Evaluation of Security Protocols for the Internet of Things

Package 2 Report #1

HB/HB+

Submitted by:

Anh Le Pham

Sinan Özdemir

Submitted To:

Prof. Frederik Armknecht

Vasily Mikhalev

# 1. Introduction

HB and HB+ schemes are shared-key authentication protocols that relies on the hardness of the learning parity with noise problem (LPN) for security proved to be NP hard. Passive attacks, such as a man-in-the-middle attack, can not be performed on devices using HB protocol for identification. The enhanced version named HB+ (Juels and Weis), however, is proven to be secure against active attacks. Due to there efficiency and simplicity these protocols are in the category of lightweight and can be used for devices with small computational capability (Katz) such as the Arduino Uno R3. For that reason, HB and HB+ protocols were implemented and evaluated in the following subsections.

# 2. HB/HB+ Algorithms

In this subsection the HB and HB+ algorithms are explained.

HB Algorithm

Figure 1 shows one iteration of the authentication of HB.

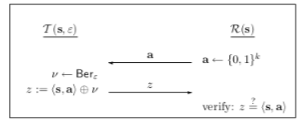


Figure 1: One iteration of the HB protocol

A tag T and a reader R share a random secret key s ∈ {0,1}k. One iteration (all of which happen in parallel) of the authentication step consists of the following: R sends a random challenge a ∈ {0, 1}k to T who in turn calculates z := ⟨s, a⟩ ⊕ v with v ← Berε. This result is sent back to R who then calculates if the iteration is successful, i.e. z = ⟨s, a⟩. Notice that even iterations of an honest tag using the correct key s can be unsuccessful with probability ε. The reader therefore accepts the authentication of the tag if the number of unsuccessful iterations is at most ≈ ε · n.

HB+ Algortihm

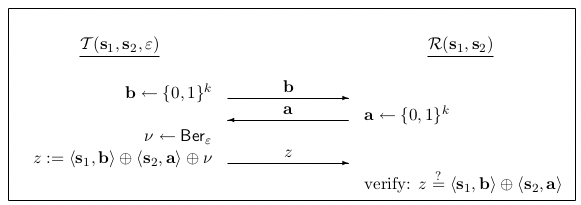
A modification of the HB protocol in order for it to be secure against an active adversary is the HB+ protocol by Juels and Weis [JW05] shown in Figure 2.

Figure 2: One iteration of the HB+ protocol

R and T now share two secret keys s1,s2 ∈ {0,1}k. One iteration of the authentication step now looks as follows: T first sends a random ”blinding 1 factor” b ∈ {0,1}k to R. The reader then, as for HB, sends a random challenge a ∈ {0,1}k to T who in turn calculates z := ⟨s1,b⟩ ⊕ ⟨s2,a⟩ ⊕ v with v ← Berε. This result is sent back to R who then calculates if the iteration is successful, i.e. z = ⟨s1, b⟩ ⊕ ⟨s2, a⟩. Again, even if T sends an honest z using the correct keys s1, s2 the iteration can be unsuccessful. Therefore, up to ≈ e · n unsuccessful iterations are allowed for the tag to still be accepted.

# 3. HB/HB+ Implementation on the Arduino Uno R3

tbd.

# 4. HB/HB+ Evaluation

This last subsection provides the evaluation results of the HB and HB+ protocols with different parameters. This evaluation is based on code size, memory size, computational complexity and the power consumption. The latter, however, provides in every version with different parameters similar results. Each set of parameter contributes in this work to a new version. In order to measure the computational complexity, the average of 10 authentication rounds is calculated and noted. The average 10 rounds of the two HB parallel versions are calculated with 1, 5, 10, 15 and 20 out of 22 maximum challenges. Whereas the average 10 authentication rounds of the two HB+ parallel versions are calculated with 1, 5, 10, and 15 out of 18 maximum challenges.

The code size of the first HB version is 4930 bytes, which corresponds to 15% of the total capacity. Second version shows the same result. The third version, with 5244 bytes, shows an increase of the code size. This trend continues along the fourth 5248 bytes and fifth version with 5292 bytes. The parallel versions of the HB protocol display the highest code size and are 5444 bytes to 5464 bytes. This means a use of 16% of the total available storage.

Memory size of the first, second and third version is 482 bytes this is 23% out of the available 2,048. Further, version four shows an increase of 514 bytes’ (25%) memory usage and version five is 578 bytes (28%). With 442 bytes (21%) the parallel versions consume the smallest amount of memory size.

For computational complexity, measured in milliseconds, can be noted that each version leads to different results. Version one starts with 12701 ms, whereas version two needs 5246 ms. Version three is the fastest and the measured time is 3340 ms. From all different set of parameter needs version four the longest with 23975 ms. Version five is 16104 ms. Bigger difference can be noted between the parallel versions. HB parallel version one is 5494 ms , whereas the second parallel version needs 13352 ms.

The code size of the HB+ protocols shows that version one and version two have both 5400 bytes in use, which corresponds to 16% of the total capacity. Version three shows an increase of the code size by 5628 bytes, version four is 5702 bytes and version five is 5694 bytes. HB+ parallel version one and two have the same code size which is 6716 bytes which is 20% of the total capacity.

Memory size for version one, two and three is 542 bytes or 26% out of the available 2,048. Whereas version four shows an increase by 580 bytes and version five shows, the highest memory size, with 644 bytes in use. This corresponds to 31% of the total memory capacity. Both parallel versions used 502 bytes out of 2048 bytes, which is 24%.

Computational complexity is 14617 ms for version one of the HB+ protocols. Version two decreases the time by 6061 ms, followed by version three with 3902 ms. Version four needs the longest amount of time with 27865 ms. Computational complexity for version five is 17780 ms. Parallel version one needs 6376 ms. Finally, the second parallel version has a computational complexity of 15446 ms.

**Evaluation table of the HB/HB+ protocols**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Protocols** | **Parametes** | **Code Size** | **Memory Size** | **Computational Complexity** | **Power Consumption Current at 200mA** |
| **HB\_ver1** | k = 512,  e = 0.25,  u = 0.348,  n = 1164 | 4930 Bytes (15%) | 482 Bytes (23%) | 12701 ms | 27.5-27.6 |
| **HB\_ver2** | k = 512,  e = 0.125,  u = 0.256,  n = 441 | 4930 Bytes (15%) | 482 Bytes (23%) | 5246 ms | 27.6 - 27.7 |
| **HB\_ver3** | k = 512,  e = 0.125,  u = 0.1875,  n = 256 | 5244 Bytes (16%) | 482 Bytes (23%) | 3340 ms | 27.8 - 27,9 |
| **HB\_ver4** | k\_x = 768  e = 0.25  u = 0.3315  n = 1494 | 5246 Bytes (16%) | 514 Bytes (25%) | 23975 ms | ... |
| **HB\_ver5** | k\_x = 1280  e = 0.125  u = 0.232  n = 574 | 5292 Bytes (16%) | 578 Bytes (28%) | 16104 ms | ... |
| **HB\_verParallel1** | k = 512  e = 0.125  u = 0.256  n = 441 | 5444 Bytes (16%) | 442 Bytes (21%) | 5494 ms | ... |
| **HB\_verParallel2** | k = 512  e = 0.25  u = 0.348  n = 1164 | 5464 Bytes (16%) | 442 Bytes (21%) | 13352 ms | ... |
| **HBP\_ver1** | k1 = 80,  k2 = 512,  e = 0.25,  u = 0.348,  n = 1164 | 5400 Bytes (16%) | 542 Bytes (26%) | 14617 ms | 26.7 - 26.8 |
| **HBP\_ver2** | k1 = 80,  k2 = 512,  e = 0.125,  u = 0.256,  n = 441 | 5400 Bytes (16%) | 542 Bytes (26%) | 6061 ms | 27.8 - 28.9 |
| **HBP\_ver3** | k1 = 80,  k2 = 512,  e = 0.125,  u = 0.1875,  n = 256 | 5628 Bytes (17%) | 542 Bytes (26%) | 3902 ms | 27.2 - 27.3 |
| **HBP\_ver4** | k1 = 128,  k2 = 768,  e = 0.25,  u = 0.3315,  n = 1494 | 5702 Bytes (17%) | 580 Bytes (28%) | 27865 ms | tbd |
| **HBP\_ver5** | k1 = 128,  k2 = 1280,  e = 0.125,  u = 0.232,  n = 574 | 5694 Bytes (17%) | 644 Bytes (31%) | 17780 ms | tbd |
| **HBP\_verParallel1** | k1 = 80  k2 = 512  e = 0.125  u = 0.256  n = 441 | 6716 Bytes (20%) | 502 Bytes (24%) | 6376 ms | tbd |
| **HBP\_verParallel2** | k1 = 80  k2 = 512  e = 0.25  u = 0.348  n = 1164 | 6716 Bytes (20%) | 502 Bytes (24%) | 15446 ms | tbd |

|  |  |  |  |
| --- | --- | --- | --- |
| Protocols | Parameter | Challenge | Computational Complexity |
| hb\_verParallel1 | k = 512  e = 0.125  u = 0.256  n = 441 | maxChallenge = 1 | 5745 ms |
|  | \* | maxChallenge = 5 | 5430 ms |
|  | \* | maxChallenge = 10 | 5456 ms |
|  | \* | maxChallenge = 15 | 5430 ms |
|  | \* | maxChallenge = 20 | 5409 ms |
|  |  | maxChallenge = 22 | - |
| hb\_verParallel2 | k = 512  e = 0.25  u = 0.348  n = 1164 | maxChallenge = 1 | 14029ms |
|  | \* | maxChallenge = 5 | 13180 ms |
|  | \* | maxChallenge = 10 | 13248 ms |
|  | \* | maxChallenge = 15 | 13110 ms |
|  | \* | maxChallenge = 20 | 13196 ms |
|  |  | maxChallenge = 22 | - |
| hbp\_verParallel1 | keyLength1 = 80;  keyLength2 = 512  e = 0.125  u = 0.256  n = 441 | maxChallenge = 1 | 6622 ms |
|  | \* | maxChallenge = 5 | 6305 ms |
|  | \* | maxChallenge = 10 | 6308 ms |
|  | \* | maxChallenge = 15 | 6269 ms |
|  |  | maxChallenge = 18 | - |
| hbp\_verParallel2 | k = 512  e = 0.25  u = 0.348  n = 1164 | maxChallenge = 1 | 16100 ms |
|  | \* | maxChallenge = 5 | 15261 ms |
|  | \* | maxChallenge = 10 | 15256 ms |
|  | \* | maxChallenge = 16 | 15169 ms |
|  | \* | maxChallenge = 18 | - |