אימות שאילתות DNS
מדריך DNSSEC

מאיר קראוסהר, dnssec@isoc.org.il

איגוד האינטרנט הישראלי ספטמבר 2024

תוכן עניינים

רקע

DNS - Domain Name System הינו מנגנון אשר פותח בראשית שנות ה- 80 במטרה לאפשר מיפוי שמות של משאבים ברשת מחשבים, לכתובות IP (או לכל ערך רצוי אחר).
DNS מורכב ממבנה נתונים ומפרוטוקול תקשורת המאפשר לתשאל את מבנה הנתונים.
מבנה הנתונים של DNS הוא היררכי ומבוזר. משמעות הדבר:

1. המידע שנמצא על שרתי DNS מפוזר בכל העולם, במיקומים רבים ושונים.
2. אין נקודה אחת שמכילה את כל המידע.
3. סידור היררכי במבנה של עץ, מאפשר חיפוש מהיר של המידע.
4. אין נקודת כשל יחידה (single point of failure).
5. אחידות של המידע. תשובה לשאילתה תהיה זהה ללא תלות במקום בו היא נשאלת

נהוג להמשיל מנגנון DNS לספר טלפונים: אם ידוע שמו של אדם או של עסק, ניתן בקלות למצוא את מספר הטלפון שלו לפי אינדקס שמי. באופן דומה, באמצעות DNS ניתן לקבל את כתובת ה- IP של משאב ברשת (אתר אינטרנט, שירות דוא"ל, מדפסת, ספריית קבצים וכו') לפי שמו.
נוכח בעיית רוחב הפס שאפיינה את רשתות המחשוב בשנות ה- 80, בעת אפיון הפרוטוקול הושם דגש על ביצועים ומהירות התגובה של המערכת, ופחות על אבטחה ופרטיות. תרמה גם העובדה שאיומי סייבר לא היו מוכרים אז כפי שהם היום.
ולכן, המחיר אותו אנו משלמים בעבור הביצועים המהירים של פרוטוקול DNS הוא היעדר פרטיות ואי יכולת לוודא את מהימנות התשובה המתקבלת בשאילתת DNS.
במהלך העשור הראשון של שנות ה- 2000 הוחל בתיקנון מנגנון אבטחת DNS ע"י ה- IETF.
המטרה היתה מצד אחד לאפשר אימות של שאילתות DNS ומצד שני להרחיב את פרוטוקול DNS, מבלי להמציאו מחדש. הפעילות שהוביל ה- IETF הביאה בסופו של דבר לתקינה של ההרחבה הקרויה "DNS Security Extensions", או בקיצור **"DNSSEC"**.
ה- RFC המעודכן של DNSSEC, מס' 9364, נמצא כאן:
https://datatracker.ietf.org/doc/html/rfc9364

מדריך זה הינו ניסיון ראשון לרכז בעברית את המשמעויות של הפעלת DNSSEC על מערכות DNS. המדריך מיועד הן למנהלים והן לאנשי IT אשר מעוניינים לאבטח את מערכות ה- DNS שלהם כנגד שימוש זדוני והונאת משתמשים.

Domain Name System Security Extensions - DNSSEC

DNSSEC הינו מנגנון אימות של המידע הניתן ע"י DNS. זהו אינו מנגנון פרטיות (הצפנה).
DNSSEC מאפשר למשתמשים לוודא כי המידע אותו קיבלו בשאילתת DNS הוא:

1. **מהימן** - מי שמציג את המידע הוא אכן הבעלים החוקיים שלו
2. **שלם** - המידע לא השתנה לאחר שהתפרסם ע"י בעליו החוקיים

שירותי DNS נתונים באופן קבוע לניסיונות מניפולציה של המידע אותו הם מחזיקים: הזנה של מידע מזויף, הרעלת cache, ועוד, במטרה להעביר למשתמשי הקצה מידע שקרי. לעיתים המוטיבציה של התוקפים היא שיבוש של השירות או השבתה שלו, אולם במקרים רבים המטרה היא להתחזות, כך שאפשר יהיה להפנות גולשים תמימים לאתרים זדוניים ובאמצעותם לגנוב מידע, סיסמאות, כסף וכו'. הפעילות הזדונית מסתמכת על העובדה שמרבית המשתמשים סומכים באופן לא מודע, על כך שהעברת הנתונים באינטרנט היא אמינה, וכי הנתונים אותם הם מקבלים בזמן הגלישה מהימנים. באופן הגיוני תורמת לכך העובדה שהגלישה כיום מתבצעת באופן כמעט מוחלט ב- TLS/SSL. יחד עם זאת, מערכת שמות המתחם (DNS) אינה חלק מתהליך בנית האמון בגלישה (SSL), ולכן לא ניתן לסמוך על כך שתעביר בוודאות נתונים נכונים. כפי שהוזכר בראשית המסמך, פרוטוקול DNS כשלעצמו, אינו מספק הגנה על התוכן: הנתונים אינם מאובטחים מפני זיוף או מניפולציה העלולים להתבצע עליהם בעת שהם מועברים ברשת בדרכם אל המשתמשים, ומוכרים גם מקרים של שינוי מידע על שרתי ה- DNS עצמם. באמצעות DNSSEC ניתן לאמת את המידע המתקבל בשאילתת DNS ובכך להקשות מאוד על גורם זדוני לבצע מניפולציה על מידע זה.

ולכן, המטרה של מנגנון DNSSEC היא
להגן על ציבור הגולשים ומשתמשי האינטרנט, ולא על נותן השירות עצמו

מנגנון DNSSEC הוא תוספת לפרוטוקול DNS, המתבססת על הרחבה הקרויה EDNS- Extended DNS. זהו חלק סטנדרטי מהפרוטוקול, ואינו מצריך נקיטת פעולה כלשהי לשם הפעלתו. הסבר מפורט על EDNS ניתן לקרוא כאן [PDF]:
https://www.isoc.org.il/wp-content/uploads/2019/01/DNS-Flag-Day-Final.pdf

קצת על קריפטוגרפיה

בתיקנון מנגנון DNSSEC הוגדר כיצד לשלב קריפטוגרפיה בפרוטוקול DNS:

- כיצד **לחתום דיגיטלית** על רשומות DNS (בצד השרת)
- כיצד לאמת **את החתימות** הנ"ל (בצד הלקוח)

מהי חתימה דיגיטלית?

בדומה לחתימה ידנית הנדרשת ע"מ לאמת צדדים בחוזה משפטי, פעולה בנקאית, וכל הליך אחר המחייב אותנטיקציה, כך גם בעת ביצוע פעולה מקוונת, לעיתים נדרש לוודא כי:

- המידע המוצג הינו אותנטי, כלומר נוצר ע"י מי שטוען שיצר אותו.
- המידע המוצג הוא מקורי ושלם: כלומר לא השתנה לאחר שנחתם ע"י בעליו החוקיים.

חתימה דיגיטלית היא אם כן פעולת חתימה ממוחשבת.

זוהי פעולה הדומה בבסיסה להצפנה. בדומה להצפנה, גם חתימה דיגיטלית היא תוצאה של ערבול המידע המקורי באמצעות מפתח קריפטוגרפי. אולם בשונה ממנה, המטרה היא לא להסתיר את המידע, אלא לאפשר אימות שלו. הדמיון הוא לתהליך הצפנה הקרוי "הצפנה א-סימטרית" כפי שיוסבר להלן:

הצפנה **סימטרית**: ערבול המידע והפענוח שלו נעשים באמצעות אותו מפתח.

זה מחייב הסכמה על סוד (מפתח) בין שני הצדדים (המצפין והמפענח).

הצפנה **א-סימטרית**: ההצפנה והפענוח נעשים ע"י מפתחות שונים.

הצד המצפין והצד המפענח לא נדרשים להסכים על סוד (לא מחליפים ביניהם מפתח).

הצפנה א-סימטרית היא כיום חלק בלתי נפרד מהפעילות ברשת האינטרנט, ומשמשת לגלישה (SSL/TLS), ניתוב (RPKI), מטבעות קריפטו, VPN, וכו'

חתימה (או הצפנה א-סימטרית) היא פעולה בין שני צדדים שאין ביניהם אמון (..ולכן היא נדרשת). פעולה זו מחייבת את אחד הצדדים (תלוי בתרחיש) לחולל זוג מפתחות, והם אלו שישמשו את הצדדים המעורבים. מפתחות אלה נקראים מפתח "פרטי" ומפתח "ציבורי":

1. **מפתח פרטי**: החלק הסודי של מי שחולל את המפתחות, עליו הוא שומר שלא יתגלה.
2. **מפתח ציבורי**: החלק הפומבי, אשר מתאים בלעדית למפתח הפרטי. מפתח זה יש לפרסם ברבים.

הערות

- מפתח אחד משמש לערבול המידע ואילו השני לפענוח, לא ניתן לבצע את שתי הפעולות עם אותו מפתח.
- למפתח הציבורי אין משמעות ללא מפתח פרטי תואם (ולהיפך).
- לא ניתן להסיק מהמפתח הציבורי דבר אודות הסוד (המפתח הפרטי).

הצפנה א-סימטרית

צד א' מעוניין לשלוח מסר מוצפן לצד ב'.

צד ב' מחולל זוג מפתחות, ודואג לפרסם את המפתח הציבורי שלו ברבים.

הצפנה: צד א' מצפין את המסר באמצעות המפתח הציבורי של צד ב'. המפתח הזה נגיש כיוון שצד ב' דאג כאמור לפרסמו. לאחר שהמסר הוצפן, המפתח הציבורי חסר ערך, לא ניתן לפענח באמצעותו את ההצפנה. לכן, אם צד ג' כלשהו מעוניין גם הוא לשלוח מסר מוצפן לצד ב', הוא ישתמש באותו מפתח ציבורי ש-ב' פרסם.

פענוח: המסר המוצפן ניתן לפענוח רק באמצעות המפתח הפרטי של ב'.

כיוון שצד ב' שומר על המפתח הפרטי בסוד, רק הוא יכול לפענח את המסר.

חתימה דיגיטלית

גם חתימה היא ערבול של המידע באמצעות מפתח אחד ופענוחו באמצעות מפתח שני, אולם הפעם זה נעשה "הפוך": ערבול המידע נעשה באמצעות המפתח הפרטי, והפענוח באמצעות המפתח הציבורי. דוגמא:

חתימה: צד א' מייצר מסמך שעליו הוא מעוניין לחתום, למשל מסמך PDF.

הוא מחולל זוג מפתחות ומערבל את המידע באמצעות המפתח הפרטי שלו.

את התוצר המעורבל הוא מצרף למסמך הגלוי, ובנוסף הוא מצרף גם את המפתח הציבורי שלו (כיוון שזה המפתח שמפענח את הפעולה).

חתימה היא אם כן המידע המקורי + המידע המעורבל + מפתח ציבורי.

את המפתח הציבורי נכון לצרף כסרטיפיקט (תעודה דיגיטלית), אשר מהווה הוכחה לנכונותו.

אימות: צד ב' מעוניין לאמת את המסמך. הוא יצטרך לפענח את המידע המעורבל, ואז להשוות את התוצר המפוענח אל מול המידע הגלוי. אם הם שווים אזי המידע הגלוי אותנטי.

הפענוח מתבצע באמצעות המפתח הציבורי של א'. כיוון שבתהליך החתימה, צד א' צירף סרטיפיקט המכיל את המפתח הציבורי שלו (ואיתו פרטים מזהים נוספים), כעת כל מאן דהוא יכול לאמת את החתימה וע"י כך לוודא את מקוריות ומהימנות המסמך.

הוכחת מהימנות: העובדה שהמפתח הציבורי של א' (ורק הוא) מצליח לפענח את המידע, משמעותה היא שרק המפתח הפרטי של א' ערבל אותו. זה אומר שצד א' הוא זה שחתם, כלומר המידע אותנטי. מכאן נגזרת החובה של צד א' לשמור על המפתח הפרטי בסוד, כך שרק הוא יוכל להפעיל אותו.

הוכחת מקוריות (שלמות): פענוח החתימה הוא כאמור תהליך הכולל השוואה של התוצר המפוענח, אל מול המידע הגלוי. כפי שנאמר קודם לכן, חתימה דיגיטלית היא תמיד מידע מעורבל בצירוף למידע הגלוי. בדוגמא שלנו, אם המסמך הנ"ל השתנה במהלך הזמן (כלומר מישהו ערך אותו לאחר שנחתם ע"י א'), אזי ניסיון לאמת אותו יכשל.
מדוע? כיוון שהתוצר המתקבל מפענוח החתימה יהיה שונה מהמסמך הגלוי. אם ההשוואה בין התוצר המפוענח למידע הגלוי נכשלת, המשמעות היא שהמסמך הגלוי השתנה, ובמילים אחרות, הוא לא מקורי. באופן דומה, אם מישהו ביצע במהלך הזמן מניפולציה כלשהי על החתימה עצמה, אז המפתח ש- א' פרסם לא יתאים, ולא ניתן יהיה לפענח אותה.
לעומת זאת, אם החתימה נפתחת, והתוצר שהתקבל זהה למידע הגלוי אזי הוכחנו את שלמותו.

ביצוע פעולות קריפטוגרפיות מחייבת שמירה על מספר עקרונות

1. שמירה על סוד

יש להגן על המפתח הפרטי!

המשמעות של דליפה או גניבת המפתח הפרטי היא **חמורה**. גורם זדוני אשר ישים ידו על המפתח יכול להתחזות לבעל המידע. הדרך המועדפת לשמירה על מפתחות פרטיים היא באמצעות "מודול אבטחה חומרתי" (HSM - Hardware Security Module). זהו רכיב פיזי אשר מתממשק לאפליקציה החותמת, ומטרתו להגן על מפתחות קריפטוגרפיים. בוודאות גבוהה מאוד לא ניתן להוציא מפתחות מ- HSM. כל גישה למפתח הפרטי (למשל בעת חתימה על מסמך) נעשית תוך התממשקות ל-HSM, כך שפעולת החתימה נעשית בתוכו. המפתח לעולם לא יוצא מה- HSM.

בשוק קיים מגוון של מוצרי HSM. הפשוטים ביותר הם כרטיס חכם עם צ'יפ (דוגמת SIM, כזה שנמצא בכל טלפון נייד ובאמצעותו ניתן לאמת את הטלפון מול רשת סלולרית), או התקן ייעודי דוגמת YubiKey, שאותו ניתן לחבר באמצעות usb/nfc למחשב. המגבלה של רכיבים אלה היא בד"כ כמות פעולות קריפטוגרפיות לשניה, שכידוע דורשות כוח עיבוד, ובנוסף איכות וכמות המפתחות הנשמרים על הרכיב.

רכיבי HSM רשתיים הם פתרונות חזקים יותר והם מקובלים בארגונים גדולים, שם יש משמעות הן לתקינה של ה- HSM והן לביצועים שלו. החיסרון של רכיבים אלה הוא עלות גבוה ותקורה תפעולית. כמו כל רכיב ברשת, גם HSM מצריך ניטור ותחזוקה.

בנוסף, שימוש ב- HSM הוא בד"כ לא טריוויאלי. הוא מחייב צוות מתפעל, הבנה של הממשק, הגדרת נהלי עבודה וסטנדרטים מגבילים של אבטחת מידע.

נקודות נוספות שיש לתת עליהן את הדעת:

א. זמינות במקרה תקלה - HSM-ים נבחרים, מציגים גם יכולת של אשכול (Cluster), במתכונת של Active Stand By.

ב. גיבוי של המפתחות הנשמרים ב-HSM - כאמור, המפתחות לעולם לא עוזבים את הרכיב, אולם יחד עם זאת אם הוא נפגם בעקבות תקלה, פגיעה פיזית, וכו", חייבת להיות יכולת לשחזר את המפתחות לרכיב חלופי. יש לכך מגוון פתרונות: החל משמירת מפתחות ב-offline בכספת נעולה, ועד למודול גיבוי ייעודי ונפרד של יצרן ה- HSM. הדרך הנפוצה והפשוטה לשמור על מפתח פרטי היא במחיצה מקומית על המחשב החותם. זהו הפתרון הקל והנוח, אולם גם הכי פחות מאובטח. לכל הפחות יש להצפין מחיצה כזו, להפעיל בקרת גישה, ולהגדיר הרשאות מתאימות (אדמיניסטרטור בלבד).

2. יכולת לחולל מפתחות חזקים

חוזק הצופן תלוי באלגוריתם ההצפנה ובאורך המפתחות.
אלגוריתם RSA/SHA256 עם מפתחות באורך 1024 לפחות הינו האופציה הוותיקה, אולם כיום, ב-DNSSEC היא פחות מועדפת. אלגוריתם ECDSA P-256 יעיל הרבה יותר, מספק הגנה טובה יותר, באמצעות מפתחות קטנים בהרבה (מפתח EC באורך 256 שקול מבחינת חוזקו למפתח RSA באורך 3072) זהו יתרון משמעותי במערכת אונליין מהירה כמו DNS.

3. החלפת מפתחות באופן מחזורי

היכולת של גורם זדוני לנחש מפתח קריפטוגרפי היא פונקציה של כוח עיבוד וזמן. לכן יש לוודא החלפת מפתחות תקופתית = "גלגול מפתחות".
חשוב לזכור: לאחר החלפת מפתחות, המפתח הציבורי החדש לא יכול לאמת חתימה שנעשתה באמצעות המפתח הפרטי הישן. לכן הפעלה של מפתח חדש מחייבת ביצוע חתימה חדשה באמצעותו. ולהיפך: כל עוד נעשה שימוש בחתימות שבוצעו באמצעות מפתח ישן, חובה להמשיך לספק את המפתח הציבורי המתאים לו (זה שיאמת את החתימה).

4. וידוא תקינות חתימה

- תוקף של חתימה: מציין עד עד מתי ניתן להשתמש בה ע"מ לאמת את המידע החתום. אחרי פקיעת התוקף לא ניתן לאמת את החתימה. לכן, יש לדעת מתי תוקף של חתימה עומד לפוג, ולדאוג לחתום על המידע שוב במידת הצורך (יש לשים לב שלא מדובר על החלפת מפתח אלא על ביצוע חתימה חדשה, באמצעות אותו מפתח). תרחיש כזה נדרש כאשר המידע המקורי משתנה, והבעלים של המידע לא ירצה לאפשר אימות של מידע ישן מדי. דוגמא לתרחיש כזה היא חתימה על רשומות DNS אשר משתנות באופן תדיר.
- תמיד יש לוודא שהמפתח(ות) הציבורי שאותו אנחנו מפרסמים מתאים לחתימות תחת אחריותנו.
- תמיד יש לוודא שלא חל שיבוש כלשהו שלא מאפשר למפתח הקיים לאמת את החתימה.

5. מנגנון אמון

כיצד ניתן לוודא כי מפתח ציבורי כלשהו, המפורסם ברבים, הוא אכן המפתח האמיתי של בעל המידע? גורם זדוני יכול להציג מפתח כלשהו בטיעון כאילו זהו המפתח הלגיטימי, לחתום על מידע שקרי באמצעות המפתח שהציג, ולגרום למי שהשתמש במפתח הזה לחשוב שהמידע אותנטי. ואכן, בסביבה שבה אין אמון בין הצדדים (למשל רשת אינטרנט), לא נכון להשתמש במפתח ציבורי לא מאומת. הפתרון המקובל הוא שימוש בתעודות דיגיטליות (סרטיפיקטים). סרטיפיקט מכיל מפתח שהוא בעצמו חתום ע"י CA (רשות מאשרת - Certificate Authority), או שרשרת של CA-ים שעליהם אנו סומכים. מנגנון האמון מאפשר לסמוך על המפתח שהוצג ולבטוח במהימנותו.
ב- DNSSEC כפי שיודגם להלן, לא משתמשים בסרטיפיקטים, אלא במפתחות. לכן DNSSEC מקיים מנגנון אמון בצורה שונה.

DNSSEC - עקרונות

1. הצד החותם הוא בעל המידע (בעל הדומיין) או בא כוחו, כלומר הגוף שמפרסם את הדומיין על שרתי ה- DNS שלו. ניתן לחתום על דומיין במגוון דרכים, כפי שיפורט בהמשך.
2. הצד המאמת הוא המשתמש. בכל גלישה לאתר או צריכה של שירות, נדרשת כתובת IP. בכל פעולה כזו המשתמש מבצע שאילתת DNS. אם התשובה לשאילתת ה- DNS תהיה חתומה, המשתמש הוא זה שמאמת אותה.
3. דומיין חתום DNSSEC הוא דומיין רגיל, אשר מתפרסם על שרת DNS שאינו מצריך משאבים נוספים (CPU/ זיכרון) או יכולות מיוחדות. כל שרת DNS שמפרסם דומיינים "רגילים" (לא חתומים), יכול לפרסם גם דומיין חתום.
4. שרת DNS שמפרסם דומיין (קרוי גם שרת DNS "אוטוריטטיבי"), תפקידו **רק** לענות על שאילתות. אם הוא מפרסם דומיין חתום, לא הוא זה שחותם עליו. תהליך החתימה מתבצע ברשת אחורית, למשל על המאסטר או על שרת חתימה ייעודי. מקובל גם לחתום באמצעות שירות בתשלום. בהמשך המסמך יש התייחסות לארכיטקטורות אפשריות.
5. חתימה ב- DNSSEC היא על רשומות, ולא על קובץ ה- zone כולו כיחידה אחת.
המשמעות: דומיין חתום מכיל רשומות DNS "רגילות", ובמקביל אליהן גם את החתימות של אותן רשומות.
6. בנוסף, דומיין חתום יכיל גם את המפתח הציבורי שאיתו ניתן לאמת את החתימות הנ"ל.
7. **גם החתימות וגם המפתחות הן בעצמן רשומות DNS.**
8. כאמור בסעיף 2, האימות של התשובה החתומה מתבצע בצד הלקוח.
בעולם ה- DNS, צד הלקוח הוא ה- DNS אשר מוגדר אצל המשתמש (ברוב המקרים בצורה אוטומטית). במחשב, בטלפון הנייד וכו' למשל:
   - שרת ה- DNS הניתן ע"י ספק האינטרנט
   - שרת ה- DNS הארגוני (למשל במקום העבודה)
   - שירות DNS ציבורי דוגמת Google 8.8.8.8, Quad9 9.9.9.9, Cloudflare 1.1.1.1

בשונה משרת DNS "אוטוריטטיבי" (אשר מפרסם את המידע), שרת ה- DNS בצד הלקוח לא מחזיק שום מידע. הוא מקבל את השאילתה מהמשתמש, ומחפש עבורו את המידע בשרתים אוטוריטטיביים. בסיום הוא מחזיר את התשובה שקיבל. אם התשובה חתומה, הוא זה שמאמת את נכונותה. הסבר מפורט יינתן בהמשך.
9. כפי שהוזכר קודם לכן, נוסף על אימות החתימה, תמיד יש לוודא גם את מהימנות המפתח החותם. המנגנון המקובל לאימות מפתח הוא תעודה דיגיטלית (סרטיפיקט). כיוון שב- DNSSEC אין סרטיפיקטים, המפתח צריך להיות חתום בעצמו.

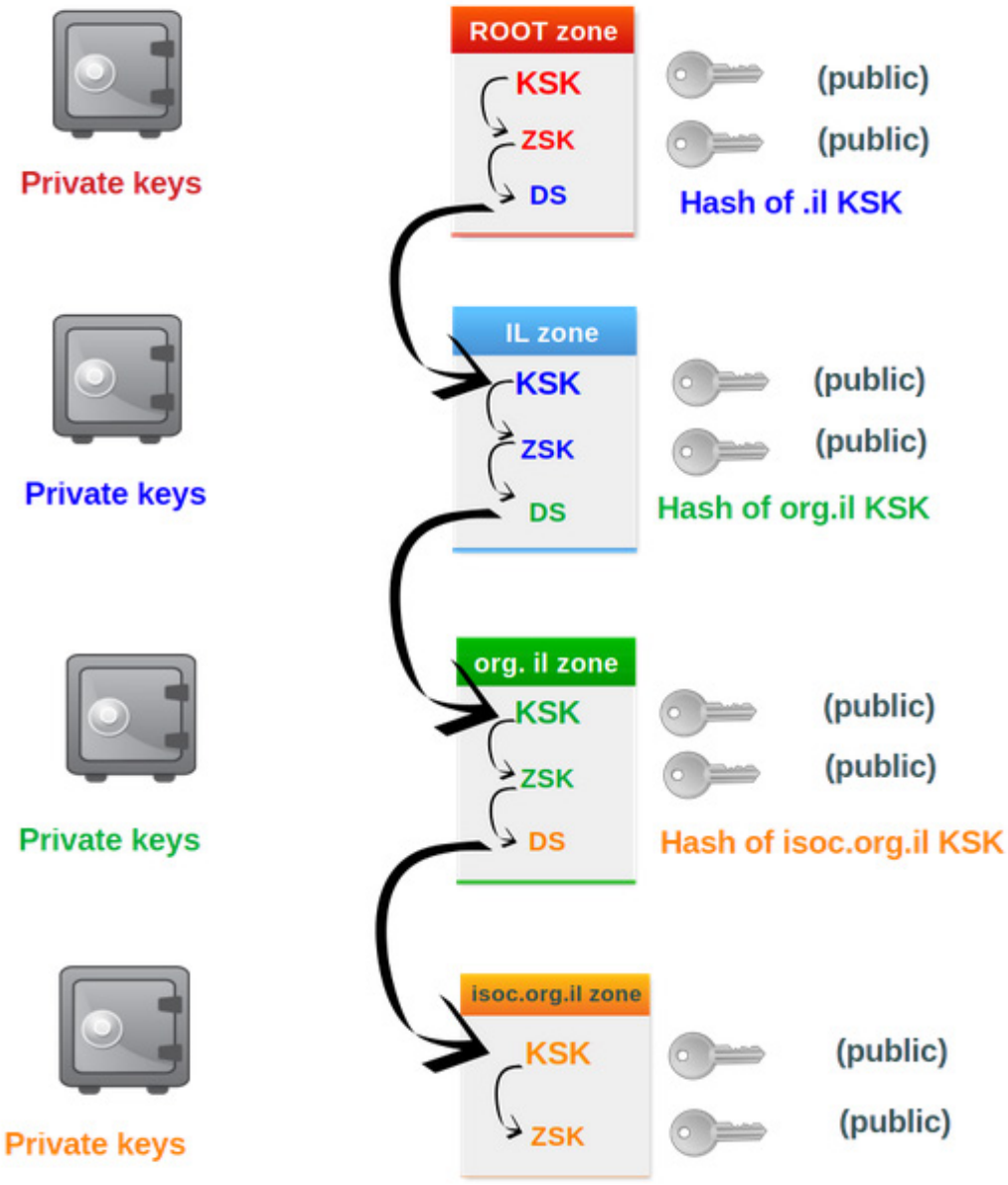
- מקובל (אם כי לא חובה) שלצורך ביצוע חתימה של מפתח, מחוללים מפתח נוסף, שזהו תפקידו היחידי: לחתום על המפתח "הרגיל".
- המפתח "הרגיל", אשר חותם על כלל הרשומות, נקרא: **ZSK** - Zone Signing Key
- המפתח הנוסף, אשר חותם רק על מפתחות, נקרא: **KSK** - Key Signing Key
- לחילופין, ניתן להשתמש במפתח אחוד הנקרא: **CSK** - Combined Signing Key

הערות:

- CSK הינו פתרון מקובל, אולם במסמך זה נתייחס לחלופה הנפוצה של ZSK ו- KSK.
- כאמור, כל מפתח (KSK, ZSK וגם CSK) וכל חתימה הן רשומת DNS לכל עניין ודבר.

10. תפקידו של ה- KSK הוא לחתום על ה- ZSK, כלומר להיות ערב לנכונותו של המפתח "הרגיל".
אולם מי ערב ל- KSK? כאן בא לידי ביטוי מנגנון האמון ב- DNSSEC, המאפשר לסמוך על ה-KSK. כיצד זה מתבצע:
מנגנון האמון מבוסס על המבנה ההיררכי של DNS. הארכיטקטורה של DNS היא כידוע במבנה של עץ: פרט לרמת השורש, לכל רמה ב- DNS יש רמה מעליה, אשר מצביעה עליה (הפניה שכזו בין רמות ב- DNS נקראת גם "דלגציה").
לכן, אם נציב ברמה שמעל הדומיין החתום מידע ייחודי אודות ה- KSK שלנו, אזי הרמה שמעל תהיה ערבה לנכונותו.
רשומת DNS, שתפקידה להצביע על ה- KSK של דומיין חתום, מתוך הרמה שמעליו, נקראת `DS - Delegation signer` . נתייחס בפרוטרוט לכל רשומה בהמשך, אולם כעת ניתן לומר כי רשומה זו מפרסמת טביעת אצבע (fingerprint) של ה-KSK שנמצא רמה מתחת (טביעת האצבע היא HASH של ה-KSK המדובר). טביעת האצבע המתפרסמת ברשומת DS חייבת להתאים ל-KSK הנ"ל.

דוגמא לשרשרת האמון בעץ ה- DNS בעמוד הבא:
בתחתית נמצא הדומיין החתום, במקרה זה isoc.org.il. הרמה שמעליו (org.il) מכילה רשומת DS שמתייחסת ל- KSK של isoc.org.il.
באופן דומה, גם ב- il יש DS שמצביע על ה- KSK של org.il, וכן הלאה.
כיוון שכל רשומת DS חתומה גם היא, יצרנו **שרשרת אמון**:
בכל רמה יש רשומת DS אשר מצביעה על ה-KSK שמתחתיה, וכל DS כזה חתום בעצמו ע"י ZSK (שבעצמו חתום ע"י KSK), וחוזר חלילה עד ל- root העולמי, עוגן האמון שעליו אנחנו סומכים. לכן, תנאי ראשון לכך שדומיין ייחתם DNSSEC הוא שהרמה שמעליו תהיה חתומה.


Private keys
ROOT zone
KSK
ZSK
DS
(public)
(public)
Hash of .il KSK
Private keys
IL zone
KSK
ZSK
DS
(public)
(public)
Hash of org.il KSK
Private keys
org. il zone
KSK
ZSK
DS
(public)
(public)
Hash of isoc.org.il KSK
Private keys
isoc.org.il zone
KSK
ZSK
(public)
(public)

דגשים

קיומה של רשומת DS מהווה הצהרה עקרונית כי **הדומיין שאליו היא מתייחסת חתום**.

יש לכך השלכות חשובות:

- **אם קיימת רשומת DS, היא חייבת להתאים ל- KSK ברמה שמתחתיה.**
- **כל עוד יש DS, הרמה שמתחת חייבת לפרסם KSK מתאים.**

אם מסיבה כלשהי אין התאמה בין ה-DS ל-KSK, או שה- KSK לא קיים (למשל כיוון שנמחק ע"י אדמיניסטרטור) המשמעות היא חמורה:

הדומיין לא ניתן לאימות ולכן נחשב כמזויף

מצב זה שקול לעצירה של השירות = הדומיין אינו נגיש!

- **תהליך ההקמה של דומיין חתום DNSSEC:**

  פרסום ה- DS הוא השלב האחרון.

  ראשית יש לחתום על הדומיין, לבצע בדיקות, ורק לאחר שהכל נמצא תקין, יש לפרסם DS.
- **כל עוד לא פורסם DS, מבחינת העולם לא הופעל DNSSEC. ניתן לבצע מניפולציות כולל מחיקה, החלפת מפתחות וכו', ללא פגיעה בשירות.**
- **במקרה שמסיבה כלשהי נדרש להפסיק DNSSEC על דומיין מסוים, התהליך הוא הפוך:**

  קודם כל יש להסיר את ה- DS ורק אז ניתן לשנות או להסיר מפתחות.

  אחרי שה- DS הוסר אפשר להתייחס לדומיין כאל דומיין "רגיל" שאינו חתום, גם אם הוא עדיין מכיל מפתחות וחתימות. רק כשאין רשומת DS ניתן למחוק מפתחות.
- **אסור! למחוק KSK לפני שה-DS שמתייחס אליו הוסר קודם לכן.**
- יש לזכור שמרגע שה- DS הוסר או הוסף יש להמתין מספיק זמן (TTL של הרשומה) על מנת שהשינוי ייקלט בכל העולם.
- הסרה של DS היא לא פעולה שמבצעים כחלק מתפעול שוטף.

זוהי פעולה שמבצעים רק אם יודעים מה עושים. יש לבצע רק במקרה של החלפת מפתח או במקרה חירום, תקלה משביתה, כמפלט אחרון.

רשומות DNSSEC

התוספות לפרוטוקול DNS ע"י DNSSEC הן:

- רשומות DNS חדשות
- הצהרות ב- DNS header ("דגלים" - flags, שלחלקם נתייחס בהמשך)

בעת ביצוע חתימת DNSSEC של דומיין, יתווספו אליו רשומות חדשות מסוג "מפתח", "חתימה" ועוד.. בפרק זה נפרט אודות רשומות אלה, ועל שאר הרשומות המתקבלות במימוש DNSSEC. רשומות אלה הן כאמור חלק מפרוטוקול DNS התקני, והן מוכרות ע"י כל שרתי ה- DNS הידועים. ולכן, בנוסף לרשומות הקלאסיות מסוג A, MX, NS, SOA, TXT וכו'.. דומיין חתום יכיל גם רשומות DNSSEC כדלקמן:

רשומה מסוג DNSKEY (מפתח ציבורי)

רשומה זו מכילה את המפתח הציבורי שיש לפרסם ע"מ לאמת חתימה. המפתח הפרטי נשמר כמובן בסוד ואותו אסור בשום אופן לחשוף. רשומה זו מתווספת לדומיין בעת הפעלת תוכנת החתימה, והיא נראית כך:

Domain name	TTL	Class	Type	Flags	Protocol	Algo	Public key
co.il.	3600	IN	DNSKEY	256	3	13	De30VOwRIfeFqHt8qVmL6kVZjK..

השדה הימני מכיל את המפתח הציבורי.

- שדה **Algo** מציין את מספר האלגוריתם הקריפטוגרפי. האלגוריתמים המקובלים הם RSA או Elliptic Curve שסימנם 8 ו- 13 בהתאמה
- שדה **Flags** הוא מספר שמייצג את סוג המפתח: **ZSK = 256, KSK = 257**

דומיין חתום יכיל שתי רשומות מסוג DNSKEY: אחת ZSK ואחת KSK

בדוגמא הנ"ל מוצג מפתח Elliptic curve מסוג ZSK של הדומיין co.il

רשומה מסוג RRSIG (חתימה)

רשומה זו מכילה את התוצר שהתקבל אחרי הפעלת מפתח, כלומר זוהי החתימה בפועל.
כפי שהוסבר קודם לכן חתימה ב- DNSSEC היא על רשומות, ולכן זוהי חתימה של רשומה כלשהי בדומיין (יוסבר איזו):

Domain name	TTL	Class	Type	Type signed	Algo	# labels	orig TTL	sig exp.	sig create	key ID	signer name	signature
co.il.	3600	IN	RRSIG	NS	13	2	3600	26200322680904	26200220670904	47665	co.il.	GrTHztnLh+axleW8m...

- השדה הימני מכיל את החתימה עצמה.
- שדה **signer name** הוא שם הדומיין שבו נמצא המפתח שיצר את החתימה
- שדה **Key ID** מציין את המזהה של המפתח שחתם (לכל מפתח יש מספר סידורי שמנוהל ע"י תוכנת החתימה)
- שדות **sig create , sig exp** מציגים את התוקף של החתימה. לפני ואחרי תאריכים אלה החתימה אינה תקפה (תוקף חתימה מוסבר בפרק המדיניות בעמ' 35)
- שדה **orig TTL** מציין את ה- TTL של הרשומה המקורית לפני שנחתמה
- שדה **labels** מציין את מספר הלייבלים המרכיבים את שם הדומיין (ב- co.il זה שניים)
- שדה **Type signed** מציין את סוג הרשומה המקורית שזאת החתימה שלה

בדוגמא הנ"ל מוצגת חתימה של רשומה מסוג NS אשר נחתמה ע"י מפתח מס' 47665
מסוג Elliptic curve, בדומיין co.il

הערה:
כאשר חותמים ב- DNSSEC על דומיין, החתימות הן כאמור של רשומות בדומיין. לכל רשומה בדומיין תהיה חתימה מתאימה.
נקודה זו יש לדייק:
למעשה, החתימה לא נעשית ממש על כל רשומה בנפרד, אלא על קבוצות של רשומות הקרויה:
RRSET = Resource record set.
נסביר בעמוד הבא.

Resource Record Set - RRSET

קבוצת רשומות שהן מאותו סוג (אותו שדה TYPE), ויש להן אותו שם (אותו דומיין), ייחשבו RRSET אחד. לדוגמא, אם בדומיין example.com יש 3 רשומות MX אזי שלושתן יקובצו לאותו RRSET.

כנ"ל רשומות מסוג NS של הדומיין, יקובצו ל- RRSET משלהן, וכו'..

לעומת זאת אם בדומיין שלנו יש משהו כזה:

```
www.example.com      IN    A     10.20.30.40
ftp.example.com      IN    A     10.20.30.41
```

אזי שתי הרשומות האלה לא ישתייכו לאותו RRSET. אמנם שתיהן מסוג "A" אבל לכל אחת יש שם אחר (סאב דומיין אחר).

ולסיכום: פעולת חתימה של DNSSEC נעשית על RRSET ולא על כל רשומה בנפרד.

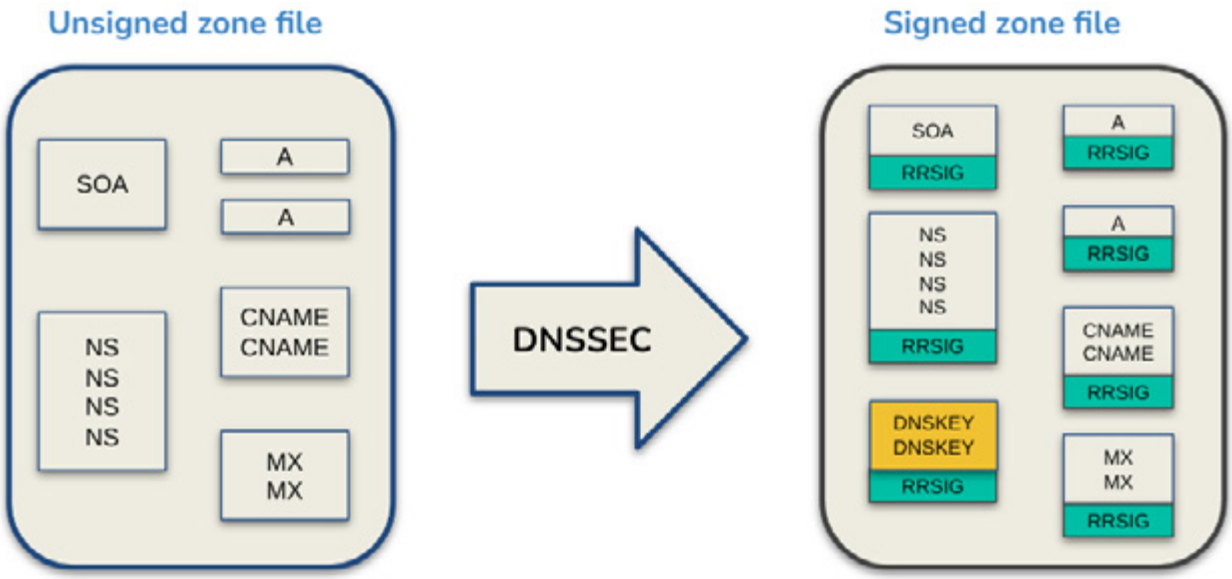
ייתכן ש- RRSET מכיל כמה רשומות וייתכן שרק רשומה אחת (למשל בדוגמא הנ"ל, כל A record יימצא ב- RRSET משלו).

בציור להלן מוצג בצד שמאל zone "רגיל", לא חתום, שבו רשומת SOA, שתי רשומות מסוג A, שתי רשומות CNAME, שני MX ו-4 רשומות NS. כל ריבוע הוא RRSET.

לאחר הפעלת DNSSEC תתווסף רשומה מסוג RRSIG (חתימה), לכל RRSET (כל ריבוע), בנוסף ניתן לראות RRSET חדש שהוא שני המפתחות (שתי רשומות מסוג DNSKEY).

ל- zone שלנו התווספו אם כן 9 רשומות חדשות:

- 2 רשומות מסוג מפתח - DNSKEY (אחד ZSK ואחד KSK)
- 7 רשומות מסוג חתימה - RRSIG



רשומה מסוג DS

במבנה ההיררכי של DNS, הצבעה מרמה אחת בעץ לרמה שמתחתיה נקראת "דלגציה", והיא נעשית באמצעות רשומות מסוג "NS".

רשומות NS תפקידן לפרסם מי הם שרתי ה-DNS של דומיין מוכל (זה ששמו מופיע בשדה השמאלי).

לדוגמא, ב- `org.il`, הדלגציה ל- `isoc.org.il` תכיל למשל את הרשומה:

```
isoc.org.il.      86400     IN       NS       ns1.isoc.org.il.
```

כאן מוגדר ששרת DNS של דומיין `isoc.org.il` הוא `ns1.isoc.org.il` והמשמעות היא שיש לפנות אליו בשאילתות לגבי הדומיין הזה (זוהי דלגציה).

אימות של דומיין חתום DNSSEC יתאפשר רק אם קיימת גם רשומה מסוג DS (דלגציה מאובטחת).

באחריות בעל הדומיין החתום ליצור רשומת DS (יוסבר בהמשך כיצד), **ולהעביר אותה לגורם המתפעל את הרמה שמעליו**.

בדוגמא שלנו : באחריות הגורם המתפעל את הדומיין `isoc.org.il` להעביר DS למי שמתפעל את `org.il`.

כפי שהוסבר בעמודים 12 - 14, רשומת DS מכילה HASH (טביעת אצבע) של מפתח (KSK) שנמצא ברמה שמתחתיה, ויש משמעות גדולה לתקינותה, לקיומה או אי קיומה של הרשומה.

רשומת DS ללא KSK מתאים תכשיל את הדומיין החתום עד כדי עצירת השירות.

זהו מנגנון הגנה מובנה כנגד אי יכולת לאמת את המפתח (מתוך הנחה שאחת הרמות- המכילה או המוכלת, נפרצה או הושחתה).

דוגמא לרשומת DS של דומיין isoc.org.il, כפי שהיא מופיעה ב- org.il

Domain name	TTL	Class	Type	Child's key tag	Algo	Digest type	Hash value
isoc.org.il.	7200	IN	DS	48693	13	2	941C5E5CBDA7343DBEFB6B7AE16583B10DE5

השדה הימני מכיל את טביעת האצבע (HASH) של ה- KSK שאליו ה-DS מתייחס

- שדה **Digest type** מציין את סוג ה- HASH. התקן היום הוא SHA2
- שדה **Algo** מציין את האלגוריתם הקריפטוגרפי של ה-KSK שאליו מתייחסים
- שדה **Child key tag** מציין את המספר הסידורי של ה- KSK שאליו ה- DS מתייחס

רשומת ה- DS הנ"ל מתייחסת ל-KSK מספר 48693 מסוג elliptic curve של דומיין isoc.org.il.

הרשומה עצמה נמצאת ברמה שמעל - org.il

ייצור רשומת DS

רשומות DNSSEC נוצרות באופן אוטומטי ע"י תוכנת החתימה ומתווספות ל- zone file בעת ביצוע חתימה על דומיין. העיקרון הזה חל על כולן, פרט לרשומת DS.

רשומת DS יש לייצר עצמאית: פעם אחת כאשר מפעילים DNSSEC לראשונה, ולאחר מכן בכל פעם שה- KSK מתחלף (דוגמאות יש בהמשך, ובעמ' 48, סעיף 6).

בשוטף, החלפת KSK אמורה להתבצע בצורה מתוזמנת מראש, ע"י תוכנת החתימה. מקובל לתזמן החלפה (גלגול) KSK פעם בשנה (תלוי בחוזק המפתח).

ניתן וצריך לדעת מראש אם ומתי התוכנה מתעתדת להחליף את המפתח, ולוודא שהמהלך התבצע במועד ובצורה תקינה, כלומר שהתווספה רשומת KSK חדשה.

תהליך החלפת KSK מתבצע בשלבים:

- בשלב הראשון, לאחר הגלגול, שני ה-KSK, החדש והישן יתפרסמו זה לצד זה.
- עד שלא מתווספת לרמה שמעל, רשומת DS שמתאימה ל- KSK החדש, יש להמשיך לחתום עם ה- KSK הישן (כיוון שרק אותו ניתן לאמת!).
- לאחר שיצרנו רשומת DS שמתאימה ל- KSK החדש (בהמשך נראה כיצד), עלינו להעביר אותה לגורם המתפעל את רמת ה-DNS שמעל. את ה- DS של דומיין `x.co.il` יש להעביר לגורם המתפעל את `co.il` ובאופן דומה, את ה- DS של `y.org.il` ל- `org.il`.
- הגורם המתפעל את מרחבי השמות תחת IL (פרט ל- gov.il ו- idf.il) הוא איגוד האינטרנט הישראלי . רשומות DS יש להעביר באמצעות הרשם המתפעל.
- יש לשים לב, כי בשלב ראשון פרסום ה- DS החדש הוא בנוסף ל- DS הישן, ולא במקומו!
- רק אחרי שעבר מספיק זמן (לא פחות מה- TTL של ה-DS, שהוא בד"כ יום) אפשר להורות לתוכנת החתימה להפסיק לחתום עם המפתח הישן, ולחתום רק עם החדש.
- אחרי פרק זמן נוסף (שוב, לפחות יום או TTL) שבמהלכו נבדק כי החתימה של ה- KSK החדש תקינה וניתנת לאימות, אפשר להורות לתוכנת החתימה למחוק את המפתח הישן.
  בסוף המסמך, בפרק "בדיקות" מוצגות אפשרויות של בדיקת DNSSEC
- רק אחרי שהמפתח הישן נמחק, יש להסיר את ה- DS הישן (שוב, ע"י פניה באמצעות הרשם המתפעל את הדומיין)

כיצד לייצר רשומת DS

1. דומיינים רבים מנוהלים אצל ספק DNS המאפשר ניהול עצמי באמצעות פורטל (למשל Wix, Cloudflare, Akamai, CloudDNS או ספקי ענן AWS, GCP וכו').
   במקרים אלה סט הפעולות הדרוש להפעלת DNSSEC הוא מצומצם וברוב המקרים כל שנדרש הוא לסמן למערכת לחתום DNSSEC, והשאר מתבצע באופן אוטומטי. יחד עם זאת, כפי שהוזכר קודם לכן, ייצור רשומת DS והעברה לרמה מעל היא באחריות לקוח הקצה או בא כוחו. לכן,

בממשק הניהול, יש לבקש לייצר רשומת DS, ואת התוצר המתקבל (שורת טקסט) יש להעביר לרשם שדרכו נרשם הדומיין.

2. ניתן לייצר רשומת DS באמצעות פקודה: **dnssec-dsfromkey** שהיא חלק מסט הפקודות בחבילת "bind9-utils" של ISC (ניתנת להתקנה חינמית).
   את הפקודה אפשר להריץ במספר אופנים (יש לעיין בתיעוד). ניתן לייצר DS עפ"י המפתח שנשמר מקומית בשרת או לחילופין לפי המפתח הציבורי המתפרסם בעולם, כלומר ממש מתוך שאילתת DNS. לדוגמא:

שאילתת DNS באמצעות פקודת dig - המראה מהם המפתחות של דומיין isoc.org.il

```
dig dnskey isoc.org.il
257  3    13   0uIaVQeiX5wtvq9yEAiJjBI58qATI56NfG2+2 ... yM/
M32ve2H2RAP9w==
256  3    13   mW6EF30JCft2c1HXiyOOFzZ/wXPszXnIcCexM ...
T8NV9rWom45Hw59Q==
```

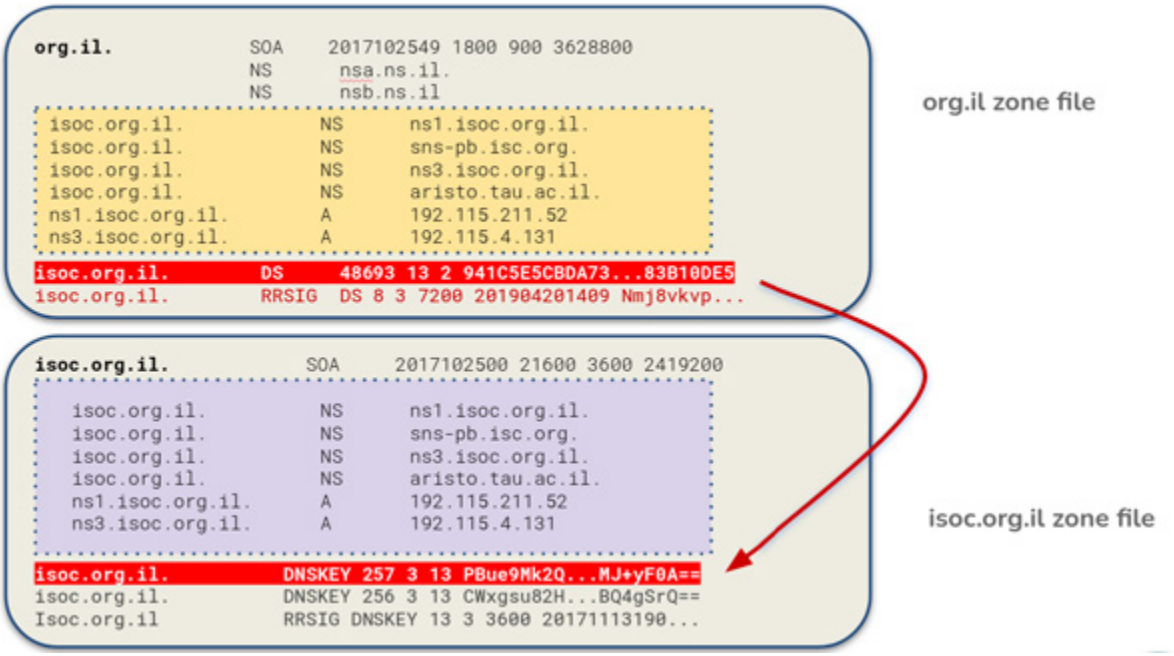
את הפלט של השאילתה הנ"ל ניתן להעביר ל- dnssec-dsfromkey ולקבל את ה- DS:

```
dig dnskey isoc.org.il | dnssec-dsfromkey -f - isoc.org.il
isoc.org.il. IN DS 27123 13 2 3FB17345C8B2DF51289DF7CE ...
BAF14B7EF98E0CD72946
```

3. ניתן לייצר רשומת DS באמצעות תוכנת החתימה, דוגמת OpenDNSsec, כפי שמתואר בעמ' 48, סעיף 6
4. ניתן לייצר רשומות DS בסקריפט פייתון, ועוד ועוד..

דוגמא 1

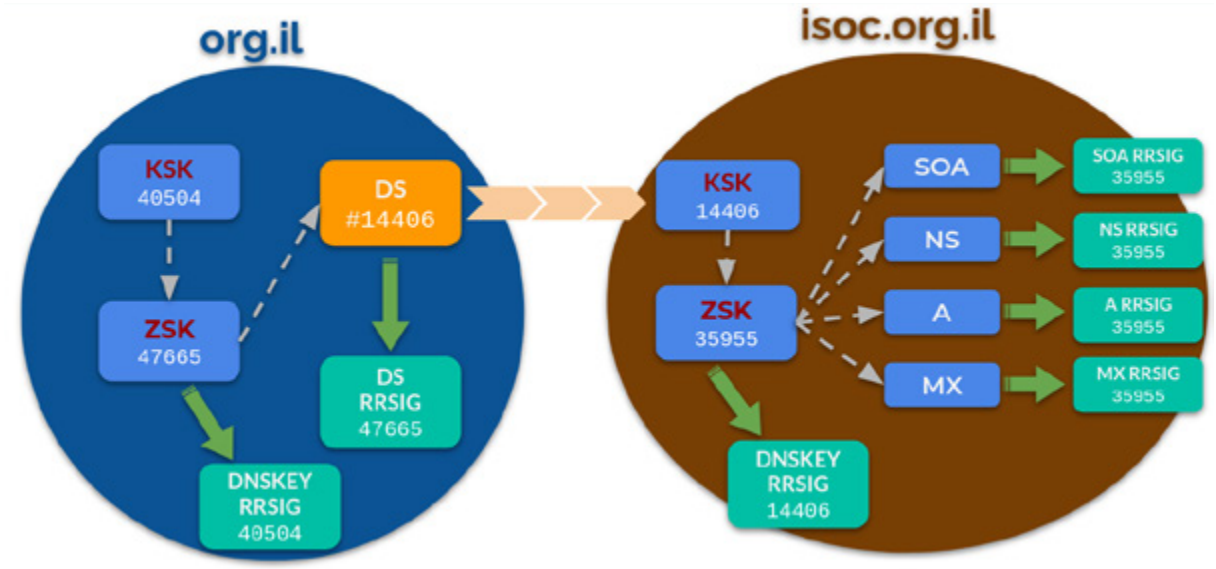
כך נראית דלגציה מאובטחת בין שתי הרמות: `org.il` ו- `isoc.org.il`



ניתן לראות את רשומות ה- NS של isoc.org.il כפי שהן מופיעות ב- org.il (דלגציה רגילה) ובנוסף את רשומת ה- DS המתייחסת למפתח KSK ברמה שמתחתיה (דלגציה מאובטחת)

דוגמא 2

תיאור מנגנון האמון: הקשר בין המפתחות וה- DS דלגציה מאובטחת בין שתי רמות חתומות



בצד שמאל: ה- zone החתום של org.il

מכיל מפתח מסוג KSK שמספרו 40504, ומפתח מסוג ZSK שמספרו 47665

שני המפתחות חתומים ע"י ה- KSK, מספר 40504

רשומת DS מצביעה על KSK מס' 14406 של isoc.org.il (דלגציה מאובטחת)

ה- DS, כמו כל רשומה אחרת, חתום גם הוא ע"י ה- ZSK של org.il

בצד ימין: ה- zone החתום של isoc.org.il

מפתח מסוג KSK שמספרו 14406, ומפתח מסוג ZSK שמספרו 35955

שני המפתחות חתומים ע"י ה- KSK, מספר 14406

כל שאר הרשומות (כל RRSET) ב- isoc.org.il חתומות ע"י ZSK מס' 35955

בין מפתח והרשומה שעליה הוא חותם יש חץ מקווקו ◄---

◄— חץ ירוק מראה את התוצר = החתימה המתקבלת

רשומה מסוג NSEC או NSEC3
(הוכחת אי קיום- proof of non existence)

מנגנון DNSSEC מאפשר להחזיר תשובה חתומה לשאילתת DNS. אולם זוהי פעולה המתאפשרת כאשר יש תוכן שעליו ניתן לחתום, כלומר כאשר יש תשובה לשאילתה. נשאלת השאלה מה קורה כאשר התשובה שלילית?

ובמילים אחרות, כיצד אפשר להוכיח שמשהו לא קיים?

במבט ראשון נראה שאין בעיה. עפ"י פרוטוקול DNS, התשובה המוחזרת במקרה של דומיין לא קיים היא "**NXDOMAIN**", ולכן ב- DNSSEC פשוט נצרף לתשובה את החתימה של המחרוזת הזאת.

לדוגמא:

נניח שהחתימה של NXDOMAIN עפ"י המפתח שלנו היא: "ABCDefgh123=="

כעת, בכל פעם שנצטרך להחזיר תשובה שלילית, נחזיר את התשובה **NXDOMAIN** בצירוף החתימה הנ"ל. ואולם, בחינה נוספת מראה שדווקא יש בעיה.

נחזור כדוגמא ל- zone שלנו, "example.com"

ונניח שיש בו 3 תת- דומיינים:

www.example.com

ftp.example.com

sub.example.com

שאילתה לגבי דומיין לא קיים, למשל: en.example.com תחזיר **NXDOMAIN** בצירוף "ABCDefgh123==". החתימה הזו לגיטימית, ניתן לאמת שהיא אכן של **NXDOMAIN**.

ואולם, גם שאילתה על דומיין אחר שאינו קיים ב- zone, למשל he.example.com תחזיר גם היא את אותה תשובה בדיוק. וכאן הבעיה: אותה תשובה מוחזרת לכל דומיין לא קיים.

גורם זדוני שיש ביכולתו ליירט או לצוטט לתקשורת הזו, יוכל להזריק את התשובה הנ"ל גם בעבור שאילתות תקינות.

כיוון שתמיד ניתן לאמת את התשובה הזו , ניתן למעשה לייצר מתקפת מניעת שירות על דומיין קיים - "Authenticated reply attack denial of service"

הפתרון של מנגנון DNSSEC לבעיה נקרא **NSEC**, (קיצור של Next Secure)

או **NSEC3** שהוא וריאנט של **NSEC** ועליהם יוסבר להלן.

1. רשומות NSEC

רשומות מסוג NSEC מתעדות את כל הדומיינים הקיימים ב- zone החתום.

כל רשומה כזאת מחזיקה שלושה נתונים:

- שמו של דומיין שאליו היא מתייחסת
- סוגי הרשומות שיש לאותו דומיין ב- zone המדובר (למשל A, SOA, MX, וכו')
- שמו של הדומיין הבא ב- zone, בסדר אלפביתי.

לדוגמא, כך ייראו רשומות NSEC ב-zone של example.com שהוצג קודם לכן:

```
example.com.        300  IN   NSEC     ftp.example.com.        SOA NS MX DNSKEY RRSIG NSEC
ftp.example.com.    300  IN   NSEC     sub.example.com.        A RRSIG NSEC
sub.example.com.    300  IN   NSEC     www.example.com.        NS DS RRSIG
www.example.com.    300  IN   NSEC     example.com.            A RRSIG NSEC
```

- השדה השמאלי מציין את שם הדומיין שאליו מתייחסים
- השדה הימני הוא סוגי הרשומות שיש לדומיין הזה
- השדה השני מימין הוא שמו של הדומיין הבא (בסדר אלפביתי)

כיצד זה פותר את בעיית אי הקיום?

כמו שניתן לראות בדוגמא, רשומות NSEC הן למעשה שרשור של שמות הדומיינים ב- zone, בסדר אלפביתי (האחרון מצביע שוב על הראשון). כאשר יש שאילתה לגבי דומיין לא קיים, למשל, מהי כתובת ה- IP של he.example.com ? שרת ה- DNS יחזיר כתשובה את רשומת ה-NSEC הבאה:

```
ftp.example.com.    300  IN   NSEC     sub.example.com.        A RRSIG NSEC
```

משמעות התשובה הזו היא: ביקשת את he.exmple.com אבל אין כזה דומיין.

יש ftp.example.com ואחריו את sub.example.com ביניהם אין שום דבר אחר. במילים אחרות, הדומיין he.example.com לא קיים. כמו כל רשומה, גם התשובה הזאת חתומה וניתנת לאימות.

באופן דומה, אם הלקוח ישאל האם יש רשומת MX לדומיין sub.example.com התשובה שתוחזר תהיה:

```
sub.example.com.    300  IN   NSEC     www.example.com.        NS DS RRSIG
```

ומשמעותה: הדומיין sub.exmaple.com קיים אבל יש לו רק רשומות מסוג DS, RRSIG ו- NS. אין MX.

תופעת לוואי של NSEC היא יכולת של צד הלקוח לנחש מה התכולה של ה- zone.
תהליך שנקרא "zone walking" הוא ביצוע הרבה שאילתות, ובהתאם לתשובות ה- NSEC המתקבלות, לצבור מספיק מידע המאפשר להרכיב תמונה מלאה של ה- zone.
על פניו זאת לא אמורה להיות בעיה כיוון שנתוני DNS הם כידוע מידע פומבי. יחד עם זאת תכולה של zone בכללותו היא במקרים רבים רגישה, ובעל המידע אינו מעוניין להנגישה לכל העולם. בעיה נוספת, רלוונטית יותר למרחבי שמות גדולים מאוד, דוגמת co.il היא העובדה שלכל סאב דומיין, גם אם הוא לא חתום יש להצמיד רשומת NSEC. זה מגדיל את ה- zone. לא ניתן לעשות "opt-out" לדלגציות שאינן חתומות.

2. רשומות NSEC3

כמו NSEC רק שבמקום לשמור ולהחזיר שמות אמיתיים של דומיינים, שומרים ומחזירים את ערך ה- HASH שלהם.
כזכור, Hash function (פונקצית גיבוב) היא פונקציה חד כיוונית. היא מפעילה אלגוריתם שהתוצר שלו הוא ערך ייחודי לכל קלט שניתן לה.

- הפונקציה תחזיר ערך שונה לכל קלט שונה.
- לא ניתן להסיק דבר מהערך שהתקבל, לגבי הקלט הראשוני שניתן לה.

מקובל היום אלגוריתם **SHA2** אם כי גם **SHA1** נמצא בשימוש במקרים מסוימים.

ב- NSEC3 מופעל SHA1 על שמות הדומיינים. התוצר הוא ערך שלא ניתן להסיק ממנו דבר על השם המקורי. שמות הדומיינים ב- example.com אחרי הפעלת פונ' SHA1 יראו כך:

Domain name	Hash value
example.com	97612fdad3f6b17f61d6912994b66ea7b7bacc99
ftp.example.com	268cf36e335c84c5b08310ef65471e66feda7349
sub.example.com	1fc71ee46c1fbf928fc30b0f2ae52463ee0fe3cb
www.example.com	b2e6b3cf1af3417e10ea73ab781c1dc687ed0c4b

ולכן, בהתאם לרשימה הנ"ל, שרשרת NSEC3 תראה כך:

```
1FC71....example.com. 300 IN NSEC3 1 0 0 -   268CF…   A RRSIG NSEC
268CF....example.com. 300 IN NSEC3 1 0 0 -   97612…   NS DS RRSIG
97612....example.com. 300 IN NSEC3 1 0 0 -   B2E6B…   SOA NS MX DNSKEY RRSIG NSEC
B2E6B....example.com. 300 IN NSEC3 1 0 0 -   1FC71…    A RRSIG NSEC
```

הפעם אין שמות אלא ערכי HASH. כמו כן המיון הוא בהתאם ל- HASH ולא לערך המקורי.

- השדה השמאלי הוא ה- HASH של הדומיין שאליו מתייחסים
- השדה החמישי משמאל מחזיק את הספרה 1 שמציינת כי הפונ' היא SHA1
- השדה השישי משמאל מציין האם יש opt-out לדלגציות שאינן חתומות. כלומר שלא תהיה רשומת NSEC3 עבור סאב-דומיין שאינו חתום (כפי שהוזכר לעיל, זה מיועד למרחב שמות גדול מאוד). 1 מציין שיש opt-out. ו- 0 שלא.
- השדה השביעי משמאל מחזיק ספרה המציינת כמה פעמים פונקצית ה- HASH תרוץ על הערך המקורי. פעם היה נהוג להפעיל מספר איטרציות של הפונקציה. למשל 10, ע"מ להקשות עוד יותר על נסיונות Dictionary attack. אולם הניסיון מלמד שזה לא רלוונטי, ומקשה על נסיונות האימות.
- השדה השלישי מימין מחזיק ערך "SALT" אם הוחלט לקיימו. זוהי מחרוזת (string) שניתן להוסיף לחישוב שמבצעת פונקצית HASH. גם זה נועד להקשות על Dictionary attack. בדוגמא מצויין "-" כלומר שאין SALT. לאחרונה הוחלט שתוספת זו מיותרת ויש לבטלה.
- השדה השני מימין הוא ה- HASH של הדומיין הבא
- השדה הימני הוא סוג הרשומות שיש לדומיין (כמו ב- NSEC רגיל)

מקובל כיום שעדיף לעבוד עם - NSEC. רק אם הנסיבות מחייבות, להשתמש ב- NSEC3. הסיבה העיקרית היא כוח עיבוד שנדרש בזמן האימות. שני שדות בעייתיים בהגדרת NSEC3: שדה מספר האיטרציות ושדה SALT יש התייחסות לפרמטרים אלה בסעיף הבא, ברשומה המגרידה אותם.

רשומה מסוג NSEC3PARAM

מגדירה את הפרמטרים של NSEC3

Domain name	TTL	Class	Type	Algo	Flags	Iterations	Salt
isoc.org.il.	7200	IN	NSEC3PARAM	1	1	15	7AE165659C6281B26

- השדה השמאלי - שם הדומיין
- שדה **Algo** מציין את סוג ה- Hash function של **NSEC3** נכון להיום זה רק **SHA1**
- שדה **Flags** מציין אם לעשות opt-out לדלגציות לא חתומות
- שדה **Iterations** מציין כמה פעמים פונקצית ה-HASH תרוץ על הערך המקורי
- שדה SALT מציין מהי התוספת שיש להוסיף ל- HASH

בדוגמה הנ"ל רשומת **NSEC3PARAM** של דומיין isoc.org.il מגדירה כי במימוש רשומת **NSEC3** יש להתעלם מדלגציות לא חתומות (opt-out), לבצע 15 איטרציות של HASH ולהוסיף לחישוב ה-HASH את הערך 7AE165659C6281B26.

כאמור, מקובל כיום כי עדיף להשתמש ב- NSEC ולא ב- NSEC3.

אם הנסיבות מחייבות בכל זאת להשתמש ב- NSEC3 (למשל כאשר רוצים למנוע zone walking) אזי הפרמטרים שיש לקבוע ב- NSEC3PARAM הם:

Domain name	TTL	Class	Type	Algo	Flags	Iterations	Salt
isoc.org.il.	7200	IN	NSEC3PARAM	1	0	0	-

כלומר, **ללא איטרציות**, **ללא SALT**, ואם אפשר גם ללא **opt-out**.

יחד עם זאת, כאשר יש הרבה דלגציות ורובן לא חתומות, ייתכן שיהיה נכון להפעיל opt-out.

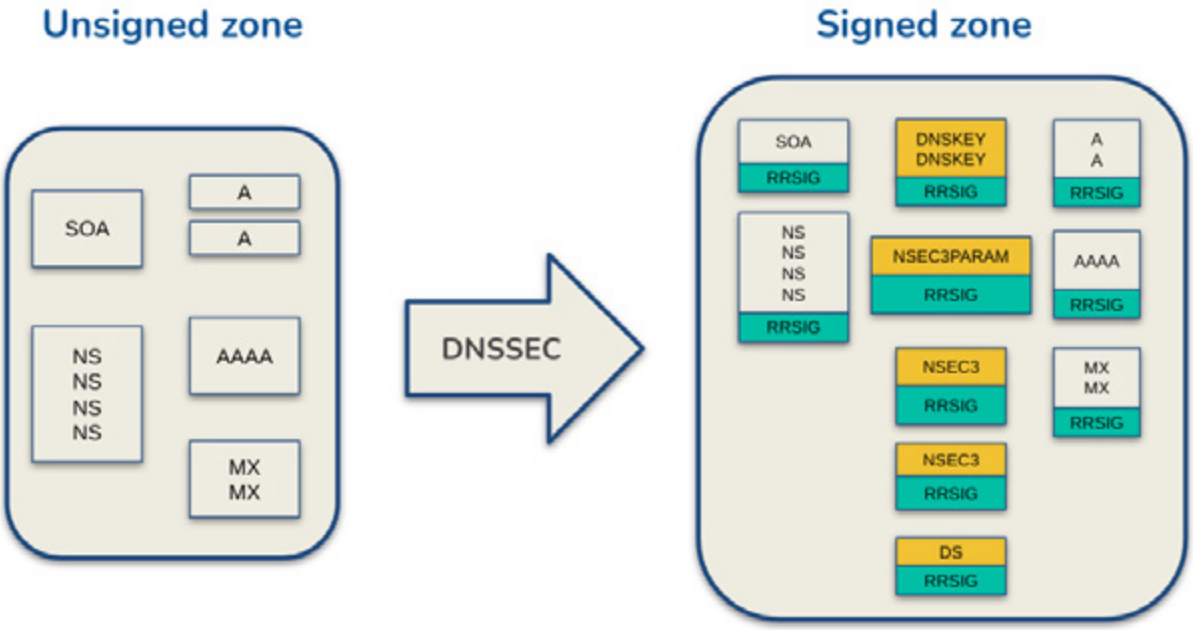
מידע על שיקולים ניתן למצוא כאן:

https://kb.isc.org/docs/dnssec-signed-zones-best-practice-guidance-for-nsec3-iterations

וכאן:

https://www.ietf.org/archive/id/draft-ietf-dnsop-nsec3-guidance-02.html#name-recommendations-for-deployi

לסיכום, לאחר הפעלת DNSSEC, תכולת ה- zone החתום תראה כך:



- לכל RRSET יש חתימה = RRSIG.
- רשומות NSEC או NSEC3 אינן RRSET אחד. כל אחת נחתמת בנפרד. כזכור, כל רשומה כזו שייכת ל- sub domain אחר (שיוך רשומות לאותו RRSET מתרחש כאשר כולן אותו TYPE ואותו שם דומיין).
- אם יש NSEC3 אז קיימת גם NSEC3PARAM + חתימה שלה.
- ולבסוף אם יש דלגציה (הפניה ל- NS) ל-zone מתחתינו, שגם הוא חתום, אז יש להציב רשומת DS שמצביעה על ה- KSK שלהם. גם כאן, לכל דלגציה רשומת DS ייחודית (רשומות DS אינן RRSET אחד) ולידה החתימה.

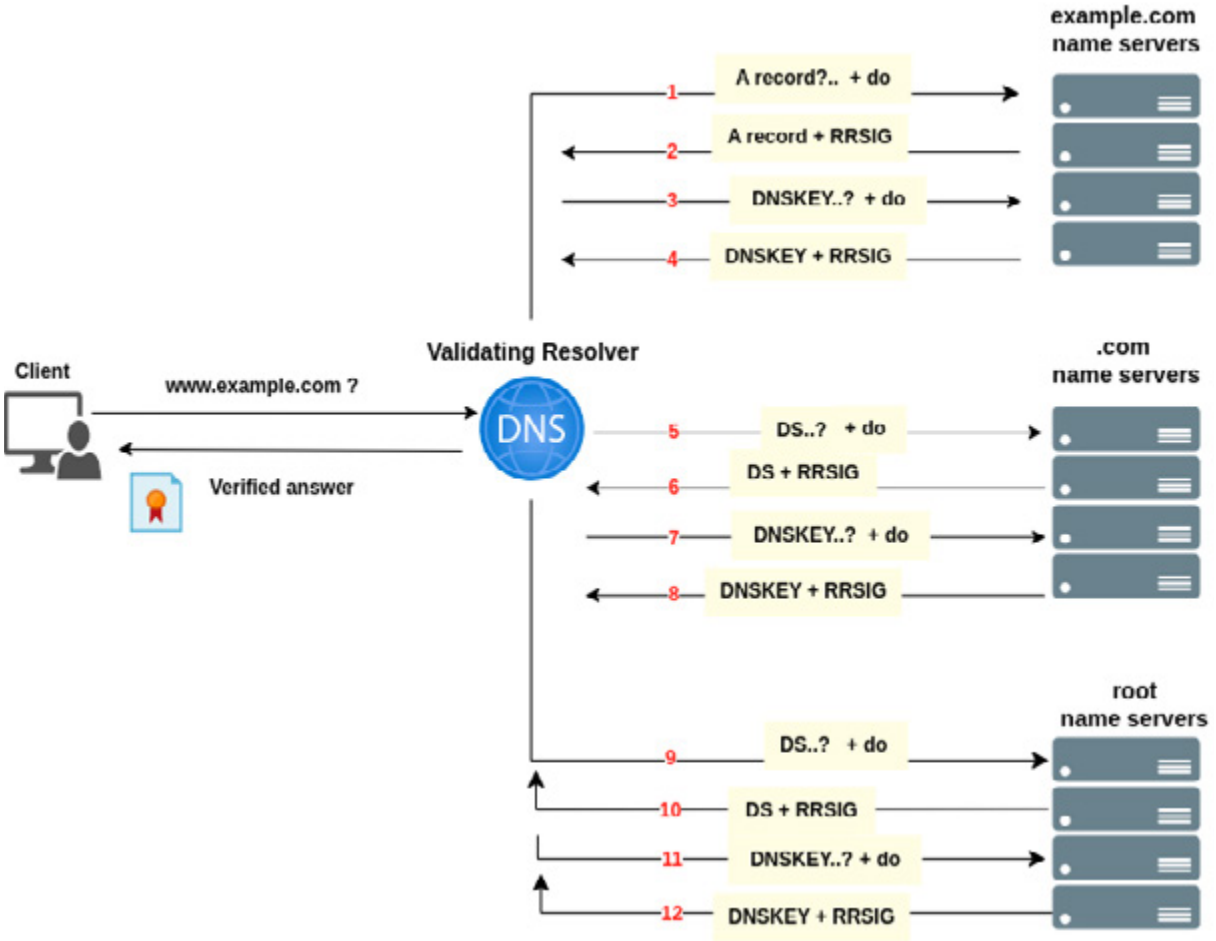
אימות המידע - DNSSEC בצד המשתמש

בסעיף "עקרונות DNSSEC" הוזכר תהליך האימות. שם צוין כי הוא מתבצע ב"צד הלקוח".
DNS בצד הלקוח הוא ה- DNS של ספק האינטרנט, או ה- DNS הארגוני שדרכו הלקוח גולש.
נפוצים מאוד גם שירותי DNS ציבוריים דוגמת 8.8.8.8 של גוגל, 1.1.1.1 של Cloudflare וכו',
כל אלה נקראים שרתי DNS "רקורסיביים" או "רזולברים".
בניגוד לשרת "אוטוריטטיבי" שמפרסם את המידע, הרזולבר הוא זה שמחפש את המידע עבור הלקוח ע"י כך שהוא מתשאל שרתים אוטוריטטיביים.
דוגמא: משתמש מעוניין לגלוש לאתר כלשהו, למשל ל- www.example.com

1. הוא מקיש את השם הנ"ל בדפדפן. מערכת ההפעלה אצלו במחשב מנסה לברר מה כתובת ה- IP של הדומיין הנ"ל. אם אין לה את המידע, היא פונה לשרת ה- DNS שמוגדר לה (ה"רזולבר").
2. הרזולבר מקבל את השאילתה: מהי כתובת ה- IP של www.example.com? אם אין לו את המידע ב- cache, הוא מנסה לאתר DNS אוטוריטטיבי שיענה על השאילתה הזו.
3. אם הרזולבר תומך ב- DNSSEC הוא מסמן לשרת ה- DNS האוטוריטטיבי שאותו הוא מתשאל, על יכולתו לאמת חתימות. הוא עושה זאת בכל שאילתה, ע"י הפעלת flag ב- DNS header שנקרא DNSSEC OK = DO.
4. שרת DNS אוטוריטטיבי שמקבל שאילתה עם דגל DO: אם הדומיין המבוקש לא חתום, מתעלם מהדגל הנ"ל ומחזיר תשובה "רגילה" (רשומה עם כתובת IP של הדומיין). אם לעומת זאת הדומיין כן חתום DNSSEC, אזי פרט לתשובה הרגילה, הוא יספק גם את החתימה של הרשומה הנ"ל.
5. לצורך אימות החתימה, הרזולבר יבקש גם את המפתח.
6. כעת הרזולבר מנסה לפענח את החתימה עם המפתח. אם היא תקינה והתוצר זהה לרשומה הגלויה, הוא יבדוק את מהימנות המפתחות (בדיקת שרשרת אמון): הוא ישווה את ה- KSK אל מול ה- DS שנמצא ברמה מעל, שוב יבצע תהליך אימות (כיוון שגם ה- DS חתום), וחוזר חלילה.
7. מכיוון שכל חתימה וכל מפתח הן רשומות, הכל נכנס ל- cache של הרזולבר. לכן, בשאילתה הבאה לאותו דומיין, תהליך האימות יהיה מיידי, ללא צורך בתשאולים או בדיקות נוספות.
8. רק אם האימות הצליח, הרזולבר מעביר את התשובה ללקוח.

להלן תרשים המתאר תהליך אימות DNSSEC. תחילת התהליך הוא איתור שרת ה- DNS האוטוריטטיבי המפרסם את הדומיין המבוקש (זהו תהליך סטנדרטי ולכן אינו מוצג בתרשים).
כאשר השרת האוטוריטטיבי אותר, הרזולבר שואל אותו: "מהו ה- A record (כתובת IP) של www.example.com", ומוסיף דגל "do" כדי לסמן שהוא תומך DNSSEC (סעיף **1** בתרשים).
השרת האוטוריטטיבי מחזיר בתשובה את הכתובת, ובנוסף את את החתימה **(2)**. כעת הרזולבר

מבקש מפתח ציבורי (ZSK) ע"מ לאמת את החתימה **(3)**. הוא מקבל אותו בצירוף החתימה עליו **(4)**. כעת יש לאמת את ה- KSK שחתם על ה- ZSK מסעיף 2. לכן הוא פונה לרמה מעל (לדומיין .com) ומבקש שם את ה- DS שמתייחס לדומיין המבוקש **(5)** בתשובה הוא מקבל את ה- DS בצירוף חתימה **(6)**. שוב, הרזולבר מבקש מפתח כדי לאמת **(7)** ושוב מקבל אותו בצירוף חתימה **(8)**. התהליך חוזר על עצמו **9,10,11,12** עד לרמת ה-root.



הערות

1. בתרשים הנ"ל מתואר תהליך מלא, **כאשר אין שום מידע ב- cache של הרזולבר**. המפתחות, החתימות וה- DS, נכנסים ל- cache של הרזולבר (כמו כל רשומה). לכן, שאילתה נוספת לדומיין הזה לא תצריך ביצוע התהליך מחדש. הן התשובה והן האימות יתקבלו מיידית.
2. ניתן לראות שתהליך האימות מתבצע מלמטה למעלה, החל מהדומיין החתום, כלפי מעלה, עד ה- root העולמי, כל הרמות משתתפות. כך באה לידי ביטוי שרשרת האמון של DNSSEC המבוססת על המבנה ההיררכי של עץ ה- DNS.
3. בהמשך לסעיף הקודם, כדי שניתן יהיה לחתום על דומיין כלשהו ב- DNSSEC, חובה שכל הרמות שמעליו יהיו חתומות לפני כן.

כל מרחבי שמות המתחם הישראליים (מרחבי IL. ו-.ישראל), כמו גם תתי-המרחבים אשר רשומים בהם - il.co, il.org, il net, il.ac, il.gov, il12.k, il.muni כולם חתומים ולכן מאפשרים לשמות מתחם אשר רשומים בהם לממש DNSSEC.

הפעלת ולידציה ב- DNS בצד הלקוח (ברזולבר)

הפעלת ולידציה ברזולבר היא פעולה פשוטה.

בדרך כלל שרת BIND עדכני מותקן עם אימות DNSSEC מופעל כברירת מחדל.

אם לא, תמיד ניתן לוודא כי בקובץ named.conf בתוך "options" יש את הערך הבא:

```
dnssec-validation auto;
```

לאחר עדכון קובץ conf יש לטעון את השינוי ע"י הרצת:

```
rndc reconfig
```

כמתואר כאן: https://kb.isc.org/docs/aa-01182

ולידציה בשרת DNS של מיקרוסופט מתוארת כאן:

https://learn.microsoft.com/en-us/windows-server/networking/dns/validate-dnssec-responses

חתימה על המידע - מימוש DNSSEC בצד האוטוריטטיבי

בניגוד לרזולבר המתשאל, שרת אוטוריטטיבי לא מבצע ולידציה. הוא רק מחזיר תשובה לפי המידע שיש לו. למעשה, שרת DNS אוטוריטטיבי הוא גם לא זה שחותם על הרשומות, אלא רק מציג אותן. עבורו- zone חתום כן או לא, לא משנה מבחינת משאבים נדרשים.

כפי שנאמר, המפתחות והחתימות הן רשומות DNS רגילות ולכן מבחינת שרת ה-DNS שמפרסם אותן, אין שום הבדל בהנגשת Zone "רגיל" או Zone חתום. מימוש DNSSEC - הגדרות מדיניות, ניהול מפתחות וביצוע החתימות עצמן כולם נעשים מאחורי הקלעים (בשרת מאסטר אחורי או בשרת חתימות ייעודי שמשתלב בתהליך הפצת ה- zone). ניתן להגדיר תהליך חתימה מתוזמן, שמופעל בקבועי זמן, או לחילופין אוטומטית בכל פעם שה- zone משתנה. יש מגוון גדול של אפשרויות וארכיטקטורות כולן כראות עיניו של ארכיטקט הרשת. לבסוף, התוצר הוא zone חתום, אשר מופץ לסלייבים כמו כל DNS zone רגיל.

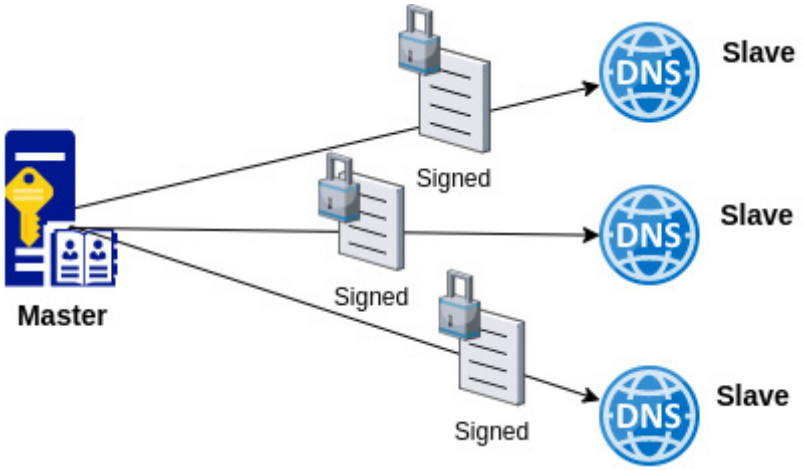
1. הדרך הפשוטה ביותר היא הפעלת שירות DNS מנוהל ע"י ספק דוגמת Cloudflare, Akamai, או ספק ענן. היתרונות ברורים, שירות מנוהל מעביר את הבעיה למישהו אחר. אולם, יש מקרים שבהם היתרון הוא בעצם חיסרון: העברת בעלות על ה- DNS הארגוני לידיים של צד ג' הוא לא נכון לכל ארגון, כיוון שזה מחייב כי גם המאסטר נמצא בשליטתו של נותן השירות. השיקולים בעד ונגד הם כמובן בהתאם לנסיבות כל מקרה לגופו.
2. מוצרים ייעודיים ל-DNSSEC דוגמת infoblox או secure64. אלו appliances אשר מיועדים ספציפית למימוש DNSSEC ופרסום zone חתום. מוצרים אלה מגיעים בד"כ עם תקינה ו- HSM מובנה (הרכיב השומר על המפתחות). יש להם ממשק משתמש נוח, וכל פעולות ניהול ה- zone, החתימה וההפצה נעשות באמצעותו. בנוסף יש ליווי ותמיכה של נותן השירות. החיסרון של appliances כאלה הוא המחיר, ולעיתים גם חוסר גמישות.
3. בדומה לסעיף הקודם, מוצרים שאינם ייעודיים אבל ניתן להשתמש בהם ע"מ לחתום על zone. דוגמת Big-Ip של F5.
4. מימוש DNSSEC באמצעות תוכנת ה- DNS:

תוכנת ה- DNS הנפוצה ביותר - BIND של ISC היא חינמית, תומכת בצורה מלאה ב-DNSSEC (כמו גם תוכנות DNS של יצרנים אחרים דוגמת "KNOT" הידועה בחדשנות והביצועים שלה). יש הרבה מאוד מידע ברשת בכלל ובאתר של ISC בפרט, על הגדרת התצורה של BIND, והפעלת יכולת DNSSEC. תיעוד של גרסת BIND האחרונה - 9.18.14 ניתן למצוא כאן: https://bind9.readthedocs.io/en/v9.18.14/index.html

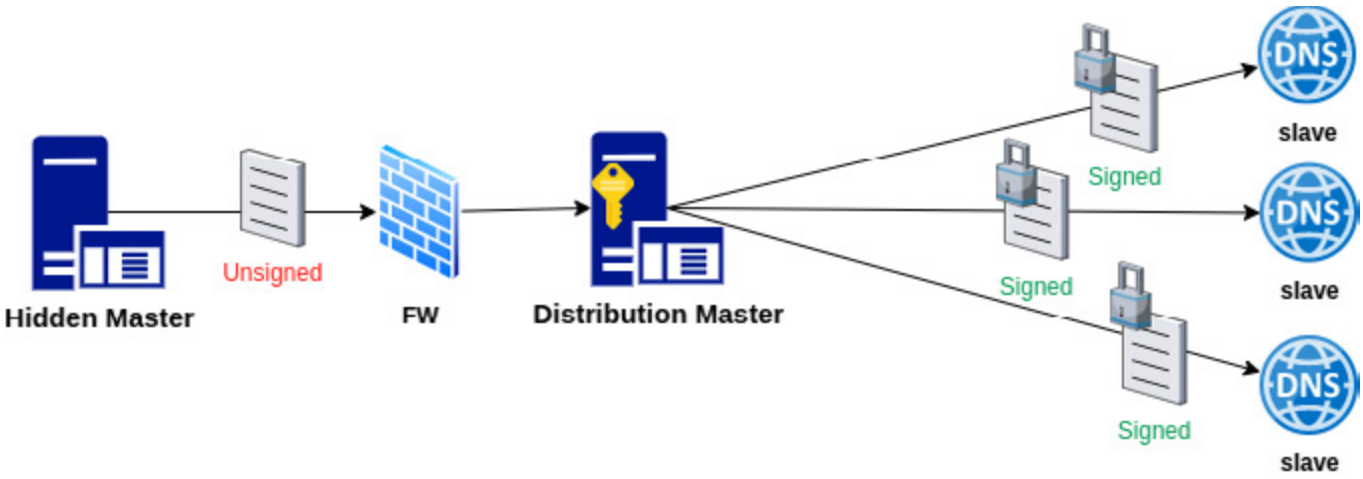
התיעוד כולל גם פרק על DNSSEC, כאן:
https://bind9.readthedocs.io/en/v9.18.14/dnssec-guide.html#

ניתן להפעיל DNSSEC עם BIND במגוון אופנים. נתמקד בשניים המוכרים:

4.1. ישירות על שרת המאסטר. כל שינוי שמתבצע ב- zone, נחתם אוטומטית, ומופץ לשרתי ה- slave מיד אחרי טעינת השינויים (פקודת **rndc reload**).



4.2. מאסטר חבוי. כלל האצבע הוא שעדיף לא לחשוף את המאסטר לעולם. לאחר ביצוע שינוי של zone על המאסטר, הוא מועבר ב- zone transfer לשרת חתימה (שהוא למעשה סלייב, שגם עליו רץ שרת DNS מסוג BIND). עם קבלת הודעת "notify" מהמאסטר, שרת החתימה טוען את ה-zone המעודכן, חותם עליו ומפיץ לסלייבים. ארכיטקטורה זו נקראת "Bump in the wire". ניתן ורצוי כמובן, לשלב FW בין כל רמה בתהליך.



4.3. תצורה של Bump in the wire ללא חשיפה של השרת החותם לעולם:

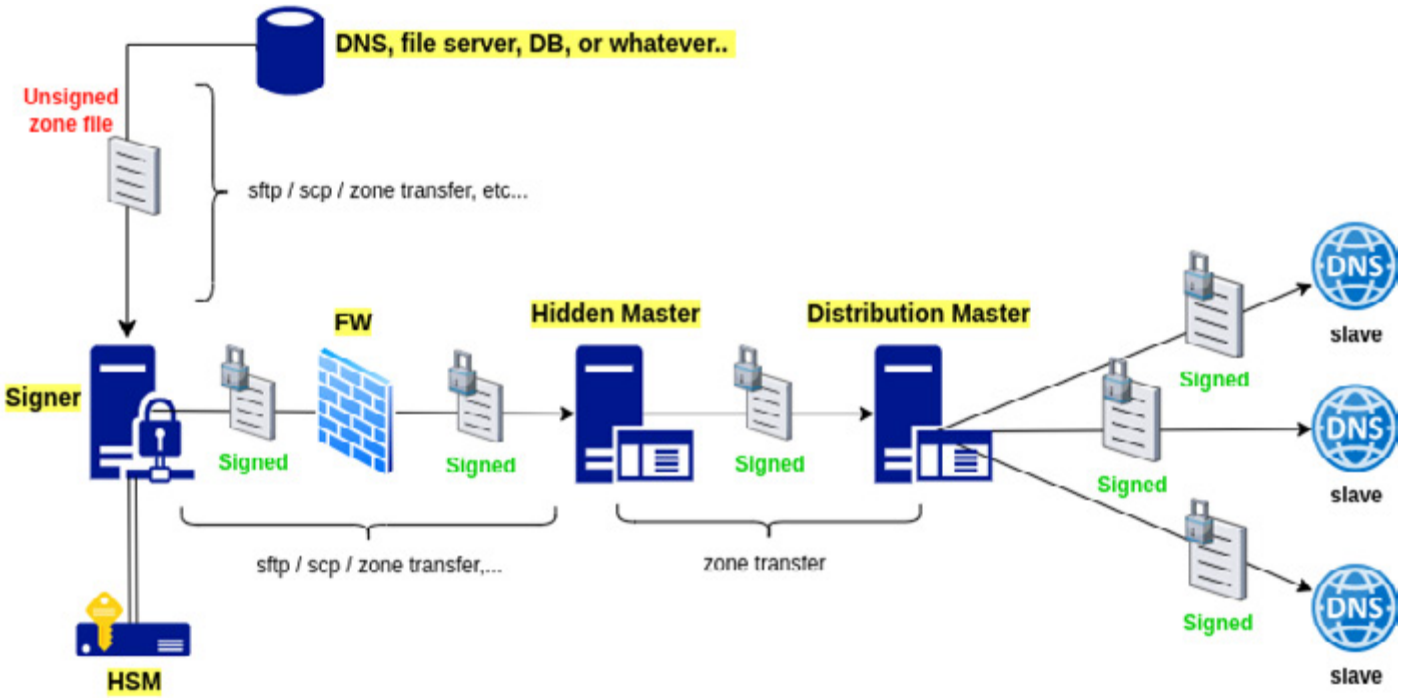
הערה: תוכנת BIND מאפשרת לממש DNSSEC גם עם HSM.

לפי היצרן, BIND תומך ב- API הסטנדרטי ל- HSM, הקרוי PKCS#11, אולם יש וודא תאימות.

קריאה נוספת כאן: https://kb.isc.org/docs/bind-9-pkcs11

5. שימוש בתוכנת חתימה ייעודית. לדוגמא OpenDNSsec. זוהי תוכנה חינמית של שרת חתימות DNSSEC, ללא תלות בסוג שרתי ה- DNS בסביבה. התוכנה מיועדת לפעול מול HSM. בהיעדר HSM אמיתי, ניתן להשתמש באמולציה הנקראת "SoftHSM" (זוהי תוכנה חינמית המדמה HSM אמיתי, כולל גישה דרך api תקני של HSM הקרוי PKCS11. המפתחות נשמרים מוצפנים על הדיסק).

תוכנת OpenDNSsec מאפשרת גמישות מלאה בארכיטקטורה: הפרדה מלאה בין סביבת החתימה לסביבת ה- DNS, ולמעשה מאפשרת לייצר כל ארכיטקטורה העולה על הדעת. לדוגמא, ניתן לשמור ולערוך את קובץ ה- zone טקסטואלית, על שרת קבצים אחורי (zone המכיל אלפי רשומות ניתן אפילו לבנות מתוך DB טבלאי רגיל), ואת התוצר להעביר לשרת החתימה בכל דרך העולה על הדעת: ftp,ssh/scp, באמצעי אוף-לייני נייד או ע"י zone transfer של DNS. לאחר החתימה ניתן להעביר את ה- zone החתום שוב, בכל דרך רצויה. תוך כדי התהליך ניתן ורצוי להוסיף בדיקות מותאמות באמצעות סקריפטים.



הגדרת מדיניות

DNSSEC פועל לפי מדיניות. לכל מוצר יש מדיניות דיפולטיבית, "ישר מהקופסא" אולם נכון לערוך אותה בהתאם לפרמטרים תפעוליים רצויים והסטנדרטים של הארגון.

מדיניות DNSSEC נקראת **KASP** - Key And Signing Policy.

בתוכנת OpenDNSsec זהו קובץ XML ואילו ב- BIND זהו חלק מה-config (החל מגרסא 9.16).

ניתן להגדיר מדיניות אחת או יותר (אם יש הבדלים בין zones שונים, ניתן לייצר לכל אחד מדיניות משלו). במדיניות KASP מוגדרים הפרמטרים שעל פיהם ממומש DNSSEC על השרת.

להלן דוגמא לחלק מהגדרות KASP של תוכנת OpenDNSsec:

1. באיזה אלגוריתם לחולל מפתחות? עדיף כמובן ECDSA, שהוא בעל מפתחות חזקים וקטנים בהרבה לעומת RSA.
2. אילו סוגי מפתחות לייצר? האם KSK + ZSK או שמא מפתח משולב מסוג CSK?

   בעידן שלפני ECDSA, אפשר היה לחולל רק מפתחות RSA גדולים. כיוון שאת ה- KSK נרצה להחליף בתדירות נמוכה ככל שאפשר (כי המשמעות של החלפת KSK היא גם החלפת ה- DS שמצביע עליו מהרמה שמעל), היה נהוג לייצר KSK גדול (באורך של לפחות 2048), ואותו להחליף פעם בשנה. לעומתו ה- ZSK קטן יותר, בגודל - 1024, ואותו התוכנה מחליפה אוטומטית כל 4 חודשים (כזכור חוזק המפתח תלוי במידה רבה באורכו). המוטיבציה של שימוש ב- ZSK קטן היא הרצון לא "לנפח" את גודל התשובה המוחזרת ע"י שרת ה- DNS, תוך החלפתו בתדירות גבוהה יותר.

   בעידן ה- ECDSA ניתן לחולל מפתחות קטנים בהרבה: P-256, שקול ל- RSA בגודל 3072. לכן, ניתן לחולל מפתח אחד -CSK שמשמש גם כ- ZSK וגם KSK ואותו להחליף פעם בשנה.
3. האם גלגול (החלפת) מפתחות מתבצע ידנית או אוטומטית ע"י התוכנה?

   גלגול ידני: נדרש פיזית להקיש את הפקודה המורה על גלגול מפתח. האחריות לתזמן ולבצע החלפת מפתח היא על מנהל המערכת. לחילופין, אם במדיניות מוגדרת החלפה אוטומטית, אזי תוכנת החתימה תגלגל את המפתחות לפי התזמון שנקבע במדיניות (בכל מקרה, תמיד יש לזכור כי אחרי גלגול KSK יש לעדכן DS ברמה מעל!). מקובל לתת את התזמונים הבאים:

- מפתח מסוג RSA באורך 1024 כל רבעון (פעם ב- 3 חודשים).
- מפתח RSA באורך 2048 פעם בשנה. עדיף לא לחולל מפתחות ארוכים מזה, ולהקפיד שרק ה- KSK הוא באורך 2048.
- מפתח ECDSA P-256 ניתן לגלגל פעם בשנה.

4. כמה זמן לאחר גלגול מפתחות יש להמשיך ולפרסם גם את המפתח הישן (במקביל לחדש)? עקרונית התשובה היא ה- TTL של החתימה. כל זמן שיש סיכוי שחתימה עם המפתח הישן עדיין קיימת, **חובה** לפרסם את המפתח המתאים לה. לכן, אם למשל ה- TTL הוא יום, אזי יש להמשיך ולפרסם את המפתח הישן יום נוסף. נהוג להשאיר מפתח ישן יום נוסף לאחר החלפתו גם אם ה-TTL נמוך יותר. **כאשר המפתח שמוחלף הוא KSK, יש לזכור לעדכן גם את ה- DS!**
5. חשוב: כל עוד ה-DS הישן ממשיך לפרסם ברמה שמעליו, יש להמשיך ולפרסם את ה-KSK שמתאים לו. רק לאחר מחיקת ה-DS הישן, ניתן למחוק את ה-KSK שמתייחס אליו.
6. מאגר המפתחות שאליו פונים (שם לוגי של המחיצה ב- HSM שבה נמצאים המפתחות).
7. מה אורך החיים של המפתחות (כמה זמן מפתח פעיל עד שהוא מוחלף)?
8. תוקף של חתימה. ניתן כאמור, להגביל את הזמן שבו חתימה היא ולידית, כך שרק בפרק הזמן הזה היא ניתנת לאימות. תוקף של חתימה נדרש במקרים שבהם המידע המקורי משתנה. במקרים כאלה לא נרצה שאפשר יהיה לאמת מידע ישן מדי. זהו בדיוק המצב בחתימה על רשומות DNS. מהו אם כן התוקף הרצוי? תוקף ארוך עלול לאפשר אימות מידע ישן ולא רלוונטי, ואילו תוקף חתימה קצר מדי עלול להוות בעיה במקרה שלא ניתן לפרסם zone מסיבה כלשהי במשך התקופה שהוגדרה (למשל במקרה של כשל כללי במערכת, כזה שמחייב הקמת התשתית מחדש). כאשר לא ניתן להפיץ או לחתום DNSSEC, עלול להיווצר מצב שרשומות ה-DNS הקיימות בעולם (בסלייבים) עברו את התוקף ואינן ניתנות לאימות. זהו מצב חמור שמשמעותו היא הפסקת שירות ה- DNS. מקובל להגדיר תוקף של חודש.
9. מתי לחדש חתימה? (בהמשך לסעיף הקודם) כמה זמן לפני שפג תוקפה של חתימה קיימת, יש לייצר חתימה חדשה? (גם כאשר לא בוצעו שינויים ב- zone שאז מן הסתם הוא חייב להיחתם מחדש). ניתן להגדיר לתוכנה לחתום מחדש גם יום לפני התוקף אבל זהו סיכון. אם לעומת זאת התוקף הוא 30 יום, ומוגדר לחדש אותה 29 יום לפני כן, אזי החתימה תתחדש כל יום, ובכך נאפשר תוקף מלא של החתימה כל יום מחדש. לחילופין אפשר גם להגדיר ערך סתמי כלשהו, ולדרוס אותו ע"י כך שכל יום חותמים ידנית ע"י הרצת פקודה או בסקריפט מתוזמן. גם כאן יתקבלו כל יום חתימות חדשות שתקפות ל- 30 יום הבאים. לכל גישה יש השלכות לכאן ולכאן. הנוהג הוא לחדש חתימה כל יום גם אם התוכן לא השתנה, ובכך לקיים חתימות עם תוקף מלא כל יום מחדש.
10. כל כמה זמן מנוע החתימה "מתעורר" ובודק האם יש zone מעודכן לחתימה. גם ערך זה ניתן למניפולציות. ניתן לתזמן את מנוע החתימה לבדוק כל קבוע זמן, או בדומה לקודם, לבצע ידנית ע"י פקודת חתימה כשידוע שמגיע לשרת zone מעודכן.
11. הגדרה של "הוכחת אי קיום" - "Proof of non-existence". כזכור, DNSSEC חותם על תשובה שלילית באמצעות **NSEC** או **NSEC3**. יש להגדיר במדיניות במי מבין השניים ייעשה שימוש.

מקובל כיום שעדיף **NSEC**, אם אין ברירה ובכל זאת נבחר להשתמש ב- **NSEC3**, אז במדיניות יוגדרו הפרמטרים של **NSEC3**: (שיפורסמו ברשומת **NSEC3PARAM**)

- כמה פעמים להריץ hash על כל שם דומיין?
- האם להוסיף SALT לחישוב?
- האם לעשות opt-out לדלגציות לא חתומות

ההגדרה המומלצת לפרמטרים אלה היא ללא איטרציות, ללא SALT ואם אפשר ללא opt-out.

הנקודות שהוצגו לעיל הן חלק מההגדרות האפשריות במדיניות של OpenDNSsec.
המדיניות של BIND שונה, גם במבנה וגם בתוכן, אולם המהות דומה.

תיעוד מדיניות OpenDNSsec כאן:
https://wiki.opendnssec.org/display/DOCS20/kasp.xml
וכאן:
https://github.com/opendnssec/odslab/blob/master/opendnssec-lab.md
תיעוד מדיניות BIND ניתן למצוא כאן:
https://kb.isc.org/docs/dnssec-key-and-signing-policy
וכאן:
https://www.sidn.nl/en/modern-internet-standards/dnssec-signatures-in-bind-named
מדריך DNSSEC של BIND:
https://bind9.readthedocs.io/en/v9.18.14/dnssec-guide.html#dnssec-guide

דוגמא למדיניות אפשרית של תוכנת OpenDNSsec מוצגת בעמוד הבא. הקטעים במדיניות ממוספרים, לכל קטע הסבר.
מפאת חוסר מקום הדוגמא מחולקת, אולם מדובר כמובן בקובץ XML אחד.

1. שם המדיניות
2. הגדרות חתימה:

- תג **Resign** = כל כמה זמן השירות מתעורר ובודק אם יש zone לחתימה
- תג **Refresh** = כמה זמן לפני התוקף יש לחתום מחדש (גם אם ה-zone לא השתנה)
- תג **validity** = תוקף חתימה (31 יום = P31D)

הערה: **resign** ו- **refresh** הם ערכים משמעותיים כאשר התהליך הוא אוטומטי. המנוע של OpenDNSSec "יתעורר" לפי קבועי הזמן המוגדרים בערכים הללו. בשורה התחתונה, אם יש קובץ zone על השרת , אזי התוכנה תתנהג לפי הערכים הללו. לחילופין, אם החתימה על ה- zone נעשית בצורה ידנית ע"י הרצת פקודת חתימה (דוגמא בהמשך), או ע"י סקריפט חיצוני, אז ערכים אלה חסרי משמעות (כמו בדוגמא).

3. תצורת **NSEC**. כאן מוגדר: **NSEC3** עם **opt-out**, ללא איטרציות נוספות וללא **SALT**

```
l version="1.0" encoding="UTF-8"?>

P>

        <Policy name="MyPolicy">
         <Description>policy for my domain</Description>
         <Signatures>
                 <Resign>P7D</Resign>
                 <Refresh>P30D</Refresh>
                 <Validity>
                         <Default>P31D</Default>
                         <Denial>P31D</Denial>
                 </Validity>
                 <Jitter>PT0S</Jitter>
                 <InceptionOffset>PT3600S</InceptionOffset>
         </Signatures>

         <Denial>
                 <NSEC3>
                         <OptOut/>
                         <Resalt>P100D</Resalt>
                         <Hash>
                                 <Algorithm>1</Algorithm>
                                 <Iterations>0</Iterations>
                                 <Salt length="0"/>
                         </Hash>
                 </NSEC3>
         </Denial>
```

1
2
3

4. מפתחות: כמה זמן אחרי גלגול מפתח אפשר לפרסם את המפתח החדש, וכמה זמן אחרי הגלגול להשאיר את הישן. התג "**SharedKeys**" מציין שניתן להשתמש באותו מפתח ע"מ לחתום על zone-ים שונים שמנוהלים ע"י המדיניות הזו (וכך לחסוך בכמות המפתחות על ה- HSM).
5. הגדרות KSK: אלגוריתם ECDSA = 13, אורך מפתח 256, תוחלת- שנה (P1Y=), והיכן נשמר (המיקום הלוגי "**MyHSM**" , יוסבר בפרק הבא).
התג "**ManualRollover**", אם היה מופעל משמעותו: לא לגלגל את המפתח בצורה אוטומטית, גם אם המועד הגיע, אלא רק ע"י פקודה ידנית מפורשת.
6. כנ"ל הגדרות ZSK, במקרה זה מתחלף כל 120 יום.
7. הגדרות רשומת SOA: רשומה זו מחזיקה את המספר הסיריאלי של ה-zone. כיוון שפעולת חתימה משנה את ה- zone, יש לעדכן את המספר הסידורי. בדוגמא מוצגת החלופה "keep" שמשמעותה: עדכון המספר הסיריאלי לא ייעשה ע"י OpenDNSsec. זה נעשה במקום אחר (וכמובן לפני החתימה, אחר כך אסור לשנות דבר!).

```
<Keys>
        <!-- Parameters for both KSK and ZSK -->
        <TTL>PT3600S</TTL>
        <RetireSafety>PT86400S</RetireSafety>
        <PublishSafety>PT86400S</PublishSafety>
        <ShareKeys/>
        <Purge>P14D</Purge>

        <!-- Parameters for KSK only -->
        <KSK>
                <Algorithm length="256">13</Algorithm>
                <Lifetime>P1Y</Lifetime>
                <Repository>MyHSM</Repository>
                <!-- ManualRollover/ -->
        </KSK>

        <!-- Parameters for ZSK only -->
        <ZSK>
                <Algorithm length="256">13</Algorithm>
                <Lifetime>P120D</Lifetime>
                <Repository>MyHSM</Repository>
        </ZSK>
                 </Keys>

<Zone>
        <PropagationDelay>PT7200S</PropagationDelay>
        <SOA>
                <TTL>PT86400S</TTL>
                <Minimum>PT86400S</Minimum>
                <Serial>keep</Serial>
        </SOA>
</Zone>
```

4 5 6 7

בשרת חתימה מבוסס BIND קובץ המדיניות שונה, אולם המהות דומה. להלן דוגמא ל-KASP שאפשר להגדיר בשרת מסוג BIND (מתוך הקישור שניתן קודם לכן):

https://www.sidn.nl/en/modern-internet-standards/dnssec-signatures-in-bind-named

```
dnssec-policy "dnssec-policy-1" {

  keys {
    ksk key-directory lifetime P1Y algorithm 8 4096;
    zsk key-directory lifetime P30D algorithm 8 2048;
    }

  purge-keys P90D;

  publish-safety PT2H;
  retire-safety PT2H;

  nsec3param iterations 0 optout false salt-length 0;

  signatures-validity P14D;
  signatures-validity-dnskey P14D;
  signatures-refresh P5D;

  max-zone-ttl P1D;
  zone-propagation-delay PT5M;

  dnskey-ttl PT1H;

  parent-ds-ttl P1D;
  parent-registration-delay
  parent-propagation-delay PT1H;

  }
```

כאן ניתן לראות שהמפתחות הם מסוג **RSA** (אלגוריתם 8), ה- KSK ענק באורך 4096, מתחלף פעם בשנה. ה-ZSK גם גדול מאוד ומתחלף כל חודש. שניהם נשמרים במחיצה מקומית על השרת. הגדרות **NSEC3**: ללא איטרציות נוספות, ללא **SALT** וללא **opt-out**. תוקף חתימה 14 יום. כיוון ששרת BIND מנהל את ה-zone, אזי שינוי המספר הסידורי נעשה על ידו בכל מקרה.

הגדרות כלליות

הגדרת מיקום קובץ KASP, תצורת החיבור ל- HSM (בדומה ל- connection string ל- DB), מיקום מפתחות, מיקום לוג, רמת התיעוד בלוג, ועוד שלל הגדרות נמצאות בקובץ קונפיגורציה כללי. גם כאן יש הבדלים בין מוצר אחד לשני, סוג ההגדרות משתנה וכמובן הפורמט. בשרת OpenDNSsec זהו קובץ שנקרא conf.xml, תיעוד ניתן למצוא כאן:

https://wiki.opendnssec.org/display/DOCS20/conf.xml

בעמוד הבא מובא קטע מתוך קובץ ה- conf אפשרי של תוכנת OpenDNSsec. שם ניתן לראות:

1. **הגדרת החיבור ל- HSM**: כל HSM ניתן לחלק למחיצות - partitions כך שכל מחיצה היא מאגר מפתחות מופרד. ניתן להגדיר מחיצה אחת ולהחזיק בה את כל המפתחות או לחילופין מספר מחיצות כך שיהיה בידול בין המפתחות (למשל כשרוצים להפריד בין סביבת ייצור לטסט או כאשר יש מספר לקוחות ונדרשת הפרדה ביניהם).
   תוכנת OpenDNSSec מאפשרת להגדיר חיבור למספר מאגרים. כל מאגר יכול להיות רכיב HSM פיזי שונה, או מחיצה על אותו HSM. בדוגמא למטה מוגדר חיבור ל- HSM בשם הלוגי "**MyHSM**". הגישה אליו היא באמצעות הדרייבר המוגדר בתגית "**Module**".
   על ה- HSM הזה יש מחיצה בשם "**HSM_PAR_PROD**", והגישה אליה היא באמצעות הסיסמא המוגדרת בתגית "**PIN**".
   תגית "**SkipPublicKey**" מציינת שבמחיצה יישמרו רק מפתחות פרטיים.
2. מיקום הלוג ורמת פירוט: הלוג נשלח ל-syslog, ברמת פירוט 3.
3. הגדרות "**Enforcer**" (ה- service האחראי על אכיפת מדיניות ופעילות שוטפת).
   מהם שם המשתמש והקבוצה שמריצים אותו, היכן ה- PID, המיקום של ה- DB של תוכנת החתימה, ועוד. בדוגמא כאן, ה- DB הוא מסוג SQLite במיקום הנתון. ניתן להגדיר בסיס נתונים אחר כגון MySql, והגדרות נוספות כמפורט בתיעוד.
4. הגדרות "**Signer**" (ה- service האחראי על ביצוע החתימה). שוב, הגדרות משתמש, רמת מיקבול, האזנה לפורט (במקרה שבוחרים לחתום על zone שמגיע ב- zone transfer), ועוד..


```
<?xml version="1.0" encoding="UTF-8"?>

<Configuration>

        <RepositoryList>

                <Repository name="MyHSM">
                        <Module>/usr/lib/libMyHSM.so</Module>
                        <TokenLabel>HSM_PAR_PROD</TokenLabel>
                        <PIN>1234</PIN>
                        <SkipPublicKey/>
                </Repository>
        </RepositoryList>

        <Common>
                <Logging>
                        <Verbosity>3</Verbosity>
                        <Syslog><Facility>local0</Facility></Syslog>
                </Logging>

                <PolicyFile>/etc/opendnssec/kasp.xml</PolicyFile>
                <ZoneListFile>/etc/opendnssec/zonelist.xml</ZoneListFile>
        </Common>

        <Enforcer>
                <Privileges>
                        <User>opendnssec</User>
                        <Group>opendnssec</Group>
                </Privileges>
                <PidFile>/var/run/opendnssec/enforcerd.pid</PidFile>
                <Datastore><SQLite>/var/opendnssec/kasp.db</SQLite></Datastore>
                <ManualKeyGeneration/>
                <RolloverNotification>P1M</RolloverNotification>
                <WorkingDirectory>/var/opendnssec/enforcer</WorkingDirectory>
        </Enforcer>

        <Signer>
                <Privileges>
                        <User>opendnssec</User>
                        <Group>opendnssec</Group>
                </Privileges>
                <!-- <PidFile>/var/run/opendnssec/signerd.pid</PidFile>
                <SocketFile>/var/run/opendnssec/engine.sock</SocketFile> -->
                <WorkingDirectory>/var/opendnssec/signer</WorkingDirectory>
                <WorkerThreads>4</WorkerThreads>

<!--             <Listener>
                        <Interface><Port>53</Port></Interface>
                </Listener> -->

                <!-- the <NotifyCommmand> will expand the following variables:

                     %zone      the name of the zone that was signed
                     %zonefile  the filename of the signed zone          -->

<!--                <NotifyCommand>/usr/local/bin/my_nameserver_reload_command</NotifyCommand> -->
<!--                <NotifyCommand>/usr/sbin/rndc reload %zone</NotifyCommand>                  -->
        </Signer>
```

חילול מפתחות וביצוע חתימה

» כמה מפתחות יש לייצר והיכן ימוקמו? נהוג לחולל מפתחות פעם אחת באירוע שנקרא "טקס חילול מפתחות". זהו אירוע שבו מתכנסים אנשי התשתית האחראים על המערכת, יחד עם אנשי רגולציה ומינהל, המתעדים אותו. במהלך הטקס מריצים את הפקודות המורות על ייצור מאגר מפתחות למשך תקופה. חישוב סך המפתחות הדרושים ייעשה ע"י התוכנה בהתאם למה שמוגדר במדיניות. אם למשל ברצוננו לחולל מאגר מפתחות ל- 10 שנים הבאות. אזי סך המפתחות הדרושים יהיה 40, כיוון שבמדיניות מוגדר שבכל שנה נדרשים 3 מפתחות מסוג ZSK ומפתח אחד מסוג KSK. סה"כ 4 מפתחות לשנה כפול 10.

כאמור, במהלך הטקס, צוות התפעול מריץ את הפקודה המורה על חילול כמות מפתחות לתקופה הרצויה (דוגמא בהמשך).

מבחינת תוכנת החתימה, המיקום בו נשמרים המפתחות שחוללנו, הוא השם הלוגי שהוגדר ב- conf בפרק הקודם, שם מוגדרים הפרמטרים הדרושים לגשת למחיצה הרלוונטית ב- HSM.

בכל פעם שהתוכנה מבצעת פעולת המצריכה גישה למפתח היא פונה למאגר. פעולות הדורשות גישה ל- HSM הן למשל פעולת חתימה או גלגול (החלפת) מפתח. במקרה של גלגול מפתח מתבצעת גישה למאגר, בחירת מפתח חדש, והוא זה שישמש את המערכת לחתימה בפרק הזמן שהוגדר במדיניות, כך "עד הגלגול הבא..".

» חתימה על zones שונים: האם לחולל מאגר מפתחות נפרד לכל zone, או להשתמש באותו מאגר מפתחות לכולם? המשמעות היא מצד אחד פחות מפתחות לחולל ולשמור, פחות "רעש" תפעולי. מצד שני החלפת המפתחות תיעשה באותו זמן לכל ה- zones. כל מקרה לגופו, יש גמישות מלאה להפעיל כמה מאגרים או אחד.

בעמוד הבא מובאת דוגמא למהלך טקס חילול מפתחות באמצעות OpenDNSsec.

הרצת הפקודות נעשית לאחר התקנת התוכנה, הגדרת user בשם "opendnssec", הגדרת קובץ KASP וקובץ conf, ובנוסף לאחר התקנת SQLlite המשמש את התוכנה. מיותר לציין שגם ה- HSM מאותחל ומוכן לשימוש.

1. בדיקת קישוריות ל- HSM

```
[root@signer]# ods-hsmutil test MyHSM
```

2. אתחול ה- DB של התוכנה. יש לשים לב לבצע עם המשתמש "opendnssec"

```
[root@signer1]# su -s /bin/bash -c /sbin/ods-enforcer-db-setup
opendnssec
```

3. הפעלת enforcer service ו- signer service

```
[root@signer]# ods-control start
Starting enforcer...
OpenDNSSEC key and signing policy enforcer version 2.1.7
Engine running.
Starting signer engine...
OpenDNSSEC signer engine version 2.1.7
Engine running.
```

4. יבוא מדיניות לתוך ה- DB. אם יש יותר ממדיניות אחת ב- KASP, הפקודה תציג את כולן ותייבא אותן לבסיס הנתונים

```
[root@signer]# ods-enforcer policy import
Created policy MyPolicy successfully
```

5. הבאת zone לחתימה. בדוגמא להלן נעתיק ב- SCP את הקובץ משרת "backend" למחיצה על שרת signer. כאמור, ניתן גם ב- zone transfer או כל דרך העולה על הדעת. אם פועלים עם קבצים יש לוודא כי ה- owner של הספריה הוא המשתמש opendnssec

```
[root@signer]# scp root@backend:/var/zones/example.com.zone /
var/opendnssec/unsigned/example.com.zone
```

6. הכנסת ה- zone למערכת: קריאה ל- enforcer להוסיף zone בשם example.com,
ולהחיל עליו את המדיניות "MyPolicy".
הקלט של הפקודה הוא קובץ zone במיקום מהסעיף הקודם, והפלט החתום גם הוא יהיה קובץ
במיקום השני (ספרית signed, עם owner שהוא המשתמש "opendnssec")

```
[root@signer]# ods-enforcer zone add -z example.com -p MyPolicy
--in-type file --input /var/opendnssec/unsigned/example.com.zone
--out-type file --output /var/opendnssec/signed/example.com.
zone.signed
input is set to /var/opendnssec/unsigned/example.com.zone.
output is set to /var/opendnssec/signed/example.com.zone.signed.
Zone il added successfully
```

אם יש zone נוסף, יש להריץ את הפקודה שוב עם הפרטים הרלוונטיים (שם zone, ומדיניות)

7. וידוא ה- zone נטען למערכת

```
[root@signer]# ods-enforcer zone list
Zones:
Zone:    Policy:    Next change:      Signer Configuration:
example.com  MyPolicy   Thu Dec 19 12:00:40 2023   /var/opendnssec/signconf/example.com.xml
```

8. חילול מאגר מפתחות ל- 10 שנים

```
[root@signer]# ods-enforcer key generate -d P10Y -p MyPolicy
```

9. בדיקת מפתחות כפי שהם ב- DB של המערכת

```
[root@signer]# ods-enforcer key list -v
```

כפי שהם ב- HSM:

```
[root@signer]# ods-hsmutil list MyHSM
```

10. התוכנה אמורה לחתום מיד, וה-zone החתום ימצא בספריה שהוגדרה בסעיף 6

פעילות שוטפת

1. פקודת חתימה ידנית

```
[root@signer]# ods-signer sign example.com
```

המשמעות של פקודה זו היא ביצוע חתימה על ה- zone example.com לפי המדיניות שהוגדרה לו. כל הרשומות ייחתמו מחדש באמצעות המפתחות הקיימים.

2. הצגת המפתחות המנוהלים במערכת

```
[root@signer]# ods-enforcer key list -v
Keys:
Zone:        Keytype: State:   Date of next transition:Size: Algorithm: CKA_ID:          Repository:   KeyTag:
example.com  KSK      active   2023-09-21 16:06:33     256  13        9f59cc70b1bca4b3b3MyHSM        54814
example.com  ZSK      active   2023-09-21 16:06:33     256  13        b9b3af94bfcd1e8343MyHSM        13275
```

3. הצגת מועדים מתוכננים לגלגול מפתחות ע"י התוכנה

```
[root@signer]# ods-enforcer rollover list
Keys:
Zone:                   Keytype:       Rollover expected:
example.com               KSK        2024-01-24 15:06:33
example.com               ZSK        2023-09-21 16:06:33
```

4. הצגת ה- zones המנוהלים במערכת

```
[root@signer]# ods-enforcer zone list
Database set to: /var/opendnssec/kasp.db
Zones:
Zone:         Policy:            Next change:                  Signer Configuration:
example.com   MyPolicy     Thu Sep 21 16:06:33 2023     /var/opendnssec/signconf/example.com.xml
example.org   MyPolicy     Thu Sep 21 16:06:31 2023     /var/opendnssec/signconf/example.org.xml
```

5. גלגול ידני של מפתח

אם מסיבה כלשהי נדרש להחליף מפתח, לא לפי התזמון המתוכנן בתוכנת החתימה בדוגמא להלן גלגלול ZSK:

```
[root@signer]# ods-enforcer key rollover -z example.com -t ZSK
```

6. ייצור רשומות DS

6.1. אחרי כל גלגול של KSK יש לייצר רשומת DS מתאימה למפתח החדש:

```
[root@signer]# ods-enforcer key export -z example.com -e publish -t ksk -d
```

ולהעביר את התוצר שהתקבל, שהוא משהו כזה:

לגורם המתפעל את הרמה מעל.

במקרה של דומיין ישראלי, למשל x.co.il שנמצא תחת co.il יש להעביר את ה- DS למרשם הישראלי, המנוהל ע"י איגוד האינטרנט הישראלי. זה נעשה באמצעות הרשם המתפעל את הדומיין.

```
example.com.    3600    IN    DS    61239 13 2 38C5B2DF51289DF7CE29CA33…
```

6.2. כעת יש ליידע את התוכנה, שה- DS החדש מתפרסם ברמה מעל. המשמעות היא שניתן להפסיק לחתום עם ה- KSK הישן

```
[root@signer]# ods-enforcer key ds-seen -z example.com -keytag 61239
```

6.3. לבסוף יש ליידע את התוכנה לאחר שה- DS הישן הוסר מהרמה מעל.

יש להקפיד להסיר רשומות DS ישנות!

```
[root@signer]# ods-enforcer key ds-gone -z example.com -keytag 54814
```

חלופה אפשרית לתוכנת OpenDNSSec היא כאמור שימוש בשרת ה- DNS עצמו, למשל BIND. תיעוד ניתן למצוא כאן:

https://bind9.readthedocs.io/en/v9.18.14/dnssec-guide.html#enabling-automated-dnssec-zone-maintenance-and-key-generation

מתוך התיעוד הנ"ל, חתימה על zone באמצעות BIND, נעשית בשרת DNS אחורי שמחזיק את ה-zone המדובר (למשל שרת המאסטר).

יש להגדיר בקובץ ההגדרות named.conf משהו כזה:

```
options {
    directory "/etc/bind";
    recursion no;
    ...
};

zone "example.com" in {
    ...
    dnssec-policy default;
    inline-signing yes;
    ...
};
```

התג inline-signing מגדיר: כל שינוי ב- zone ייחתם מיידית.

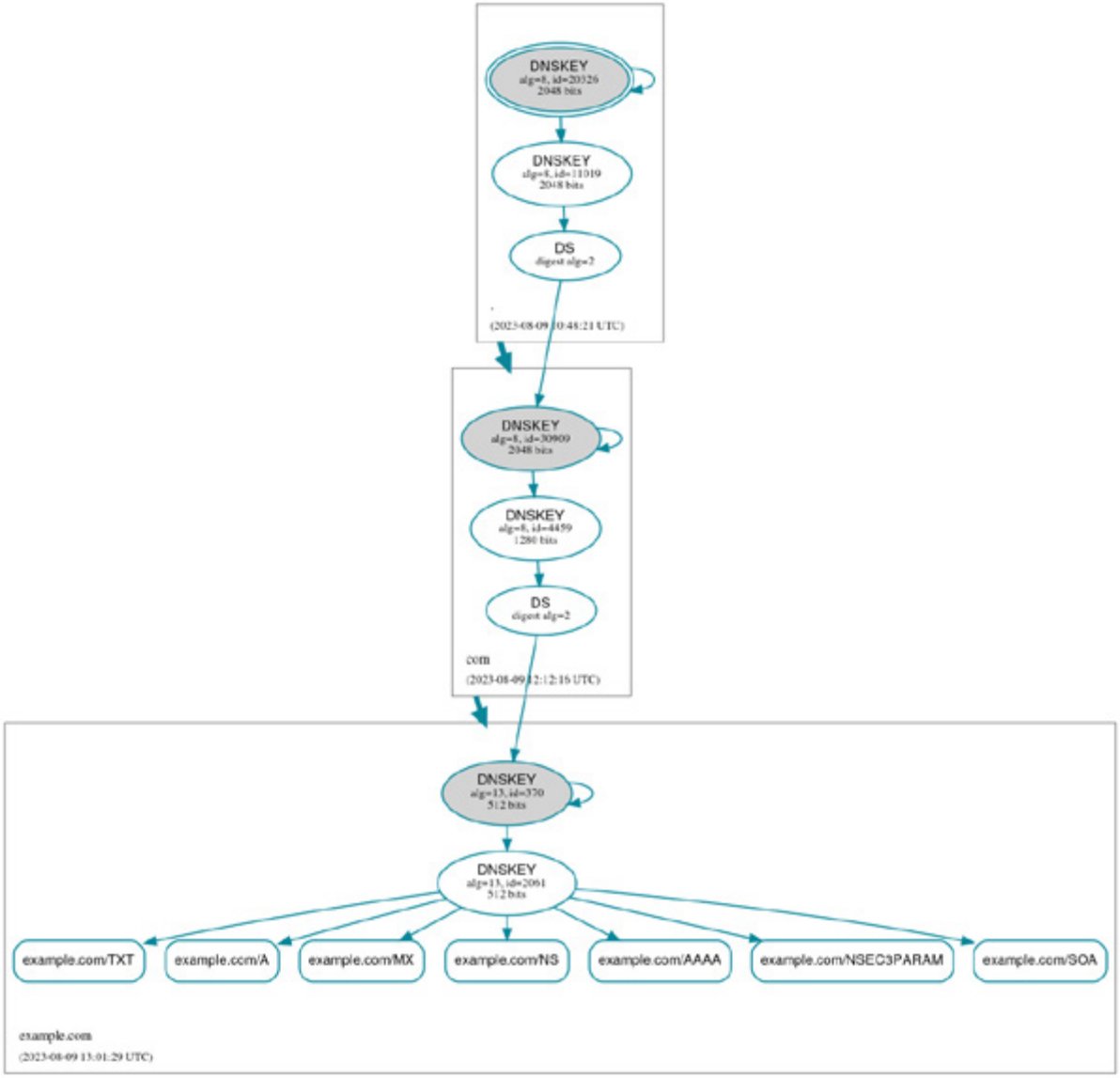
החתימה תיעשה בהתאם להגדרות במדיניות הדיפולטיבית שמגיעה עם השרת. ניתן כמובן להגדיר ידנית מדיניות נוספת כפי שהוזכר קודם לכן .

מיקום ההגדרות הללו *בתוך* המקטע המגדיר את ה- zone הספציפי (במקרה זה example.com) משמעותו שרק הוא ייחתם. אפשר למקם את התגים האלה במקטע options הכללי, ולהחיל את החתימה על כל ה- zones המנוהלים ע"י השרת.

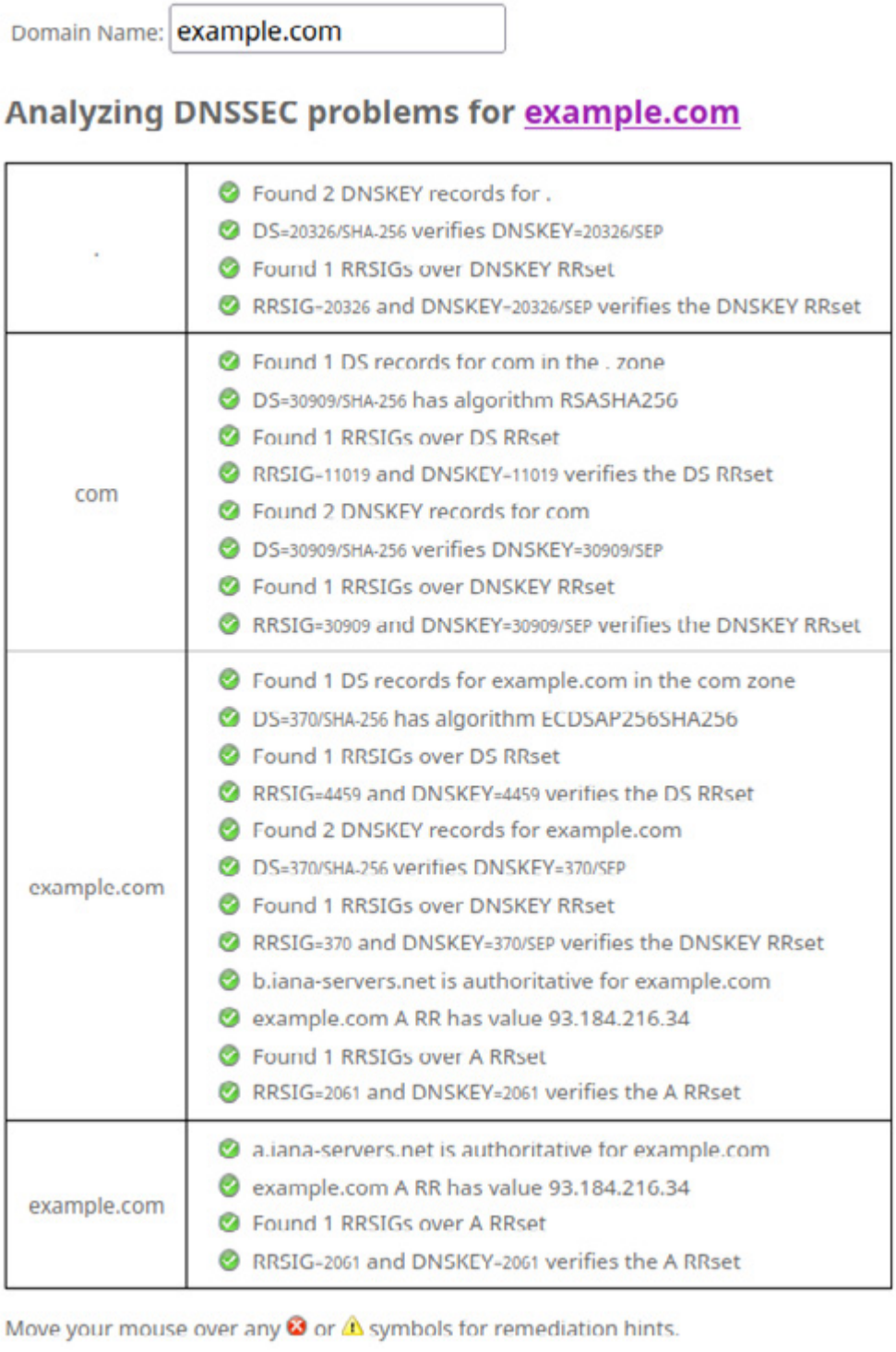
בדיקות

אלמנט חשוב במימוש DNSSEC הוא ביצוע בדיקות תקינות. קיימים אתרי אינטרנט המאפשרים ויזואליזציה של התוצאה. שני האתרים המובילים הם:

1. https://dnsviz.net/ המציג את התוצאות בצורת עץ עם מקרא. אם יש בעיות כלשהן הן יסומנו באדום וצהוב (לפי החומרה):



2. https://dnssec-analyzer.verisignlabs.com/ המציג את התוצאה בצורה טבלאית. גם כאן בעיות מודגשות בצבע. ניתן להגדיל את רמת הפירוט המוצגת בטבלה:


Domain Name: example.com

Analyzing DNSSEC problems for example.com

.	Found 2 DNSKEY records for . DS=20326/SHA-256 verifies DNSKEY=20326/SEP Found 1 RRSIGs over DNSKEY RRset RRSIG-20326 and DNSKEY-20326/SEP verifies the DNSKEY RRset
com	Found 1 DS records for com in the . zone DS=30909/SHA-256 has algorithm RSASHA256 Found 1 RRSIGs over DS RRset RRSIG-11019 and DNSKEY-11019 verifies the DS RRset Found 2 DNSKEY records for com DS=30909/SHA-256 verifies DNSKEY=30909/SEP Found 1 RRSIGs over DNSKEY RRset RRSIG=30909 and DNSKEY=30909/SEP verifies the DNSKEY RRset
example.com	Found 1 DS records for example.com in the com zone DS=370/SHA-256 has algorithm ECDSAP256SHA256 Found 1 RRSIGs over DS RRset RRSIG=4459 and DNSKEY=4459 verifies the DS RRset Found 2 DNSKEY records for example.com DS=370/SHA-256 verifies DNSKEY=370/SEP Found 1 RRSIGs over DNSKEY RRset RRSIG=370 and DNSKEY=370/SEP verifies the DNSKEY RRset b.iana-servers.net is authoritative for example.com example.com A RR has value 93.184.216.34 Found 1 RRSIGs over A RRset RRSIG=2061 and DNSKEY=2061 verifies the A RRset
example.com	a.iana-servers.net is authoritative for example.com example.com A RR has value 93.184.216.34 Found 1 RRSIGs over A RRset RRSIG-2061 and DNSKEY-2061 verifies the A RRset

Move your mouse over any ⊗ or ⚠ symbols for remediation hints.

בדיקות DNSSEC ב- command line

סט פקודות "dig" הן המקובלות, הנוחות והנפוצות ביותר. ניתן להתקין dig על כל מעה"פ.

סט פקודות "drill" מיועד ספציפית לבדיקות DNSSEC ומיועד רק ל- Linux.

במערכת חלונות ניתן גם לבצע באמצעות Powershell.

1. תשאול רזולבר, למשל את Cloudflare (בכתובת 1.1.1.1) מהי כתובת ה- IP של דומיין isoc.org.il? כיוון שהרזולבר הזה מבצע ולידציה של DNSSEC הוא מציין שהתשובה מאומתת, ע"י הוספת דגל "ad" שמשמעותו "Authenticated Domain":

```
~ dig @1.1.1.1 isoc.org.il

; <<>> DiG 9.18.17 <<>> @1.1.1.1 isoc.org.il
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12473
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;isoc.org.il.                   IN      A

;; ANSWER SECTION:
isoc.org.il.            261     IN      A       192.115.211.55
```

2. אם נרצה לראות את החתימה עצמה, נוסיף +dnssec:

```
~ dig @1.1.1.1 isoc.org.il +dnssec

>> DiG 9.18.17 <<>> @1.1.1.1 isoc.org.il +dnssec
 server found)
lobal options: +cmd
ot answer:
>>HEADER<<- opcode: QUERY, status: NOERROR, id: 31841
lags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

PT PSEUDOSECTION:
NS: version: 0, flags: do; udp: 1232
UESTION SECTION:
c.org.il.                   IN      A

NSWER SECTION:
.org.il.            300     IN      A       192.115.211.55
.org.il.            300     IN      RRSIG   A 13 3 300 20230903175011 20230813165011 18460 isoc.org.il. a4cEHkq8190K11w
```

3. אם נרצה לראות את המפתחות נבקש "dnskey" ובנוסף נציין +multi ע"מ לקבל את המידע בצורה קריאה:

```
meir@007 ~ dig @1.1.1.1 dnskey isoc.org.il +multi

; <<>> DiG 9.18.17 <<>> @1.1.1.1 dnskey isoc.org.il +multi
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26283
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;isoc.org.il.           IN DNSKEY

;; ANSWER SECTION:
isoc.org.il.           3472 IN DNSKEY 256 3 13 (
                              mW6EF30JCft2c1HXiyOOFzZ/wXPszXnIchJyiEe6nLy1
                              3NnZkCxqNdYvN7buuNV9rWom7KtxsE2UCexMT8w59Q==
                              ) ; ZSK; alg = ECDSAP256SHA256 ; key id = 18460
isoc.org.il.           3472 IN DNSKEY 257 3 13 (
                              0uIaVQeiX5wtvq9yEAiJjBI58qATI56NfG2+2/QAn6d/
                              kOy0zsuEAY5lUevNtBuPzxIWytyM/M32ve2H2RAP9w==
                              ) ; KSK; alg = ECDSAP256SHA256 ; key id = 27123
```

4. ושוב, אם נרצה לראות גם את החתימה של המפתחות נוסיף +dnssec:

```
meir@007 ~ dig @1.1.1.1 dnskey isoc.org.il +multi +dnssec

; <<>> DiG 9.18.17 <<>> @1.1.1.1 dnskey isoc.org.il +multi +dnssec
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46328
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;isoc.org.il.           IN DNSKEY

;; ANSWER SECTION:
isoc.org.il.           3217 IN DNSKEY 256 3 13 (
                              mW6EF30JCft2c1HXiyOOFzZ/wXPszXnIchJyiEe6nLy1
                              3NnZkCxqNdYvN7buuNV9rWom7KtxsE2UCexMT8w59Q==
                              ) ; ZSK; alg = ECDSAP256SHA256 ; key id = 18460
isoc.org.il.           3217 IN DNSKEY 257 3 13 (
                              0uIaVQeiX5wtvq9yEAiJjBI58qATI56NfG2+2/QAn6d/
                              kOy0zsuEAY5lUevNtBuPzxIWytyM/M32ve2H2RAP9w==
                              ) ; KSK; alg = ECDSAP256SHA256 ; key id = 27123
isoc.org.il.           3217 IN RRSIG DNSKEY 13 3 3600 (
                              20230904082151 20230814072151 27123 isoc.org.il.
                              ZbiMY4nv5lJGkH2x61zTGFzmh+qd330ubNs9+EagT99v
                              sd3y/hPnX+PjfGXjBtCgQKvrPoUy1SZWs6ELA8C3aw== )
```

ZSK

KSK

חתימה

ניתן לראות, על המפתחות חותם ה- KSK, מס' 27123, כמצופה.

5. כאשר יש בעיה באימות DNSSEC התשובה המוחזרת היא "SERVFAIL":

```
~ dig @1.1.1.1 dnssec-failed.org

; <<>> DiG 9.18.17 <<>> @1.1.1.1 dnssec-failed.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 61670
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; EDE: 9 (DNSKEY Missing): (no SEP matching the DS found for dnssec-failed.org.)
;; QUESTION SECTION:
;dnssec-failed.org.             IN      A
```

6. ניתן לוודא שזו אכן בעיית DNSSEC ע"י הוספת דגל +cd (קיצור של "Checking Disabled").
זוהי הוראה לא לנסות לאמת את החתימה

```
~ dig @1.1.1.1 dnssec-failed.org +cd

; <<>> DiG 9.18.17 <<>> @1.1.1.1 dnssec-failed.org +cd
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48516
;; flags: qr rd ra cd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; EDE: 9 (DNSKEY Missing): (no SEP matching the DS found for dnssec-failed.org.)
;; QUESTION SECTION:
;dnssec-failed.org.             IN      A

;; ANSWER SECTION:
dnssec-failed.org.      300     IN      A       96.99.227.255
```

מאיר קראוסהר, dnssec@isoc.org.il

איגוד האינטרנט הישראלי ספטמבר 2024

איגוד האינטרנט הישראלי ISOC-IL

.il

.ישראל