Distance Hijacking Attacks on Distance Bounding Protocols – Abstract

Cas Cremers
ETH Zurich
Information Security group
Zurich, Switzerland
cas.cremers@inf.ethz.ch

Kasper B. Rasmussen
University of California, Irvine
Computer Science Dept.
Irvine, California
kbrasmus@ics.uci.edu

Srdjan Čapkun
ETH Zurich
Systems Security group
Zurich, Switzerland
capkuns@inf.ethz.ch

Using a *distance bounding protocol*, a device (the verifier) can securely obtain an upper bound on its distance to another device (the prover) [1]. A number of distance bounding protocols were proposed in recent years, which provide different performance and security guarantees. So far, several distance-bounding protocols were implemented, some using digital processing and short symbols, whereas others rely on analog processing and use signal streams [4].

The security of distance-bounding protocols was so far mainly evaluated by analyzing their resilience to three attack types: *Distance Fraud*, *Mafia Fraud* and *Terrorist Fraud*. In Distance Fraud, a sole dishonest prover convinces the verifier that he is at a different distance than he really is. In Mafia Fraud, the prover is honest, but an external attacker tries to modify the measured distance by interfering with the communication. In Terrorist Fraud, a dishonest prover colludes with an attacker that is closer to the verifier, to convince the verifier of a wrong distance to the prover. So far, it was assumed that distance bounding protocols that are resilient against these three attack types, are indeed secure.

However, we show that many of these protocols, irrespective of their physical-layer implementation, are vulnerable to a fourth type of attack, which we coin *Distance Hijacking*. In Distance Hijacking attacks a dishonest prover P convinces the verifier V that P is at a distance at which some other honest prover P' resides, which differs from the actual distance from V to P. For example, one of the ways in which P can achieve this is by hijacking the distance measurement phase of a distance bounding protocol from an honest (closer or further) prover P' and inserting his own identity into messages that are not time-critical. This type of attack can pose a serious threat in many practical scenarios, including in situations where other attacks, e.g. Terrorist Fraud, may not be a concern. In Table 1 we list protocols that are vulnerable to Distance Hijacking attacks.

We propose two classes of effective and generic countermeasures that make Brands and Chaum and related protocols secure against Distance Hijacking. Our countermeasures are inexpensive, i.e., they do not require introducing additional messages or cryptographic operations.

Additionally, we show that all distance bounding protocols, including those based on the Hancke and Kuhn protocol [2], are vulnerable to Distance Hijacking if run alongside another distance bounding protocol. This can occur if more than one distance bounding protocol is used in the same environment. This result can be seen as an extension of the Chosen Protocol Attack [3]. We also generalize Distance Hijacking to Location Hijacking, and show that it is possible to hijack locations at which no other provers reside.

Protocol	Year
Brands and Chaum (Fiat-Shamir) [1]	1994
Brands and Chaum (Schnorr) [1]	1994
Brands and Chaum (signature) [1]	1994
MAD	2003
Meadows et al. for $\langle NV, NP \oplus P\rangle$	2007
Noise resilient MAD	2007
WSBC+DB	2010
WSBC+DB Noent	2010
Kuhn, Luecken, Tippenhauer	2010
CRCS [4]	2010

Table 1. Vulnerable protocols

References

[1] S. Brands and D. Chaum. Distance-bounding protocols. In *EUROCRYPT '93*, volume 765 of *LNCS*, pages 344–359. Springer, 1994.

[2] G. Hancke and M. Kuhn. An RFID distance bounding protocol. In *Proc. of IEEE/CreatNet SecureComm*, pages 67–73, 2005.

[3] J. Kelsey, B. Schneier, and D. Wagner. Protocol interactions and the chosen protocol attack. In *Proc. 5th International Workshop on Security Protocols*, volume 1361 of *LNCS*, pages 91–104. Springer, 1997.

[4] K. Rasmussen and S. Čapkun. Realization of RF distance bounding. In *USENIX Security 2010*, 2010.