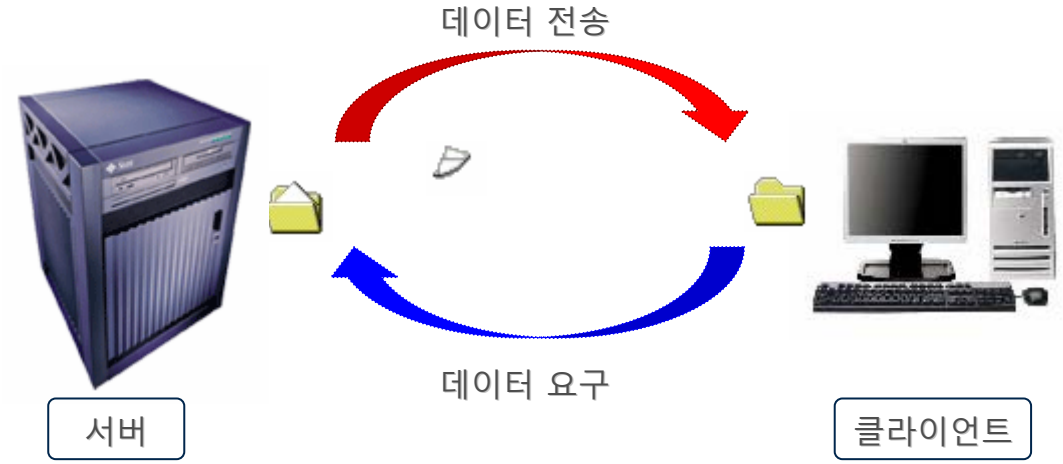
- ActiveX 바이너리 조작 및 후킹.
- ActiveX 바이너리 조작을 하지않는 범용적 COM 후킹. (모든 COM 모듈을 후킹할 수 있음)
- 최근 언론에 보도되는 키보드 후킹(키로거)은 크래커가 가장 직관적으로 접근하는 해킹수단. (언급은 하지만 논의에서 제외함)
- 키보드 후킹의 한계는 사용자 입력을 예상하기도 힘들고 정확히 어떤 행동(Action) 중인지 포착하는 인공지능적 해킹이 어렵다. 즉, 저수준 해킹에 머물게된다. 저수준 입출력 해킹의 한계는 항상 그 포착 시점의 탐지를 찾아내는 알고리즘의 문제가 수반된다.
- 논의점 - 키보드 후킹은 스니핑과 다를바 없으므로 방지하기위해서 다른 시스템을 강화한다고 했을때, ActiveX 와 같은 COM 에 의존하면 심각한 구조적 문제에 직면한다. (마치 궁극적으로 키보드 후킹을 막을 수 없고 ARP 구조에의해 스니핑을 막을 수 없는 것처럼)
- 먼저 웹의 발전배경부터 알아본다.

웹의 발전배경

- 웹의 발전배경은 매우 지루한 논의 이지만 왜 이렇게 발전할 수 밖에 없었는가에 대해서 논의하는데 필연적인 이유를 보여준다. 우리가 웹 발전사에서 눈 여겨 보아야 할 것은 MS 의 기술이다.

- 현재 우리가 사용하고 있는 인터넷 서비스인 웹의 모습은 탄생이후로 두번의 큰 혁신을 거쳤다.

- 첫번째 혁신은 사용자의 입력을 효율적으로 처리하기위한 서버측 기술문제를 해결하는 과정에서 나타났으며 이를 CGI 로 압축설명 할 수 있다.



- 클라이언트는 요청을 하고 서버는 단순히 데이터를 전송해주는 역할이 최초 웹의 모습이다. 사용자의 요구가 다양해지고 서버에서 더 많은 요구를 처리해야할 필요성이 대두되어 CGI 가 발전하기 시작하였다.

웹의 발전배경

❑ 두번째 혁신은 서버와 클라이언트 모두 발전을 거듭했다. 그 이전까지 서버지향적이던 웹은 사용자 PC 의 자원을 활용할 수 있게 됨으로써 더 동적이고 화려해졌으며 이 기술에서 탄생한 것이 자바진영의 Applet (애플릿) 과 Microsoft 사의 ActiveX(액티브엑스) 기술이다.

웹의 발전배경

❑ **현재는 세번째 혁신기이자 시대적 트랜드인 웹 2.0 으로 가는 과도기적 단계에 이르렀다. 클라이언트를 동적 구조로 만들어 웹 서비스 자체가 하나의 어플리케이션화 되는 것이 웹2.0이다. MS 에서는 웹2.0 시대를 주도하기위해 ActiveX 를 대체할 기술로 스마트 클라이언트(Smart Client) 를 밀어붙이고 있음. (닷넷기반)**

서버의 발전과정

Ajax 로 대표됨

서버에서 처리하던 CGI 기술이 점차 클라이언트로 이동. 서버는 클라이언트가 보내는 데이터를 신뢰할 수 있는 기술 필요. 보안취약성 대두됨.

- 웹2.0 기술흡수
기존 CGI 들 PHP/JSP/ASP
-MS 의 닷넷 프레임워크
ex. 비스타에서 기본탑재됨

데이터 전송

데이터 요구

서버로의 처리요구는 상대적으로 줄어듬

클라이언트의 발전과정

서버의 도움없이도 웹서비스 자체는 하나의 어플리케이션 처럼 사용하게 됨 (리플래쉬가 거의없음)

-클라이언트 발전
Ajax 기반 웹 2.0
-MS 의 닷넷프레임 워크 기반, 웹상에서 닷넷기반 어플리케이션 실행가능

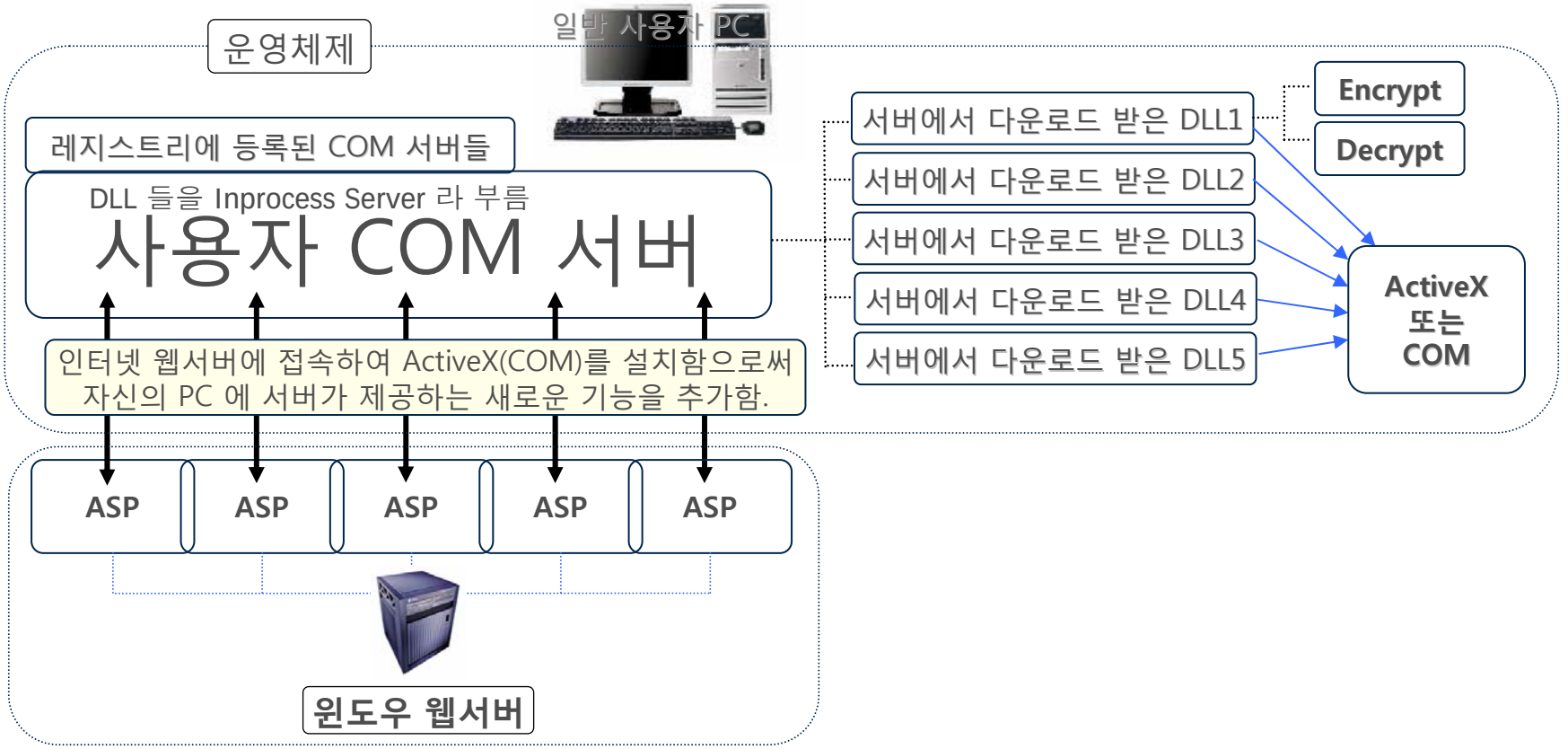
*리플래쉬: 클라이언트가 서버로 새로운 웹 페이지를 요구하는 것으로 이때 깜빡임이 발생하며 클라이언트의 기존정보는 기본적으로 모두 삭제된다. 인터넷 익스플로어에서 F5 키를 누르는 것을 의미하기도 함.

리버스 엔지니어링의 대두

- 웹의 발전배경에서 본 것처럼 자바진영의 Applet 을 누르고 승자는 ActiveX 가 되었다.
- MS 는 기존의 웹 환경에서 처리하기 힘든 사용자 PC 의 자원을 클라이언트 분산기술인 ActiveX 로 확보하려고 하였다. (현재는 닷넷으로 전향)
- 즉, 사용자의 PC 를 서버의 노예화 시키는 것이 그 목표라고 할 수 있다. (현재의 닷넷 역시 동일한 의도, MS 는 표준을 지키지 않고있다.)
- 그러나 MS 의 ActiveX 기술은 서버의 기능확장만을 고려한 나머지 클라이언트를 심각한 보안문제로 빠뜨리고 말았다.
- MS 는 이미 이러한 심각한 보안문제를 사전에 알았을 것이다. 그러나 결단을 감행한 것은 MS 가 여지껏 서버시장에서 입지를 굳히지 못하고있기 때문인 것으로 보인다.
- 이러한 클라이언트 ActiveX 의 심각한 보안결함을 수면위로 부상시킨 것이 바로 리버스 엔지니어링이다.

ActiveX 의 설치과정

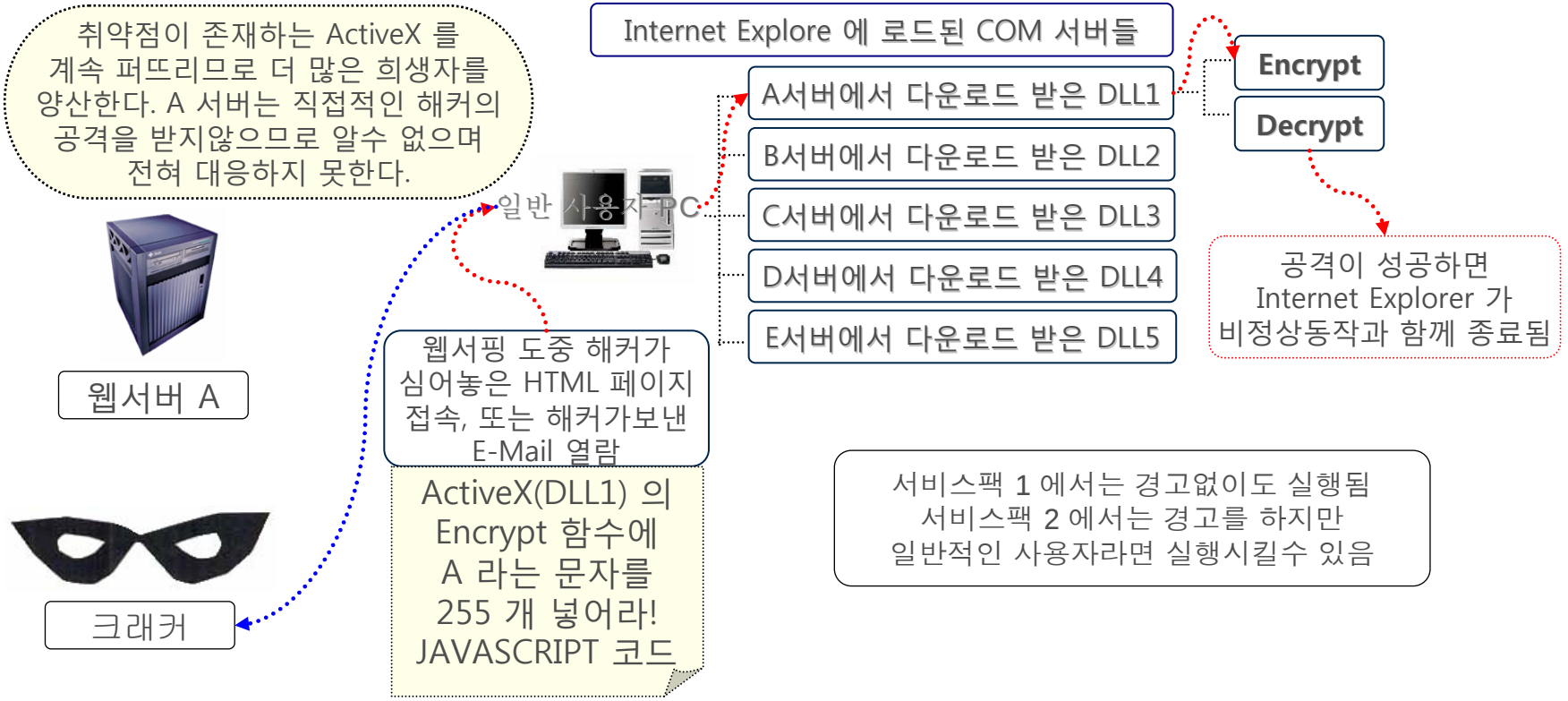
❑ 서버로부터 확장기능을 부여받기 위해 클라이언트는 ActiveX (COM) 를 다운로드 받아 시스템 레지스트리에 Control 로써 등록한다. 모든 ActiveX 는 이 등록절차를 반드시 자동화해야 한다는 프로그래밍 법칙이 있으며, 사용자의 설치허가가 떨어지면 나머지는 추가개입 없이 자동으로 시스템에 등록된다.



ActiveX 문제의 시발점

PANIC SECURITY
www.panicsecurity.com

❑ **주인도 몰라보는 ActiveX 의 문제점. 웹서버A 가 보내는 HTML 코드에서만 실행되어야 함에도 불구하고 크래커가 마음만 먹으면 자신의 코드를 보내 누구나 상대방 PC 의 ActiveX 코드를 실행시킬 수 있다. 또한, BOF 공격까지 감행할 수 있다.**



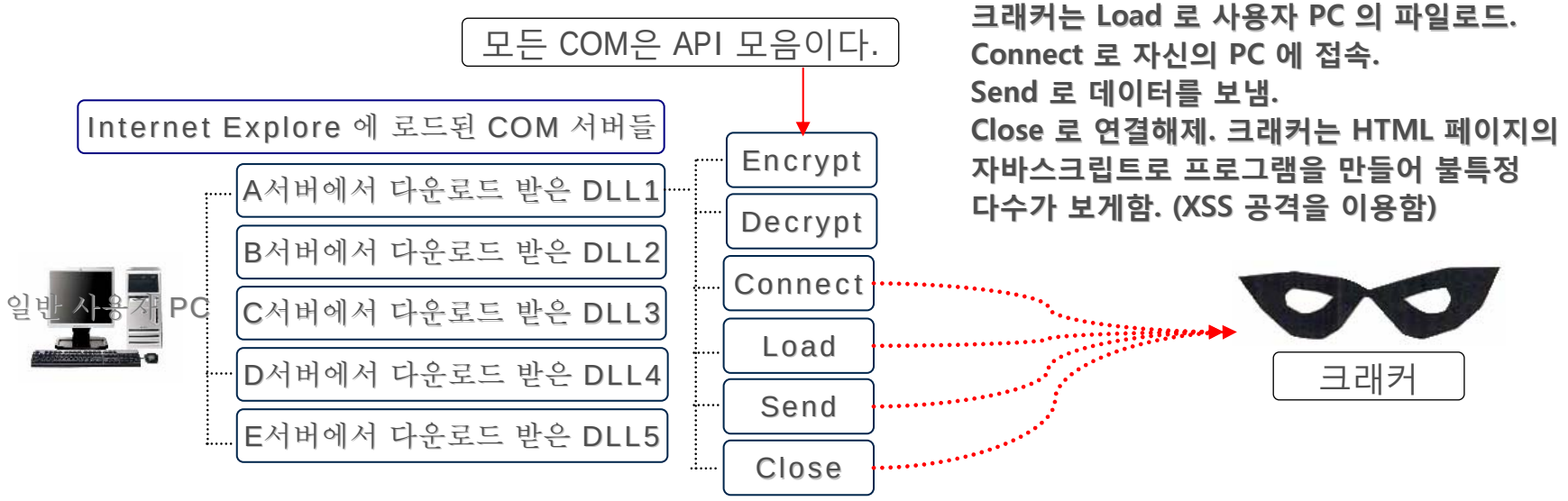
문제해결 대안으로 발생한 또다른 문제점

- 이러한 취약점을 막을 수 있는 대안은 존재했다. 그것은 미연에 점검을 하는 것이다.
- 자동화 공격툴(일명 Fuzzer: 퍼져)을 이용하여 미연에 점검을 할 수 있다.
- 그러나 또 다른 심각한 문제점이 존재한다. 그것은 ActiveX 의 정상적인 기능들이 취약한 경우이다.
- 최근에 이러한 또 다른 문제점(정상기능을 악용)이 화두에 오르기 시작하였으며 Jikto 라는 자바스크립트로 만든 웹 스캐너가 그 대표격이라 할 수 있다.

정상적인 함수에서 발생하는 문제점

- 다음과 같은 정상 함수들이 UltraEdit 를 설치해도 ActiveX 로 등록된다.

- 최근에 이슈가 된 Jikto(직토) 를 보면 다음의 시나리오와 별반 다를바 없는 구조이다. Ajax 에서 필수적인 XMLHttp 가 바로 시나리오에서 보는 것과 비슷한 함수들을 갖고있다. Connect, Send 등등의 함수들을 이용하여 사용자 PC 를 웹서버 공격을 위한 원격 웹스캐너로 만드는게 Jikto 이다. 단, 엄청난 차이점은 Ajax 가 XMLHttp 를 사용할때는 사용자에게 경고없이 실행된다. (시나리오에서 사용자 PC 의 파일을 로딩하는 취약점은 가상의 시나리오임.)



정상적인 함수의 취약성 판별(모니터링)

- 정상적인 함수가 호출될때 그것이 정상적인 호출인지 확인하는 방법의 필요성.
- 즉, 함수가 호출될때 그 매개변수(인자)로 넘어가는 것이 타당한지 확인할 필요성이 대두됨.
- 이를 실현하기 위한 기술로써 COM 후킹방법이 필요하다.
- COM 후킹의 방법에 있어서 두가지의 대표적 후킹방식이 존재한다.
  - 코드인젝션 후킹 (Code Injection: 기존의 후킹이론에 대응)
  - COM 범용랩퍼 후킹 (모든 COM 의 가상함수 테이블을 가로채는 후킹이론)
- 사실상 이러한 대표적 후킹방식은 큰 틀의 후킹방법론 중 일부분에 불과하며 수많은 후킹방법이 존재한다.
- 먼저, 기존의 후킹방식을 ActiveX (COM DLL 파일) 에 적용하려면 COM 이 무엇인가에 대해서 알 필요가 있다. 그 이유는 후킹할 함수의 정확한 위치(주소)를 바이너리에서 찾아내야 하기 때문이다.

COM 이란 무엇인가?

- COM 은 악마의 자식이다. (-_-);

- 악마의 자식이라고 표현한 이유는 악용될 수 있기 때문이란 의미가 아니라 COM 프로그래밍 자체가 너무 난해해서 악마적인 프로그래밍이라해도 과언이 아니므로 만드는 사람은 악마이고 그로인해 태어난 COM 모듈은 악마의 자식이라는 엉뚱한 생각을 해볼수 밖에없다. 그 정도로 COM 은 처음 접하는 자에게 난해하다.

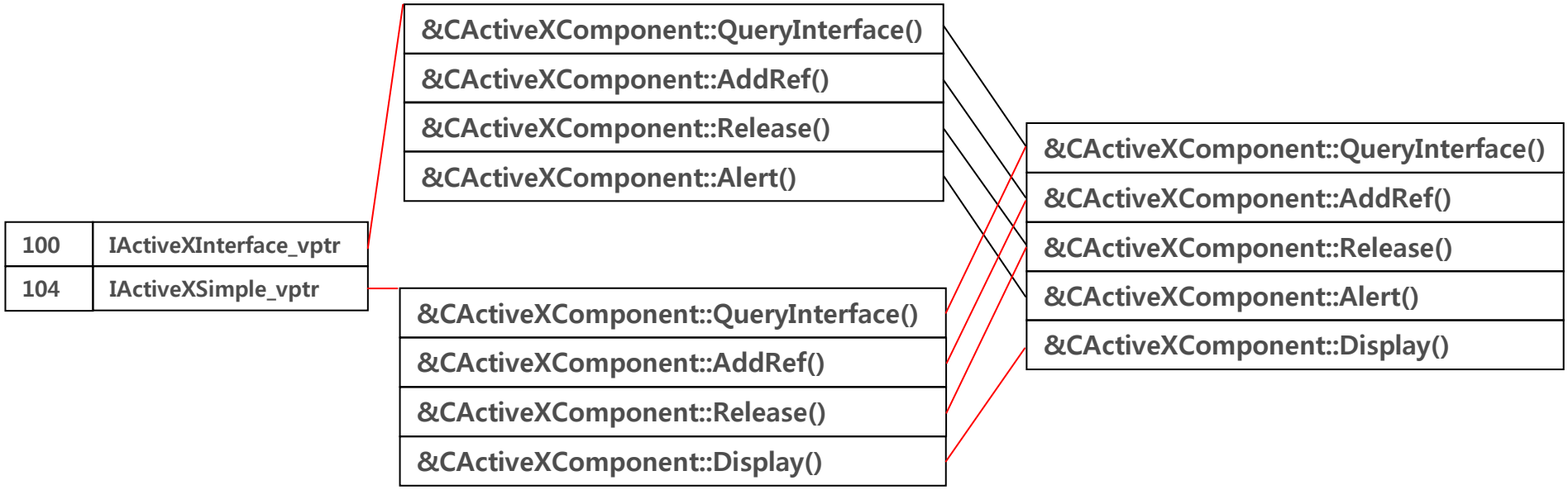
- 여기서 COM 을 한 마디로 요약하기에는 너무 많은 특징들이 있으므로 COM 책을 꼭 읽어보길 권한다.

- 본 발표자의 주관적인 생각으로 COM 을 한마디로 함축하라고 한다면 COM 이란 윈도우의 모든 것 그 자체라고 말하고 싶다.

- 한 가지 확실히 짚고 넘어가야할 것은 COM 은 C++ 과 그 구조적인 모습이 같다. COM 은 C++ 라이브러리이자 API 묶음인 동시에 그 스스로가 완벽히 객체이며 매우 유기적으로 통합되는 말 그대로 컴포넌트 객체모델이다.

COM 의 가상함수 관계도

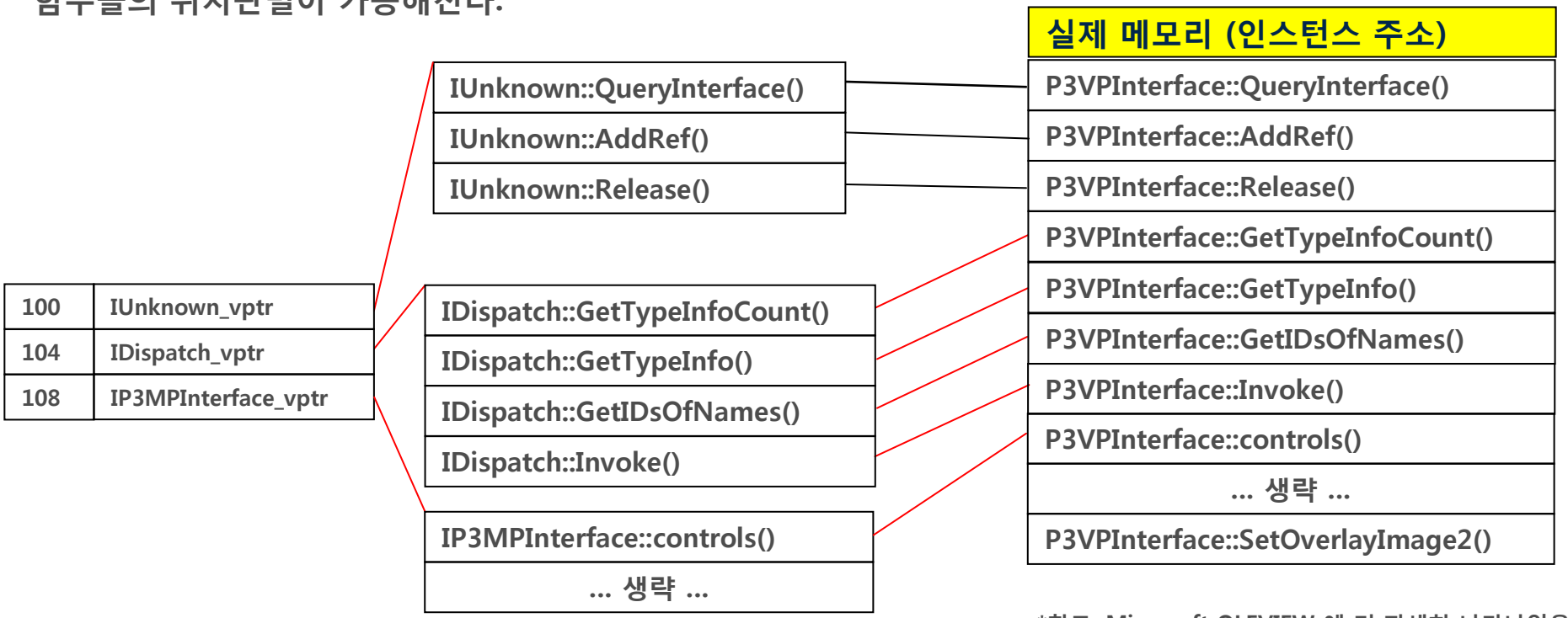
(COM 이 되기 위해서 필요한 조건)
최소한 한개이상의 IUnknown 인터페이스를 상속 받아야 함



*참고: Beginning ATL COM Programming 49 페이지 그림 편집 (정보문화사 출판)

실질적인 ActiveX 의 가상함수 관계도

- ActiveX 를 대상으로한 실질적인 가상함수 테이블의 모습.
- 즉, P3VPInterface 라는 ActiveX(COM 모듈) 의 COM 인스턴스를 만들면 메모리에 다음의 인스턴스가 생성되며 가상함수 테이블을 메모리에서 찾아낼 수 있다.
- 인스턴스의 주소에서 가상함수 테이블에서 함수들의 주소들을 뽑아낼 수 있으며 ActiveX 바이너리에서 실제 함수들의 위치판별이 가능해진다.

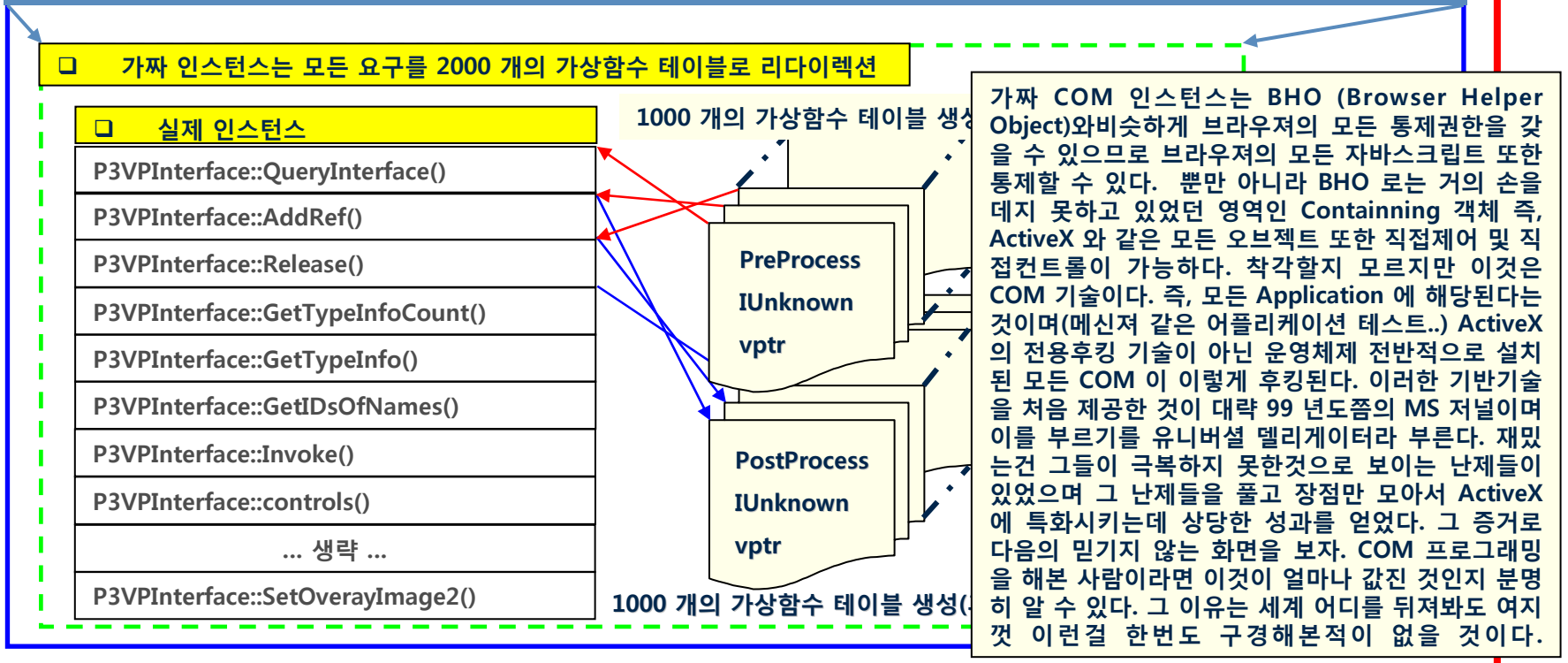


*참고: Microsoft OLEVIEW 에 더 자세히 나타나있음

❑ 판도라의 상자안에는 뭐가 있을까?

[클라이언트] COM 서버를 탑재하며 대표적 클라이언트는 MS Internet Explorer 이다. 이를 호스트 또는 컨테이너라 부름

MSHTML.DLL – 실제 인터넷을 하도록 브라우져 역할(COM 모듈임), OBJECT 태그를 만나면 ActiveX를 로딩 시스템에 등록된 정보로부터 COM 을 자신의 메모리영역에 갖음. (자바스크립트 로딩은 다름)



가짜 COM 인스턴스는 BHO (Browser Helper Object)와비슷하게 브라우져의 모든 통제권한을 갖을 수 있으므로 브라우져의 모든 자바스크립트 또한 통제할 수 있다. 뿐만 아니라 BHO 로는 거의 손을 데지 못하고 있었던 영역인 Containning 객체 즉, ActiveX 와 같은 모든 오브젝트 또한 직접제어 및 직접컨트롤이 가능하다. 착각할지 모르지만 이것은 COM 기술이다. 즉, 모든 Application 에 해당된다는 것이며(메신져 같은 어플리케이션 테스트..) ActiveX 의 전용후킹 기술이 아닌 운영체제 전반적으로 설치된 모든 COM 이 이렇게 후킹된다. 이러한 기반기술을 처음 제공한 것이 대략 99 년도쯤의 MS 저널이며 이를 부르기를 유니버셜 델리게이터라 부른다. 재밌는건 그들이 극복하지 못한것으로 보이는 난제들이 있었으며 그 난제들을 풀고 장점만 모아서 ActiveX 에 특화시키는데 상당한 성과를 얻었다. 그 증거로 다음의 믿기지 않는 화면을 보자. COM 프로그래밍을 해본 사람이라면 이것이 얼마나 값진 것인지 분명히 알 수 있다. 그 이유는 세계 어디를 뒤져봐도 여지껏 이런걸 한번도 구경해본적이 없을 것이다.

- 바이너리 OPCODE 를 뜯어보지 않고도 범용랩퍼로 간단히 몇초만에 판도라 상자를 열게되다.
- 상당히 고무적인 기술일 수 밖에 없다. PPT 를 만들기위해 3시간에 걸친 바이너리 후킹작업을 단 10 초만에 한다.

해킹과 보안은 양날의 검

PANIC SECURITY
www.panicsecurity.com

- 앞서 설명한 것과 같이 COM 범용랩퍼 후킹, 다시말해 ActiveX 에 특화된 COM 범용랩퍼 후킹기술을 연구한 것은 보안을 위해 꼭 필요했던 요소이다. (현재로써 Jikto 를 막을 수 있는 유일한 방법일 수도 있다.)
- 그러나 뜻하지 않게 이 기술은 어둠의 요소까지 포함하고 있었다. 즉, 날이 한 개인 보안이라는 도인줄 알았으나 역시 양날이 존재하는 검이었다.
- 한쪽에 날이 서 있다는 것을 발견한 것은 오래전부터 그룹에서 같이 있었고 또 자신의 회사로 불러주신 해커분이셨다. 그 취약성은 곧 경험이 풍부한 개발자분의 탄력을 받았고 기술을 구현하였다.
- 우리는 중대한 문제점을 발견했으며 그 것은 ActiveX 또는, COM 전체에 심각한 구조적 문제를 불러올 수 있는 문제라고 판단하여 조심하고 또 조심하였다.
- 하지만 현재 키보드 보안과 같이 계속해서 보안에 관련된 문제점들이 부각되면서 정작 보아야 할 것을 보지 못하며 크래커의 칼에 시야가 잘려나가고 있다는 것을 느꼈다.
- 이에 언더적인 사고방식을 가진 리버스 엔지니어링 세미나에서 이 주제를 자세히 언급하기로 결정하였다.
- 다음의 화면은 실제 화면이며 문제의 소지를 없애기 위하여 철저하게 가려졌음을 알린다.

해킹과 보안은 양날의 검

PANIC SECURITY
www.panicsecurity.com

- 본 취약성은 특정한 금융권의 문제로 종속되지 않으며 특정한 솔루션에 종속적이지도 않은 범용적인 문제를 내포하고 있어 취약점이 아닌 취약성이라 부른다.
- 현재 이러한 취약성은 우리 사회에 있어서 중요한 부분을 차지하고있는 모든 금융권에 포괄적으로 존재한다.
- 뿐만 아니라, 금융권을 제외한 모든 민감한 정보를 ActiveX 로 다루고 있는 기관들이 오늘의 논의대상에서 벗어날 수 없다.
- 해킹은 양날의 검이다. 한쪽 면인 보안과 다른 한쪽 면인 해킹은 둘다 날카롭다. 다만, 그 바라보는 방향이 틀릴뿐 해킹을 하지 못하는 사람이 보안을 할 수 없듯이 해킹기술이 발전하지 않으면 보안기술은 발전할 수 없다.
- 충분히 감추고 숨기려하면 할 수 있겠으나 이러한 기술자체는 이미 해외에서 공개되고 사용하고 있다. 다만, 그 들이 사용하고있는 목적이 달라서 다른 이면에 존재하는 칼날을 보지못하고 있을 뿐이다.
- 만약, 해외에서 먼저 다른 칼날을 발견하고 관심을 갖는다면 그땐 누가 막을 것인가? 대책없이 당하는 것보다 최선의 방책을 세우도록 하는데 있어서 패닉시큐리티가 해야할 것은 바로 해킹기술과 보안기술을 연구하는 것이다. 그 이후의 판단은 여러분의 몫입니다.

진실은 저 넘어에..

㈜패닉시큐리티는 언더그라운드 해커들로부터 출발하였습니다. 국내 보안기술발전에 도움이 되는 리버스엔지니어링 세미나가 되었길 바라는 바입니다.

(주)패닉시큐리티
www.panicsecurity.com