

Capire gli ECM/EMM v4.00 Add-on II : raccolta di vario materiale

From FAQ v3.02

Sono state osservate, ma **non confermate**, strane conseguenze sul contenuto della card a seguito dell'invio di INS con zero dati per il decrypt:

- Alterazione dell'area record (creazione record anomali e presunto funny-bug 2)
- Status 9028 ottenuti con ECM (a cadenza "casuale")
- Alterazione della struttura provider
- Alterazione dell'ATR (ultimi 4 byte)

From Tirannia Digitale (Bramino)

Al fine di comprendere il meccanismo che sta alla base di tutto il discorso è opportuno riprendere il significato del bug 38/36.

Nel seka 1 la C1 38 richiedeva la presenza di pseudonimi in posizioni fisse:

C1 38 P1 P2 1A **16** D1 **2A** D2 D3 **2B** D4 D5 **86** x1 x2 x3 x4 x5 x6 x7 x8 82 signa

Con il seka 2 non è più necessario che i nomi 16 / 2B / 86 siano presenti effettivamente (vengono dati per scontati) e vi ricordo che il corpo di un C1 38 può essere trasformato in C1 40 ed eseguito come tale..

La coppia di ins C1 38/36 ha molte somiglianze con la C1 34/32 entrambe infatti possono chiedere record alla carta e detti record vengono restituiti dalla carta come echo alla successiva C1 36 o 32.

A differenza della C1 34 però la C1 38 può essere inviata criptata e inizializzata con P1 come le C1 40 e richiede la signature.

Il parametro D1 è equivalente al primo byte di richiesta dati del C1 34, mentre i D2 D3 al secondo e terzo

Come se fosse: C1 34 00 00 03 D1 D2 D3

Ne consegue che:

code:

D1	D2	D3	
0	0	0	Provider package bitmap record
1	0	0	Provider PPV credits record
3	x	x	Provider PPV event record
4	0	x	Seka record
6	x	x	Given record (number)

La costruzione di un C1 38 ottenuto da un C1 40 valido sarà di questo tipo:

C1 38 P1 P2 len P3 + dati p3 + byte in Se + 90 byte in sse

Supponiamo che il comando decrittato sia questo **(attenzione noi non vediamo il comando in chiaro)**

C1 38 P1 P2 LEN 71 89 01 02 03 04 05 06 43 01 3e 00 00 e4 d3 d4 21 11 22 33 44
55 66 77 88 82 s1 s2 s3 s4 s5 s6 s7 s8 + byte completamento LEN

Nel comando sopracitato:

71 = P3
89 01 02 03 04 05 06 = byte dati di p3 (89=p4)
01 = D1
00 00 = D2 D3
d3 d4 = D4 D5

Questo comando quindi richiede un provider ppv credit record

Il C1 36 seguente al C1 38 assume un significato analogo al C1 32 che segue un C1 34 con delle particolarità.

Nella sua formulazione il C136 deve avere:

P1= con nibble basso uguale a quello del C1 38 e nibble alto uguale a 2 0 3 (ad esempio 21 o 31)

P2= indicante key e tabelle algo da utilizzare in particolare se il bit 7 di p2 è settato viene richiesta una risposta crittata se non è settato viene richiesta una risposta in chiaro

Ad esempio : Se chiediamo la risposta in chiaro potremmo ottenere:

C1 36 21 00

36 (ACK)

86 11 22 33 44 55 66 77 88 (dove 11 22 33 44 55 66 77 88 = decrypt dei byte dal 10° al 17° dopo dati p3)

84 2f ee 00 aa bb d0 13 00 (dove 2f ee 00 aa bb d0 13 00 = ppv record)

04

82 ff fino a LEN

Se chiediamo la risposta criptata potremmo ottenere:

C1 36 21 90 LEN

36 (ACK)

1C (len dei comandi restituiti)

7b 31 aa 00 23 11 d4 d3 23

f3 7b ad 4e 16 c7 1c cd a9

b8

82 21 33 45 32 45 64 34 90 (dove 21 33 45 32 45 64 34 90 = signa)

Se ne deduce che

7b 31 aa 00 23 11 d4 d3 23 f3 7b ad 4e 16 c7 1c cd a9 b8

è il cript di

86 11 22 33 44 55 66 77 88 84 2f ee 00 aa bb d0 13 00 04

INS LUNGHE, METODI PER OTTENERLE E PUNTI DI INGRESSO

Abbiamo capito perciò fino adesso che per ottenere in risposta dalla 36 delle ins lunghe entra in gioco il parametro D1 e D2 e D3 ne determineranno i record letti, ovviamente vi ricordo che anche se si trovassero le condizioni ideali per avere delle risposte lunghe ma richiederete l'ins 36 con una len non sufficiente a contenerne la risposta otterrete una "normale" ins corta.

Detto questo dobbiamo capire insomma come far assumere a D1 il valore da noi desiderato.

Per far ciò sarà sufficiente prendere un ins corta ed eseguirvi un brute sui byte post signa e non sotto SSE, così facendo l'ins cambierà il proprio cript e prima o poi incontreremo un D1 "adatto" (non tutte le ins devono necessariamente assumere un cript che dia in chiaro un D1 desiderato).

Tutte le ins corte tranne le P3 8x (se noterete il parametro D1 cade sul nano 82) possono restituire un ins lunga, però ognuna di queste, a seconda del P3, ha dei suoi pro e dei suoi contro.

Infatti le corte da 3x a 6x hanno i parametri D1- D2 - D3 sotto SE e non ne conosciamo il valore e questo ci impedirà di conoscerne l'esatto punto d'inizio lettura dei record, ma questo ci permetterà anche di non essere vincolati troppo "dal punto di ingresso" ma di poter spaziare con la stessa ins in diversi modi, leggendo magari prima da un record e poi da un altro ma sempre nella casualità dei cript ottenuti. Mentre le corte con P3 7x presentano 4 parametri (D2 - D3 - D4 - D5) nella signa, e siccome la signa, a differenza del seka 1 viene applicata sul comando criptato e non subisce nessun processo di cript e decrypt, perciò possiamo conoscere a priori il reale valore dei parametri che cadranno nella firma.

Metto una tabellina chiarificatrice:

C1 38 P1 P2 LEN

P3 P4 x0 x1 x2 x3 x4 x5 x6 x7 82 s0 s1 s2 s3 s4 s5 s6 s7 m0 m1 m2 m3 m4 m5 m6 ecc...

P3	d1	d2 d3	d4 d5	b0 b1 b2 b3 b4 b5 b6 b7
1y	x1	x3 x4	x6 x7	s0 s1 s2 s3 s4 s5 s6 s7
2y	x2	x4 x5	x7 82	s1 s2 s3 s4 s5 s6 s7 m0
3y	x3	x5 x6	82 s0	s2 s3 s4 s5 s6 s7 m0 m1
4y	x4	x6 x7	s0 s1	s3 s4 s5 s6 s7 m0 m1 m2
5y	x5	x7 82	s1 s2	s4 s5 s6 s7 m0 m1 m2 m3
6y	x6	82 s0	s2 s3	s5 s6 s7 m0 m1 m2 m3 m4
7y	x7	s0 s1	s3 s4	s6 s7 m0 m1 m2 m3 m4 m5
8y	82	s1 s2	s4 s5	s7 m0 m1 m2 m3 m4 m5 m6

Allora il metodo più semplice e funzionale per ottenere delle ins lunghe è di usare delle ins corte 7x.

Ad esempio

C1 36 21 FE 14 (36) 13 7D E1 93 67 B5 3E 92 F9 A8 72 82 **5A 45** 46 6B 3F 9E 95 B0
[90 00]

Traduciamola in c138

C1 38 21 fe len 7D E1 93 67 B5 3E 92 F9 A8 72 82 **5A 45** 46 6B 3F 9E 95 B0 00 00
00 00 00 00 00 00 00 +[89 x00 + 09]

7D E1 93 67 B5 3E 92 F9 = P3 byte dati

A8 72 = 2 byte liberi "eseguibili" per il decrypt ed essendo meno di 8 scatta il famoso byte/reset bug

Andiamo a vedere il significato dei byte in funzione della C1 38:

[7D E1 93 67 B5 3E 92 F9] vengono saltati come argomento di P3

[A8 72] di questi due valori il secondo (72) rappresenta il cript di D1 poiché avviene il reset bug detto valore cambierà ad ogni reset

Se viene ad assumere un valore pari a i valori compatibili (00 01 03 04 06) al C1 36 successivo otterremo una lunga.

[5A 45] sono i valori di D2 D3 IN CHIARO infatti sono extra envelope ed extra superencryption

[6B 3F] sono i valori di D4 D5 IN CHIARO per lo stesso motivo

[95 B0 00 00 00 00 00 00] sono i byte della finestra del c136

Andando per tentativi incontreremo un D1 "giusto" per leggere lungo e chiediamo perciò il C1 36 con una len sufficiente a contenere una risposta lunga

La risposta lunga si ottiene quando il decrypt variabile del termine 72 (72 è il parametro D1 crittato) assumerà uno dei valori citati sopra

Nel caso di D1 03 (parametro D1 valore 03 non predeterminabile, ma perseguibile a tentativi) il punto di lettura dei record B1 partirà dall'event ID 5A 45 in avanti. Se non si ha un evento con quell'event id si leggerà da quello con event id appena superiore come valore.

Il numero dei record che dumperà dipenderà dai record presenti sulla carta e dalla len della 36 con cui richiederete i dati, se sufficiente a contenerne il numero desiderato.

Prendendo come esempio l'ins 7x di sopra e se sulla carta avessimo i seguenti B1:

code:

```
B1 00 06 B7 FF C5 00 00 FF FF 00 00
B1 00 07 9D FF 89 00 00 FF FF 32 8E
B1 00 08 02 FF F8 00 00 FF FF 32 8E
B1 00 35 90 FF 21 00 00 FF FF 32 8E
B1 00 3E 22 FF 6C 00 00 FF FF 32 8E
B1 00 72 E1 FF C2 00 00 FF FF 32 8E Punto d'ingresso e primo record B1
B1 00 AC 59 FF C7 00 00 FF FF 32 8E
B1 00 B1 93 FF 9F 00 00 FF FF 32 8E
B1 00 B1 CA FF E0 00 00 FF FF 32 8E
B1 00 B2 D7 FF 9A 00 00 FF FF 32 8E
B1 00 B6 CE FF 5C 00 00 FF FF 32 8E
B1 00 B6 CE FF 5C 00 00 FF FF 32 8E
B1 00 ED 7C FF 67 00 00 FF FF 32 8E
B1 00 F7 97 FF 37 00 00 FF FF 32 8E
```

In questo caso se noi richiedessimo la 36 con una len sufficiente a contenere almeno 4 record B1 avremmo una ins 36, in chiaro, di questo tipo:

```
C1 36 P1 P2 LEN 86 95 B0 00 00 00 00 00 00 B1 00 72 E1 FF C2 00 00 FF FF 32 8E
B1 00 AC 59 FF C7 00 00 FF FF 32 8E B1 00 B1 93 FF 9F 00 00 FF FF 32 8E B1 00 B1
CA FF E0 00 00 FF FF 32 8E 03 82 S1 S2 S3 S4 S5 S6 S7 S8
```

Una Ins lunga di 47 H di len interna, composta da 8 ottetti e 6 byte spaiati, di cui possiamo richiederne successivamente il cript per avere i P3 e P4 desiderati, proprio come per le altre INS lunghe.

Ricordando che cambiando il ciclo di reset la stessa ins non darà più l'ins lunga ottenuta prima, pertanto è bene prestare attenzione in questa fase se non si vuole perdere il lavoro fatto fin qui.

Questo era il caso di parametro D1 incontrato = 03

Vediamo invece il caso di parametro D1 = 06 che cosa succederebbe, data sempre un INS 7x, ovviamente con valori D2 e D3 più bassi, perché l'ingresso non sono più gli event Id, bensì i numeri dei record.

C1 36 21 FE 14 (36) 13 7D E1 93 67 B5 3E 92 F9 A8 72 82 **00 13** 46 6B 3F 9E 95 B0
[90 00]

code:

P=01 Rec.0006 [F0] FF FF FF FF FF FF FF FF ED 00 [81] Primary ***
P=01 Rec.0007 [50] FF FF FF FF FF FF FF FF 63 00 [C1] Secondary ***
P=01 Rec.0008 [51] FF FF FF FF FF FF FF FF 4B 00 [81] Primary ***
P=01 Rec.0009 [51] FF FF FF FF FF FF FF FF DF 00 [C1] Secondary ***
P=01 Rec.000A [50] 42 FA 35 7E F3 F8 C7 90 FC E4 [E1] **** PPV record
P=01 Rec.000B [6D] 19 F2 13 2B 06 14 2B 00 00 00 [A1] Preview Record
P=01 Rec.000C [5E] FF FF FF FF FF FF FF FF A6 00 [81] Primary ***

Punto di ingresso e primo record D2 letto:

P=01 Rec.001F [00] E1 72 FF C2 00 00 FF FF 8E 32 [B1] PPV Record

P=01 Rec.0020 [5C] FF FF FF FF FF FF FF FF B4 00 [81] Primary ***
P=01 Rec.0021 [5D] FF FF FF FF FF FF FF FF 89 00 [81] Primary ***
P=01 Rec.0022 [00] 9D 07 FF 89 00 00 FF FF 00 00 [B1] PPV Record
P=01 Rec.0023 [00] 02 08 FF F8 00 00 FF FF 00 00 [B1] PPV Record
P=01 Rec.0024 [00] 90 35 FF 21 00 00 FF FF D4 31 [B1] PPV Record
P=01 Rec.0025 [00] 97 F7 FF 37 00 00 FF FF 8E 32 [B1] PPV Record
P=01 Rec.0026 [00] B7 06 FF C5 00 00 FF FF 00 00 [B1] PPV Record
P=01 Rec.0027 [00] 22 3E FF 6C 00 00 FF FF 00 00 [B1] PPV Record
P=01 Rec.0028 [00] D7 B2 FF 9A 00 00 FF FF 8E 32 [B1] PPV Record
P=01 Rec.0029 [00] CE B6 FF 5C 00 00 FF FF 8E 32 [B1] PPV Record
P=01 Rec.002A [00] CA B1 FF E0 00 00 FF FF 8E 32 [B1] PPV Record
P=01 Rec.002B [00] 93 B1 FF 9F 00 00 FF FF 8E 32 [B1] PPV Record
P=01 Rec.002C [00] 59 AC FF C7 00 00 FF FF 8E 32 [B1] PPV Record

In questo caso se noi richiedessimo la 36 con una len sufficiente a contenere almeno 4 record D2 avremmo una ins 36, in chiaro, di questo tipo:

C1 36 P1 P2 LEN 86 00 00 00 00 00 00 00 00 00 00 D2 00 1F 00 E1 72 FF C2 00 00 FF FF
8E 32 B1 00 00 D2 00 20 5C FF FF FF FF FF FF FF FF B4 00 81 00 00 D2 00 21 5D FF
FF FF FF FF FF FF FF 89 00 81 00 00 D2 00 22 00 9D 07 FF 89 00 00 FF FF 00 00 B1
00 00 03 82 S1 S2 S3 S4 S5 S6 S7 S8

IMPORTANZA DI AVERE UN P3 = 7X e CONSERVAZIONE PLAIN

Sappiamo che con le ins lunghe di certi tipi ed un P3 adeguato possiamo conoscere a priori il nostro comando e cosa eseguiremo. Questo P3 è 7x perché questo ci farà schippare un ottetto intero (P3 + 7 byte dati) e ci permetterà di "conservare" parte della nostra plain di cui conosciamo il valore in chiaro.

Ma cerchiamo anche di capirne il perché succede questo:

Premessa necessaria, la nuova superncript opera sempre per ottetti ma a ritroso. Senza andare nei dettagli diciamo che una ipotesi che rispecchia la realtà sperimentale è la seguente

Immaginiamo un comando di questo tipo in chiaro

A1 A2 A3 A4 A5 A6 A7 A8 B1 B2 B3 B4 B5 B6 B7 B8 C1 C2 C3 C4 C5 C6 C7 C8 D1 D2 D3

Nel cript quello che succede è una cosa di questo tipo

Con il residuo viene calcolata una pseudochiave legata alla somma in modulo 4000 dei byte C1 C2 C3 C4 C5 C6 C7 C8 vengono criptati con un meccanismo stile sekai ed ulteriormente elaborati dalla chiave modulo 4000 ricavata dai residui si ottiene così un ottetto c1c2c3c4c5c6c7c8 criptato

Con c1c2c3c4c5c6c7c8 viene calcolata una pseudochiave legata alla somma in modulo 4000 dei byte B1 B2 B3 B4 B5 B6 B7 B8 vengono criptati con un meccanismo stile sekai ed ulteriormente elaborati dalla chiave modulo 4000 ricavata dall'ottetto criptato precedentemente si ottiene così un ottetto b1 b2 b3 b4 b5 b6 b7 criptato

Con b1 b2 b3 b4 b5 b6 b7 viene calcolata una pseudochiave legata alla somma in modulo 4000 dei byte A1 A2 A3 A4 A5 A6 A7 A8 vengono criptati con un meccanismo stile sekai ed ulteriormente elaborati dalla chiave modulo 4000 ricavata dall'ottetto criptato precedentemente così un ottetto a1 a2 a3 a4 a5 a6 a7 a8 criptato

A questo punto abbiamo finito di trattare gli ottetti a ritroso, ma manca ancora un passaggio :

Con a1 a2 a3 a4 a5 a6 a7 a8 viene calcolata una pseudochiave legata alla somma in modulo 4000 dei byte.

Si prendono a questo punto gli ultimi 8 byte del comando ossia :

c4c5c6c7c8 D1 D2 D3 che vengono con un meccanismo stile sekai ed ulteriormente elaborati dalla chiave modulo 4000 ricavata dall'ottetto criptato precedentemente così un ottetto c4' c5' c6' c7' c8' d1 d2 d3

il cript sarà quindi

a1 a2 a3 a4 a5 a6 a7 a8 b1 b2 b3 b4 b5 b6 b7 c1c2c3 c4' c5' c6' c7' c8' d1 d2 d3

Metto una situazione con un esempio "facilotto" di una ins con + ottetti e cosa accadrebbe con un P3 7x (è un semplice esempio per rendere il concetto):

6 ottetti + byte spaiati (r):

A B C D E F r

Cript:

r genera una chiave che cripterà F

F genera una chiave che cripterà E

E genera una chiave che cripterà D

D genera una chiave che cripterà C

C genera una chiave che cripterà B

B genera una chiave che cripterà A

A genera una chiave che cripterà gli ultimi 8 bytes (r + parte di F)

Decript con P3 7x e perciò ottetto A saltato (il procedimento proseguirà all'inverso):

B genera una chiave che decripterà gli ultimi 8 bytes (r + parte di F)

C genera una chiave che decripterà B

D genera una chiave che decripterà C

E genera una chiave che decripterà D

F genera una chiave che decripterà E

r genera una chiave che decripterà F

Questo vi farà capire che saltando il primo ottetto avremo un sicuro cambiamento sugli ultimi 8 bytes legati appunto ai primi 8 e anche ai penultimi 8 perché legati all'ultimo 8 che è stato modificato in principio, ma vi avrà fatto vedere come i 3 ottetti centrali in questo caso siano stati decriptati dalla stessa chiave e perciò questi conservati e non sotto al cosiddetto "effetto bordo"

TIPI DI INS LUNGHE e EFFETTO BORDO

PARAMETRO D1 = 03 DUMP RECORD B1 (attenzione P3 deve essere tipo 7x)

INS LUNGA 2B H = 2 RECORD B1 = 4 OTTETTI + 2 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 37 H = 3 RECORD B1 = 5 OTTETTI + 6 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 43 H = 4 RECORD B1 = 7 OTTETTI + 2 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 4F H = 5 RECORD B1 = 8 OTTETTI + 6 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 5B H = 6 RECORD B1 = 10 OTTETTI + 2 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 67 H = 7 RECORD B1 = 11 OTTETTI + 6 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 73 H = 8 RECORD B1 = 13 OTTETTI + 2 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 7F H = 9 RECORD B1 = 14 OTTETTI + 6 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

B1 00 EV EV FF BC 00 00 FF FF PP PP

03 82 S1 S2 S3 S4 S5 S6 S7 S8

Con il parametro D1 = 03 si puo continuare con INS + lunghe se si dispone di parecchi record B1, si possono avere i seguenti casi:

- INS LUNGA 8B H = 10 RECORD B1 = 16 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA 97 H = 11 RECORD B1 = 17 OTTETTI + 6 BYTE SPAIATI
- INS LUNGA A3 H = 12 RECORD B1 = 19 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA AF H = 13 RECORD B1 = 20 OTTETTI + 6 BYTE SPAIATI
- INS LUNGA BB H = 14 RECORD B1 = 22 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA C7 H = 15 RECORD B1 = 23 OTTETTI + 6 BYTE SPAIATI
- INS LUNGA D3 H = 16 RECORD B1 = 25 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA DF H = 17 RECORD B1 = 26 OTTETTI + 6 BYTE SPAIATI
- INS LUNGA EB H = 18 RECORD B1 = 28 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA F7 H = 19 RECORD B1 = 29 OTTETTI + 6 BYTE SPAIATI

PARAMETRO D1 = 06 DUMP RECORD D1(attenzione P3 deve essere tipo 7x)

INS LUNGA 35 H = 2 RECORD D1 = 5 OTTETTI + 4 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 46 H = 3 RECORD D1 = 7 OTTETTI + 5 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 57 H = 4 RECORD D1 = 9 OTTETTI + 6 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 68 H = 5 RECORD D1 = 11 OTTETTI + 7 BYTE SPAIATI

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 79 H = 6 RECORD D1 = 14 OTTETTI

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

INS LUNGA 8A H = 7 RECORD D1 = 16 OTTETTI + 1 BYTE SPAIATO

P3 11 22 33 44 55 66 77 **xx**

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

D2 00 NR FF FF FF FF FF FF FF FF FF FF FF TR 00 00

03 82 S1 S2 S3 S4 S5 S6 S7 S8

Con il parametro D1 = 06 si possono ottenere anche i seguenti casi:

- INS LUNGA 9B H = 8 RECORD D1 = 18 OTTETTI + 2 BYTE SPAIATI
- INS LUNGA AC H = 9 RECORD D1 = 20 OTTETTI + 3 BYTE SPAIATI
- INS LUNGA BD H = 10 RECORD D1 = 22 OTTETTI + 4 BYTE SPAIATI
- INS LUNGA CE H = 11 RECORD D1 = 24 OTTETTI + 5 BYTE SPAIATI
- INS LUNGA DF H = 12 RECORD D1 = 26 OTTETTI + 6 BYTE SPAIATI
- INS LUNGA F0 H = 13 RECORD D1 = 28 OTTETTI + 7 BYTE SPAIATI

LEGENDA E NOTE VARIE

XX = Primo byte eseguito, nono byte dell'ins, ultimo byte della finestra dell'ins 36

EV = Byte eventi id di un record B1

BC = Byte Counter di un record B1

PP = Byte PPV spot di un record B1

NR = Numero Record quando si dumpa con D1 = 06

TR = Numero Record quando si dumpa con D1 = 06

Byte in grassetto = Byte predeterminabili dell'ins

Si presuppone ovviamente che i P3 delle ins siano del tipo 7x atti a scavalcare un ottetto completo.

Negli esempi con D1 = 06 i record sono messi tutti a FF, ma dipenderà dal tipo di record dumpato il loro reale valore.

Non si può superare un ins di lunghezza 9B H lunga , perché in fase di costruzione non riusciremmo a piazzare entro 256 byte i 90 bytes in SSE.

I record che si dumpano e l'ordine dipende dai parametri D2 e D3, e nel caso specifico di D1 = 06 sono situazioni molto "personali" che dipendono da carta a carta.

I byte predeterminati sono stati segnati in base al presupposto che si perdono informazioni solamente del primo, degli ultimi 2 ottetti e dei byte spaiati quando ci sono.

Esiste un solo caso dove non vi sono i byte spaiati e la non pre-determinabilità è al minimo di 24 bytes (contando anche il primo ottetto saltato), questo caso è la 79 h (con D1 = 06), infatti questa contiene 14 ottetti completi e zero byte spaiati.

Il massimo di non pre-determinabilità si ha con i casi 68 H e F0 H = 31 bytes non determinabili. L'ins più lunga che si può ottenere anche se non sfruttabile è la F7 h con D1 = 03

TIPS & TRICKS PER "SCRIVERE e AGEVOLARE IL PARSING"

Come si è visto fino adesso ci sono casi dove la plain del comando di partenza in fase di decrypt conserva il valore dei byte che la compone.

Detto questo vediamo dei trucchettini per poter piazzare i nanocomandi voluti, avere un parsing corretto e azzerare i rischi in fase di scrittura.

- Byte XX

Il primo byte "sfruttabile" al 100% è il primo nano eseguito (quello segnato con xx nelle tabelle).

Questo perché quel byte è il nono byte dell'ins nonché l'ultimo della finestrella della 36, perciò settabile a nostro piacimento in fase di ricerca. Perciò questo byte può essere opportunamente settato per scrivere o parsare nella posizione desiderata.

- PPV SPOT

Il ppv spot può essere molto utile anche lui, soprattutto per un motivo, quello che noi possiamo modificarlo a nostro piacimento (richiedendo delle ins lunghe, quando avremo D1 = 03, il ppv spot assume i valori dei parametri D4 e D5). Perciò anche questi byte possono essere sfruttati per scrivere o parsare.

- Record Number e Tipo Record

In caso di ins lunghe con D1 = 06, anche questi valori possono essere usati per agevolare il parsing, ci sono rari casi che assumano valori di nano comandi "sfruttabili" per scrivere.

- Record AX

Questi record possono essere molto utili, essendo dei record che ci possiamo far scrivere in qualsiasi momento e avendo dei byte facilmente variabili. Perciò sfruttabili per avere un parsing corretto o piazzare dei nani desiderati.

- Byte counter del record B1

Non molto versatile come byte, nel senso che quello che si becca si becca, ma delle volte può tornare utile per agevolare il parsing

Nota: Se si costruiscono ins che vanno a parsare dentro nei byte non predeterminabili, si otterranno anche dei 96 00 nella costruzione dei cript e ovviamente si andrà in conto a dei rischi.

Mentre se andremo a parsare prima dell'ultimo byte, anche se in zona non predeterminabile, incontreremo ancora dei 96 00 in fase di ricerca cript, ma i rischi saranno azzerati perché l'unico byte che porterà a buon fine il parsing sarà del tipo 0x.

Mentre per comandi che andranno a parsare in zona determinabile fino al nano 82 si otterrà solo dei 90 00 e azzeramento totale dei rischi anche qui.

(...)

Riporto qui sotto degli spezzoni che contengono dei begli esempi di applicazione del metodo:

Domanda : come fai nella 3C a far cadere inizio dell'argomento del nano D1 proprio sull'ottetto ?

Risposta:

Il nano 04 deve precedere il Nano D1 altrimenti non funziona.

Spieghero' come ho fatto a costruire le C1 3C : mi sono "arrangiato" con quello che avevo, cioe' questa risposta lunga (dvmp con dl=6):

```
86 xx xx xx xx xx xx xx yy
D2 00 21 00 43 7C FF 89 00 00 FF FF 00 00 B3 00 00
D2 00 22 00 00 D2 FF 17 00 00 FF FF 00 00 B3 00 00
D2 00 23 B1 00 21 D1 FF CA 00 5A 24 F3 67 E3 00 00
D2 00 24 A6 FF FF FF FF FF FF FF FF DE 00 C3 00 00
D2 00 25 00 4E 08 FF 0C 00 00 FF FF 00 00 B3 00 00
D2 00 26 04 FF FF FF FF FF FF FF FF 04 00 83 00 00 03 ...
```

- I record 0021, 0022, 0025 sono dei Bx creati "a caso", cioe' senza controllo sui valori di fvfmt-ID e numero visioni

- I record 0024, 0026 sono delle kfy, create con PPV-spot "90 04" e "91 A6"... non chiedetene il valore, non lo conosco

- Il record 0023 e' un record Ex creato "ad hoc" con dentro il Nano D1. Potevo infilarlo in un PPV-spot, ma dato che avevo un evento con un byte di quel valore, perche' non sfruttarlo per generare un record "buono"?

Per lo 0023, in realta', mi sono preparato due comandi di creazione record Ex (nano B0), tramite dvmp con dl=03 (lettura eventi)...

- uno crea un record "B1 00 21 D1 FF CA 00 5A 24 F3 67 E3"

- uno crea un record "B1 00 21 D1 FF CA 00 B8 0D 1B 93 E3"

Trattandosi entrambi di record con primo byte = "B1" i due record si sovrascrivono a vicenda.

A questo punto e' partita la ricerca dei PPV-spot per un parsing "decente", ho trovato questi

```
86 xx xx xx xx xx xx yy
D2 00 21 00 43 7C FF 89 00 00 FF FF D9 CB B3 00 00
D2 00 22 00 00 D2 FF 17 00 00 FF FF 04 9C B3 00 00
D2 00 23 B1 00 21 D1 FF CA 00 5A 24 F3 67 E3 00 00
D2 00 24 A6 FF FF FF FF FF FF FF DE 00 C3 00 00
D2 00 25 00 4E 08 FF 0C 00 00 FF FF 00 00 B3 00 00
D2 00 26 04 FF FF FF FF FF FF FF 04 00 83 00 00 03 ...
```

Se riesco a cadere su "D9" va tutto bene! Il parsing teorico sarebbe:

```
86 xx xx xx xx xx xx yy
D2 00 21 00 43 7C FF 89 00 00 FF FF(D9)CB B3 00 00
D2 00 22 00 00 D2 FF 17 00 00 FF FF(04|9C)B3 00 00
D2 00 23 B1 00 21(D1)FF CA 00 5A 24 F3 67 E3 00 00
D2 00 24 A6 FF FF(FE)FF FF FF FF FF DE 00 C3 00 00
D2 00 25 00 4E 08 FF 0C 00 00 FF FF 00 00 B3 00 00
D2 00 26 04 FF(FE)FF FF FF FF FF FF 04 00 83 00 00 03 ...
```

L'ultimo (FF), quello del record 0026, non e' un FF "vero": cadendo entro i byte della coda assume valore non predicibile.

Come cadere su D9? It's easy, ma non troppo ...

Serve un nano "Cx" iniziale, ma non posso fare lo stesso gioco delle C1 38/40, cioe' di assegnare direttamente tale valore al byte yy della finestra del Nano 86. Infatti, a differenza delle C1 38/40, nelle C1 3C vengono decr|ptati 8 byte in meno (quelli in testa, subito dopo i dati di P3). Allora ho fatto questo:

- a- ho cercato una risposta "lunga" 0x79 (il dvmp dei 6 record)
- b- ho generato 256 risposte lunghe variando 'yy' da 00 a FF (e variando altri byte fuori finestra-nano 86, per mantenere lo status 9000)
- c- dalle 256 risposte ho estratto quella con un nano Cx in testa al secondo ottetto
- d- a partire da questa ho variato i byte precedenti ad "yy" fino a trovare P3 = 7n e P4 = m1 oppure m9 (per il gioco del masking, cambia solo il primo ottetto e l'ultimo)

-e- ho inviato la C1 3C cosi' costruita
-f- se lo status e' 9000 invio la C1 3A, altrimenti torno al punto -d- (ripeto:
-d-).

Tutta la trafila e' eseguita per ciascuna prova (non spaventatevi! WinExplorer
fa miracoli)

1' prova con record 0023 uguale a
B1 00 21 D1 FF CA 00 5A 24 F3 67 E3

2' prova con record 0023 uguale a
B1 00 21 D1 FF CA 00 B8 0D 1B 93 E3

3' prova con record 0023 uguale a
B1 00 21 D1 FF CA 00 B8 0D 1B 93 E3
ma kfy 0F sicuramente diversa!

4' prova con record 0023 uguale a
B1 00 21 D1 FF CA 00 5A 24 F3 67 E3
e stessa kfy 0F delle prove 1,2

Per cambiare il secondo ottetto nella prova 4 ho semplicemente cancellato il
record 0024 (e' una kfy, basta costruire un comando sfruttando un PPV-spot "10
x6"), avendo anche un record 0027 il dvmp di giusta lunghezza e' comunque
assicurato...

PRIMA:

86 xx xx xx xx xx xx xx yy
D2 00 21 00 43 7C FF 89 00 00 FF FF(D9)CB B3 00 00
D2 00 22 00 00 D2 FF 17 00 00 FF FF(04|9C)B3 00 00
D2 00 23 B1 00 21(D1)FF CA 00 5A 24 F3 67 E3 00 00
D2 00 24 A6 FF FF(FF)FF FF FF FF FF DE 00 C3 00 00
D2 00 25 00 4E 08 FF 0C 00 00 FF FF 00 00 B3 00 00
D2 00 26 04 FF(FF)FF FF FF FF FF FF 04 00 83 00 00 03 ...

DOPO:

```
86 xx xx xx xx xx xx xx yy
D2 00 21 00 43 7C FF 89 00 00 FF FF(D9)CB B3 00 00
D2 00 22 00 00 D2 FF 17 00 00 FF FF(04|9C)B3 00 00
D2 00 23 B1 00 21(D1)FF CA 00 5A 24 F3 67 E3 00 00
D2 00 25 00 4E 08(FF)0C 00 00 FF FF 00 00 B3 00 00
D2 00 26 04 FF FF FF FF FF FF FF FF 04 00 83 00 00
D2 00 27 00 FA(EE)FF CA 00 00 FF FF 1F 00 B3 00 00 03 ...
```

ESEMPIO DI SCRITTURA DI UNA PPUA

```
C1 36 21 00 40 (36) 86 40 21 00 00 00 00 23 41 B1 21 67 EE FF EE 00 00 FF FF 0A
15 B1 34 65 EE FF EE 00 00 FF FF 0A 15 B1 55 A8 EE FF EE 00 00 FF FF 0A 15 03 82
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

Criptata:

```
C1 36 21 F0 40 (36) 37 72 D1 1C 7E A9 BB 5F 6C 5C D9 33 15 BE 8E 4B CF B6 82 20
88 2D E4 C7 9D 5D 84 24 C8 50 2D D1 C7 EA 9B C5 CD 93 31 5B E8 E4 BC FB 68 22 08
82 DE 4C 79 D5 D8 42 4C 85 FF FF
```

Risultato scrittura PPU* : B1 21 67 EE come da comando in chiaro.

Ma se si gioca con differenti valori di D1 (06) si ottengono altre cose interessanti... con il processo di concatenazione che produce una sign lunga in presenza di una nano 16 con dato 06 e con un sufficiente numero di record presenti sulla sc.

Esempio:

Situazione dei record del provider 00 x0 a partire dal record number 0030 :

```
P=01 Rec.0030 [00] A9 A0 FF 7A 00 00 FF FF PP PP [B1] PPV Record
P=01 Rec.0031 [8B] FF FF FF FF FF FF FF FF 9C 00 [C1] Secondary ***
P=01 Rec.0032 [AB] FF FF FF FF FF FF FF FF 77 00 [81] Primary ***
P=01 Rec.0033 [23] FF FF FF FF FF FF FF FF 9F 00 [81] Primary ***
P=01 Rec.0034 [50] FF FF FF FF FF FF FF FF CA 00 [81] Primary ***
P=01 Rec.0035 [00] AC 1E FF 0D 00 00 FF FF PP PP [B1] PPV Record
```

PP PP valore del ppv spot

Una c1 38 che abbia questa caratteristica in chiaro :

c1 38 P1 P2 LL 16 06 2a 00 30 2b xx xx 86 b1 b2 b3 b3 b4 b6 b7 DD 82 s1 s2... s8

Produrrà una risposta in chiaro del tipo :

86 11 22 33 44 55 66 77 DD

D2 00 30 00 A9 A0 FF 7A 00 00 FF FF PP PP B1 00 00

D2 00 31 8B FF FF FF FF FF FF FF FF 9C 00 81 00 00

D2 00 32 AB FF FF FF FF FF FF FF FF 77 00 81 00 00

D2 00 33 23 FF FF FF FF FF FF FF FF 9F 00 81 00 00

D2 00 34 50 FF FF FF FF FF FF FF FF CA 00 81 00 00

D2 00 35 00 AC 1E FF 0D 00 00 FF FF PP PP B1 00 00

03 82 11 22 33 44 55 66 77 88

dove PP PP valore del ppv spot

Ovviamente non avremo una risposta in chiaro dalla c1 36 ma si può affermare che la risposta della c1 36 nella forma criptata ha questa caratteristica :

Lunghezza della risposta = 0x79;

Numero byte da considerare =

Numdati = Len_c136 - byte di sign = 0x79 - 9 = 0x70

Dando un'occhiata a quanti byte spaiati ci sono si vede che

Byte_spaiati = Numdati/8 = 14 , quindi nessun byte spaiato.

L'assenza di byte spaiati porta a questa situazione di predeterminabilità :

ci ci ci ci ci ci ci ci DD

cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc

cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc

cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc

cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc

cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc

cc cc ci ci ci ci ci ci ci ci ci ci ci ci ci ci ci ci

ci 82 si si si si si si si si

ci= cripted byte di cui non conosciamo il valore
cc = cripted byte di cui conosciamo il valore in chiaro.
DD = ottavo byte dei byte curiosi... (questo sara il primo nano eseguito)
Di fatto i primi 8 byte saranno saltati dal p3=7x , quindi rimarranno solo gli
ultimi 16 byte ad essere non determinabili..

Detto che conosciamo DD , conosciamo tutti i byte della risposta meno gli ultimi
8 prima della sign facciamo un passo ancora... ,
Forziamo il ppv spot del primo record (PP PP) in modo che diventi ad esempio 90
A5 (il perche del 90 mi sembra chiaro...)

Avremo quindi :

D2 00 30 00 A9 A0 FF 7A 00 00 FF FF 90 A5 B1 00 00 da cui la situazione della
nostra carta diverrà :

```
86 11 22 33 44 55 66 77 DD
D2 00 30 00 A9 A0 FF 7A 00 00 FF FF 90 A5 B1 00 00
D2 00 31 8B FF FF FF FF FF FF FF FF 9C 00 C1 00 00
D2 00 32 AB FF FF FF FF FF FF FF FF 77 00 81 00 00
D2 00 33 23 FF FF FF FF FF FF FF FF 9F 00 81 00 00
D2 00 34 50 FF FF FF FF FF FF FF FF CA 00 81 00 00
D2 00 35 00 AC 1E FF 0D 00 00 FF FF PP PP B1 00 00
03 82 11 22 33 44 55 66 77 88
```

Ora sappiamo come detto sopra che anche DD e predeterminabile quindi facciamo
assumere a DD il valore 0xC0 produciamo come detto una c1 36 con p3 = 7x e
vediamo che succede.... (metto sempre i dati in chiaro perche e piu semplice
capire , in bold dove cade il parsing , ?? sono i byte indeterminati.) :

```
7x x1|9 xx xx xx xx xx xx C0
D2 00 30 00 A9 A0 FF 7A 00 00 FF FF 90 A5 B1 00 00
D2 00 31 8B FF FF FF FF FF FF FF FF 9C 00 C1 00 00
D2 00 32 AB FF FF FF FF FF FF FF FF 77 00 81 00 00
D2 00 33 23 FF FF FF FF FF FF FF FF 9F 00 81 00 00
D2 00 34 50 FF FF FF FF FF FF FF FF CA 00 81 00 00
D2 00 35 00 AC 1E FF 0D 00 00 ?? ?? ?? ?? ?? ?? ??
?? 82 11 22 33 44 55 66 77 88
```

Utilizzando questa risposta come c1 40 avremo...

P3=7x quindi skip sul byte C0 che diventa un nano 0 di lunghezza C quindi skip sul byte 90 che diventa un nano 90 quindi scrivi chiave di indice A5... e via con il parsing evidenziato con l'uso del bold.

Il parsing predeterminato termina sui valori ?? ?? ?? che in origine erano 00 00 03 ma che ora come detto sono sconosciuti , quindi una singola c1 40 non e probabile produca un 9600... , occorre produrre quindi molte risposte lunghe che convertite in c1 40 daranno alla fine sicuramente almeno una c1 40 con status 9000 e quindi una scrittura di una chiave , ovvio che se il ppv spot valeva B0 xx avremmo avuto un'altra cosa e via di seguito...

Occhio!!! in questo esempio rimangono 3 byte che possono generare comunque una cancellazione chiave , soluzione ideale e trovare una configurazione di record che abbia un parsing che spiova giusto sul nano 82 oppure al massimo sul byte precedente , cosa che non comporta pericoli per ovvi motivi.

Semplici Esempi:

Vediamo :

Scrittura nani 4x:

Parametro D1 = 03 Ins lunga 37 H

86 11 22 33 44 55 66 77 xx

B1 00 EV EV FF DD 00 00 FF FF P1 P2

B1 00 EV EV FF DD 00 00 FF FF P3 P4

B1 00 AC 1E FF DD 00 00 FF FF P5 P6

03 82

Piazzare al posto dei byte xx il nano che si desidera eseguire, i 4 byte seguenti saranno il loro argomento.

Scrittura Nani Dx

Parametro D1 = 03 Ins lunga 43 H

```
86 11 22 33 44 55 66 77 xx
B1 00 EV EV FF DD 00 00 FF FF P1 P2
B1 00 EV EV FF DD 00 00 FF FF P3 P4
B1 00 EV EV FF DD 00 00 FF FF P5 P6
B1 00 EV EV FF DD 00 00 FF FF P7 P8
03 82
```

Piazzare al posto dei byte xx il nano che si desidera eseguire, i 16 byte seguenti saranno il loro argomento.

Scrittura nani 2x

Parametro D1 = 03 Ins lunga 37 H Condizione necessaria primo B1 dumpato event id 1x Oppure event id = 0x 0x

```
86 11 22 33 44 55 66 77 xx
B1 00 1x EV FF DD 00 00 FF FF P1 P2
B1 00 EV EV FF DD 00 00 FF FF P3 P4
B1 00 EV EV FF DD 00 00 FF FF P5 P6
03 82
```

Piazzare al posto dei byte xx il nano che si desidera eseguire, i 2 byte seguenti saranno il loro argomento.

Scrittura nani 3x

Parametro D1 = 03 Ins lunga 43 H. Condizione necessaria primo byte eseguito Ax e secondo B1 dumpato con event id 1x xx Oppure event id = 0x 0x

```
86 11 22 33 44 55 66 77 Ax
B1 00 EV EV FF DD 00 00 FF FF 3x P2
B1 00 1x EV FF DD 00 00 FF FF P3 P4
B1 00 EV EV FF DD 00 00 FF FF P5 P6
B1 00 EV EV FF DD 00 00 FF FF P6 P7
03 82
```

From DigitalFreeSat (wlg4)

I byte dei record Bx hanno tutti un significato ben preciso e il valore di alcuni byte dipende da quali nano sono stati utilizzati per caricare tali record.

Ad esempio, ho trovato molto utile (forse indispensabile) caricare un record Bx tramite l'accoppiata 31 e 15 (ins 3C).

In sostanza, inviando un ins del tipo :

```
C1 30 00 02 09 00 00 00 00 00 00 00 00 FF
C1 3C 21 9C LL P3 P4 ... [31] x1 x2 x3 [15] D3 ....[82] ...
```

Viene caricato il seguente record Bx

```
02 x1 x2 x3 00 00 D3 00 00 ev sp B1
```

Rinviando la stessa ins

```
C1 30 00 02 09 00 00 00 00 00 00 00 00 FF
C1 3C 21 9C LL P3 P4 ... [31] x1 x2 x3 [15] D3 ....[82] ...
```

Il record viene modificato nel seguente modo

```
02 x1 x2 x3 00 01 A6 00 00 ev sp B1
```

Ad ogni rinvio, si somma il valore del nano 15 (in questo specifico caso "D3").

NOTA : L'INS 3C dopo una transizione pin va a buon fine anche senza il D1.

From DarkSat (ilsorcio)

procedura per preview infinita

Premessa: La descrizione delle procedure in questo doc sono sommarie e non approfondiscono gli argomenti correlati, per approfondimenti e chiarimenti fare riferimento alla Faq.

Vediamo un esempio pratico di come usare ins 'very long' per pilotare la scrittura su card.

Come esercizio consideriamo il problema della 'preview infinita'.

Cenni sul funzionamento della preview.

Quando sintonizziamo un canale che gestisce la preview le ins 3C che arrivano al decoder sono diverse da quelle che arrivano su un channel gestito da bitmap, se potessimo mettere in chiaro queste ins noteremmo la presenza di questi nani:

Nano 31 : con argomento event ID (2 byte) associato la programma trasmesso e numero di diffusione(1 byte).

Nano 19 : con argomento counter (1 byte) che determina la durata della preview.

Nano 27: con argomento data (2 byte) rappresenta la data attuale.

Ovviamente sono presenti altri nani ma per noi i più interessanti sono quelli sopra riportati.

La logica preview è gestita interamente dal nano 19 di cui riporto il flusso per sommi capi :

- 1 Controllo presenza nel buffer ID EVENT valido (>0) altrimenti esci con errore
- 2 Controlla se esiste già un record (AX) con EVENT ID (argomento nano 31) se no salta al punto 6
- 3 Confronta data attuale (argomento nano 27) con data record se non data non valida (data record > data attuale) esci con errore, se data attuale = data record vai al punto 5
- 4 Cancella tutti i record con data inferiore a data attuale.
- 5.Decrementa il contatore (memorizzato nel record) associato al EVENT ID se contatore = 0 esci con errore altrimenti aggiorna il record ed esci con status 9027 (visione consentita)
6. Se non vi è spazio per creare il record esci con status 9006 altrimenti crea un record AX con EVENT ID e DATA corrispondenti all'argomenti dei nano 31 e 27 ed esci con status 9027.

Questo flusso è verificabile dal disassemblato della v6 di H.A. , faccio presente che nelle card 4.1 la preview era gestita in maniera diversa (cancellazione degli AX gestito dal nano 27).

Il nostro obiettivo è creare dei comandi che cancellino tutti i record AX, per quanto riportato sopra possiamo abbozzare una strategia.

Innanzitutto dobbiamo creare una Ins 3C con i nani 27 31 e 19, i nano 27 e 31 devono precedere il 19 (il 19 si aspetta che i nani 27 e 31 siano già stati eseguiti ed i corrispondenti argomenti presenti nel buffer).

Vincoli:

- L'argomento del 27 deve essere inferiore alla data di scadenza dell'abbonamento (altrimenti becchiamo un errore) e contemporaneamente deve essere maggiore della data associata agli Ax presenti sulla SC, per cui dobbiamo spostare la data dell'abbonamento ad un valore che sia sufficientemente grande, per fare ciò prepareremo una ins C1 40 che contenga un nano di aggiornamento data .

- sull'argomento del nano 31 l'unico vincolo è event id>0

- l'argomento del nano 19 (counter) deve essere > 0

L'invio di una 3C così costruita andrebbe a cancellare tutti gli AX con data inferiore all'attuale (nano 27) ma ne creerebbe uno nuovo associato all'event id argomento del nano 31 e con data attuale, questo è un problema dato che questo AX impedirebbe il corretto funzionamento della preview, bisogna quindi cancellarlo ..., per fare ciò abbiamo bisogno di una C1 40 con nano 28, che va a cancellare gli AX corrispondenti alla data associata al nano 28 , quindi il 27 della 3C e il 28 della 40 devono avere lo stesso argomento.

Partiamo dalla costruzione della 3C .. ci serve una lunga generata come risposta di una corta 7X (vedi thread di st7ing) , cambio word sino a quando d1=03.

La risposta in chiaro è la seguente:

```
C1 36 21 00 LEN
86 b0 b1 b2 b3 b4 b5 b6 b7
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
...
....
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
04
```

in pratica il crypt di n-record Bx corrispondenti ad event-id >= ai primi due byte della segnatura della corta 7X (bisogna prima cercare una 7x adeguata), preceduti da 86 + otto bytes relativi a quelli in chiaro della 38 (comprendenti agli ultimi 2 bytes della signature + 6 bytes in chiaro post signature)

una particolarità di ins lunghe così composte è che trasformandole in 38 o 40 (rispettando i vincoli noti !) e facendo in modo che il primo ottetto venga 'saltato' scegliendo P3 = 7X , il decrypt dei bytes che seguono corrisponde al valore in chiaro , questo limitatamente solo ad un certo numero di ottetti successivi al primo saltato (più info nelle FAQ) .

Nel nostro caso risultano noti i valori del decrypt corrispondenti alla parte non evidenziata:

```
[86 b0 b1 b2 b3 b4 b5 b6] b7
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
...
...
B1 xx xx xx xx xx xx [xx xx xx xx xx
B1 xx xx xx xx xx xx xx xx xx xx xx
04]
```

per ins sufficientemente lunghe (e P3=7x) il primo byte del decrypt è a nostra scelta (b7)

ci sono altre considerazioni da fare

Dato che con la risposta va costruita una 3C, le chiavi che si possono usare per codificarla sono le operative OC-OD-OE , a noi fa comodo usare la OE dato che non è soggetta a cambiamento mensile (almeno sino ad oggi ...) inoltre c'è da considerare che il bug ci consente di firmare la nostra risposta solo in UA, per cui la 3C può essere utilizzata solo sulla sc su cui è stata creata , quindi purtroppo ognuno dovrebbe crearsi la propria.

NOTA: Nelle 3C il primo ottetto non è codificato

Impostando quindi quindi b7=27 abbiamo scritto il primo nano a noi necessario corrispondente ad una data B1 00, per fare in modo che l'argomento del nano 28 del comando di cancellazione (c1 40) dell'ax sopra descritto faremo la stessa scelta b7=28 in modo da di essere sicuri di cancellare proprio l'ax voluto (l'ax creato ha la data = all'argomento del nano 28).

Vediamo ora come impostare i 2 nani che mancano.. per fare ciò dovremo modificare il PPV spot di alcuni record bx (sul metodo per modificare il PPV spot fare riferimento alla FAQ)

Per impostare il nano 31 dovremo scegliere un PPV spot uguale a 31 XX.

Infine per impostare il nano 19 serve un PPV spot = 19 XX

A questo punto abbiamo finito la procedura manca solo da verificare la correttezza del parsing ..

Premesso che nel 90 % dei casi un bx (corrispondente ad un evento già visionato) contiene valori dei byte tali da fare in modo automaticamente che il parsing cada sul primo byte del PPV spot, spesso tuttavia risulta comodo saltare un intero bx (facendo in modo che il parsing cada sul nano B1), una certa forma di controllo sul parsing lo abbiamo attraverso la scrittura dei PPV spot e attraverso l'impostazione dei crediti (memorizzati nel 6° e 7° byte del bx) tramite la procedura descritta ultimamente da W1G4.

Tuttavia a volte è inevitabile che il parsing cada all'interno della coda (ultimi bytes di cui non si conosce il decrypt) e risulta necessario ottenere nuovi crypt modificando i bytes in chiaro b0 .. b6 (mantenendo i vincoli su P3)

Seguendo la stessa tecnica è possibile creare il comando di cancellazione dell'ax ed il comando di spostamento della data.