חולשות ב-SSL מהפן האפליקטיבי

נכתב ע"י ישראל חורז'בסקי / Sro (ראש צוות בדיקות אבטחה ב-AppSec-Labs)

הקדמה

מאמר זה הינו מאמר המשך למאמר הקודם שעסק ב-SSL. בעוד שבמאמר הקודם הנושא היה תשתיתי ומצא שבצד ה-Server שרתי Web מאפשרים הצפנות חלשות, ובצד ה-Client דפדפנים מעודכנים מוודאים שה-Tunnel מוצפן כראוי. מאמר זה הינו אפליקטיבי ועוסק בשפות תכנות, בבדיקה האם בעת שאנחנו שולחים החוצה בקשת HTTPS ספריות הקוד השונות מוודאות שהתווך מוצפן כראוי (שזהו תפקידו של מנגנון ה-SSL), או שהן מאפשרות חיבור בפונקציות חלשות - כך שתוקף המבצע מתקפת MITM (באמצעות ARP Poisoning, DNS Spoofing וכד') יוכל לפענח את התעבורה.

שני דגשים שחשוב לתת כבר בשלב זה:

1. כדי שהלקוח (Client) ישתמש מול השרת (Server) בהצפנה חלשה, על השרת לתמוך בהצפנה זו.
2. המאמר חוקר את פרוטוקול HTTPS משום שהוא נפוץ יותר, אולם ניתן לחקור באותה צורה את פרוטוקולים מבוססי SSL נוספים, כגון FTPS.

סביבת העבודה

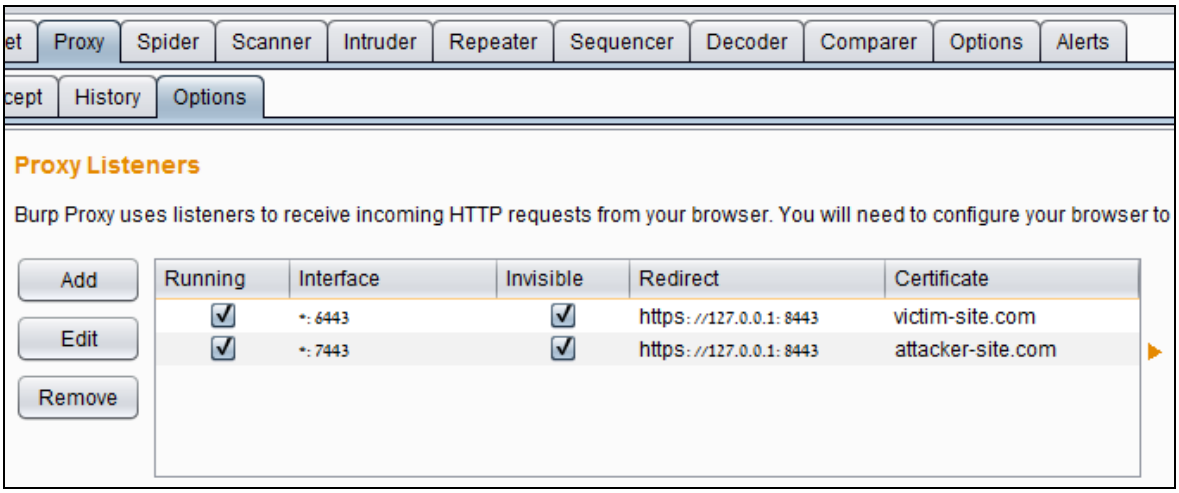
הערה: פרק זה של סביבת העבודה מפרט על האופן הטכני שבו בוצעו הבדיקות והוא עלול להיות קשה להבנה בקריאה "קלילה", תוכל/י לדלג על פרק זה ולעבור לפרק הבא "הרצה אחרונה לפני בדיקות".

כדי לבחון את התנהגות הקודים מול כל אלגוריתם הצפנה לחוד, הרמתי שרת Apache 2.4.2 (ספיציפית השתמשתי ב-WAMP על Win7), חוללתי מפתחות הצפנה אישי וציבורי תוך שימוש בדומיין victim-site.com (הדומיין שייך לחברת AppSec-Labs והוא משמש עבור הדגמות שונות, ביצעתי הפניה לוקאלית של הדומיין לכתובת 127.0.0.1), לאחר מכן קינפגתי את השרת שיאזין לפורטים שונים ושלכל פורט ישתמש באלגוריתם הצפנה מסויים.

כדי לחסוך ממך את המאמץ, תוכל/י להוריד את קבצי הקופיגורציה (כולל מפתחות ההצפנה) ואת הסקריפטים שיוצרים אותם מכווצים בקובץ בודד, לינק בסוף המאמר.

כיוון שהתעודה הינה מסוג Self signed, היא אינה מוכרת, ולכן הגדרתי בקטעי הקוד השונים שלא תתבצע בדיקות אימות האם התעודה חוקית.

כדי לבדוק האם שפות התכנות מבצעות בדיקת אימות, הן לתעודה - האם היא מוכרת, והן הצלבה של שם מתחם (דומיין) התעודה ושם המתחם המבוקש, יצרתי שני הפניות באמצעות הפרוקסי המוכר Burp.



בתצלום הקונפיגורציה הנ"ל ניתן לראות שהשרת (Apache) מאזין על פורט 8443, תעודתו כזכור אינה מוכרת עבור אימות. את התעודה של Burp כן התקנתי על המחשב (כ-CA, הוראות התקנה).

פורט 6443 מפנה ל-8443 עם שם המתחם victim-site.com כך שגלישה לכתובת:
https://victim-site.com:6443/

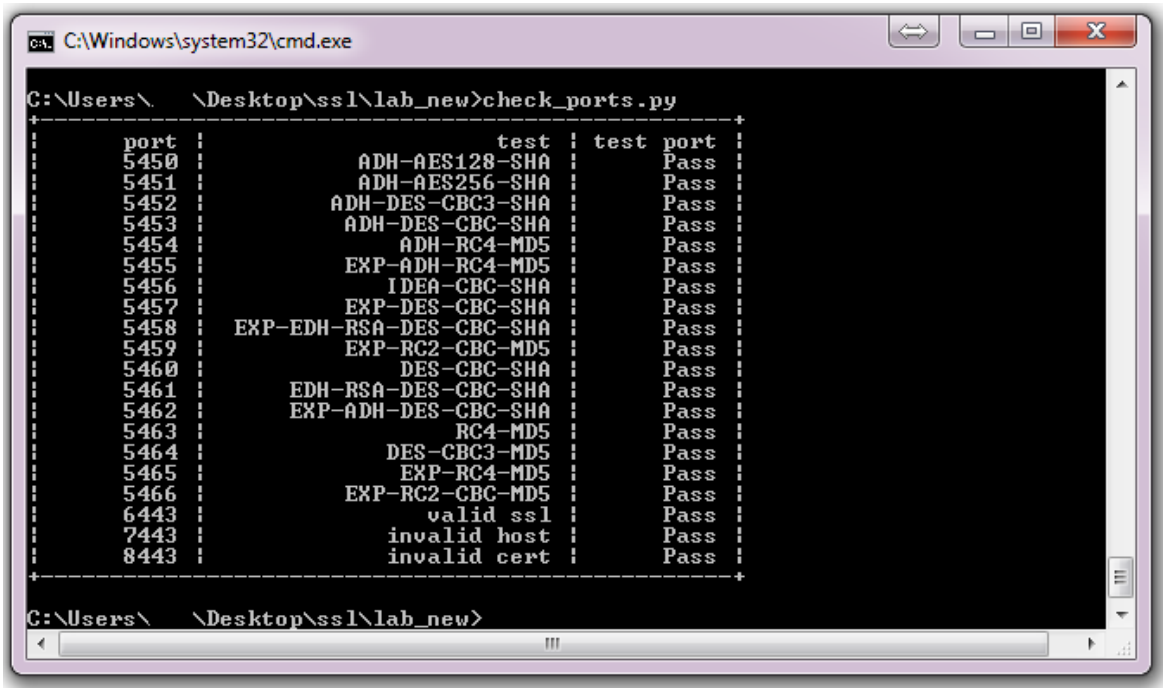
מהמחשב תקלט כתקנית ותאומת בהצלחה (שימו לב שמתבצעת גלישה ישירה לפורט של הפרוקסי ולא באמצעות הגדרה של Burp כפרוקסי. לכן בקונפיגורציה מסומן Invisible ומוגדרת כתובת Redirect).

גלישה לפורט 7443 אמורה לגרום לשגיאה של שם מתחם לא תואם (attacker-site.com גם הוא של AppSec שאישרו את השימוש של הדומיינים עבור המחקר) כיוון שהתעודה נחתמה עבור victim-site.com ולא עבור attacker-site.com, לעומת זאת התעודה עצמה כן מוכרת היות ו-Burp מוגדר כ-Trusted CA והוא מאשר אותה.

כך שהקוד שפונה ישירות לשרת מוגדר לא לבצע בדיקת SSL, והקוד שפונה לפורטים 6443, 7443, 8443 כן מוגדר לבצע אימות. אימות של התעודה לא נוגע (לפחות לא אמור) בשאלה האם לאשר אלגוריתם מסויים, כפי שנראה שהספריות השונות מאשרות / לא מאפשרות שימוש באלגוריתמים מסויימים. הארכתי כאן מעט, בשביל השלמת התיעוד.

הרצה אחרונה לפני בדיקות

כדי לוודא שהשרת מאזין על שלל הפורטים הרלוונטים וש-Burp מאזין גם הוא, נריץ את הסקריפט check_ports.py:



בתצלום ניתן לראות את רשימת הפורטים, סוג הבדיקה שהפורט מוודא, והטור האחרון מוודא שקיימת האזנה על הפורט.

בדיקת שפות סקריפט

כאמור, מעבר לבחינה הנפוצה של SSL כתשתית (נושא זה נחקר במאמר הקודם), כאן אנחנו חוקרים את SSL מהפן האפליקטיבי שבו בודקים האם ברמת הקוד שאנחנו כותבים והספריה שאנחנו משתמשים איתה, אנחנו מתירים לסקריפט להתחבר אך ורק עם הצפנות איכותיות, ושגם אם השרת מבחינתו מאשר (גם) הצפנה חלשה (התוקף שמבצע MITM, יגרום לנו לראות שהשרת מאזין רק להצפנה זו), אנחנו לא נתחבר אליו, כמו שלא נתחבר ב-http ללא הצפנה כלל.

אמרנו קוד - השלב הראשון שבדקתי היה שפות סקריפט, הן vbs הזכור ממאמר קודם, והן python, ruby וכו'. הבדיקה בוצעה על הספריות והפונקציות הנפוצות עבור קריאת דפי Web.

טבלת הבדיקות הראשונה הינה:

ספריה	שפה	ספריה	שפה
Nethttps	Ruby	MSXML2ServerXMLHTTP	VBS
Openuri	Ruby	xhttplib2	Python
		xurllib2	Python

כתבתי קובץ קוד קצר לכל אחד מהספריות הנ"ל שמקבל בפרמטר של Command Line כתובת (כגון: https://victim-site.com:5450) ומנסה לקרוא אותה, עם try ו-exception, אם הקריאה מצליחה מודפסים עשרת התווים הראשונים (שבמקרה זה מדובר ב-<?xml vers) ואם הוא נכשל הוא מדפיס ERROR.

הסקריפט (שהוא וכל יתר הקבצים הרלוונטים ניתנים להורדה, לינק בסוף המאמר) מציג טבלה של תוצאות הרצת קבצי הקוד על מגוון הצפנות חלשות ו/או תעודות לא מאומתות ו/או תעודות שחתומות עבור דומיין שונה:

```
C:\Users\mac\Desktop\ssl\lab_new>test_scripts.py > output2.txt

C:\Users\mac\Desktop\ssl\lab_new>
```

port	test	vbs.MSXML2	xhttplib2	xurllib2	nethttps	openuri
5450	ADH-AES128-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5451	ADH-AES256-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5452	ADH-DES-CBC3-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5453	ADH-DES-CBC-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5454	ADH-RC4-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
5455	EXP-ADH-RC4-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
5456	IDEA-CBC-SHA	ERROR	ERROR	ERROR	<?xml vers	<?xml vers
5457	EXP-DES-CBC-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5458	EXP-EDH-RSA-DES-CBC-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5459	EXP-RC2-CBC-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
5460	DES-CBC-SHA	ERROR	ERROR	ERROR	<?xml vers	<?xml vers
5461	EDH-RSA-DES-CBC-SHA	ERROR	ERROR	ERROR	<?xml vers	<?xml vers
5462	EXP-ADH-DES-CBC-SHA	ERROR	ERROR	ERROR	ERROR	ERROR
5463	RC4-MD5	<?xml vers	<?xml vers	<?xml vers	<?xml vers	<?xml vers
5464	DES-CBC3-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
5465	EXP-RC4-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
5466	EXP-RC2-CBC-MD5	ERROR	ERROR	ERROR	ERROR	ERROR
6443	valid ssl	<?xml vers	ERROR	<?xml vers	ERROR	ERROR
7443	invalid host	ERROR	ERROR	<?xml vers	ERROR	ERROR
8443	invalid cert	ERROR	ERROR	<?xml vers	ERROR	ERROR

ננתח את תוצאות הסריקה, ונתחיל מפורט 6443 (שורה שלישית מלמטה) - Valid SSL, אומר שהתעודה מאומתת באמצעות CA שמוגדר במחשב, כך שהיא תקינה לגמרי. שורה זו היא היחידה שהקודים אמורים לקרוא אותה. VBS קרא את הדף בהצלחה, xurllib2 קרא גם הוא. השאלה הנשאלת היא למה יתר הקודים נכשלו. התשובה היא שיש להם רשימת CA משלהם שמוגדרת ב-Open SSL, ואינו מושפע מה-Key store של מערכת ההפעלה.

למה זה חשוב? כי זה אומר שאם הקוד קורא מכתובת פנימית ברשת (כפי שפעמים רבות קורה), להתקין תעודה של השרת תהיה יותר ארוכה מהסטנדרט של דאבל-קליק, מה שאומר שסיכוי סביר שהמתכנת

יגדיר בקוד לא לבצע ווידוא לחוקיות התעודה. ולמעשה, אם נבדוק באתרים כמו StackOverFlaw נמצא שמרבית הפתרונות המוצעים הם לבטל את אימות התעודה, במקום להתקין את התעודה כראוי.

נמשיך עם היתר, Invalid host ו-Invalid Cert, ספריה שלא בודקת אותם, לא מדובר על הצפנה חלשה שיש צורך בכח עיבוד בשביל לבצע MITM ולפענח את התעבורה, מדובר ב-MITM בסיסי וקליל, כך שממצאים אלה הינם חמורים.

RC4-MD5, הצופן הזה לא עומד בתקנים מחמירים, אבל אישית קשה לי לחשוב איך בפועל ניתן לשבור צירוף שלהם, כך שזה לא טוב, אבל פחות חמור. יתר הממצאים בטבלה הם של CBC. שבירת צפני CBC (הידועה כ-BEAST Attack) לא נחקרה עדיין לעומק על מגוון האלגוריתמים. ראשית יש לבדוק אם הספריות הללו מאפשרות שליחת בקשות נוספות על אותו Connection, במידה ויימצא שכן, סביר שניתן במאמץ כזה או אחר לנצל אותן, וגם זה במידה והתוקף יכול לשלוט בבקשות שהשרת שולח.

אבטחת אתרי Web

הדבר הבא שעניין אותי הוא אתרי Web. הארכיטקטורה הנפוצה היא שיש דפדפן, הדפדפן פונה לאתר, והאתר פונה לשרת אחר (במקרה והוא צריך לשלוף ממנו מידע). מה שבדרך כלל ייבדק, זה הצפנת התעבורה בין הדפדפן לאתר. מעניין לבדוק את את איכות ההצפנה של החלק השני – בין האתר לשרת האחר. האם אפליקטיבית כשקוד באתר פונה לשרתים אחרים הוא מאובטח. בדקתי שתי פונקציות נפוצות של PHP: file_get_contents ו-curl, ואת הספריה HttpWebRequest של C#:

```
C:\Users\___\Desktop\ssl\lab_new>test_web.py > output3.txt

C:\Users\___\Desktop\ssl\lab_new>
```

output3.txt

port	test	file_get_contents	curl	HttpWebRequest
5450	ADH-AES128-SHA	ERROR	ERROR	ERROR
5451	ADH-AES256-SHA	ERROR	ERROR	ERROR
5452	ADH-DES-CBC3-SHA	ERROR	ERROR	ERROR
5453	ADH-DES-CBC-SHA	ERROR	ERROR	ERROR
5454	ADH-RC4-MD5	ERROR	ERROR	ERROR
5455	EXP-ADH-RC4-MD5	ERROR	ERROR	ERROR
5456	IDEA-CBC-SHA	<?xml vers	<?xml vers	ERROR
5457	EXP-DES-CBC-SHA	<?xml vers	<?xml vers	ERROR
5458	EXP-EDH-RSA-DES-CBC-SHA	<?xml vers	<?xml vers	ERROR
5459	EXP-RC2-CBC-MD5	<?xml vers	<?xml vers	ERROR
5460	DES-CBC-SHA	<?xml vers	<?xml vers	ERROR
5461	EDH-RSA-DES-CBC-SHA	<?xml vers	<?xml vers	ERROR
5462	EXP-ADH-DES-CBC-SHA	ERROR	ERROR	ERROR
5463	RC4-MD5	<?xml vers	<?xml vers	<?xml vers
5464	DES-CBC3-MD5	ERROR	ERROR	ERROR
5465	EXP-RC4-MD5	<?xml vers	<?xml vers	ERROR
5466	EXP-RC2-CBC-MD5	<?xml vers	<?xml vers	ERROR
6443	valid ssl	<?xml vers	ERROR	<?xml vers
7443	invalid host	<?xml vers	ERROR	ERROR
8443	invalid cert	<?xml vers	ERROR	ERROR

כפי שניתן לראות, ב-PHP הפונקציות file_get_contents ו-curl, פגיעות באופן חמור, למי שזוכר מהמאמר הקודם, האלגוריתמים שמתחילים ב-EXP הינם קצרים בעלי 40 Bits, היום המינימום שיכול להיחשב למאובטח הוא 128 ביט.

ב-C#, המצב יחסית תקין. יאמר לשבחה של Microsoft, שבנוגע ל-SSL היא יוצאת טוב לאורך כל הדרך.

תוכנות צד שלישי

הדבר האחרון שנבדוק הפעם הוא תוכנות צד שלישי. לצורך העניין בדקתי שתי תוכנות שמאפשרות להוריד דפי אינטרנט, url2file ו-wget. שניהם – קבצי exe של ווינדוס, אולם באותה מידה יש לבדוק את החבילות של לינוקס.

```
C:\Windows\system32\cmd.exe
C:\Users\    \Desktop\ssl\lab_new>test_apps.py > output4.txt
C:\Users\    \Desktop\ssl\lab_new>
```

port	test	url2file	wget
5450	ADH-AES128-SHA	ERROR	ERROR
5451	ADH-AES256-SHA	ERROR	ERROR
5452	ADH-DES-CBC3-SHA	ERROR	ERROR
5453	ADH-DES-CBC-SHA	ERROR	ERROR
5454	ADH-RC4-MD5	ERROR	ERROR
5455	EXP-ADH-RC4-MD5	ERROR	ERROR
5456	IDEA-CBC-SHA	ERROR	<?xml vers
5457	EXP-DES-CBC-SHA	ERROR	<?xml vers
5458	EXP-EDH-RSA-DES-CBC-SHA	ERROR	<?xml vers
5459	EXP-RC2-CBC-MD5	ERROR	<?xml vers
5460	DES-CBC-SHA	ERROR	<?xml vers
5461	EDH-RSA-DES-CBC-SHA	ERROR	<?xml vers
5462	EXP-ADH-DES-CBC-SHA	ERROR	ERROR
5463	RC4-MD5	ERROR	<?xml vers
5464	DES-CBC3-MD5	ERROR	ERROR
5465	EXP-RC4-MD5	ERROR	<?xml vers
5466	EXP-RC2-CBC-MD5	ERROR	<?xml vers
6443	valid ssl	<?xml vers	ERROR
7443	invalid host	<?xml vers	ERROR
8443	invalid cert	ERROR	ERROR

כפי שניתן לראות, url2file מכיל בעיה קריטית - הוא לא מוודא ששם הדומיין זהה לתעודה, כך שכל בעל תעודת SSL מוכרת (קרי: כל בעל אתר שיש לו תעודת SSL) יכול לחתום את הבקשות ו-url2file לא יזהה כלל בעיה כלשהי. יתר ההצפנות "לא נפרצו" אבל הסיבה היא שהתעודה של השרת לא מותקנת במחשב ו-url2file לא מאפשר לסמן לו לא לבדוק את התעודה. כך שבבדיקה מול שרת אמיתי שמאפשר אלגוריתמים חלשים, סביר שנמצא הצפנות שהוא כן יאשר.

Wget מאפשר גם הוא שימוש באלגוריתמים חלשים מאוד. שתי התוכנות לא יצאו טוב בקטע של SSL...

עדכון לגבי המאמר הקודם

למי שעוקב, יצא עדכון גרסה של SSL Vulnerabilities Analyzer, פרטים:

https://appsec-labs.com/SSL_Analyzer

SSL Vulnerabilities Analyzer מאפשר לסרוק אתר (כתובת IP / דומיין), כדי לדעת האם הוא מאפשר חיבורי SSL שמשתמשים באלגוריתמים חלשים. אם תמצאו אתרים רגישים במדינת ישראל כפגיעים, על כך ועל כל יתר המאמר אני מדגיש: הכלי והמאמרים מיועדים לידע תיאורתי, כל שימוש שהוא על אחריותך בלבד.

לינק להורדת קטעי הקוד והקופיגורציה

קובץ הזיפ מכיל בתוכו את כל קבצי הקוד והקונפיגורציה הסופיים ששומשו לצורך המחקר:
http://digitalwhisper.co.il/files/Zines/0x26/SSL_lab_files.zip

סיסמה לקוץ: digitalwhisper.

לסיכום

במאמר זה סקרתי בקצרה מגוון ספריות וקטעי קוד, תוך בחינת החיבור שלהם לשרתי Web. כפי שראינו, ישנן ספריות מאובטחות יותר וישנם כלים שה-SSL שלהם כמוהו כמעט כ-Clear text, וניתן לפענוח בקלות. יש עוד מגוון תחומים לחקור בנישה זו, הן את פרוטוקול FTPS ודומיו, הן ספריות נוספות, הן דרכי הקשחה רלוונטים ועוד. יש עוד כמה קווים מעניינים על אותה נישה שניתן לפתח, אבל את זה אשאיר לפעם הבאה...

אשמח לקבל פידבק (Israel@appsec-labs.com), רעיונות לכיווני מחקר יתקבלו בברכה, תודה מראש. אני שמח לתרום את המאמר לקהילה, ומקווה שהתרומה הישראלית לקהילה תתרחב.

ישראל חורז'בסקי [Sro.co.il]

ראש צוות Penetration Testing ב-AppSec.