# NFC - קצת תיעוד

## crc.py

בקובץ הזה נשתמש בחישוב CRC לDATA שנרצה לשלוח לרכיב NFC (כמובן שאין שום תלות בין קובץ הCRC לNFC, והוא גנרי).

יוצרים אובייקט CRC ועליו מפעילים את המתודה העיקרית (compute) שמחזירה את הCRC.

מבחינת הקוד עצמו, מצאתי באינטרנט מה הלוגיקה שצריכה לקרות בחישוב CRC בהינתן כל הפרמטרים הרלוונטיים (דיי מקביל לשימוש בחבילה crcmod).

החישוב יודע להתייחס לערך התחלתי של החישוב, פולינום, שיקוף ביטים, חישוב xor סופי ואינדיאניות...

כרגע תומך בעיקר בCRC16 ולכן לא מספיק גנרי (בשינויים מינימליים אפשר לתמוך בעוד גרסאות).

## crc\_brute\_force.py

* מוכן להרצה

בקובץ הזה יש מנייה על ההודעה שנתונה לנו בצורה חלקית (אבל CRC שלם) ולכן אפשר למנות על כלל האופציות (חסרים 2 בתים, , 65536 אופציות) להודעה, עד שמגיעים להודעה שהCRC שלה, שווה לCRC הנתון.

ההודעה החלקית שיש לנו מתחילה בקוד 1b שלפי התיעוד זהו הקוד לPWD\_AUTH, ז.א לאחר הצלחת המנייה יהיה לנו את הסיסמא של הרכיב ואז נוכל לפנות לרכיב גם עם פקודות שדורשות אימות קודם (לדוגמא, קריאה של תוכן מוגן בסיסמא).

## nfc\_sender.py

* מוכן להרצה

זהו הקובץ העיקרי, מממש מחלקה של NFC בעזרתה נוכל לתקשר עם הרכיב.

בבנאי נגדיר את כל הפרמטרים של התקשורת ולאחר מכן נוכל להפעיל מתודות לפי הפקודות האפשריות בפרק 10 [כאן](https://www.nxp.com/docs/en/data-sheet/NTAG213_215_216.pdf) ולפי הפקודה הנוספת שנתונה בתרגיל.

בקצרה, לכל פקודה מוגדר לו מה הmsg code הרלוונטי ואיזה פרמטרים הוא צריך (או יכול) לקבל (לדוגמא, ניתן לעשות לפני כן התאמתות עם סיסמא, אבל לא תמיד הכרחי).

אפשר לקרוא תוכן, לכתוב, לקרוא קונפיגורציה (ברמות פירוט שונות) ועוד...

הפונקציה הרלוונטית לפתרון התרגיל (כל השאר נמצאות תחת תיעוד) היא read\_protected שמשיגה מהקונפיגורציה של הרכיב, את הpage שהחל ממנו התוכן מוגן על ידי סיסמא וקוראת את התוכן (לאחר התאמתות עם הסיסמא מהמנייה) החל מאותה נקודה עד הסוף (במקרה שלנו, 4 pageים == 16 בתים כמו שנתון בתרגיל לאורך הסיסמא לבונקר).

* לא הרצתי ובדקתי את הפונקציה של כתיבה לרכיב כי לא ידעתי אם לגיטימי, ואם עוד אנשים עובדים במקביל (ולכן יש עליה סימן תיעוד כפול).

## nfc\_consts.py

קובץ קבועים לnfc\_sender.py.

מצורף לרפו גם הPDF הרלוונטי למחקר.