|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | |  | | KITRI 모의해킹 28기 | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | 작성:문관필 | | |  | |
|  | <Security Misconfiguration> | | | | | | |  |
|  | | Cross-Origin Resource Sharing(AJAX) | | | |  | | |
|  | |  | | | |  | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |
|  | | |  | |  | | | |

|  |
| --- |
| 1. 개요(공격개념/원리/취약점/공격 시 예상피해 등) |
|  |
| Security Misconfiguration   * OWASP TOP 10 A5(잘못된 보안 구성) * 서버 / 시스템 / DB / 네트워크 장비 / 웹 설정 오류로 인하여 발생하는 취약점   EX) Directory listing, Error Page, Web Page 주석, Web Server 기본 설정, Adobe Flash 취약점  Dos / DDos 공격, 특정 서비스에 대한 Reverse\_TCP 공격, 로컬 권한 취약점, 백업 / 임시 / Robot 파일     * 설정 오류 / 미비 항목들 중 대표적인 예   EX) 클라우드 서비스 접근 권한 설정이 올바르지 않음  필요하지 않거나 감춰야 할 서비스 / 기능을 설치했거나 활성화함(서버 포트 외 다른 포트 개방, 불필요한 슈퍼계정 / 권한, 불필요한 HTTP Method, Debug 모드 등등)  애플리케이션 서버(Tomcat, Apache) / 프레임워크(Spring, ASP.Net)의 보안 설정을 세팅하지 않음  Application Stack 내에서 오래되거나 취약점이 있는 버전을 사용함.  유저들에게 서버의 디렉토리 구조를 모두 혹은 일부 노출시킴.  웹 서버의 프로그램 혹은 버전을 노출시킴  기타 등등..  AJAX(Asynchronous Javascript And XML)   * 다른 도메인의 서버 URL을 호출하여 데이터를 가져올 수 있는 기능을 수행한다. * 사용자와 웹페이지와 자유롭게 상호작용 할 수 있는 기능을 수행한다. * 별도의 프로그램 없이 웹페이지를 다시 로딩할 필요 없이 내용, 메뉴 등을 제공한다. * 서버 처리를 대기하지 않고, 비동기 요청이 가능하다.   SOP(Same-Origin Policy)   * 동일 출처 정책이라고 하며, 이는 자바스크립트(XMLHTTPRequest)로 다른 웹페이지에 접근 시 동일 출처(프로토콜, 호스트 이름, 포트)의 웹페이지에만 접근이 가능한 기능이다. * 이를 통해서 동일 출처 관계에서만 AJAX 요청을 실시하게 된다. * 단, 외부 API 호출이 많이 사용되는 웹 같은 경우에는 SOP 정책으로 인해서 문제가 발생하기 때문에 CORS 정책을 사용하여 다른 출처 관계에서도 호출이 가능하도록 해야 한다.   CORS(Cross-Origin Resource Sharing)   * 웹 페이지의 제한된 자원을 외부 도메인에서 요청을 허용해주는 기능을 수행한다. * 즉, 브라우저와 서버 간에 AJAX 요청/응답 시 Cross-Origin 요청을 허용/거부 할 것인지를 결정하는 동작을 수행한다.   Access-Control-Allow-Origin   * Access-Control-Allow-Origin: \* <- 모든 도메인에 대해서 AJAX 요청에 대한 CORS를 허용한다. * Access-Control-Allow-Origin: 도메인 <- 특정 도메인에 대해서만 AJAX 요청에 대한 CORS를 허용한다. |
|  |

|  |
| --- |
|  |
| 2. 공격과정 (필요시 가정이나 예상 시나리오 포함) |
| |  |  |  | | --- | --- | --- | | 실습 환경 | IP Address | OS | | 공격자 | 192.168.1.102/24 | Kali Linux | | 희생자 | 192.168.1.205/24 | Ubuntu(Bee-box) |  1. 환경 구성   **WEB SERVER**  **HACKER**  192.168.1.102 192.168.1.205 |
| 1. 공격  * 보안 레벨 선택 및 시나리오 선택      * 악의적인 사이트의 AJAX 요청을 사용하여 Neo의 비밀을 획득하라는 내용 확인 * Secret 클릭      * Neo의 비밀 확인 |
| * Burp Suite에 Intercept -> Intercept is on 으로 변경      * 다시 secret 클릭 * Burp Suite에 Intercept 내용 확인 -> Forward 버튼 클릭      * Burp Suite에 Intercept 응답 내용 확인 * Access-Control-Allow-Origin: \* 확인      * Intercept -> Intercept is off로 변경 * Neo의 비밀 확인      * sm\_cors.php 파일 내용 확인     -중간 생략-  switch($\_COOKIE["security\_level"])  {  case "0" :  $hero = "Neo";  $file = "secret-cors-1.php";  break;  case "1" :  $hero = "Wolverine";  $file = "secret-cors-2.php";  $hint = "HINT: the secret is only available to requests from intranet.itsecgames.com...";  break;  case "2" :  $hero = "Johnny";  $file = "secret-cors-3.php";  $hint = "HINT: never trust local intranet zones!";  break;  default :  $hero = "Neo";  $file = "secret-cors-1.php";  break;  }   * secret-cors-1.php 파일 확인      * secret-cors-2.php 파일 확인      * secret-cors-3.php 파일 확인 |

|  |
| --- |
|  |
| 3. 공격 결과 |
|  |
| 어떤 URL에서든 이 페이지에 요청을 보내면 접근을 허용한다는 것을 알 수 있다.  그러나 난이도를 중으로 하고 시도를 하면 비밀을 알 수가 없다.  왜냐하면 위에 secret-cors-2.php 파일을 보면 알 수 있듯이 특정 사이트에서 오는 요청만 허용하기 때문에 차단당한다. |
|  |
|  |
|  |

|  |
| --- |
|  |
| 4. 보안진단 및 대책 |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
|  |
| 5. 보안대책 적용시 공격결과 (최종 매뉴얼에 포함) |
|  |