안전한 Web 환경을 위한 HTTP/2 취약점에 관한 연구

류정현, 문서연, 박종혁*
서울과학기술대학교 컴퓨터공학과
e-mail : {jh.ryu, moon.sy0621, jhpark1}@seoultech.ac.kr

A HTTP/2 Security Vulnerability for the Secure Web Environment

Jeong Hyun Ryu, Seo Yeon Moon, Jong Huyk Park*
Dept. of Computer Science and Engineering, Seoul National University of Science and Technology (SeoulTech),Seoul, 139-743, REPUBLIC OF KOREA
e-mail : {jh.ryu, moon.sy0621, jhpark1}@seoultech.ac.kr

요 약

Web 환경이 급격히 변화함에 따라 HTTP 프로토콜의 변화도 요구되었다. 이를 보완하기 위한 비동기 메커니즘, Ajax 등이 제시되었고 최근 사물인터넷, 클라우드 등을 활용한 웹 어플리케이션이 주목 받고 있다. 이러한 패러다임의 변화로 웹은 여러 가지 기능이 필요하게 되었으며 HTTP/1 의 단점을 보완한 HTTP/2 가 개발되었다. HTTP/2 는 웹 어플리케이션 및 Hypertext page 변화를 위해 복합적인 기능들이 추가 되었으나 이러한 추가적인 요소에 대해 또 다른 보안 취약점이 나타났다. 웹 어플리케이션은 사용자의 서비스에 직접적인 영향을 미치기 때문에 보안 위협 및 그 피해가 매우 치명적일 수 밖에 없다. 따라서 이러한 보안 취약점에 대한 보안 대책이 시급하다. 본 논문에서는 HTTP/2 의 주요 취약점에 대해 분석하고 네 가지 보안 위협에 대해 기술하여 앞으로의 HTTP/2 에서의 웹 보안 대책 및 연구에 기여하고자 한다.

1. 서론

HTTP 는 World Wide Web 의 주를 이루는 요소 중 하나이다. 이는 간단한 요청-응답 방식의 프로토콜이며, 현재까지 유지되고 있다. 하지만 해가 거듭되면서, 인터넷은 급격히 변화하고 있으며 이를 포용하기 위해 다양한 비동기 메커니즘과 Ajax 등이 수십, 수백 개의 자원으로 이루어진 웹 페이지들 간의 Hypertext page 들을 변환하기 위해 제시되었다 [1]. 뿐만 아니라 사물인터넷, 클라우드가 점차 보편화 되고 있으며 일반적으로 사용자에게 웹 어플리케이션을 통해 서비스된다. 이러한 지속적인 필요성에 의해 HTTP/2 프로토콜은 새로운 패러다임에 대해 적용 및 확장 할 수 있도록 차세대 프로토콜로 디자인되었다 [2].

새로운 HTTP/2 는 HTTP/1.x 의 기본 방식이 일부 유지되고 있지만, HTTP/2 프로토콜은 완벽한 기술과 새롭고 획기적인 메커니즘으로 평가된다. 하지만 기존의 많은 예시들에 의해 이러한 획기적인 변화와 복합성의 증대는 그만큼 많은 공격과 위협의 여지가 확대됨을 의미한다. 여러 기능 및 방식들이 추가됨에 따라 HPPT/2 는 새로운 취약점을 나타내게 되어있다. 웹 어플리케이션은 사용자의 서비스에 직접적인 영향을 미치기 때문에 보안 위협 및 그 피해가 매우 치명적일 수 밖에 없다. 따라서 이러한 보안 취약점에 대한 보안 대책이 시급하다. 따라서 본 논문에서는 HTTP/2 의 주요 취약점을 분석 및 살펴보고 네 가지 보안 위협에 대해 기술한다.

2. HTTP/1.x 와 다른 HTTP/2

기존의 HTTP/1.x 는 요청-응답 방식으로, 클라이언트가 서버에 요청을 보내고 서버로부터의 응답을 기다린다. 이러한 특성 때문에 대역폭에 상관없이 Round-trip time 이 길어져 응답시간이 지연된다. 이에 반해 HTTP/2 는 네 가지의 주요 변경 사항으로 응답 시간을 획기적으로 줄였다. HTTP/2 의 변경된 새로운 요소는 다음과 같다 [2].

1. 단일 TCP 연결을 통한 다중 스트림을 동시에 전송 가능
2. Compression 프로토콜을 통한 헤더 압축
3. 서버 푸시
4. 스트림 우선순위 및 의존성

위의 HTTP/2 프로토콜은 세 개의 논리계층 구조로 이루어져 있다. 먼저 The Transmission Layer 는 스트림,

프레임, 흐름 제어등을 하며 HPACK 은 바이너리 인코딩과 압축 프로토콜 기능을 가진다 [3]. 마지막으로 The Sematic Layer 는 HTTP/1.1 의 서버보다 강화된 Push 기능 가졌다.

또한 HTTP/1.1 과 마찬가지로 HTTP/2 는 TCP/TLC 계층 위에 구현되었다. HTTP/2 의 주요 블록은 다음과 같이 변환된다.

1. HTTP/2 Frames : HTTP/2 의 기본 데이터 유닛은 HTTP/1.1 의 그것을 대체한다. 기존 버전의 ASCII 인코딩에 반해 HTTP/2 는 Binary 인코딩을 사용하며, 프레임들은 Header frame, Data Frame, Setting Frame 으로 나뉜다.
2. HTTP/2 Streams : HTTP/2 는 기존의 request-respond 프로토콜을 대체한다. HTTP/2 의 스트림은 전송된 프레임들의 쌍방채널이다.
3. TCP Connection : HTTP/2 는 서버의 TCP 연결 사용을 줄이기 위해 다중 스트림을 전송하기 위해 단일 TCP 연결을 허용한다.

HTTP/2 의 스트림과 흐름제어방식은 HTTP/1.x 방식과 차이가 있으며 네 가지 메커니즘을 가진다. Stream concurrency 은 다중 스트림을 단일 TCP 연결로 전송한다. Flow control 은 수신자는 발신자에게 데이터의 한계치를 신호를 통해 알려주며, Stream priority 와 Stream dependency 는 각각 발신자는 수신자에게 스트림 사이의 우선순위(Priority)와 의존성(Dependency)에 대해 신호를 통해 교환한다 [4].

Flow control: 이 메커니즘은 연결 설정에서 쓰일 수 있는 파라미터들을 가지고 있으며 TCP 계층에서 연결에서 가능한 동시 스트림 수를 설정하는 SETTINGS_MAX_CONCURRENT_STREAMS 와 발신자가 수신자에게 전송할 수 있는 데이터의 최대치를 알리는 WINDOW_UPDATE 및 등을 포함한다. 스트림 우선순위나 의존성 같은 다른 파라미터들은 새 스트림을 초기화 할 때 발신자에 의해 보내진다.

이러한 새로운 흐름 제어 방식은 발신자에게 전송 프로토콜에 영향을 미치는 강한 권한을 심어주게 되는데 일반적인 경우 이는 사용자 권한을 향상시키게 된다. 하지만, 이를 악용하는 경우 서버에 극심한 자원의 소비를 초래하여 DoS 공격에 대해 노출 및 피해를 심화 시킬 수 있다는 위험이 존재한다 [5][6].

HPACK: 헤더 압축 프로토콜은 헤더 전송의 대역폭 소비를 줄여 응답 시간을 향상시킨다. 이 프로토콜은 주로 개인 정보 유출인 보안 이슈에 있어 공격 및 위협을 막기 위해 일정 수준의 압축을 제공했다. HPACK 프로토콜은 스태틱 인코딩 테이블과 다이나믹 인코딩 테이블에 의존하는데, 발신자가 다이나믹 테이블의 사이즈를 설정하지만 수신자가 설정한 최대 사이즈를 초과할 수는 없다. 헤더의 이름이나 값을 전송할 때, 발신자는 ASCII 인코딩 또는 허프만 압축을 통해 보낼건지, 다이나믹 테이블 안의 값의 위치를 표시할건지 선택한다. 발신자는 다이나믹 테이블의 크기를 컨트롤한다.

Server Push: HTTP/2 의 서버 Push 메커니즘은 클라이언트가 리소스들을 필요로 한다고 확신할 때, 서버가 클라이언트의 요청에 대한 대기 없이 사전에 리소스들을 미리 보낼 수 있도록 허용한다. 이는 Round-trip 시간과 페이지 로딩 시간을 줄여 사용자 경험 향상에 도움을 준다. 그림 1 은 HTTP/1.1 과 HTTP/2 의 Push 차이점을 보여준다.

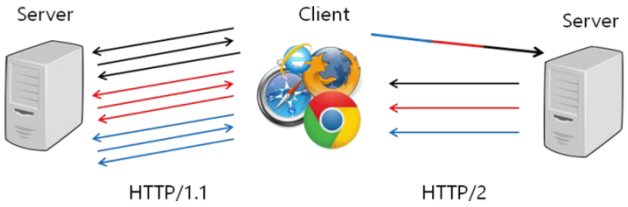

그림 1. HTTP/1.1 과 HTTP/2 의 Push

3. HTTP/2 의 취약점

ISS, APACHE, NGINX 가 포함된 주요 어플리케이션 서버들을 가진 브라우저들은 HTTP/2 를 지원한다. W3Techs 에 따르면 모든 웹사이트의 6.5%, 트위터, 페이스북, 구글과 같은 초대형 웹사이트를 포함한 상위 1000 위의 웹사이트 13.5%가 HTTP/2 를 사용 중이다. HTTP/2 를 사용함으로써 사용자 경험이 향상될 수 있는 반면에 서버와 클라이언트가 새로운 취약점에 노출될 가능성을 가진다 [2].

HTTP/2 의 기본적인 프로토콜의 보안 방안을 수립하였으나, 여전히 HTTP/2 의 보안 취약점들이 된다. 예를 들면, 흐름 제어 메커니즘에서 사용되는 파라미터 중 하나인 WINDOW_UPDATE 를 DoS 공격을 위해 오용하는 경우가 있다. 또한 HPACK 에서 매우 큰 헤더 블록을 처리하는 과정에서 서버에 큰 메모리 소비를 유발할 수도 있어, 이를 이용한 공격이 가능하다는 점이다. 따라서 HTTP/1.x 에서 HTTP/2 로 전환하기 위해 기존의 웹 소스 코드 변경 및 재작성은 공격자들에게 또 알려진 취약점을 통한 보안 공격 기회를 제공할 수 있다.

본 논문에서 분석한 HTTP/2 의 취약점은 네 가지가 있으며 이를 이용한 공격 형태는 표 1 과 같다.

<표 1> HTTP/2 에서 보안 취약점

HTTP/2 취약점	공격 유형	피해
Slow Read attack	악성 클라이언트를 통한 Slowloris DDos 같은 형태	웹 성능 저하 및 서비스 마비
HPACK Bomb attack	스미싱, 트로이목마 등을 통한 악성파일 감염	백신프로그램 무력화 메모리 리소스 점유
Dependency Cycle attack	흐름 제어의 의존성에 대해 무한 루프 적용	지속적으로 소모되는 리소스와 접근시 치명적인 오류를 유발
Stream Multiplexing Abuse	다중 스트림 기능을 통한 공격	서버 조작 및 context 밖의 프레임 제어

Slow Read attack : 악성 클라이언트에 의해 행해지며, 읽기 응답이 매우 느려진다. 201 년 미 금융협회를 강타했던 Slowloris DDoS 공격과 매우 유사하며, 이 공격은 최신 웹 프로토콜에서도 유효하다 [7]. 최근 이 공격 방식은 웹 어플리케이션 계층에서 이루어지고 있으며 Apache, IIS, Jetty 를 포함한 많은 서버사이드에서 가능하다. HTTP/2 의 윈도우 메커니즘은 과거 Zero-Window 공격과 Slow read 공격의 목표가 되었던 TCP 의 윈도우 메커니즘과 상당히 유사한데, 이는 큰 리소스를 요청하는 동안 유입될 윈도우 사이즈를 줄이거나 재설정하고, 악성 클라이언트가 매우 긴 시간 또는 무한정 연결을 유지하고 서버의 리소스를 소비할 때 나타난다.

HPACK Bomb attack : 이 공격은 과거의 'Zip bomb'과 유사한 형태이다. 악성 압축파일이 프로그램이나 시스템을 파괴하고 백신 소프트웨어를 무력하게 만들도록 고안되었다 [8]. 대체로 크기가 작으며 보안 위협이 없어 보이는 메시지가 서버 안에서 폭발하여 모든 서버의 메모리 리소스를 점유하고 오프라인상태로 만들어버린다.

Dependency Cycle attack : HTTP/2 는 네트워크를 최적화하기 위해 새로운 흐름 제어 메커니즘을 선보였는데, 공격자가 이 우선순위와 의존성을 이용하여 무한 루프 형태의 dependency cycle 을 만들 수 있다. 이 경우에 흐름 제어 메커니즘이 공격자가 만든 Cycle 을 처리하려고 하면 문제가 발생한다.

Stream Multiplexing Abuse : 이 공격은 가장 주요하고 위험한 공격 방법이다. 공격자가 다중 스트림 기능의 원리를 이용할 때 발생한다. 이 공격은 서버를 망가뜨릴 수도 있으며, 정당한 사용자의 접근을 거부하는 결과를 낳는다. 이 메커니즘의 위험은 접속 파티션이 완전히 논리적이며, 서버를 조작하거나 Context 밖의 프레임을 전송할 수 있다는 사실에 기인한다. HTTP/2 의 스트림은 하나의 요청-응답 사이클만을 사용하며, 각 스트림 식별자는 같은 연결을 사용할 수 없어야 하지만 공격자는 같은 스트림에 두개의 요청을 보내 서버로 하여금 심각한 오류를 발생하게 만든다.

4. 결론

HTTP/2 는 기존의 프로토콜의 단점을 보완하기 위해 새롭고 효율적인 메커니즘으로 무장하여 나타났다. 복합된 여러 기술로 향상된 UX 를 보여주고 있으며 개선된 웹 환경을 만들어 주었고 이 프로토콜의 점유율은 날로 증가하는 추세이다. 하지만 기존의 많은 사례들과 같이 새로운 기술 요소에는 새로운 보안 위협이 존재하며 피해 규모 또한 더욱 크게 예측된다.

따라서 새롭게 선보이는 HTTP/2 의 여러 요소들은 공격자들에게 오히려 새로운 공격 가능성을 열어준다고 볼 수 있다. 이런 이유로, 보안에 대한 대책 마련 없이 HTTP/2 의 사용을 무작정 확장해서는 안될 것으로 생각되며, 본 논문에서 기술한 취약점들과 또 다른 보안 위협 및 대책 마련을 통해 보안 피해를 줄일 수 있는 연구가 앞으로 이루어져야 한다.

Acknowledgement

이 논문은 2016 년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임
(No. 2016R1A2B4011069).

참고문헌

[1] M. Belshe, R. Peon, "Hypertext Transfer Protocol Version 2 (HTTP/2)", Internet Engineering Task Force (IETF), 2015
[2] Imperva, "HTTP/2: In-depth analysis of the top four flaws of the next generation web protocol", 2016
[3] R. Peon, H. Ruellan, "HPACK: Header Compression for HTTP/2", Internet Engineering Task Force (IETF), RFC: 7541, 2015
[4] Duc V. Nguyen, Hung T. Le, "Request adaptation for adaptive streaming over HTTP/2", Consumer Electronics (ICCE), 2016
[5] Erwin Adi, Zubair Baig, "Low-Rate Denial-of-Service Attacks against HTTP/2 Services", IT Convergence and Security (ICITCS), 2015
[6] Erwin Adi, Zubair A. Baig, "Distributed denial-of-service attacks against HTTP/2 services", Cluster Computing, 2016
[7] S McGregory, "Preparing for the next DDoS attack", Network Security, 2013
[8] Chithra Selvaraj, "A survey on Security Issues of Reputation Management Systems for Peer-to-Peer Networks", Computer Science Review, 2012