

Adaptive security and privacy for mHealth sensing

Shrirang Mare¹, Jacob Sorber¹, Minh Shin², Cory Cornelius¹, and David Kotz¹

¹*Institute for Security, Technology, and Society, Dartmouth College, USA*

²*Department of Computer Engineering, Myongji University, South Korea*

Abstract

As healthcare in many countries faces an aging population and rising costs, mobile Health (mHealth) sensing technologies promise a new opportunity. However, the privacy concerns associated with mHealth sensing are a limiting factor for their widespread adoption. The use of wireless body area networks pose a particular challenge. Although there exist protocols that provide a secure and private communication channel between two devices, the large transmission overhead associated with these protocols limit their application to low-power mHealth sensing devices. We propose an adaptive security model that enables use of privacy-preserving protocols in low-power mHealth sensing by reducing the network overhead in the transmissions, while maintaining the security and privacy properties provided by the protocols.

1 Introduction

Recent improvements in mobile computing and developments in miniature medical sensors have enabled mobile Health (mHealth) sensing. An mHealth sensing system can deliver continuous health monitoring to patients throughout their daily activities with the potential to simultaneously reduce cost and improve the quality of healthcare [6, for example]. However, there are strong privacy and security concerns associated with mHealth sensing. For instance, a woman wearing a wireless fetal heart monitor, during her daily activities, may not want people around her knowing that she has such a device or observing the data transmitted by the device. In a wireless network, an adversary can observe a device's transmissions and may learn the contents of the transmissions or the type of the device [5, 9]. Thus, mHealth sensing requires protocols that provide strong privacy and security guarantees.

There exist some privacy-preserving protocols that provide security and privacy properties like unlinkability, data confidentiality, data integrity and data authentication; most such protocols are designed for Wi-Fi because the privacy problem is well known in the Wi-Fi setting. These protocols, however, add various forms of overhead; although the overhead may not be significant in a Wi-Fi setting where the payloads are large, in

an mHealth sensing network, the payloads are small and this overhead becomes significant.

Often the protocol overhead is related to the security of the protocol. For instance, the message authentication code (MAC), which is sent with the packet for data authentication, contributes to the packet overhead. The MAC length determines the strength of the protocol against message authentication attacks; for example, broadly speaking, to forge a packet with a 128-bit MAC, by brute-force, the adversary would have to try about 2^{128} times,¹ but to forge a packet with a 16-bit MAC, the adversary would only have to try about 2^{16} times, which is not sufficient protection. Thus, simple ways to reduce the overhead would also reduce the security of the protocol.

We propose an adaptive security model in which nodes change the size of packet overhead dynamically when a node detects a possible forgery attack, thereby providing strong security and privacy in the presence of an adversary, but otherwise using low overhead to minimize the energy consumption of the network.

2 Privacy-preserving protocols

Due to the broadcast nature of wireless networks, there are many ways in which a user's privacy is leaked; for example, an adversary can infer some information about the data being transmitted, or the device transmitting the data, based on the address in the transmitted packet, other identifying information in the packet, the size of the packet, or the timing of the packet sequence. Thus, the goal of a privacy-preserving protocol is to obfuscate such information so that the adversary cannot infer any sensitive information about the data or about the device transmitting the data. There are several privacy-preserving protocols for wireless networks proposed in the literature [1, 8, and their references]. Such protocols provide several properties: *data confidentiality* (an adversary should not be able to learn the contents of the underlying data in the packet), *data authenticity* (an adversary should not be able to forge a packet without being detected), *data integrity* (an adversary should

¹More precisely, to forge a n -bit MAC the adversary would have to try about 2^{n-1} times (on average); for simplicity we will use 2^n as the work required to forge a n -bit MAC by brute force.

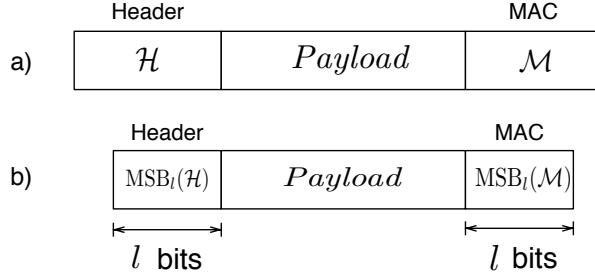


Figure 1. a) Generalized packet format of a privacy-preserving protocol, b) Adaptive packet format

not be able to modify a packet without being detected), and *unlinkability* (given two packets transmitted to/from the same node, the adversary should not be able to link the packets).

Figure 1a shows a generalized packet format of a typical privacy-preserving protocol. Standard encryption techniques provide confidentiality and authenticity by encrypting the payload and appending a message authentication code (MAC). To achieve unlinkability, the privacy-preserving protocols ensure that the header \mathcal{H} , the payload, and the MAC \mathcal{M} change with each packet such that the entire packet appears as a pseudorandom string to an adversary. The receiver, however, is able to identify packets addressed to itself efficiently based on \mathcal{H} . We assume that the privacy-preserving protocols use strong cryptographic algorithms to compute \mathcal{H} and \mathcal{M} , and the only way for the adversary to forge \mathcal{H} and \mathcal{M} is by a brute-force attack.

The packet overhead (\mathcal{H} and \mathcal{M}) of existing protocols is usually high (e.g., SlyFi [1] overhead is 32 bytes). This overhead is not significant in a Wi-Fi setting, where payloads are large (e.g., 512 bytes). But for mHealth sensing, where the sensor data is small (1-50 bytes), 32 bytes of overhead is significant.

3 Adaptive security

In privacy-preserving protocols the receiver node uses the header \mathcal{H} to filter incoming packets (that is, to determine whether the incoming packet is addressed to this node), and it uses the MAC \mathcal{M} for data authentication. This overhead (\mathcal{H} and \mathcal{M}) in the protocols is usually fixed, and for strong security, the protocols choose long \mathcal{H} and long \mathcal{M} . However, a node is not always in a hostile environment, so using large overhead all the time is inefficient. In most mobile devices, the network interface is the most energy-consuming part of the system, so we need to reduce network overhead.

In adaptive security, the parameter that we change is the size of the header and the MAC sent in the packet. Thus, instead of using a fixed large overhead, the nodes use a small packet overhead that expands dynamically if the node detects the presence of an adversary who is trying to forge a packet.² This approach provides strong

security when required (e.g., when there is an attack), but saves energy, at both the sender node and the receiver node, when not required.

3.1 Adapting: how

In a typical body-area sensor network for mHealth sensing, there will be one *mobile node* (MN, e.g., mobile phone), and one or more *sensor nodes* (SNs). Let us consider, for simplicity, a network with one MN and one SN.

We propose to shorten the packet header and MAC. As shown in Figure 1b, the actual bits sent as header and as MAC are the l most significant bits of \mathcal{H} and \mathcal{M} , respectively. In our proposal, this length adapts to potential forgery attacks; thus, the MN and SN need to agree on l and how it changes. Let σ_{MN} and σ_{SN} be the sets of values that MN and SN use, respectively, to determine the header and the MAC lengths for an incoming or an outgoing packet; for example, if $\sigma_{MN} = \{l_0\}$, then for every received packet, the MN will assume the header and MAC lengths as l_0 . If set σ_{MN} contains more than one element, then MN will parse the packet multiple times, each time with a different value from σ_{MN} , till either the packet is accepted (after header and MAC verification) or the packet is discarded. Thus, if the SN sends a packet p with header and MAC of size l each, then the MN will accept the packet only if $l \in \sigma_{MN}$. In our adaptive protocol, the size of set σ_{SN} is always one; thus, the SN can communicate with the MN only if $\sigma_{SN} \subseteq \sigma_{MN}$.

Figure 2 shows how the MN and the SN change the sets σ_{mn} and σ_{sn} so that they can communicate with each other while adapting to the new header and new MAC length l . At time t_0 , during discovery, the MN and the SN share the initial length l_0 ; that is, at time t_0 , $\sigma_{MN} = \sigma_{SN} = \{l_0\}$. In our proposal, only the MN makes the decision to change l . Later at some time t_1 , the MN decides that it needs to adapt and change the header and MAC length to l_1 , and so it adds l_1 to σ_{MN} . After time t_1 , when the MN receives a packet from the SN (the header and MAC size for this packet will be l_0), it accepts the packet (because $l_0 \in \sigma_{MN}$), and piggybacked on the ACK returned to the SN, it indicates its preference for l_1 , so SN adds l_1 to σ_{SN} and removes the old value (l_0) from σ_{SN} . However, the SN may not receive the ACK (as shown in the figure), in which case the SN will continue to send packets with header and MAC of size l_0 (each), which is okay because the MN can accept those packets, as $l_0 \in \sigma_{MN}$. Eventually, at time t_2 , the SN will receive the ACK, and it will change σ_{SN} (as shown in the figure). Now, after time t_2 , when the MN receives a packet from the SN (at time t_3), it would know that the SN has updated its σ_{SN} , and so the MN will remove the old value l_0 from σ_{MN} . Thus, in short, while adapting to the new header and MAC size, the MN maintains the set σ_{MN} such that σ_{SN} will always be its subset, and this

whenever header-collision counts rise too large, which may simply indicate the presence of a neighboring network; in the worst case, the overhead never exceeds that of the non-adaptive protocol.

²As described below, our method conservatively reacts

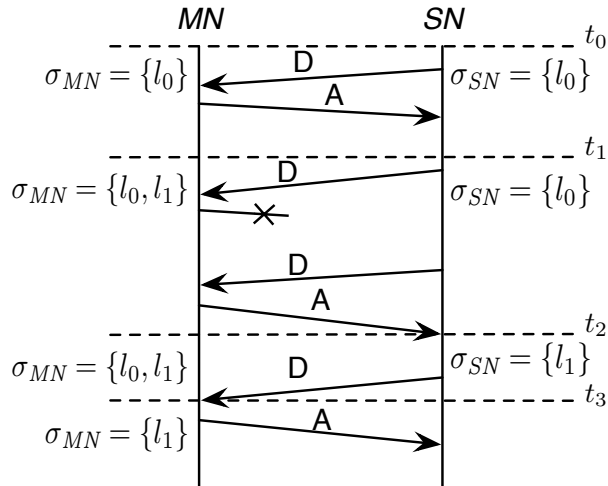


Figure 2. Adaptive security model

ensures that the MN and the SN remain in sync and can communicate while adapting.

3.2 Adapting: when

In this section we describe one simple condition that triggers the MN to adapt. One can add more conditions as per the application and network requirement. Due to space constraints, we will provide only the intuition behind the proof for choosing this specific condition.

Smaller header and smaller MAC decrease the work required to forge a packet. So the MN increases the header and the MAC length in presence of an adversary who is trying to forge a packet, thereby making it harder for the adversary to succeed in forging a packet.

The MN uses the header to filter incoming messages, so depending on the network traffic (due to transmissions from neighboring BASNs), a smaller header will cause more collisions in the MN's hash table than a larger header. A hit in the hash table implies that the incoming message is from the SN, or that the message is of a neighboring BASN that happens to use a header matching one of the headers in the MN's hash table, or that the message is a forgery attempt. It is hard for the MN to distinguish between the latter two cases, in which MAC verification fails. We take a cautious approach, and consider a failed MAC verification to be a forgery attempt, and hence presence of an adversary.

The MN's hash table contains w headers. For packets with header and MAC lengths l , the adversary has to transmit about $2^{2l}/w$ times to successfully forge a packet. The MN counts the number of forgery attempts, and when the number of the such forgery attempts gets larger than a threshold α (we derive α in Section 4), the MN increases the header and MAC lengths, which in turn increases the difficulty of forgery. Thus, the security against forgery increases as the adversary works harder. The MN resets the forgery counter, and the header and MAC size to l_0 after certain time T , which is determined by the MN.

4 Security and privacy analysis

After adapting a privacy-preserving protocol, it is important that we preserve the security and privacy properties achieved by that protocol. As described above, the properties achieved by a typical privacy-preserving protocol are unlinkability, data confidentiality, data authenticity and data integrity. Data authentication is a stronger property than data integrity; by achieving data authentication one also achieves data integrity.

Unlinkability. In a privacy-preserving protocol, the \mathcal{H} , payload, and \mathcal{M} portions of a packet are pseudorandom strings (from an adversary's perspective), and they change with each packet. This property provides unlinkability — preventing packets from being linked to a sender, a receiver, or even to each other. In the adaptive security model we use a truncated \mathcal{H} as the header and a truncated \mathcal{M} as the MAC. It is easy to prove that any substring of an unlinkable string is also unlinkable (because if a substring is linkable, then it implies that the adversary can link the full strings, by using just the linkable substring). Thus, the smaller header and smaller MAC used in the adaptive security model are also pseudorandom strings from an adversary's perspective; hence, the packets in the adaptive security model are also unlinkable.

Data confidentiality. The payload is not dependent on the number of header or MAC bits transmitted as part of the packet, and neither the size of the header or MAC is dependent on the payload. Thus, in the adaptive security model the header and the MAC do not give away any more information about the payload than the full header \mathcal{H} and full MAC \mathcal{M} used by the privacy-preserving protocol. Thus data confidentiality is preserved under the adaptive security model.

Data authentication. In the adaptive security model we wish to provide security against existential forgery with a probability of $2^{-\delta}$; that is, at any given time, the probability that the adversary can forge a packet (where the underlying message may or may not be meaningful to the receiver) is never greater than $2^{-\delta}$, where $\delta (> 0)$ is set by the MN.

The header and the MAC are pseudorandom strings from the adversary's perspective, and to successfully forge a packet, the adversary has to guess the right header (among the w headers in MN's hash table) and the right MAC for the packet. Thus, if l represents the header and MAC lengths, the probability that the adversary would forge a packet in x consecutive attempts is

$$P = 1 - \left(1 - \frac{w}{2^{2l}}\right)^x \quad (1)$$

We need the probability P to be lower than $2^{-\delta}$. Thus, solving the inequality $P < 2^{-\delta}$ for x , we get the following inequality for x

$$x < \frac{\log(1 - 2^{-\delta})}{\log(1 - \frac{w}{2^{2l}})} \quad (2)$$

Whenever this inequality becomes invalid (that is, the

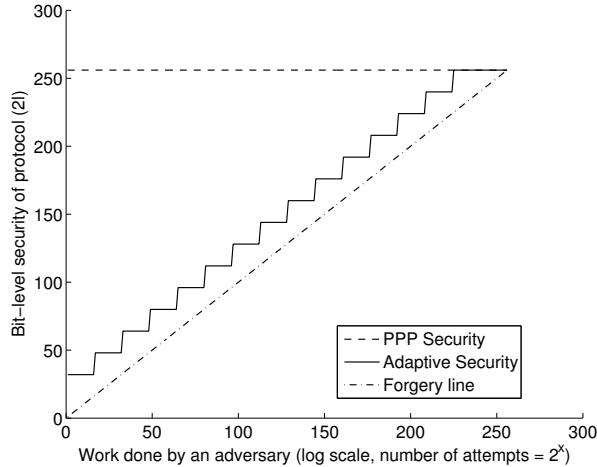


Figure 3. Adaptive security against a forgery attack

adversary attempts more than $\frac{\log(1-2^{-\delta})}{\log(1-\frac{w}{2^{2l}})}$ times), the MN increases the header and MAC lengths (i.e., l) to maintain the inequality in Equation 2.

Figure 3 gives some intuition about how the MN will adapt in the presence of an adversary who is trying to forge a packet. The logarithmic x -axis represents the work the adversary does to forge a packet (i.e., number of forgery attempts = 2^x), and the y -axis represents the bit-level security of the protocol, which is equivalent to the size of header and MAC (i.e., $2l$). The adversary is likely to be successful in forging a packet if the line representing the security of the protocol falls below the forgery line ($y = x$). In the plot, the forgery probability is set to 2^{-16} (i.e., $\delta = 16$), and the initial l value is 16 bits (i.e., $l_0 = 16$). The plot also presents the security level of a typical privacy-preserving protocol (PPP) that has a overhead of 256 bits, and an equivalent security level. Thus, as the adversary tries harder to forge the packet, via brute-force attempts, the security increases, making it harder for him to succeed. As you can see, in worst-case scenario (i.e., when the adversary keeps trying or there is continuous enough noise) the MN will choose the highest header and MAC length. Thus, in worst-case scenario, the adaptive protocol behaves similar to the non-adaptive privacy-preserving protocol.

5 Related work

Our adaptive-security approach adapts the security level of the protocol dynamically in response to network activity that implies the presence of an adversary (or a forgery attack). There are adaptive security models in the literature but they differ in how the protocols adapt and under what conditions they adapt.

Prasad et al. [3] suggest using three modes of security: low-level, medium-level, and high-level security. Depending on the user location (e.g., home vs. public place) or which devices the user is contacting (e.g., trusted vs. untrusted), the model will choose an appropriate security level. The different levels of security

achieve different sets of properties, and they use different cipher algorithms. However, choosing the right level of security is not easy; for example, a familiar location (e.g., home) does not necessarily imply the absence of an adversary. Portialla et al. [2] propose changing ECC parameters to provide different levels of security depending on the energy budget of the node. The TLS cipher suite [4] is a real-world example where nodes can negotiate and decide on the security level by selecting algorithms (authentication, encryption, and message authentication code). However, all these methods focus on adapting cryptographic primitives (i.e., encryption, MAC algorithms) to provide different levels of security rather than on reducing network overhead, which would provide more energy savings (especially in a wireless network) than the proposed computational adaptation.

The hybrid security model proposed by Shon et al. [7] is probably the closest work to our adaptive security model. In their adaptive scheme they propose using MAC of different sizes to provide different levels of security. In their scheme, they choose a security level by the network characteristics (public, commercial, or private) and the data characteristics (application data or control data). Classifying data sensitivity and determining which level of security would be reasonable for a given data type, while optimizing energy, can be tricky. For mHealth sensing, where the medical data is considered sensitive, their approach would choose the higher-level security, which would be energy inefficient. Our adaptive model, however, adapts the security level dynamically in response to an attack by an adversary, and in absence of an adversary it adapts to optimize the overhead; thus, the application does not have to worry about classifying data sensitivity or choosing a reasonable security level.

6 Conclusion

We demonstrate how to apply the adaptive security model to a class of privacy-preserving protocols to reduce their transmission overhead while preserving the security and privacy properties of those protocols, so that these protocols can be used in low-power mHealth sensors. A more complete presentation of the protocol, and its evaluation will appear in a future paper.

Acknowledgements

We would like to thank the reviewers for their helpful comments. We also thank the members of Trust Lab at Dartmouth College for their feedback.

This research results from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under Grant Award Number 0910842 and by the Department of Health and Human Services (SHARP program) under award number 90TR0003-01. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

7 References

- [1] B. Greenstein, D. McCoy, J. Pang, T. Kohno, S. Seshan, and D. Wetherall. Improving wireless privacy with an identifier-free link layer protocol. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 40–53. ACM, June 2008. DOI 10.1145/1378600.1378607.
- [2] J. Portilla, A. Otero, E. de la Torre, T. Riesgo, O. Stecklina, S. Peter, and P. Langendörfer. Adaptable security in wireless sensor networks by using reconfigurable ECC hardware coprocessors. *International Journal of Distributed Sensor Networks*, 2010. DOI 10.1155/2010/740823.
- [3] N. R. Prasad and M. Alam. Security framework for wireless sensor networks. *Wireless Personal Communications*, 37:455–469, 2006. DOI 10.1007/s11277-006-9044-7.
- [4] The transport layer security (TLS) protocol. RFC 5246.
- [5] M. Salajegheh, A. Molina, and K. Fu. Privacy of home telemedicine: Encryption is not enough. *Journal of Medical Devices*, 3(2), Apr. 2009. Online at <http://www.cs.umass.edu/~kevinfu/papers/salajegheh-DMD09-abstract.pdf>.
- [6] L. A. Saxon, D. L. Hayes, F. R. Gilliam, P. A. Heidenreich, J. Day, M. Seth, T. E. Meyer, P. W. Jones, and J. P. Boehmer. Long-term outcome after ICD and CRT implantation and influence of remote device follow-up: The ALTITUDE survival study. *Circulation*, 122(23):2359–2367, Dec. 2010. DOI 10.1161/CIRCULATIONAHA.110.960633.
- [7] T. Shon, B. Koo, H. Choi, and Y. Park. Security architecture for IEEE 802.15.4-based wireless sensor network. In *Proceedings of the International Symposium on Wireless Pervasive Computing (ISWPC)*, pages 1–5, Feb. 2009. DOI 10.1109/ISWPC.2009.4800607.
- [8] D. Singelée and B. Preneel. Location privacy in wireless personal area networks. In *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, pages 11–18. ACM, 2006. DOI 10.1145/1161289.1161292.
- [9] C. V. Wright, L. Ballard, S. E. Coull, F. Monroe, and G. M. Masson. Uncovering spoken phrases in encrypted voice over IP conversations. *ACM Transactions on Information and System Security (TISSEC)*, 13(4), Dec. 2010. DOI 10.1145/1880022.1880029.