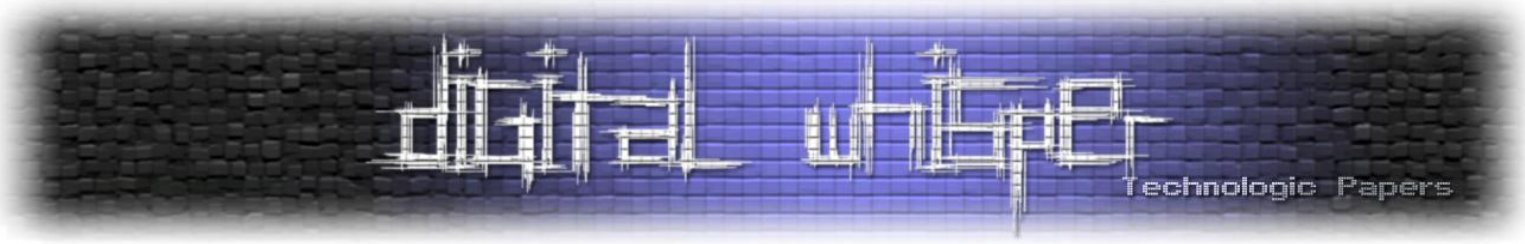

HTAccess

מאת אפיק קסטיאל (cp77fk4r)

רכיב ה-HTAccess הוא אחד ממרכיבי הקונפיגורציה הבסיסית של שרתי ה-Apache. קובץ זה אחראי על הקונפיגורציה המקומית של התיקיה אליה אנחנו ניגשים: הוא קובע מי יוכל לגשת לאיזה תיקיה, איך היא תתנהג: איך היא תציג לנו את הקבצים, איזה קבצים יהיו נגישים ואילו ידרשו סיסמא לפני הכניסה אליהם, התייחסות שונה לפרמטרים ב- URL, קביעת דפי שגיאה (403\404 וכו') מוגדרים מראש, אילו מתודות יפעלו על תוכן התיקיה וכו'. במאמר זה נכיר את ה-HTAccess ואפשרויות שונות שבו.

באופן מעשי, HTAccess הוא קובץ טקסט המכיל מספר שורות הקובעות למערכת ההרשאות איך להתנהג במקרה ספציפי.

מומלץ לנסות את זה בבית!

כדי להבין טוב את ההסברים חיזרו על ה דוגמאות השונות בעצמכם. לצורך כך, יש צורך בשרת Apache מותקן על המחשב. מערכת נוחה ומומלצת לניהול ותפעול שרת Apache היא XAMPP, הזמינה בכתובת: http://www.apachefriends.org/en/xampp-linux.html

אחרי התקנתה, תמצאו בתיקיה XAMPP תת-תיקיה בשם htdocs. תיקיה זו היא תיקית-השורש שלכם ("wwwroot"). כל הקבצים שתשימו בתיקיה זו יהיו נגישים דרך האינטרנט.

תפעילו את השרת ותכנסו בדפדפן לכתובת: http://localhost
אתם אמורים לקבל את דף הבית של השרת שלכם שאומר שהשרת הותקן בהצלחה.

נקודות חשובות לגבי קבצי HTAccess:

- שם הקובץ הוא: htaccess. (נקודה ואז "htaccess").
- קובץ זה הוא קובץ טקסט. (משמע- במידה ויאוכסן בו מידע רגיש, בעת פריצה לשרת המידע בו יהיה זמין לתוקף)
- כאשר ישנה התנגשות בין שני הגדרות של קבצי htaccess קובץ ה-htaceess העליון יותר יקבע (הקובץ שנמצא בספריה הקרובה יותר לשורש).

Directory Listing

כאשר נכנסנו ל-http://localhost/ קפץ לנו דף ברירת מחדל שהותקן ביחד עם השרת . נכנס ל-htdocs ונמחק אותו , ונכנס שוב לכתוב http://localhost/ דרך הדפדפן . אנו נראה את כל הקבצים בתיקיה הראשית שלנו - מה שנקרא "Directory Listing" או "Directory Browsing". מצב זה לא תקין – אפשרות לראות את רשימת הקבצים נותנת לפורצים לאסוף מידע יקר על תצורת הקבצים והשרת, להוריד לסייר ולראות את כל הקבצים הנגישים בשרת וכו'.

מה בעצם קורה כאן? כאשר נכנסים לספריה בשרת, השרת בודק אם קיים קובץ באחד מהשמות הבאים index.html, index.php או קבצים דומים (הרשימה המדוייקת משתנה בין שרת לשרת). אם קובץ כזה קיים, הוא מוצג באופן אוטומטי , אך במידה ולא מוגדר לשרת שום קובץ לטעינה- או שמוגדר אך הקובץ לא קיים, התוצאה תיהיה מה שראינו.

רשימת הקבצים שאותם השרת ינסה לטעון באופן אוטומטי כאשר המשתמש ינסה לגשת לתיקיה נמצאת בקובץ httpd.conf באיזור הבא: IfModule dir_module. כברירת מחדל, האיזור נראה כך:

```
<IfModule dir_module>
    DirectoryIndex index.php index.php4 index.php3 index.cgi index.pl
index.html index.htm index.shtml index.phtml
</IfModule>
```

הסדר בו מופיעים הקבצים ברשימה זהו הסדר בו השרת מחפש אותם. (כלומר, אם קיים index.php וגם index.html, השרת יציג את index.php מכיוון שהוא מופיע לפניו ברשימה זו).

כאמור, אנחנו מאוד לא מעוניינים שהשרת יציג את תוכנן של תיקיות האתר שלנו, ולכן אנחנו ניצור קובץ htaccess. בתיקית ה-root שלנו ונכתוב בו את הפקודה הבאה:

```
Options -Indexes
```

פקודה זו תגרום לשרת , בכל פעם שיבקשו ממנו להציג תוכן של תיקיה , להציג עמוד שגיאה "403" – "Access forbidden". אם נרצה שהשרת כן יציג תוכן של תיקיה מסויימת, ניצור בתוך התיקיה הספצית קובץ htaccess. ובתוכו נכתוב:

```
Options +Indexes
```

כל עוד לא נמקם באותה תיקיה שום קובץ index, כאשר משתמש יבקש להציג את תוכנה של התיקיה השרת אכן יציג לו אותה.

ישנה גם אפשרות להגיד לשרת להציג את תוכנה של תיקיה מסויימת חוץ מסוג מסויים או מקבץ של קבצים ספציפים, לדוגמא, אם נכתוב בקובץ ה-htaccess. שלנו את הפקודה:

```
IndexIgnore *.php *.conf
```

ומשתמש יבקש להציג את תכולת התיקיה- השרת יציג לו את תכולת התיקיה ללא הקבצים שהגדרנו.

אם נרצה שהשרת יציג קובץ מסויים כאשר המשתמש יבקש להציג את תוכנה של התיקיה (למשל קובץ המבצע- Directory List שאנחנו יצרנו), נוכל לכתוב:

```
DirectoryIndex FILE
```

בכל פעם שמשתמש יבקש להציג את תכולתה של התיקיה- השרת יטען לו את הקובץ הנ"ל.

Password Protected Folder

בעזרת htaccess. אנחנו יכולים לאפשר כניסה לתיקיה מסויימת רק לאחר ביצוע הזדהות. זהו כלי מאוד מומלץ ונוח לחסימת תיקית ממשק ניהול האתר וחלקים אחרים שלא אמורים להיות ציבוריים. בכדי לבצע את החסימה הזאת אנחנו צריכים להשתמש בשני קבצים:

1. קובץ ה-htaccess. - יגדיר את מאפייני התיקיה
2. קובץ htpasswd. - יאכסן את פרטי המשתמשים , שמותיהם וסיסמאותיהם (מוצפנות) ויעבוד תחת ה-htaccess.

הגדרת ה-htaccess.:

ניצור קובץ htaccess. בתיקיה אותה נרצה לחסום, ונכתוב בו:

```
AuthUserFile [Location/.htpasswd]
AuthName [BANNER]
AuthGroupFile /dev/null
AuthType Basic
require user [USERNAME]
```

הסבר:

- **השורה הראשונה** מצביעה על מיקום קובץ ה-htpasswd., כאשר זה מתאפשר- עדיף מאוד לאכסן אותו מחוץ לתיקית השרת (מתחת ל-wwwroot).
- **השורה השניה** מאכסנת את הבאנר שיוצג, מה שיהיה כתוב בחלון ההזדהות.

- **השורה השלישית** מגדירה את קבוצת ההזדהות (הקובץ יכיל את שם הקבוצה, ":" (נקודותיים) ואת שמות המשתמשים השייכים לאותה הקבוצה מורשת הגישה)
- **השורה הרביעית** מגדירה את סוג ההזדהות (מדוברת בהזדהויות מבוססות HTTP, כמו למשל גם Digest)
- **שורה חמישית** קובעת איזה משתמש מה-htpasswd. יהווה אימות לסיסמא, בקובץ ה-htpasswd. אפשר להגדיר מספר משתמשים, והשורה החמישית תבחר משתמש ספציפי מתוך כלל המשתמשים (בכדי להגדיר יותר ממשתמש אחד פ שוט שכפלו את השורה עם שינוי שמו של המשתמש).

הגדרת ה-htpasswd.:

צרו קובץ בשם htpasswd. מחוץ לספריות שרת (במידה והדבר אפשרי) וכיתבו בו את שמות המשתמשים שאתם רוצים לאפשר להם להתחבר לתיקיה, באופן הבא:

```
[User]:[Crypted-Password]
```

מצד שמאל של הנקודותיים נכתוב את שם המשתמש, ומצד ימין שלהם את הסיסמא המגובבת ע"י פונקציית Crypt (כך שה-Salt הוא שתי התווים ראשונים של הסיסמא). זיכרו לשמור את הקובץ במיקום שקבעתם קודם לכם ב-AuthUserFile.

אם ביצעתם הכל כשורה, כאשר תנסו לגשת לכל קובץ הקיים בתיקיה שבה הכנסתם את קובץ ה-htaccess. תתבקשו להקיש סיסמא.

Custom Errors Pages

איך דפי שגיאה יכולים לגרום לכשלי אבטחה? א'- כמעט כל דפי השגיאה כיום שמציגים את כתובת העמוד הלא קיים שאותו אתה מבקש להציג חשופים למתקפת XSS ואם לא ל-XSS אז למתקפת UTF7-XSS (פשוט מאוד ע"י הכנסת וקטור התקיפה בתוך ה-URL). ב'- דפי השגיאה האלה עוזרים לכל סורקי החשיפות (כגון Acunetix, AppScan, SSS וכו') מבוססי התבניות לדעת איפה ממוקמות התיקיות הרגישות שלנו. בעזרת ה-htaccess. נוכל לגרום לשרת לפלוט דף שגיאה 404 (Page Not Found) גם לתיקיות שהיו מחזירות לסורק/תוקף שגיאת 401 (Unauthorized) או 403 (Forbidden) וכך למנוע מהם למפות את התיקיות הרגישות שלנו (תיקיות הכוללות קבצים פרטיים, ממשקי ניהול וכו').

הרעיון הוא לגרום לשרת להתנהג בצורה זהה במצב של 404 ובמצב של 403 וכו', וכך תוקף לא יוכל לדעת מתי העמוד לא קיים או מתי העמוד קיים ואין לו הרשאות גישה אליו.

בכדי לבצע זאת, צרו קובץ htaccess. בתיקיות ה-root של השרת, וכתבו בו:

```
ErrorDocument 404 /404.html
ErrorDocument 403 /404.html
```

צרו קובץ שגיאה בשם 404.html על תיקית ה-root של השרת ומעכשיו, כל מי שינסה לבדוק האם אכן קיימת התיקיה הפרטית שלנו (והיא אכן קיימת), הוא יופנה ישירות לעמוד שגיאה והשרת ישמח לבשר לו שהתיקיה אינה קיימת כלל.

URL Filtering

קבצי ה-htaccess. יכולים לעזור לנו בעוד נקודה חשובה לא פחות - סינון תווים ב-URL. כיום כמעט כל המתקפות כנגד Web Application כגון: Sql-Injection, XSS, R/LFI, Path Traversal, וכו' מבוצעות על-גבי ה-URL, כאשר מדובר, כמובן, על קלט מבוסס GET. בעזרת סינון התווים של קובץ ה-htaccess. נוכל להוסיף עוד שכבה באבטחה על אפליקציות אלו. ישנן שתי דרכים לבצע בדיקה האם קלט תקין:

- **Black-List:** קבלת כל קלט שהוא מלבד תווים ספציפים ("מזיקים").
- **White-List:** אי-קבלת של שום קלט מלבד תווים ספציפים ("לא-מזיקים").

מבחינת אבטחת-מידע האפשרות השניה הרבה יותר עדיפה, למה? כי יש כל כך הרבה קומבינציות להכניס תו אחד, אם חסמנו את התו ">" תוקף יוכל להכניס אותו בעזרת הקידוד ההקס דצימלי שלו. חסמנו קידוד זה? תוקף יוכל להכניס את קידוד היוניקוד שלו. חסמנו את היוניקוד? יהיה אפשר להכניס אותו בעזרת ה-UTF-7 שלו, וכך עד (כמעט) אינסוף. לכן עדיף לנו להמנע מכל הסיפור ולאפשר להכניס רק את התווים שאנחנו יודעים שאנחנו נשתמש בהם (אותיות גדולות, קטנות, מספרים ושאר תווים "לא מזיקים") ואת שאר התווים נזרוק.

יש מספר נקודות שאנחנו חייבים לדעת, לדוגמא- בכדי לבצע Path Traversal, תוקף יהיה חייב להכניס איזה וריאציה של "..\" אז מן הסתם כדאי שנחסום את התווים האלה (נקודה וסלאש) אבל אנחנו לא יכולים לחסום אותם מפני שהם קיימים גם בכתובת URL תיקנית! לכן נהיה חייבים לאפשר את "." ואת "\", אבל נוכל לחסום את התבנית "..\" (לצורותיה). לעומת זאת, אף פעם אין שימוש חוקי בתו "<" או בתו ">" ולכן נוכל לקבוע בוודאות שכאשר מישהו מנסה לגשת לכתובת על השרת שלנו עם אחד מהתווים האלה- מדובר פה במשהו מסריח – ולכן נעדיף לזרוק אותו.

חבילת ה-Xampp לא מאפשרת כברירת מחדל להשתמש במודול שמבצע מניפולציות ב-URL (שמו- "mod_rewrite") וכדי לאפשר את זה אנחנו צריכים שוב לגשת לקובץ ההגדרות (httpd.conf), ולהוריד את הסולמית (#) המופיעה לפני השורה:

```
#LoadModule rewrite_module modules/mod_rewrite.so
```

אחרי שתבצעו Refresh לשרת אפשר להמשיך.

איך אנחנו קובעים לשרת להתעלם מכל URL שכולל בתוכו את אחד משני התווים "<" ו- ">" (או את שניהם ביחד)? צרו קובץ htaccess. וכיתבו בו:

```
RewriteEngine on
RewriteCond %{QUERY_STRING} (\<|%3c|%3e|\>) [NC]
RewriteRule ^.*$ - [F]
```

- **השורה הראשונה** מדליקה את האפשרות של Rewrite_mod.
- **השורה השניה** בודקת האם השאילתה (ה-GET) כוללת בתוכנה את אחד או כמה מהתוים המופיעים בתוך הסוגריים, שימו לב ל-[NC] – אומר לשרת "No Case Sensitive", ולכן גם "%3E" ו-"%3C" לא יתקבלו.
- **השורה השלישית**- "אזי"- אומרת מה יקרה אם ה-"RewriteCond" מתקיים- פה היא נגמרת ב-"[F]", מה שאומר "Forbidden" – תציג הודעת 403 שתגיד למשתמש שאין לו גישה לבצע את השאילתה הזאת. המחרוזת "^.*$" היא ביטוי רגולרי שמשמעו בעצם "כל שאילתה".

יש מתקפות XSS שאפשר לבצע גם בלי הסוגריים המשולשות, ולכן כדאי שנוריד גם את התווים הבאים: "(,),:,%,*,,\,/,+,-,;,',",," וכו'. לצורך כך נכניס אותם בתוך המקטעים של ה-OR ב-RewriteCond.

ביצענו סינון קלט מבוסס "Black-List". כאמור זו הדרך הפחות טובה לסינון קלט . כדי לבצע סינון קלט מבוסס "White-List" נוכל להשתמש ב-"NOT" – אופרטור סימן קריאה (!), בכדי לבדוק האם הקלט לא כולל רק אותיות ומספרים ותווים נחוצים, בואו נחשוב איזה תווים אנחנו כן צריכים להעביר דרך ה-GET:

- אותיות קטנות/גדולות? **כן.**
- מספרים? **כן.**
- נקודה? **כן.** (לסיומת של הקובץ)
- סלאש? **כן.** (חוצץ בין תיקיות).
- סולמית? **אפשר.** (מצבים בהם מצביעים על אמצע תוכן של טקסט)
- סימן שאלה? **חובה.** (משתנים)
- שווה (=)? **חובה.** (הצבת ערכים במשתנים)
- אמפרסנד (&)? **חובה.** (תמיכה במספר משתנים)
- עוד משהו? **לא.** פשוט נזהר לא לקרוא לתיקיות בשמות עם מקף (-) ואם בכל זאת כללנו תו יחודי, נוסיף אותו ל-white list.

החוק שלנו אמור לבדוק האם **השאילתה כוללת תווים אשר לא מופיעים ב-White-List** שלנו. ואם כן- נחזיר למשתמש Forbidden. נכתוב את זה כך:

```
RewriteEngine on
RewriteCond %{QUERY_STRING} [^a-zA-Z0-9&?/#=\.]
RewriteRule ^.*$ - [F]
```

אנחנו צריכים לזכור שבעזרת ".." אפשר לבצע במצבים מסויימים מספר מתקפות שאותן החוק שלנו מאפשר, ולכן נוסיף בדיקה האם יש לנו שתי נקודות אחת אחרי השניה, ככה נאפשר להשתמש בנקודה אחת, אבל לא נאפשר להשתמש בשתי נקודות, נשנה את השורה השניה לשורה הבאה:

```
RewriteCond %{QUERY_STRING} ([^a-zA-Z0-9&?/#=\.]|\.\.)
```

ניתן, כמובן, לשכלל רבות את סינון זה . הדוגמא שהוצגה היא דוגמה בסיסית בלבד לניפוי קלט בעזרת שימוש ב-White-List.

IP Filtering

אפשרות נוספת שמציע לנו המודול mod_rewrite הוא חסימת גישה למקומות ספציפיים ע"פ כתובת ה-IP של המשתמש. כך לדוגמא אפשר לאפשר רק לכתובת מסויימת להכנס לממשק האדמין, לתת "באנים" למשתמשים, למנוע מתקפות DoS/DDoS וכו'. צרו קובץ htaccess. בתיקיה אותה אתם מעוניינים לחסום (תיקית ממשק הניהול, תיקית הפורום, תיקית השורש וכו') וכיתבו בו כך:

```
RewriteEngine on
RewriteCond %{REMOTE_HOST} IP
RewriteRule .php$ [URL] [R=301,L]
```

- **השורה השניה** היא תנאי הבדיקה שלנו, שימו לב שאנחנו פועלים על: %{REMOTE_HOST} ואנחנו משווים אותו לפי לכתובת ה-IP שאותה אנחנו רוצים לחסום. שימו לב שאם למשל כתובת ה-IP שאותה אנחנו רוצים לחסום היא: 94.123.33.58
  אנחנו נכתוב אותה באופן הבא: 94\.123\.33\.58
  נכתוב כך כי מדובר בביטוי רגולרי, חשוב לזכור זאת כל הזמן.

- **השורה השלישית** היא הביצוע של התנאי. כתבנו שהתנאי יתבצע רק כאשר כתובת ה-IP שקבענו תנסה לגשת לעמוד PHP, נוכל גם לקבוע שאותה כתובת לא תוכל לגשת בכלל, ע"י שינוי השורה, לשורה הבאה:

```
RewriteRule .*$ [URL] [R=301,L]
```

דוגמא לשימוש:

```
RewriteRule .*$ http://disneyland.disney.go.com [R=301,L]
```

כך כל כתובת IP שתנסה לגשת לכל מידע שקיים באותה התיקיה שחסמנו תשלח ישירות לאתר הנחמד של דיסנילנד. הוספנו [R=301] כי מדובר בהפנייה (Redirect).

יש סוגי קונפיגורציה שבכדי להשתמש בהפניות בצורה הזאת בתוך הקובץ עצמו, תאלצו להוסיף בתחילת הקובץ את השורה:

```
Options FollowSymLinks
```

Disable HTTP Methods

בעזרת ה-htaccess. אפשר למנוע מהשרת להגיב לסוגים שונים של בקשות HTTP. לדוגמא, אם נרצה לחסום אפשרות להשתמש ב-HTTP OPTIONS Request (בקשת פירוט ה-Methods שהשרת תומך בהן) נוכל להשתמש בתג <Limit> או בתג <LimitExcept>. הראשון מיישם סינון מבוסס Black-List והשני מיישם סינון בעזרת White-list. לדוגמא, הקוד הבא:

```
<Limit OPTIONS>
deny from all
</Limit>
```

<u>ימנע רק</u> את האפשרות לשימוש ב-HTTP OPTIONS REQUEST באותה רמה שבה ממוקם הקובץ. לעומת זאת, הקוד הבא:

```
<LimitExcept POST GET>
Require valid-user
</LimitExcept>
```

<u>יאפשר רק</u> את האפשרויות לשימוש ב-HTTP OPTIONS REQUEST באותה רמה שבה ממוקם הקובץ.

<u>מספר דברים שחשוב לדעת:</u>

- שמות ה-Methods ב-htaccess. הינם Case sensitive (רגישות לאותיות גדולות וקטנות) ובשרת ה-Apache הם לא, מה שמחייב דרשני כאשר משתמשים בסינון מבוסס רשימה שחורה (Limit), כי במידה ונרצה לחסום למשל את OPTIONS נאלץ לחסום את כל הקומבינציות (מבחינת אותיות גדולות וקטנות) שאפשר להרכיב.
- בעזרת Limit ו-LimitExept לא ניתן לחסום את המתודה TRACE. בכדי לחסום את המתודה הנ"ל יש לעשות שימוש בתג TraceEnable באופן הבא:

```
TraceEnable off
```

- המתודה GET "כוללת בתוכה" את המתודה HEAD, במקומות שהמתודה GET תאופשר, גם המתודה HEAD תאופשר, במקומות שהמתודה GET תבוטל- גם כך המתודה HEAD.

סיכום

הכוח של htaccess. הוא רב מאוד ושימוש נכון בתכונותיו יכול להועיל רבות באבטחת המערכת שלכם. הצגתי בטקסט זה מספר דוגמאות לשימוש אך קיימות עוד דוגמאות רבות. ישנם עוד שימושים לקובץ הזה- למשל, מקדמי אתרים משתמשים בו בכדי לקצר את הכתובות לדפים שלהם ולעשות אותם נוחים ל-Crawlers של מנועי החיפוש (בעיקר של גוגל). אני מקווה שלמדתם דברים חדשים שישמשו אתכם.

נקודה חשובה מאוד שהייתי רוצה להזכיר לפני סוף המאמר היא ששימוש ב -htaccess. לא מחליף את האבטחה של המערכת שלכם. לדוגמא- נכון שבעזרתו אפשר לסנן קלט, אך הדבר לא אומר שכעת אין לממש מנגנון קלט במערכת עצמה. כמו שכתבתי בתחילת המאמר, הקובץ ישמש רק כרמה נוספת, ושימוש בו לא פותר אתכם מלפתח מערכות מאובטחות. פגשתי מספר לא קטן של מקרים שבהם אבטחה של מערכת מסויימת הסתמכה על הגדרות המעטפת של אותה מערכת ומפני ששינו או הגדירו מחדש את המעטפת ולא הקדישו לאבטחה מחשבה - המערכת נשארה פרוצה. מכיוון שכך, אדגיש שוב לסיום, **אין להסתמך על הנושא כפתרון אבטחה יחידי.**