Data Is a Stream

Security of Stream-Based Channels

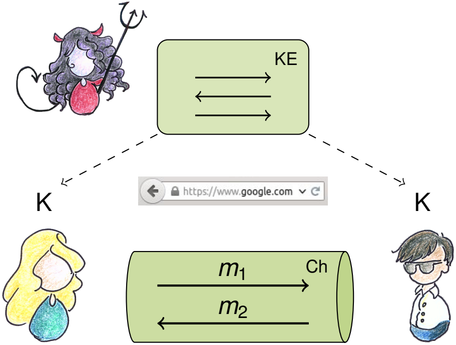
Felix Günther

Technische Universität Darmstadt, Germany

joint work with Marc Fischlin, Giorgia Azzurra Marson, and Kenneth G. Paterson

Secure Communication Needs Secure Channels

TECHNISCHE UNIVERSITÄT DARMSTADT



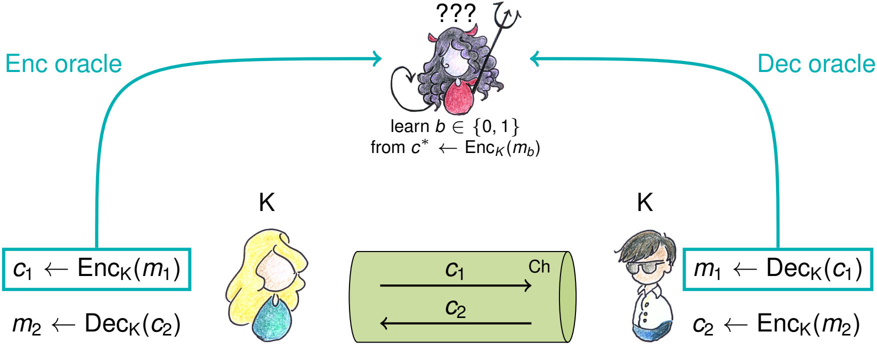
What's that secure channel precisely?

drawings by *Giorgia Azzurra Marson*

On the Origin of Channel Models

Encryption

TECHNISCHE UNIVERSITÄT DARMSTADT



IND-CPA

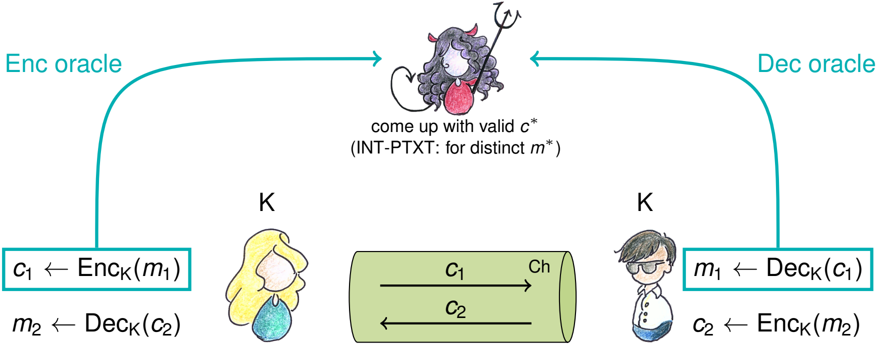
(Goldwasser, Micali 1984)

IND-CCA

(Naor, Yung 1990), (Rackoff, Simon 1991)

On the Origin of Channel Models

Integrity



Authenticated Encryption

IND-CPA + INT-CTXT

($\Rightarrow$ IND-CCA)

INT-PTXT

(Bellare, Namprempre 2000)

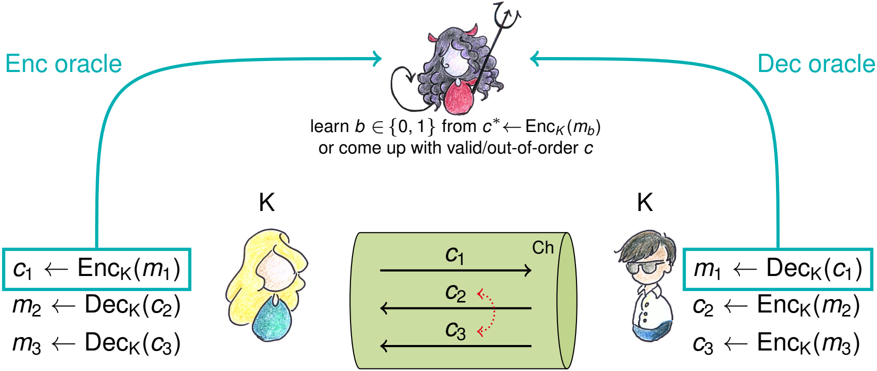
INT-CTXT

(Bellare, Rogaway 2000)

On the Origin of Channel Models

Stateful Authenticated Encryption

TECHNISCHE UNIVERSITÄT DARMSTADT



Stateful Authenticated Encryption

IND-sfCCA

used to analyze SSH

INT-sfCTXT

(Bellare, Kohno, Namprempre 2002)

INT-sfPTXT

(Brzuska, Smart, Warinschi, Watson 2013)

On the Origin of Channel Models

(Stateful) Authenticated Encryption+

TECHNISCHE UNIVERSITÄT DARMSTADT

- Authenticated Encryption with **Associated Data** — AEAD
  (Rogaway 2002)
  - ciphertext carries additional unencrypted, but authenticated data field

- **Length-Hiding** Authenticated Encryption (with AD) — LH-AEAD
  (Paterson, Ristenpart, Shrimpton 2011)
  - hides message length up to some granularity (padding)
  - used to analyze TLS record layer (within ACCE framework)

Stateful Length-Hiding Authenticated Encryption
is the accepted security notion for channels to date, so we're done?

Attack on SSH

TECHNISCHE UNIVERSITÄT DARMSTADT

Albrecht, Paterson, Watson 2009: **plaintext recovery attack against SSH**
(SSH Binary Packet Protocol with CBC-mode Encode-then-Encrypt&MAC)

- basic idea:
  - packet length field encrypted in first ciphertext block
  - MAC verification depends on decrypted length value
  - adversary feeds ciphertext in *block-wise* (via TCP fragmentation)
  - observable MAC failure leaks content of length field
  - put arbitrary ciphertext block as first block to leak $|\text{len}|$ bits
- clearly breaks confidentiality

Wait. . .

- SSH was proven IND-sfCCA and INT-sfCTXT secure! (BKN 2002)
- . . . but these only allow *atomic* ciphertexts in Dec oracle

On the Origin of Channel Models
Symmetric Encryption Supporting Fragmentation

TECHNISCHE UNIVERSITÄT DARMSTADT

Paterson, Watson 2010

- new model to analyze SSH(-CTR), IND-“buffered stateful decryption”-CCA

Boldyreva, Degabriele, Paterson, Stam 2012:

Symmetric Encryption Supporting Fragmentation

- general security model for ciphertext fragmentation
- security notion: IND-sfCFA (chosen-fragment attack)
  - standard Enc algorithm (and left-or-right oracle)
  - Dec algorithm obtains ciphertext fragments, outputs messages separated with ¶
  - (focuses on confidentiality)

Are we there yet?

Attacks on TLS

Truncating Connections and Cutting Cookies

TECHNISCHE UNIVERSITÄT DARMSTADT

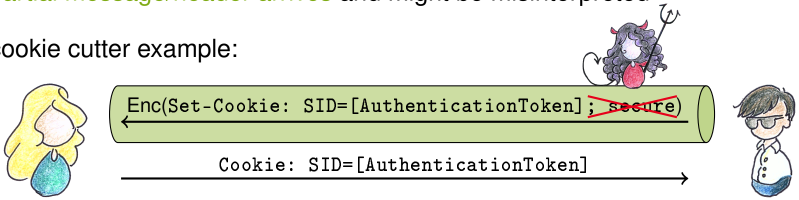
Smyth, Pironti 2013: **truncation attack**

- attacker truncates TLS connection by closing underlying TCP connection
- thereby drops (parts of) messages, potentially corrupting web application logic

Bhargavan, Delignat-Lavaud, Fournet, Pironti, Strub 2014: **cookie cutter attack**

- attacker forces part of the HTTP header (e.g., cookie) to be cut off
- partial message/header arrives and might be misinterpreted

- cookie cutter example:



Wait. . . Deleting message parts within ciphertext—how can this be possible?

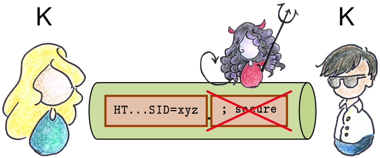
Cookie Cutter Attack

A Closer Look

TECHNISCHE UNIVERSITÄT DARMSTADT

$c \leftarrow \mathrm{Enc}($`HTTP/1.1 200 OK`

```
    ...
    Set-Cookie: SID=xyz; secure)
```




```
HTTP/1.1 200 OK
...
Set-Cookie: SID=xyz
```

```
#include <openssl/ssl.h>
SSL_write("HTTP/1.1 200 OK
        ...
        Set-Cookie: SID=xyz; secure")
```

HTTP/1.1 200 OK ... Set-Cookie: SID=xyz	; secure

2 TLS records

Data Is a Stream!

TECHNISCHE UNIVERSITÄT DARMSTADT

- That behavior is actually okay—and specified:

  6.2.1. Fragmentation
  The record layer fragments information blocks into TLSPlaintext records [...]. Client ***message boundaries are not preserved*** *in the record layer (i.e., multiple client messages of the same ContentType MAY be coalesced into a single TLSPlaintext record, or a single message MAY be fragmented across several records).*

  RFC 5246 TLS v1.2

- TLS never promised to treat messages atomically!

- au contraire: 2^{14} bytes maximum message length will lead to fragmentation
- some implementations don't even guarantee to send at all on `SSL_write`, but have a separate `flush` command (e.g., MS.NET)

Data Is a Stream!

. . . and TLS is not alone

- many important channel protocols treat data as a stream
  - TLS
  - SSH tunnel-mode
  - QUIC
- meant as secure drop-in replacement for TCP (which works on streams)
- channel models so far don’t capture this behavior exposed to the application

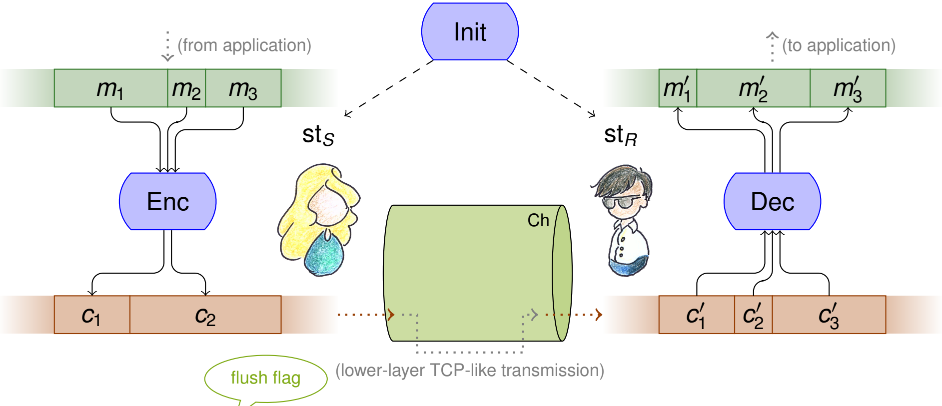
Stream-Based Channels

Overview & Syntax

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\mathsf{Init}(1^\lambda) \to \mathsf{st}_S \in \mathcal{S}_S, \mathsf{st}_R \in \mathcal{S}_R$$

$(K \in \mathsf{st}_S/\mathsf{st}_R)$



$$\mathsf{Enc}(\mathsf{st}_S, m \in \{0,1\}^*, f \in \{0,1\}) \to c \in \{0,1\}^* \qquad \mathsf{Dec}(\mathsf{st}_R, c \in \{0,1\}^*) \to m \in \{0,1\}^* \cup \mathcal{E}$$

Stream-Based Channels
Properties

TECHNISCHE UNIVERSITÄT DARMSTADT

- no particular input/output behavior stipulated on sender side
  - allow for buffering (e.g., optimization for lower layer)
    output c can even be empty
  - `flush` command modeled with flush flag $f \in \{0,1\}$
    $f = 1 \Rightarrow$ all message fragments sent out instantaneously

Correctness

$$\text{if } \| \mathbf{c} = \| \mathbf{c}' \text{ then } \| \mathbf{m}[1, \ldots, i] \preceq \| \mathbf{m}' \preceq \| \mathbf{m}$$

received message stream is **prefix** of sent stream

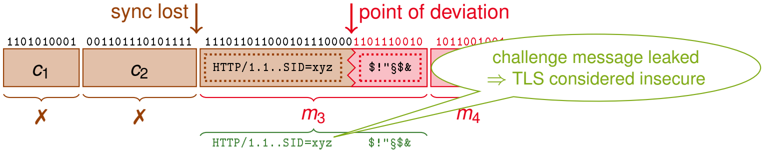
received message stream contains **everything upto last flush**

for

- sent/received ciphertext (fragments) $\mathbf{c}/\mathbf{c}'$
- sent/received message fragments $\mathbf{m}/\mathbf{m}'$
- i-th Enc the last flushing call ($f = 1$)

Stream-Based Channels
Confidentiality

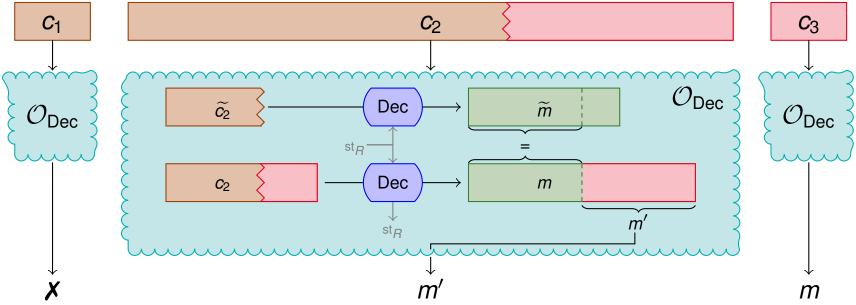
- CPA case straightforward: left-or-right oracle allowing to control flush flag
- CCA case more complex:
  - general idea: allow as much decryption as possible, but no trivial attacks
  - Bellare-Kohno-Namprempre approach: Dec oracle $\mathcal{O}_{\mathrm{Dec}}$ can be in/out of sync
    - in sync (original ciphertext stream): no output
    - out of sync (deviation from original stream): Dec output given to adversary
  - But where exactly shall $\mathcal{O}_{\mathrm{Dec}}$ / ciphertext stream be considered out-of-sync?
    - BDPS 2012: at ciphertext boundaries



Stream-Based Channels

Confidentiality

- key insight: there is no inherent structure on a stream!
  - think: Enc generates ciphertext stream as "message stream $\oplus$ keystream"
- $\mathcal{O}_{\mathsf{Dec}}$ behavior
  - in-sync / already out-of-sync cases as always: output nothing / everything
  - loosing sync: strip longest common prefix with output of genuine ciphertext part

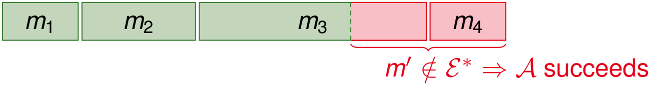


Stream-Based Channels

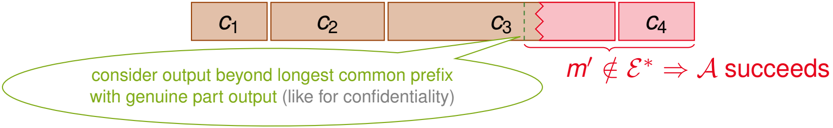
Integrity

(first consideration of integrity in non-atomic setting)

- plaintext-stream integrity
  no adversary can make received message stream deviate from sent stream



- ciphertext-stream integrity
  no adversary can make message bits being output after point of deviation



! stream-based confidentiality/integrity allow (genuine) **“partial message” output**
(would be considered as breaking security in atomic (and BDPS 2012) setting)

Relations & Composition Result

TECHNISCHE UNIVERSITÄT DARMSTADT

Classic implications hold:

- chosen ciphertext-fragment confidentiality $\Rightarrow$ chosen plaintext-fragment conf.
- ciphertext-stream integrity $\Rightarrow$ plaintext-stream integrity

Classic composition result: IND-CPA + INT-CTXT $\Rightarrow$ IND-CCA (BN 2000)

- idea: when $\mathcal{A}$ gets any $\mathcal{O}_{\mathsf{Dec}}$ output, it broke integrity; let $\mathcal{B}$ always return $\perp$
- multi-error setting: need additional "error invariance" property (BDPS 2013)

  at most one error with non-negl. probability

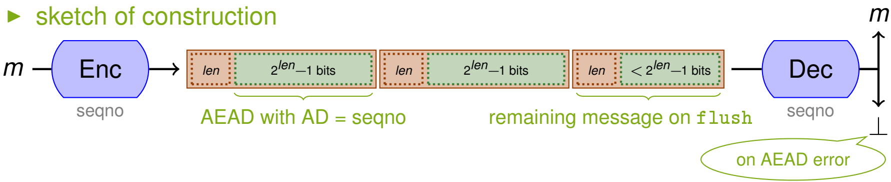
- composition in stream-based setting:
  - inherently "multi-error": Dec output on deviating ciphertext can be $\perp$ *or* empty
  - we require predictability of errors by an efficient algorithm (given sent/received ciphertext stream and next ciphertext fragment)
  - sounds strong, but is achievable by natural constructions
  - also extends to atomic setting with multiple non-negligible errors

Generic Construction

TECHNISCHE UNIVERSITÄT DARMSTADT

- secure stream-based channels can be built
  - based on authenticated encryption with associated data (AEAD)
  - achieving strong (CCA-like) confidentiality
  - achieving strong (CTXT-like) integrity

- sketch of construction



- example scheme satisfying error predictability (composition theorem used)
  unencrypted length field allows to predict when error $\perp$ is output

- close to TLS record layer design using AEAD (providing some validation)
  - ✓ unsent sequence number as authenticated AD
  - ✓ sent length field, unauthenticated (in TLS 1.3)
  - ✗ TLS additionally includes: version number, content type (sent + authenticated)

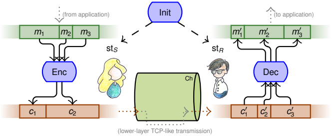
Summary

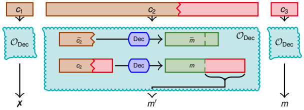
TECHNISCHE UNIVERSITÄT DARMSTADT

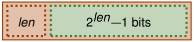
Data is a stream!

We

- formalize stream-based channels
- give adequate security notions and a composition result
- provide an AEAD-based construction







Ongoing / Future Work

- explore exact relation between atomic and stream-based notions
- what is length-hiding on a stream?
- multiplexing several data streams into one channel
- how to safely encode atomic messages in a stream?

Thank You!

References I

[1] M. R. Albrecht, K. G. Paterson, and G. J. Watson.
Plaintext recovery attacks against SSH.
In *IEEE Symposium on Security and Privacy (S&P 2009)*, pages 16–26. IEEE Computer Society, 2009.

[2] M. Bellare, T. Kohno, and C. Namprempre.
Authenticated encryption in SSH: provably fixing the SSH binary packet protocol.
In *ACM Conference on Computer and Communications Security, CCS 2002*, pages 1–11. ACM, 2002.

[3] M. Bellare and C. Namprempre.
Authenticated encryption: Relations among notions and analysis of the generic composition paradigm.
In *Advances in Cryptology - ASIACRYPT 2000*, pages 531–545. Springer, 2000.

[4] M. Bellare and P. Rogaway.
Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography.
In *Advances in Cryptology - ASIACRYPT 2000*, pages 317–330. Springer, 2000.

[5] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Pironti, and P. Strub.
Triple handshakes and cookie cutters: Breaking and fixing authentication over TLS.
In *IEEE Symposium on Security and Privacy, SP 2014*, pages 98–113. IEEE Computer Society, 2014.

[6] A. Boldyreva, J. P. Degabriele, K. G. Paterson, and M. Stam.
Security of symmetric encryption in the presence of ciphertext fragmentation.
In *Advances in Cryptology - EUROCRYPT 2012*, pages 682–699. Springer, 2012.

References II

TECHNISCHE UNIVERSITÄT DARMSTADT

[7] A. Boldyreva, J. P. Degabriele, K. G. Paterson, and M. Stam.
On symmetric encryption with distinguishable decryption failures.
In *Fast Software Encryption - 20th International Workshop, FSE 2013*, pages 367–390. Springer, 2013.

[8] C. Brzuska, N. P. Smart, B. Warinschi, and G. J. Watson.
An analysis of the EMV channel establishment protocol.
In *ACM SIGSAC Conference on Computer and Communications Security, CCS'13*, pages 373–386. ACM, 2013.

[9] T. Dierks and E. Rescorla.
The Transport Layer Security (TLS) Protocol Version 1.2.
RFC 5246 (Proposed Standard), Aug. 2008.

[10] S. Goldwasser and S. Micali.
Probabilistic encryption.
J. Comput. Syst. Sci., 28(2):270–299, 1984.

[11] M. Naor and M. Yung.
Public-key cryptosystems provably secure against chosen ciphertext attacks.
In *ACM Symposium on Theory of Computing*, pages 427–437. ACM, 1990.

[12] K. G. Paterson, T. Ristenpart, and T. Shrimpton.
Tag size does matter: Attacks and proofs for the TLS record protocol.
In *Advances in Cryptology - ASIACRYPT 2011*, pages 372–389. Springer, 2011.

References III

[13] K. G. Paterson and G. J. Watson.
Plaintext-dependent decryption: A formal security treatment of SSH-CTR.
In *Advances in Cryptology - EUROCRYPT 2010*, pages 345–361. Springer, 2010.

[14] C. Rackoff and D. R. Simon.
Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack.
In *Advances in Cryptology - CRYPTO '91*, pages 433–444. Springer, 1991.

[15] P. Rogaway.
Authenticated-encryption with associated-data.
In *ACM Conference on Computer and Communications Security, CCS 2002*, pages 98–107. ACM, 2002.

[16] B. Smyth and A. Pironti.
Truncating TLS connections to violate beliefs in web applications.
In *7th USENIX Workshop on Offensive Technologies, WOOT '13*. USENIX Association, 2013.