draft-tiloca-6tisch-robust-scheduling-00

Authors: Marco Tiloca

Simon Duquennoy

Gianluca Dini

Motivation

- Cell utilization patterns are predictable in TSCH
  - Even if security is used at the link layer
- An external adversary can easily:
  - Derive the communication pattern of a victim node
  - Selectively jam the exact cells of the victim's schedule
- The attack is:
  - Easy and efficient to perform
  - Highly effective and with very low exposure

What makes the attack easy?

- Periodicity property --- Every cell re-uses the same sequence of channels, with period (N_C x N_S)
- Usage property --- Within each period, all cells use all channels, once each
- Offset property --- All cells follow the same sequence of channels, with a certain offset
- Predictability property --- The sequence of channel is predictable, given a pair (timeslot, channel)
  - Timeslots repeat periodically on a same channel
  - One can compute the remaining channel hopping sub-sequence
- Attack rationale
  - ASN = (s + T x N_S) ; a cell uses channel f and timeslot s on slotframe T
  - Solve $f = [(s + T \times N\_S + c) \bmod N\_C]$ in c (Equation 1)
  - Find the channels used by the cell in the next slotframes
  - The exact ASN is not needed! One can re-number slotframes from an arbitrary one

Attack outline

- Start the attack at a starting-slotframe $t = 0$
- Determine the timeslots in which the victim transmits
  - Pick a channel f^* at random
  - Listen on f^* for N_C consecutive slotframes
- Find the channels used by the victim in the next slotframes
  - Solve Equation 1 in c for each found timeslot
  - Now f can be computed for any $t > 0$ and every timeslot s
- The adversary knows the full victim schedule
  - Selective jamming against the victim cells only
  - Staying quiet otherwise

Attack example

- Starting-slotframe t = 0 is slotframe T = 1
- Listen to f^* = 1 for 4 slotframes
  - Used in s = 1 and s= 2 of t = 0
  - Used in s = 0 of t = 1
  - All used timeslot are found
- Solve Equation 1 in c for each s found
  - Able to derive future f from s and t
- At each slotframe t > 3, i.e. T > 4
  - Derive f for each timeslot s
  - Jam traffic during s over f

```
|==|===================================================================|
|Ch|                              ASN                                  |
|  |===================================================================|
|Of| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |10 |11 |12 |13 |14 |15 |16 |
|==|===================================================================|
|0 |   |   |f=2|   |   |f=1|   |   |f=0|   |   |f=3|   |   |f=2|   |   |
|--|-------------------------------------------------------------------|
|1 |   |f=2|   |   |f=1|   |   |f=0|   |   |f=3|   |   |f=2|   |   |f=1|
|--|-------------------------------------------------------------------|
|2 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|--|-------------------------------------------------------------------|
|3 |f=3|   |   |f=2|   |   |f=1|   |   |f=0|   |   |f=3|   |   |f=2|   |
|==|===================================================================|
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
   |s=0|s=1|s=2|s=0|s=1|s=2|s=0|s=1|s=2|s=0|s=1|s=2|s=0|s=1|s=2|s=0|s=1
   |           |           |           |           |           |
   |   T = 0   |   T = 1   |   T = 2   |   T = 3   |   T = 4   |   T = 5
               |
               \__ t = 0
```

Equation 1: $f = [(s + t \times 3 + c) \bmod 4]$

Solution – Overview

- Prevent the attack by construction
  - Alter the communication pattern of nodes at every slotframe
  - The resulting used pattern must be unpredictable for the adversary
- At each slotframe T:
  - All nodes pseudo-randomly permute the original schedule for $T + 1$
  - Separate permutation of timeslot usage (optional) and channel offset usage
  - All nodes locally compute the same permutation
  - The resulting schedule is consistent and collision-free
- Pseudo-random number generator
  - val = random(K, z) = E(K, z) - Encrypt a fresh value z with a key K
  - AES-CCM-16-64-128 must be supported

Solution – Key material

- Permutation key *K_s*
  - Used to permute the timeslot utilization pattern
  - Provided upon joining, e.g. using the 6TiSCH Join Protocol (CoJP)
- Permutation key *K_c*
  - Used to permute the channelOffset utilization pattern
  - Provided upon joining, e.g. using the 6TiSCH Join Protocol (CoJP)
- Counter *z_s*
  - Used to permute the timeslot utilization pattern
  - At the beginning of *T*, *z_s* is equal to the ASN of the first timeslot of *T*
- Counter *z_c*
  - Used to permute the channelOffset utilization pattern
  - At the beginning of *T*, *z_c* is computed from the ASN of the first timeslot of *T*

Solution steps

- At the beginning of each slotframe, each node:
  - Takes the original schedule for the next slotframe
  - Performs the steps below to permute the schedule using the Fisher-Yates algorithm
  - Provides the permuted schedule to TSCH, to send/receive traffic in the next slotframe

- Step 1 – Permute the timeslot utilization pattern (optional)
  - N_s invocations of random(K, z)
  - $K = K\_s$; $z = z\_s$;
  - z_s incremented after each invocation

- Step 2 – Permute the channelOffset utilization pattern
  - N_c invocations of random(K, z)
  - $K = K\_c$; $z = z\_c$;
  - z_c incremented after each invocation

Key provisioning

- *K_s* and *K_c* MAY be provisioned with the minimal security framework
  - The JRC provides the pledge with *K_s* and *K_c* upon joining
- Additional COSE_KeySet in the Join Response
  - If two keys are present, the first key is *K_s* and the second key is *K_c*
  - If one key is present, it is *K_c* (permute only channelOffset utilization patterns)
- Details need to be updated to the latest Join Response format
  - A dedicated COSE_KeySet seems still the best option

Summary and next steps

- Preventive approach against selective jamming
  - Agnostic of the specific scheduling algorithm
  - Preserve collision-free and consistent schedules
  - Efficient pseudo-random shuffling of cells
  - No communication overhead

- Next steps
  - Get comments and feedback
  - Align the text on key provisioning with the latest *draft-ietf-6tisch-minimal-security*

Thank you!

Comments/questions?

https://gitlab.com/crimson84/draft-tiloca-6tisch-robust-scheduling