Deniable Encryption
and its applications

Konrad Durnoga
kdr@mimuw.edu.pl

Faculty of Mathematics, Informatics and Mechanics
University of Warsaw

June 12, 2010
What is "Deniability"?

Term referring to ability of rejecting accusations for some shady activities.

Political example:

- USA government allegeably planned an assassination of Cuba’s leader
- Kennedy’s administration: "we knew nothing about this!"
- no hard evidence found
What is "Deniability"?

Term referring to ability of rejecting accusations for some shady activities.

Political example:

- USA government allegeably planned an assassination of Cuba’s leader
- Kennedy’s administration: "we knew nothing about this!"
- no hard evidence found

This is plausible deniability.
Deniability in Cryptography

Quite the opposite:

- using cryptography and concealing keys can be highly suspicious
- ciphertext/signatures form perfect commitment
- long-lived keys
- non-repudiability

We demand repudiability and perfect forward secrecy!
Deniability in Cryptography

Quite the opposite:

- using cryptography and concealing keys can be highly suspicious
- ciphertext/signatures form perfect commitment
- long-lived keys
- non-repudiability

We demand *repudiability* and *perfect forward secrecy*!
Quite the opposite:

- using cryptography and concealing keys can be highly suspicious
- ciphertext/signatures form perfect commitment
- long-lived keys
- non-repudiability

We demand repudiability and perfect forward secrecy!
Borisov, Goldberg, Brewer: Why not to use PGP in private communication.
Somewhat "traditional" approach to cryptosystems: adversary should not get any significant information about the encrypted message from the ciphertext.
Somewhat "traditional" approach to cryptosystems: adversary should not get any significant information about the encrypted message from the ciphertext (semantic security).
Somewhat "traditional" approach to cryptosystems: adversary should not get any significant information about the encrypted message from the ciphertext (semantic security).

Fundamental principle

Cryptosystem security should rely exclusively on the secrecy of keys.
Somewhat "traditional" approach to cryptosystems: adversary should not get any significant information about the encrypted message from the ciphertext (semantic security).

Fundamental principle
Cryptosystem security should rely exclusively on the secrecy of keys.

And what if key gets compromised?
Somewhat "traditional" approach to cryptosystems: adversary should not get any significant information about the encrypted message from the ciphertext (semantic security).

Fundamental principle

Cryptosystem security should rely exclusively on the secrecy of keys.

And what if key gets compromised? Usually no security is guaranteed.
People are the weakest link the system security.

Kevin Mitnick
Motivation

*People are the weakest link the system security.*

---

Kevin Mitnick

Passwords and keys are likely to be disclosed:

- social engineering
- "black-bag" cryptoanalysis (trojans, key-loggers)
- extortion/coercion
- rubber-hose cryptoanalysis
Often users are not even aware when their essential data is compromised.
In other cases...
Often users are not even aware when their essential data is compromised. In other cases...
Concern not entirely unjustified:
- in certain countries public authorities can demand handing over encryption keys, e.g. *Regulation of Investigatory Powers Act* in UK
Campaigners hit by decryption law

By Mark Ward
Technology correspondent, BBC News website

Animal rights activists are thought to be the first Britons to be asked to hand over to the police keys to data encrypted on their computers.

The request for the keys is being made under the controversial Regulation of Investigatory Powers Act (RIPA).
Coercing Keys

Often users are not even aware when their essential data is compromised. In other cases...

Concern not entirely unjustified:

▶ in certain countries public authorities can demand handing over encryption keys, e.g. *Regulation of Investigatory Powers Act* in UK

▶ political activists in non-democratic countries subject to rubber-hose cryptoanalysis
Encrypted File Systems

Remedies:

- hidden writings – steganography
Encrypting File Systems

Remedies:

- hidden writings – steganography
- encrypting file systems (e.g., TrueCrypt) with hidden partitions – at least two layers: a decoy and the confidential one
Encrypting File Systems

Remedies:

- hidden writings – steganography
- encrypting file systems (e.g. TrueCrypt) with hidden partitions – at least two layers: a decoy and the confidential one
- Rubberhose – arbitrary number of mixed hidden "partitions"
Remedies:

- hidden writings – steganography
- encrypting file systems (e.g. TrueCrypt) with hidden partitions – at least two layers: a decoy and the confidential one
- Rubberhose – arbitrary number of mixed hidden "partitions"

Implementations of plausible deniability but can we trust these programs? B. Schneier says we should not!
A fresh idea – use "distributed" self-destructing storage:

1. Encrypt data with a random key K.
2. Use M of N secret sharing to split K into parts.
3. Distribute key pieces (distributed hash table) over P2P network.
4. Keep the data but throw out local copy of K.
5. Key parts are erased after timeout – encrypted data is rendered useless afterwards.
A fresh idea – use "distributed" self-destructing storage:

- encrypt data with a some random key $K$

$K_1$, $K_2$, $K_3$, ..., $K_N$
Vanish

A fresh idea – use "distributed" self-destructing storage:

▶ encrypt data with a some random key $K$

▶ use $M$ of $N$ secret sharing to split $K$ into parts
A fresh idea – use "distributed" self-destructing storage:

- encrypt data with a some random key $K$
- use $M$ of $N$ secret sharing to split $K$ into parts
- distribute key pieces (distributed hash table) over P2P network

Konrad Durnoga

Applications of the Deniable Encryption Scheme
A fresh idea – use "distributed" self-destructing storage:

- encrypt data with a some random key $K$
- use $M$ of $N$ secret sharing to split $K$ into parts
- distribute key pieces (distributed hash table) over P2P network

- keep the data but throw out local copy of $K$
- key parts are erased after timeout – encrypted data is rendered useless afterwards
Alice is approached by the adversary that has previously intercepted a whole communication:

- Alice is asked (not too kindly) to reveal plaintext data
- the adversary requests encryption keys to be presented (together with all random choices involved)
Alice is approached by the adversary that has previously intercepted a whole communication:

- Alice is asked (not too kindly) to reveal plaintext data
- the adversary requests encryption keys to be presented (together with all random choices involved)

Is there a possibility of disclosing fake plaintexts?
Alice is approached by the adversary that has previously intercepted a whole communication:

- Alice is asked (not too kindly) to reveal plaintext data
- the adversary requests encryption keys to be presented (together with all random choices involved)

Is there a possibility of disclosing fake plaintexts?
Not quite possible in popular cryptosystems.
Not feasible at all if we additionally require that fake messages must not be senseless.
Alice is approached by the adversary that has previously intercepted a whole communication:

- Alice is asked (not too kindly) to reveal plaintext data
- the adversary requests encryption keys to be presented (together with all random choices involved)

Is there a possibility of disclosing fake plaintexts? Not quite possible in popular cryptosystems. Not feasible at all if we additionally require that fake messages must not be senseless. But this is what deniable encryption is all about!
We have a message $M$ encrypted with algorithm $E$ that uses a randomness $r$: $C = E(M, r)$.

How can we deliberately pick $r'$ and construct $M' \neq M$ such that $E(M', r') = C = E(M, r)$?
We have a message $M$ encrypted with algorithm $E$ that uses a randomness $r$: $C = E(M, r)$.

How can we deliberately pick $r'$ and construct $M' \neq M$ such that $E(M', r') = C = E(M, r)$?

We should we limit ourselves?

Deniable encryption by Canetti, Dwork, Naor and Ostrovsky:

- allows virtually any opening, i.e. almost every $M'$ matches a given ciphertext $C$
We have a message $M$ encrypted with algorithm $E$ that uses a randomness $r$: $C = E(M, r)$.

How can we deliberately pick $r'$ and construct $M' \neq M$ such that $E(M', r') = C = E(M, r)$?

We should we limit ourselves?

Deniable encryption by Canetti, Dwork, Naor and Ostrovsky:

- allows virtually any opening, i.e. almost every $M'$ matches a given ciphertext $C$
- $M$ can be decided at the moment of the attack
We have a message $M$ encrypted with algorithm $E$ that uses a randomness $r$: $C = E(M, r)$. How can we deliberately pick $r'$ and construct $M' \neq M$ such that $E(M', r') = C = E(M, r)$?

We should we limit ourselves? Deniable encryption by Canetti, Dwork, Naor and Ostrovsky:

- allows virtually any opening, i.e. almost every $M'$ matches a given ciphertext $C$
- $M$ can be decided at the moment of the attack
- formally provable deniability property
- but not quite robust and rather impractical
Translucent Sets

$t$ – some parameter

Translucent Set – an informal definition

Set $S \subset \{0, 1\}^t$ together with a trapdoor information $d$ is said to be translucent iff

- $S$ is of "moderate" size
Translucent Sets

$t$ – some parameter

**Translucent Set – an informal definition**

Set $S \subset \{0, 1\}^t$ together with a trapdoor information $d$ is said to be translucent iff

- $S$ is of "moderate" size
- one can easily draw random elements from $S$
Translucent Sets

$t$ – some parameter

**Translucent Set – an informal definition**

Set $S \subset \{0, 1\}^t$ together with a trapdoor information $d$ is said to be translucent iff

- $S$ is of "moderate" size
- one can easily draw random elements from $S$
- it is hard to tell apart a random element of $\{0, 1\}^t$ from random element of $S$
Translucent Sets

$t$ – some parameter

Translucent Set – an informal definition

Set $S \subset \{0, 1\}^t$ together with a trapdoor information $d$ is said to be translucent iff

- $S$ is of "moderate" size
- one can easily draw random elements from $S$
- it is hard to tell apart a random element of $\{0, 1\}^t$ from random element of $S$
- having $d$ it is easy to decide whether given $x \in \{0, 1\}^t$ belongs to $S$ or not
Translucent Sets

$t$ – some parameter

Translucent Set – an informal definition

Set $S \subset \{0, 1\}^t$ together with a trapdoor information $d$ is said to be translucent iff

- $S$ is of "moderate" size
- one can easily draw random elements from $S$
- it is hard to tell apart a random element of $\{0, 1\}^t$ from random element of $S$
- having $d$ it is easy to decide whether given $x \in \{0, 1\}^t$ belongs to $S$ or not

When a random $x \in S$ is picked one can obtain a convincing proof that $x$ is generated is this way!
Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
From Translucent Sets to Deniable Encryption

$n$ – some parameter

Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
- generate a $n$-tuple:

\[
\bigcirc \bigcirc \ldots \bigcirc \bigcirc \bigcirc \ldots \bigcirc
\]
$n$ – some parameter

Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
- generate a $n$-tuple:

\[ \ldots \quad S \quad \ldots \quad R \quad \ldots \quad S \]

$i$
$n$ – some parameter

Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
- generate a $n$-tuple: first $i$ terms are $S$-elements,

\[ S \text{-element – drawn uniformly from } S \]
\[ R \text{-element – drawn uniformly from } \{0, 1\}^t \]
$n$ – some parameter

Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
- generate a $n$-tuple: first $i$ terms are $S$-elements, the remainder is composed of $R$-elements
$n$ – some parameter
Deniable encryption of a single bit $b$ only:

- pick $i \in \{0, 1, \ldots, n\}$ such that $i \equiv b \pmod{2}$
- generate a $n$-tuple: first $i$ terms are $S$-elements, the remainder is composed of $R$-elements

- dishonest opening: it was $i - 1$ that was picked, not $i$
- lying not possible in case where $i = 0$
Scenario:

- adversary demands a private signing key from Alice
- adversary forces Alice to issue a signature on a message of his choice
Secretly Embedded Extortion Warning

Scenario:

- adversary demands a private signing key from Alice
- adversary forces Alice to issue a signature on a message of his choice

Alice cannot simply refuse – she has to present some real-looking data. Adversary will surely check for inconsistencies.
Scenario:

- adversary demands a private signing key from Alice
- adversary forces Alice to issue a signature on a message of his choice

Alice cannot simply refuse – she has to present some real-looking data.
Adversary will surely check for inconsistencies.

Handing over a fake key will not do. Can we offer Alice any form of cryptographical protection?
Give Alice a possibility of putting a special secret message in a signature indicating that signature is forced. Such a warning is only readable by a fixed trusted party – Savior of the Damned. The trusted can call the Police and send some help to Alice.
Give Alice a possibility of putting a special secret message in a signature indicating that signature is forced. Such a warning is only readable by a fixed trusted party – *Savior of the Damned*. The trusted can call the Police and send some help to Alice.

Ideen:
- introduce an additional key $K$
Give Alice a possibility of putting a special secret message in a signature indicating that signature is forced. Such a warning is only readable by a fixed trusted party – *Savior of the Damned*. The trusted can call the Police and send some help to Alice.

Ideas:

- introduce an additional key $K$

Alice may very well lie about the second key $K$ – without any severe consequences.
Secretly Embedded Extortion Warning

Give Alice a possibility of putting a special secret message in a signature indicating that signature is forced. Such a warning is only readable by a fixed trusted party – *Savior of the Damned*. The trusted can call the Police and send some help to Alice.

Ideas:
- introduce an additional key $K$
- use the deniable encryption obviously

Alice may very well lie about the second key $K$ – without any severe consequences.
Secretly Embedded Extortion Warning

Give Alice a possibility of putting a special secret message in a signature indicating that signature is forced. Such a warning is only readable by a fixed trusted party – *Savior of the Damned*. The trusted can call the Police and send some help to Alice.

Ideas:

- introduce an additional key $K$
- use the deniable encryption obviously
- transfer potential warnings transparently via subliminal channel

Alice may very well lie about the second key $K$ – without any severe consequences.
What can we expect?

- anyone is able to verify whether signature is valid

- voluntary signatures are indistinguishable from coerced ones

- trusted party can extract extortion message sent subliminally

- the adversary cannot craft a signature that will considered as a voluntary one

Signatures issued by the adversary are perfectly acceptable for ordinary receivers (but not the trusted party)!
What can we expect?

- anyone is able to verify whether signature is valid
- voluntary signatures are indistinguishable from coerced ones
Signature Properties

What can we expect?

- anyone is able to verify whether signature is valid
- voluntary signatures are indistinguishable from coerced ones
- trusted party can extract extortion message sent subliminally

Konrad Durnoga
Applications of the Deniable Encryption Scheme
Signature Properties

What can we expect?

- anyone is able to verify whether signature is valid
- voluntary signatures are indistinguishable from coerced ones
- trusted party can extract extortion message sent subliminally
- the adversary cannot craft a signature that will considered as a voluntary one
Signature Properties

What can we expect?

- anyone is able to verify whether signature is valid
- voluntary signatures are indistinguishable from coerced ones
- trusted party can extract extortion message sent subliminally
- the adversary cannot craft a signature that will considered as a voluntary one

Signatures issued by the adversary are perfectly acceptable for ordinary receivers (but not the trusted party)!
What we have so far:

- $K$ – a shared secret for Alice and the trusted party
- the trusted party possesses a trapdoor information $d$ for decrypting deniable ciphertexts
- Alice employs an arbitrary signature of the form $(\text{Sig}(M, R), \text{Ver}(M, \sigma))$
Signature Overview

What we have so far:

- \( K \) – a shared secret for Alice and the trusted party
- the trusted party possesses a trapdoor information \( d \) for decrypting deniable ciphertexts
- Alice employs an arbitrary signature of the form \((\text{Sig}(M, R), \text{Ver}(M, \sigma))\)

\( H \) – random hash function

Signing \( M \):

- compute a deniable encryption: \( R := E(H(M \oplus K)) \)
What we have so far:

- $K$ – a shared secret for Alice and the trusted party
- the trusted party possesses a trapdoor information $d$ for decrypting deniable ciphertexts
- Alice employs an arbitrary signature of the form $(\text{Sig}(M, R), \text{Ver}(M, \sigma))$

$H$ – random hash function

Signing $M$:
- compute a deniable encryption: $R := E(H(M \oplus K))$
- $R$ is quasi-random
What we have so far:

- $K$ – a shared secret for Alice and the trusted party
- the trusted party possesses a trapdoor information $d$ for decrypting deniable ciphertexts
- Alice employs an arbitrary signature of the form $(\text{Sig}(M, R), \text{Ver}(M, \sigma))$

$H$ – random hash function

Signing $M$:

- compute a deniable encryption: $R := E(H(M \oplus K))$
- $R$ is quasi-random
- Alice computes the inner signature: $\sigma := \text{Sig}(M, R)$

A complete signature is $(M, R, \sigma)$. 
Signature verification phase:

- ordinary verification – use verification of the inner signature $\text{Ver}(M, \sigma)$
- opening subliminal warning message – deciphering $R$ using trapdoor $d$ and comparing the result with $H(M \oplus K)$
Interesting applications of the deniable encryption:
- electronic voting protocol – protection against vote buying
- secure multiparty computation