

# RFID Privacy Models

This presentation is a summary of Serge Vaudenay's paper "*On Privacy Models for RFID*"



# An RFID Scheme

- $\text{SetupReader}(1^s)$  → Generates private/public key pair  $(K_S, K_P)$  according to  $s$
- $\text{SetupTag}_{K_P}(\text{ID})$  → Returns  $(K, S)$  for the tag ID
- A polynomial time interactive protocol btw a reader and a tag: Reader ends with **Output**
- The Output is said to be *correct* if  $\text{Output} = \text{ID}$  and the tag is legitimate, or  $\text{Output} = \text{Null}$  and the tag not legitimate

# Adversary Oracles

- An *Adversary* is an algorithm that takes the public input  $K_p$  and runs by using following oracles:
  - > 1. **CreateTag<sup>b</sup>(ID)**: If  $b=1$ , creates a free legitimate tag with ID. Uses  $\text{SetupTag}_{K_p}$  algorithm
  - > 2. **DrawTag(distr)  $\rightarrow$  (vtag<sub>1</sub>, b<sub>1</sub>, ..., vtag<sub>n</sub>, b<sub>n</sub>)**: Returns a vector of fresh identifiers (vtag<sub>1</sub>, ..., vtag<sub>n</sub>) in which vtag<sub>i</sub> is legitimate if  $b_i = 1$  for  $i=1, \dots, n$
  - > 3. **Free(vtag)**: Moves the vtag to the set of free tags
  - > 4. **Launch  $\rightarrow \pi$** : Makes reader launch a new protocol instance  $\pi$ .

# Adversary Oracles

- > 5. **SendReader**( $m, \pi$ )  $\rightarrow m'$ : (Resp. **SendTag**( $m, vtag$ )  $\rightarrow m'$ ) Sends the message  $m$  to the reader (resp.  $vtag$ ) in the protocol instance  $\pi$ , and gets the answer  $m'$ .
- > 6. **Execute**( $vtag$ )  $\rightarrow (\pi, transcript)$ : Runs a complete protocol consisting of one Launch query and successive SendReader and SendTag queries. It returns the protocol transcript
- > 7. **Result**( $\pi$ )  $\rightarrow x$ : When  $\pi$  is completed, it returns 1 if Output  $\neq$  Null, and 0 otherwise.
- > 8. **Corrupt**( $vtag$ )  $\rightarrow S$ : Returns the current state of the tag. If  $vtag$  is not used after this call, we say that it is *destroyed*.

# Adversary Classes

- The adversary *plays* a *game* which starts with the setting up of the RFID system and feeding the adversary with the public key. According to the class of the adversary, it uses some of the oracles obeying some *rules* and produces some output. Depending on the rules, adversary *wins* or *loses*.
- The classes are ordered as: STRONG, DESTRUCTIVE, FORWARD, and WEAK.
- All of the above classes are reduced to less capable ones using the NARROW adjective. A NARROW-X adversary is a X adversary who does not use Result oracle

# Adversary Classes

<b>Normal Classes</b>	CreateTag	DrawTag	Free	Launch	SendReader - SendTag	Result	Corrupt
<b>STRONG</b>	+	+	+	+	+	+	+
<b>DESTRUCTIVE</b>	+	+	+	+	+	+	1 times for a unique tag
<b>FORWARD</b>	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	+
<b>WEAK</b>	+	+	+	+	+	+	-

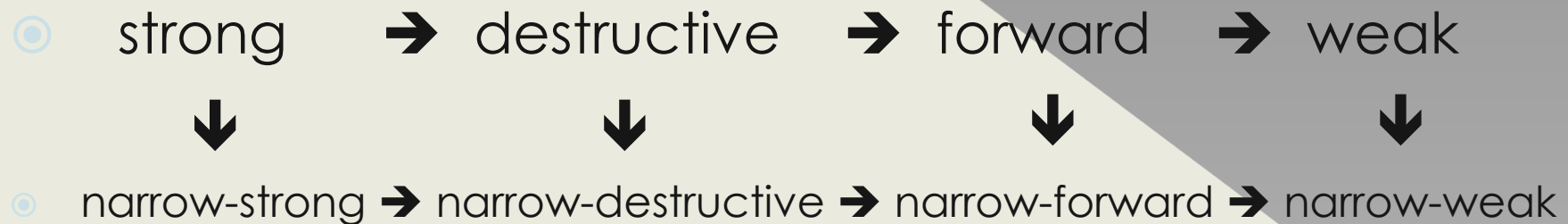
<b>Weak Classes</b>	CreateTag	DrawTag	Free	Launch	SendReader - SendTag	Result	Corrupt
<b>NARROW-STRONG</b>	+	+	+	+	+	-	+
<b>NARROW-DESTRUCTIVE</b>	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	-	1 times
<b>NARROW-FORWARD</b>	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	!Corrupt → +	-	+
<b>NARROW-WEAK</b>	+	+	+	+	+	-	-

# Privacy of RFID Schemes

- *Privacy* is defined in terms of ability to infer non-trivial ID relations from protocol messages. This generalizes the notion of *anonymity* and *untraceability*.
- A *Blinder*  $B$  for an adversary  $A$  is a polynomial time algorithm which uses the same messages as  $A$  and simulates the **Launch**, **SendReader**, **SendTag**, and **Result** queries. Blinder does not have access to reader's secret key nor the database.
- A *blinded adversary*  $A^B$  is itself an adversary who does not use above oracles

# Privacy of RFID Schemes

- For an adversary  $A$ , if there exist a blinder  $B$  s.t.  $|\Pr[A \text{ wins}] - \Pr[A^B \text{ wins}]|$  is negligible, we say that  $A$  is *trivial*. Informally, this means adversary gain no advantage using protocol messages. Since tag corruption always compromises its privacy, so below table shows privacy relations only on wireless setting:



# Privacy of RFID Schemes

strong ↗ Note 10		destructive ↗ Note 18	≠ <sup>Note 10</sup>	forward ↗ Note 18	≠ <sup>Note 14</sup>	weak ↗ Note 18
narrow-strong	≠ <sup>Note 16</sup>	narrow-destructive	≠ <sup>Note 17</sup>	narrow-forward	≠ <sup>Note 14</sup>	narrow-weak

- The non-implications show that these definitions are pairwise different.
- The missing non-implication from narrow-strong to strong privacy is equivalent to the feasibility of destructive privacy, which is an open problem.
- In a *secure* RFID scheme, the success probability of any adversary which tries to pass a protocol instance  $\pi$  and an uncorrupted tag ID with no matching conversation is negligible

# Privacy of RFID Schemes

- ◉ Lemma: In a **secure** RFID scheme, FORWARD and NARROW-FORWARD (resp. WEAK and NARROW-WEAK) PRIVACY are equivalent.
- ◉ Theorem: A destructive-private RFID scheme is **NOT** narrow-strong private
- ◉ → So STRONG Privacy is **impossible** (since strong → destructive and strong → narrow-strong)
- ◉ Theorem: A narrow-strong private RFID scheme can be transformed into a *Secure Key Agreement Protocol* (in polynomial time). Here Alice simulates SetupTag and reader, whereas Bob simulates the tag.

# Privacy of RFID Schemes

- 1. Alice:  $(K_S, K_P) \leftarrow \text{SetupReader}(1^s)$
- 2. Alice:  $(K_0, S_0) \leftarrow \text{SetupTag}_{K_P}(ID_0)$
- 3. Alice:  $(K_1, S_1) \leftarrow \text{SetupTag}_{K_P}(ID_1)$
- 3. Alice sends  $(K_P, S_0, S_1)$  to Bob and simulates the reader protocol with database  $\{(ID_0, K_0), (ID_1, K_1)\}$
- 4. Bob simulates the tag protocol with state  $S_b$  and interact with Alice
- Alice sets  $a$  such that  $ID_a = \text{Output}$

- **Theorem:** A narrow-forward private stateless RFID scheme can be transformed into a secure key agreement protocol with the same number of rounds (message rounds)

- **Source:**

- Vaudenay, S.: On Privacy Models for RFID. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 68–87. Springer, Heidelberg (2007)