

# Information Hiding, Anonymity and Privacy: A Modular Approach

Vitaly Shmatikov

SRI International

Dominic Hughes

Stanford University

# Is This Protocol “Anonymous”?

$A \rightarrow \text{clinic} \quad \{ \text{"hi"}, A, K_A \}_{pk(\text{clinic})} \quad K_A \text{ is a fresh key}$   
 $A \leftarrow \text{clinic} \quad \{ \text{"ok"} \}_{K_A} \quad \begin{array}{l} \text{if } A \text{ is a patient} \\ \text{if } A \text{ is not a patient} \end{array}$   
*nonce*

- What does “anonymity” mean?
- What does “privacy” mean *for this protocol*?
- What information should be hidden?
- How to verify that it is indeed hidden?

# Information Hiding

---

- Security often requires information hiding
- Secrecy
  - “Can someone learn this credit card number?”
- Anonymity
  - “Who sent this email?”
- Privacy
  - “Is Bob a patient of this doctor?”
- Information flow
  - “Is the *classified* process observable?”

# Attacker's Knowledge

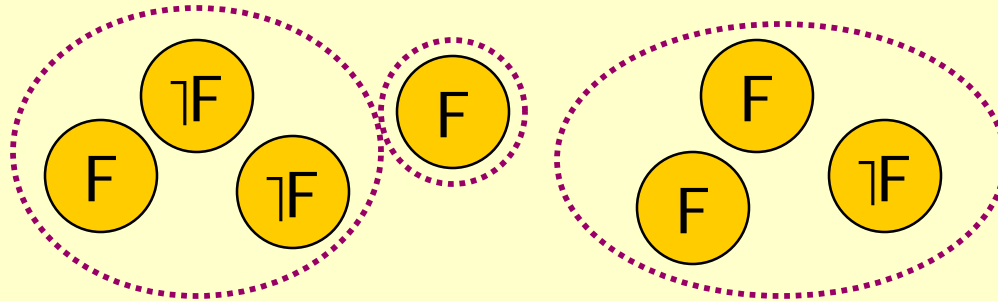
---

- Security properties can be expressed by stating what the attacker should not know
- Secrecy:                      some value
  - Shouldn't know *CardNumber* value
- Anonymity:                    msg→name function
  - Shouldn't know *senderOf(msg)*
- Privacy:                      name×name predicate
  - Shouldn't know *patientOf(name, name)*
- What does it mean to **not know a function?**

# Reasoning about Knowledge

[Hintikka, Halpern, etc.]

- Attacker's observations partition the space of system states into equivalence classes



- “Doesn't know  $F$ ” = cannot distinguish states where  $F$  is true and where  $F$  is false
- Functions have more properties than logic formulas. What does it mean to **know  $f(x)$** ?

# Partial Function Knowledge (I)

---

What does it mean to  
(partially) “know” a function?

Traditional domain theory approach [Scott et al.]:  
approximation is a subset of input-output behavior

**a** approximates **f**:  $X \rightarrow Y$  iff **a** is partial function

$$\mathbf{a} \subseteq \mathbf{f} \subseteq X \times Y$$

# Partial Function Knowledge (II)

---

What else can one know about a function?

- **Graph**

- Input-output behavior (subset of  $X \times Y$ )

- **Image**

- Subset  $\underline{\text{im}} f = \{f(x) \mid x \in X\} \subseteq Y$

- **Kernel**

- Equivalence relation (quotient) induced by  $f$   
 $(x, x') \in \underline{\text{ker}} f \iff f(x) = f(x')$

# Our Representation of Knowledge

$\langle \mathbf{F}, \mathbf{I}, \mathbf{K} \rangle$  = partial knowledge of function  $f$

- $\mathbf{F}$  is *graph knowledge*
  - Binary relation  $\mathbf{F} \subseteq X \times Y$  s.t.  $f \subseteq \mathbf{F}$
- $\mathbf{I}$  is *image knowledge*
  - Subset  $\mathbf{I} \subseteq Y$  s.t.  $\mathbf{I} \subseteq \underline{\text{im}} f$
- $\mathbf{K}$  is *kernel knowledge*
  - Equivalence relation  $\mathbf{K}$  on  $X$  s.t.  $\mathbf{K} \subseteq \underline{\text{ker}} f$

**Refinement:**  $\langle \mathbf{F}, \mathbf{I}, \mathbf{K} \rangle \leq \langle \mathbf{F}', \mathbf{I}', \mathbf{K}' \rangle$  if  $\mathbf{F}' \subseteq \mathbf{F}$ ,  $\mathbf{I} \subseteq \mathbf{I}'$ ,  $\mathbf{K} \subseteq \mathbf{K}'$

Poset of triples  $\langle \mathbf{F}, \mathbf{I}, \mathbf{K} \rangle$  ordered by  $\leq$  forms a distributive **lattice**

# Function Views

**Lineup** is a set of candidate functions

$$\text{lineup}(k) = \{f: X \rightarrow Y \mid k \leq f\}$$

For any partial knowledge  $k$ , this is the set of functions consistent with  $k$

$$\text{know}(I) = \{\langle \cup f, \cap_{\text{im}} f, \cap_{\text{ker}} f \rangle \mid f \in I\}$$

For any set of candidate functions  $I$ , this is partial knowledge implied by  $I$

1. Galois connection:  $I \subseteq \text{lineup}(k) \iff k \leq \text{know}(I)$
2. For any knowledge  $k$ , closure  $\underline{k} = \text{know}(\text{lineup}(k))$
3. A **view** of  $f: X \rightarrow Y$  is  
any closed knowledge  $k$  consistent with  $f$

# Opaqueness

- Let  $v = \langle \mathbf{F}, \mathbf{I}, \mathbf{K} \rangle$  be a view of  $f: X \rightarrow Y$ 
  - What information about  $f$  is hidden in this view?
- $v$  is **k-value opaque** if  $\forall x \ |\mathbf{F}(x)| \geq k$ 
  - On any input,  $f$  has at least  $k$  plausible values
- $v$  is **image opaque** if  $\mathbf{I}$  is empty
  - Attacker doesn't know any point in image of  $f$
- $v$  is **kernel opaque** if  $\mathbf{K}$  is equality
  - Attacker doesn't know any two inputs on which  $f$  produces equal outputs

# From Opaqueness to Verification

Easy to reason about knowledge

❶ Specify the security property as *opaqueness* of certain functions

❷ Predicates on attacker's equivalence classes

Hold *if and only if* the property is satisfied

Easy to specify system behavior

❸ Specify the system in your favorite process calculus

❹ Observational equivalence relation on system states

Any calculus provides an equivalence relation "for free"

⊕

❺ Checkable verification conditions for anonymity, privacy, etc.

# Observational Equivalence

---

$$P_1(\dots) \sim P_2(\dots)$$

- Equivalence relation between processes
  - Bisimilarity [spi-calculus]
  - Computational indistinguishability [p.p. calculus]
  - Simulatability [cryptography]
- Proof techniques available!

# Deriving Verification Conditions

- $\sim$  partitions state space of the system into observational equivalence classes  $C, C', \dots$
- $\sim$  induces a *view* of every function
  - $\text{lineup}_C(f) = \{f_{C'}: X \rightarrow Y \mid C' \sim C\}$
- Does the system hide information?
  - If and only if the views of system functions induced by  $\sim$  are opaque
- Security properties  $\Rightarrow$  opaqueness of functions  $\Rightarrow$  checkable predicates on equivalence classes

# Example: Anonymity and Privacy

---

## What the attacker shouldn't know

"Blender" anonymity

Shouldn't know sender-recipient pairs  
(Ok to know all senders, recipients)

Sender anonymity

Shouldn't know sender's identity

Recipient anonymity

... recipient's ...

(persistent pseudonyms are Ok)

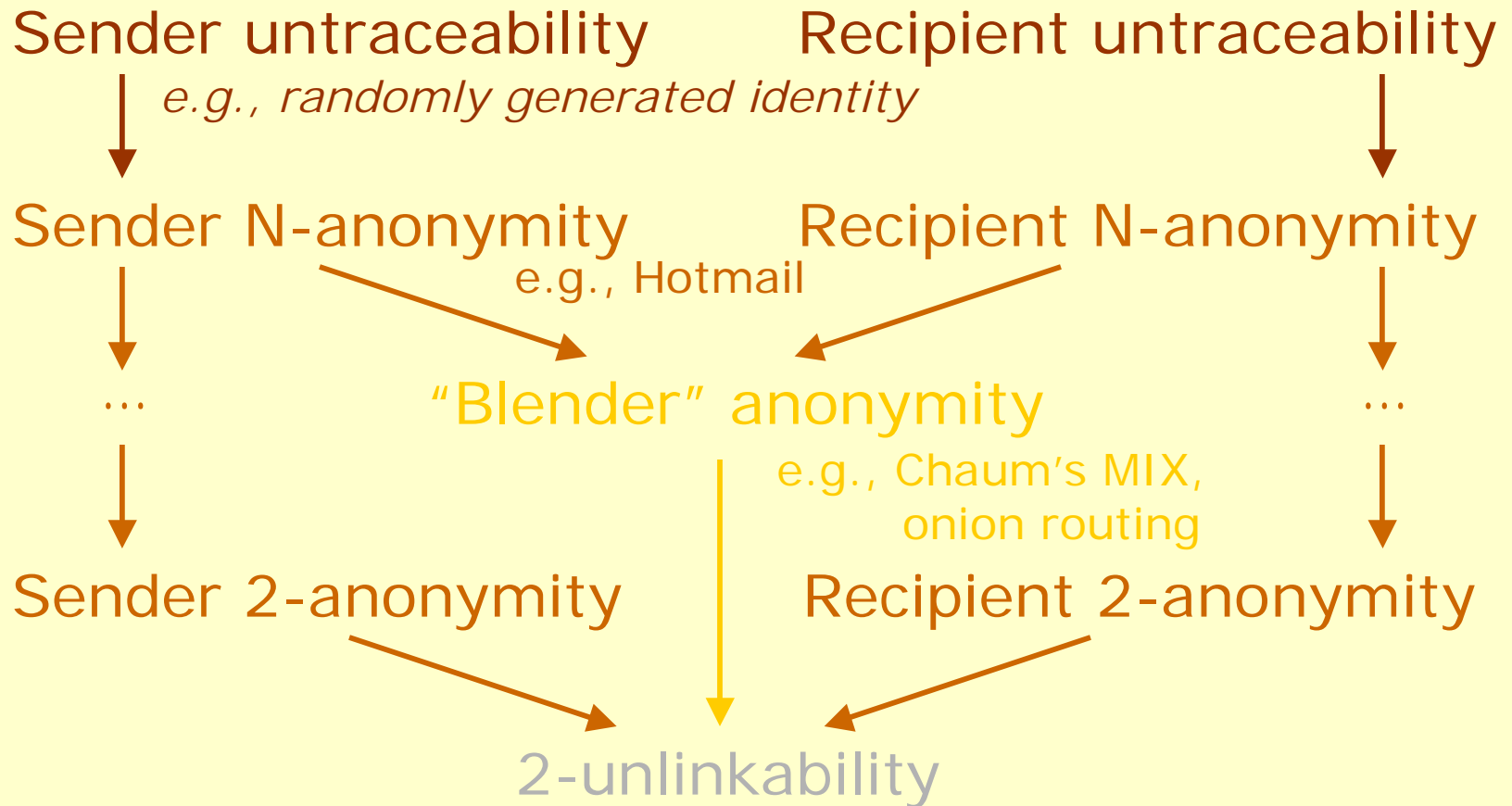
Untraceability

Shouldn't know that the same agent  
appears in different conversations

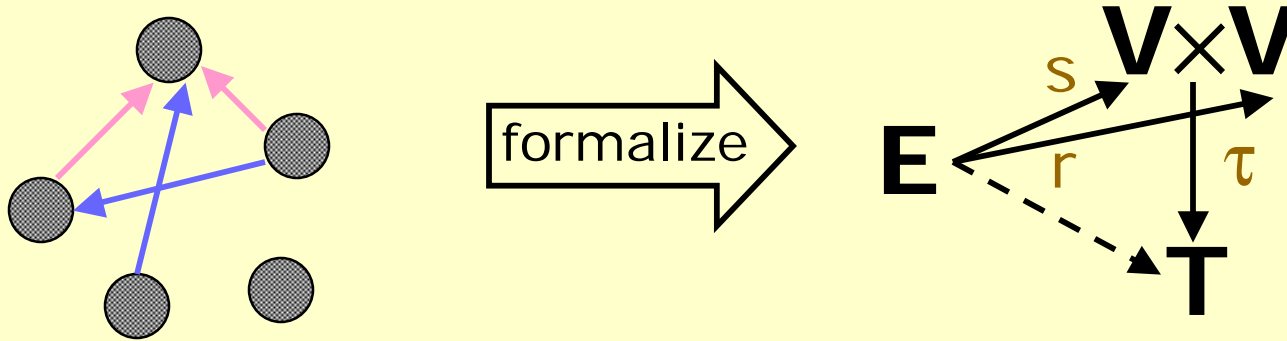
Privacy

Shouldn't know relations betw. agents

# Anonymity Properties



# Abstract Communication Graphs



- Vertices  $V$  are agents, edges  $E$  are messages
- $s: E \rightarrow V$  assigns senders,  $r: E \rightarrow V$  assigns recipients
  - **Anonymity** = opaqueness of  $s, r$
- $\tau: V \times V \rightarrow T$  models relations between agents
  - e.g.,  $\tau(\text{ph498}, \text{net}) = \text{subscriber}$ ,  $\tau(\text{ph112}, \text{net}) = \text{roaming}$
  - Different protocols may be run for different values of  $\tau$
  - **Privacy** = opaqueness of  $\tau$

# Example: "Blender" Anonymity

## Informal

Attacker knows all senders & recipients but can't find **both** sender & recipient for any msg



## Opaqueness-based definition

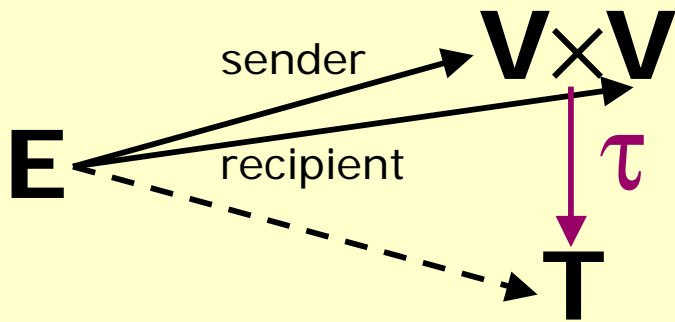
$sr_C$  is 2-value opaque, where  $sr_C: E \rightarrow V \times V$  is the sender/recipient function from edges to vertices of the communication graph  $C$

## Predicate on equivalence classes

$\forall C \forall m \exists C' \text{ s.t. } sr_C(m) \neq sr_{C'}(m) \wedge P(C) \sim P(C')$

Verifiable  
(maybe)

# Formal Model for Privacy



$\tau: V \times V \rightarrow T$  models relations between agents

For example:

- $\tau(\text{person}, \text{clinic}) \in \{\text{patient}, \text{nonpatient}\}$
- $\tau(\text{phone}, \text{service}) \in \{\text{subscr}, \text{roam}, \text{unauth}\}$

$\tau(A, B)$  determines which protocol A and B run

- Privacy = opaqueness of  $\tau$ 
  - e.g., attacker cannot find out if  $a \in V$  is a patient  
*even if the clinic-patient and clinic-nonpatient protocols differ*
- From opaqueness, derive predicate on equivalence
  - $\forall C \forall \tau \forall a, b \in V \exists C' \exists \tau' \text{ s.t. } \tau'(a, b) \neq \tau(a, b) \wedge P(C, \tau) \sim P(C', \tau')$

# Privacy vs. Anonymity

- Anonymity does not guarantee privacy

$A \rightarrow \text{clinic}: \{ \text{"hi"}, A, K_A \}_{pk(\text{clinic})}$   $K_A$  fresh

$\text{clinic} \rightarrow A: \{ \text{"ok"} \}_{K_A}$  if A is a patient  
random otherwise

Protocol protects A's identity, but for any agent attacker can learn whether she is a patient

- Privacy does not require anonymity

$A \rightarrow \text{clinic}: \{ \text{"hi"}, A \}_{pk(\text{clinic})}$

$\text{clinic} \rightarrow A: \{ \text{"ok"}, N'_C \}_{pk(A)}$  if A is a patient  
random otherwise

Attacker can learn A's identity, but not whether she is a patient

# Summary

---

- Formal definitions for all kinds of information hiding properties
  - Better understand what different properties mean and how they relate to each other
- Independence of anonymity and privacy
- Fundamental concept: function opaqueness
  - Related to equivalence in process calculus
  - Works with any process calculus formalism
  - Verifiable (but not as a trace property!)