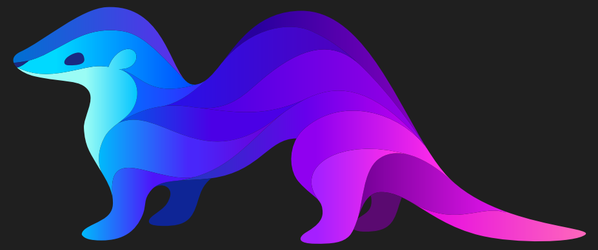
TLP:AMBER



lutra security

KOBOLD LETTERS AND OTHER MISCHIEF

HOW EMAILS CAN DECEIVE YOU

lutra security

In a world where we taught people how to spot phishing...

lutra security

In a world where we taught people how to spot phishing...
...attackers would write more convincing phishing emails.

lutra security

KONSTANTIN WEDDIGE

- Co-founder of Lutra Security
- Mathematician and Security Researcher
- Local Politician BA Milbertshofen

lutra security

WHY DOES PHISHING EVEN WORK?

lutra security

PHISHING EXPLOITS TRUST

- in the situation
- in the sender
- in the medium (e.g. email)

lutra security

KOBOLD LETTERS

- Conditional formatting of emails
- Different content before and after forwarding

lutra security

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter {
      display: none;
    }

    .moz-text-html>div>.kobold-letter {
      display: block !important;
    }
  </style>
</head>

<body>
  <p>This text is always visible.</p>
  <p class="kobold-letter">This text will only appear after forwarding.</p>
</body>

</html>
```

lutra security

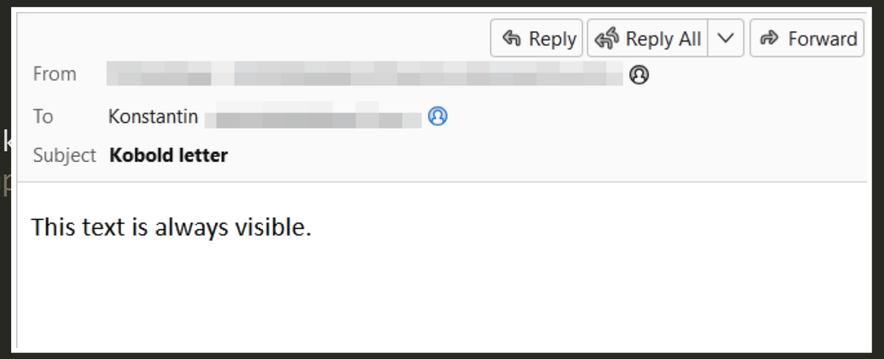
```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter {
      display: none;
    }

    .moz-text-html>div>.k
      display: block !imp
    }
  </style>
</head>

<body>
  <p>This text is always visible.</p>
  <p class="kobold-letter">This text will only appear after forwarding.</p>
</body>

</html>
```



lutra security

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter
      display: non
    }

    .moz-text-html
      display: blo
    }
  </style>
</head>

<body>
  <p>This text is
  <p class="kobold
</body>

</html>
```

File Edit View Insert Format Options Security Tools Help

Send | Encrypt | S/MIME | Spelling | Save | Attach | CardBook

From Konstantin Weddige [redacted] | Cc Bcc

To [redacted]

Subject Fwd: Kobold letter

Paragraph | Variable Width

fyi

-------- Forwarded Message --------

Subject:	Kobold letter
Date:	Fri, 25 Oct 2024 07:32:58 +0000
From:	[redacted]
Reply-To:	[redacted]
To:	Konstantin [redacted]

This text is always visible.

English (United Kingdom)

lutra security

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter
      display: non
    }

    .moz-text-html
      display: blo
    }
  </style>
</head>

<body>
  <p>This text is
  <p class="kobold
</body>

</html>
```

File Edit View Insert Format Options Security Tools Help

Send

CardBook

From Me

To Me

Reply Forward

Subject **Fwd: Kobold letter**

fyi

-------- Forwarded Message --------
Subject: Kobold letter
Date: Fri, 25 Oct 2024 07:32:58 +0000
From:
Reply-To:
To: Konstantin

This text is always visible.

This text will only appear after forwarding.

Paragraph

fyi

-------- Fo

Subject:
Date:
From:
Reply-To:
To:

This text i

English (United Kingdom)

lutra security

WHAT CAN WE DO AGAINST IT?

- Get rid of HTML email
  ⇨ nice, but unrealistic
- Remove complicated formatting
  ⇨ e.g. Thunderbirds Simple HTML
- Forbid `<style>` blocks in email
  ⇨ reliable, but will ruin most newsletters

lutra security

THE GOOGLE CASE

lutra security

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter {
      display: none;
    }
  </style>
</head>

<body>
  <p>This text is always visible.</p>
  <p class="kobold-letter">This text will only appear after forwarding.</p>
</body>

</html>
```

lutra security

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .kobold-letter
      display: none
    }
  </style>
</head>

<body>
  <p>This text is a
  <p class="kobold-
</body>

</html>
```

Kobold letter Inbox ×

[redacted]
to Konstantin

Hello,

We had issues to transfer the money for the transaction 1234.

Could you please forward this to your accountant.

Mr. Hacker

K [redacted]

Please deal with this

---------- Forwarded message ---------
From: [redacted]
Date: Thu, 24 Oct 2024 at 16:53
Subject: Kobold letter
To: Konstantin [redacted]@gmail.com>

Hello,

We had issues to transfer the money for the transaction 1234.

Could you please forward this to your accountant.

Mr. Hacker

Send

lutra security

```
<!DOCTYPE ht
<html>

<head>
  <style>
    .kobold-
      displa
    }
  </style>
</head>

<body>
  <p>This te
  <p class="
</body>

</html>
```

Kobold letter Inbox ×

to Konstantin

From Konstantin Weddige @googlemail.com> Reply Forward Archive Junk Delete More

To Me 24.10.2024, 16:54

Subject **Fwd: Kobold letter**

Please deal with this.

---------- Forwarded message ---------
From:
Date: Thu, 24 Oct 2024 at 16:53
Subject: Kobold letter
To: Konstantin @gmail.com>

Hello Konstantin,

I just got confirmation that the drawing I told you about last week is a genuine van Gogh.

We had it carbon-dated and our expert found no issues. If you manage to transfer the money by tonight, I can secure it for you. Please use "VG-000" as the reason for the transaction.
The amount would be as discussed 1234000€ and the account DE02100100100006820101.

Could you please give my regards to your wife? I look forward to playing golf with you both this weekend to celebrate this deal. But be prepared to add a few more losses to your score. I have been playing golf with my accountant twice a week.

Mr. Hacker

Send

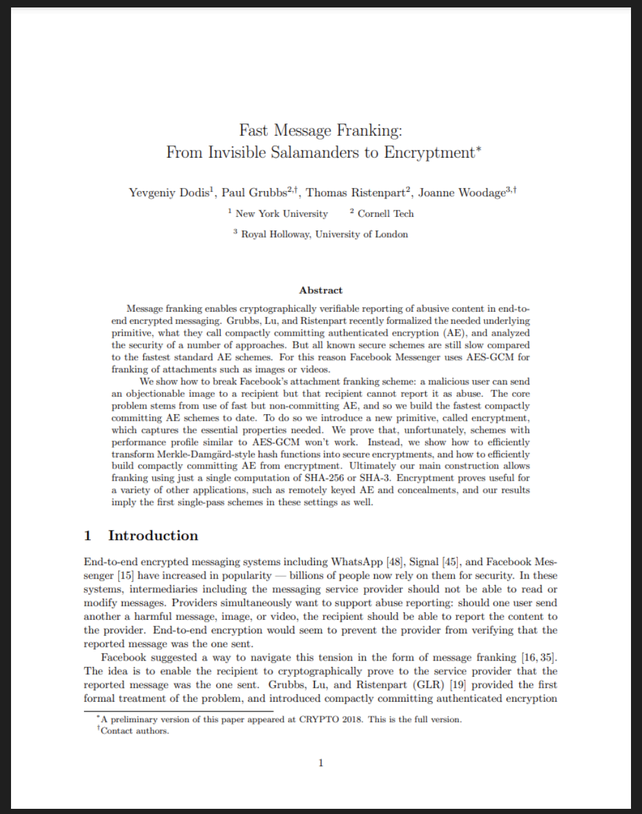
lutra security

SALAMANDER MIME

- Exploitation of missing key commitment in S/MIME
- Different content for each recipients

lutra security

INVISIBLE SALAMANDERS



Fast Message Franking:
From Invisible Salamanders to Encryptment*

Yevgeniy Dodis[1], Paul Grubbs[2,†], Thomas Ristenpart[2], Joanne Woodage[3,†]
[1] New York University [2] Cornell Tech
[3] Royal Holloway, University of London

Abstract

Message franking enables cryptographically verifiable reporting of abusive content in end-to-end encrypted messaging. Grubbs, Lu, and Ristenpart recently formalized the needed underlying primitive, what they call compactly committing authenticated encryption (AE), and analyzed the security of a number of approaches. But all known secure schemes are still slow compared to the fastest standard AE schemes. For this reason Facebook Messenger uses AES-GCM for franking of attachments such as images or videos.

We show how to break Facebook's attachment franking scheme: a malicious user can send an objectionable image to a recipient but that recipient cannot report it as abuse. The core problem stems from use of fast but non-committing AE, and so we build the fastest compactly committing AE schemes to date. To do so we introduce a new primitive, called encryptment, which captures the essential properties needed. We prove that, unfortunately, schemes with performance profile similar to AES-GCM won't work. Instead, we show how to efficiently transform Merkle-Damgård-style hash functions into secure encryptments, and how to efficiently build compactly committing AE from encryptment. Ultimately our main construction allows franking using just a single computation of SHA-256 or SHA-3. Encryptment proves useful for a variety of other applications, such as remotely keyed AE and concealments, and our results imply the first single-pass schemes in these settings as well.

1 Introduction

End-to-end encrypted messaging systems including WhatsApp [48], Signal [45], and Facebook Messenger [15] have increased in popularity — billions of people now rely on them for security. In these systems, intermediaries including the messaging service provider should not be able to read or modify messages. Providers simultaneously want to support abuse reporting: should one user send another a harmful message, image, or video, the recipient should be able to report the content to the provider. End-to-end encryption would seem to prevent the provider from verifying that the reported message was the one sent.

Facebook suggested a way to navigate this tension in the form of message franking [16, 35]. The idea is to enable the recipient to cryptographically prove to the service provider that the reported message was the one sent. Grubbs, Lu, and Ristenpart (GLR) [19] provided the first formal treatment of the problem, and introduced compactly committing authenticated encryption

*A preliminary version of this paper appeared at CRYPTO 2018. This is the full version.
†Contact authors.

1

Fast Message Franking: From Invisible Salamanders to Encryptment

Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage

https://ia.cr/2019/016

- Attack against Facebook's attachment franking scheme
- Missing key commitment
- AES-GCM

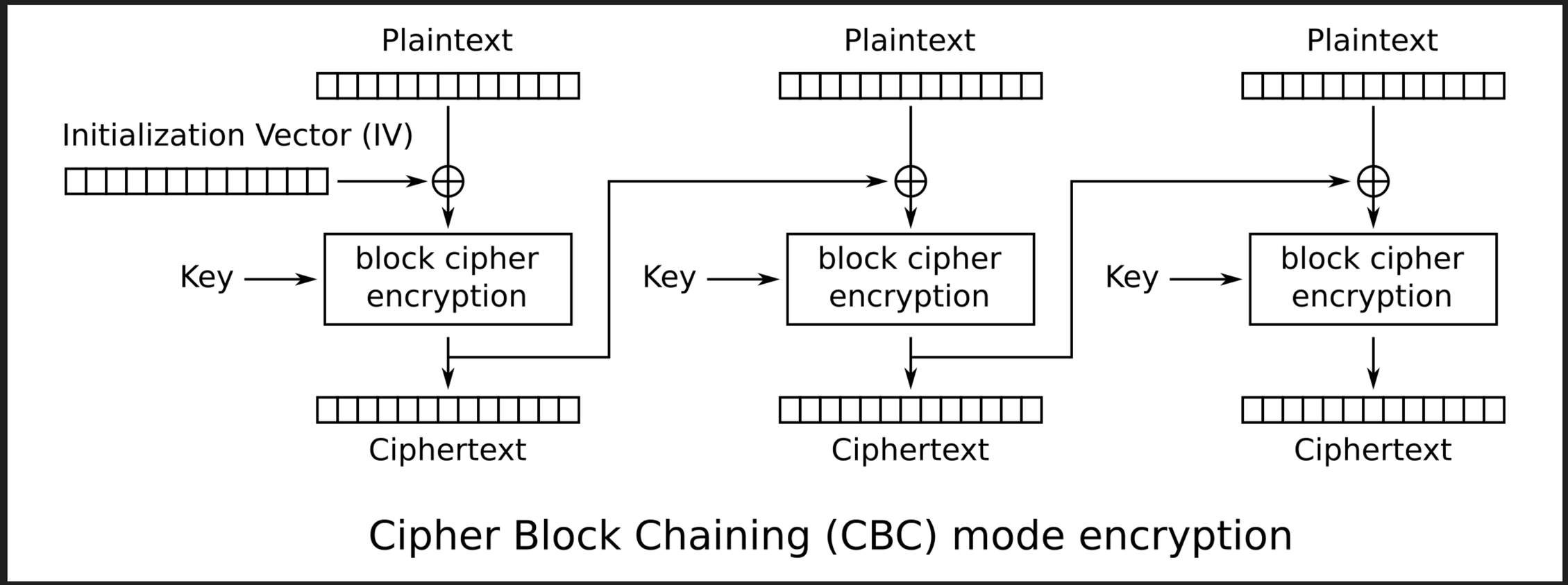
lutra security

ADOPTION FOR S/MIME

- We'll use AES-CBC instead of AES-GCM
- Our TODOs:
  1. Adapt the concept for CBC
  2. Construct suitable emails
  3. Assemble everything

lutra security

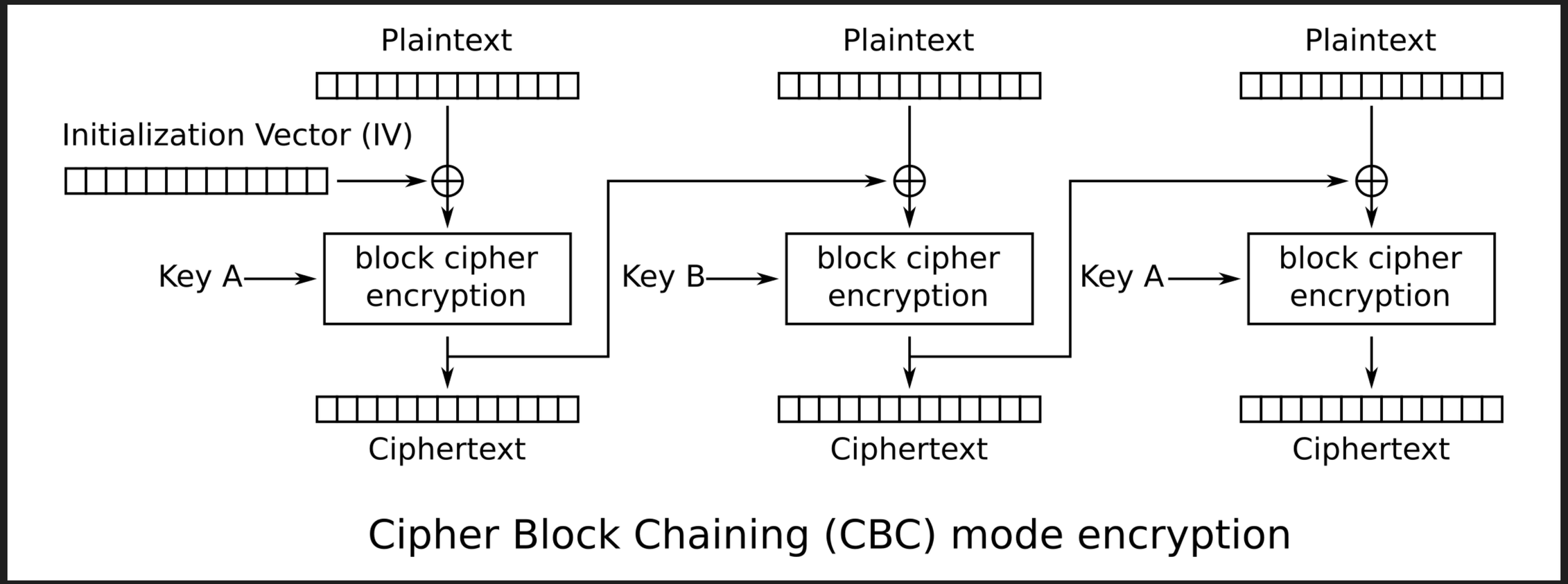
ADOPTION FOR S/MIME



Cipher Block Chaining (CBC) mode encryption

lutra security

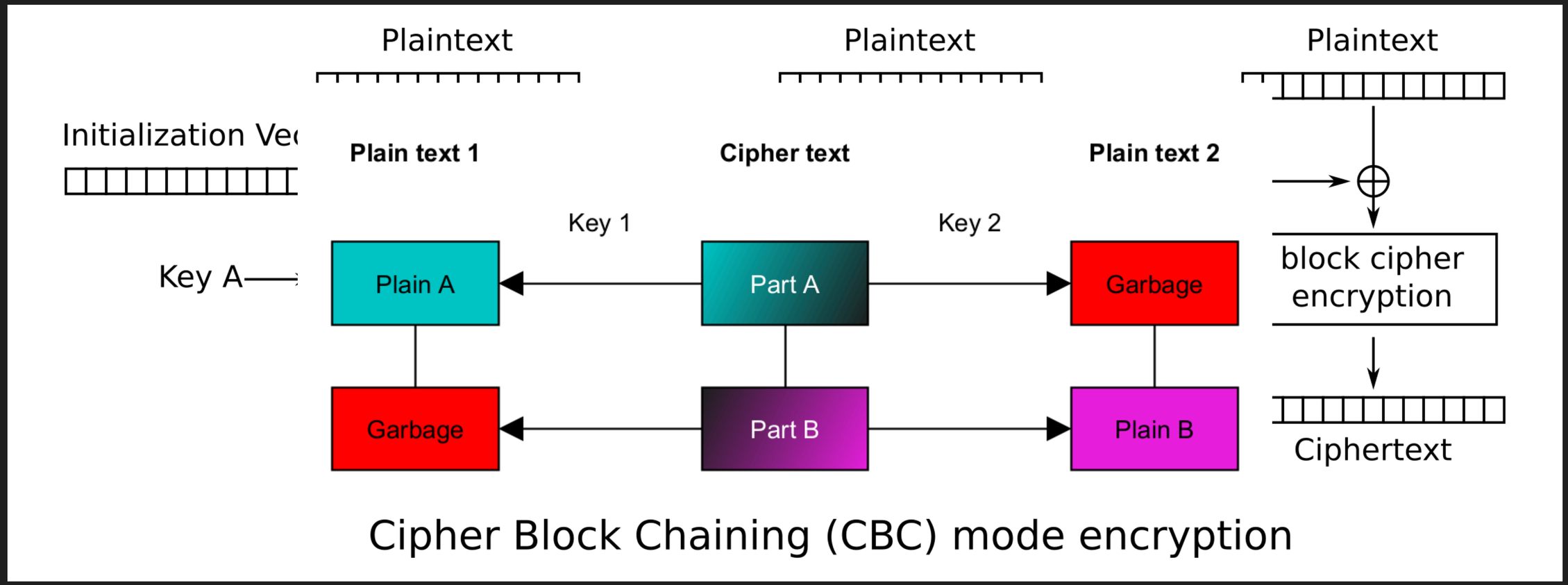
ADOPTION FOR S/MIME



Cipher Block Chaining (CBC) mode encryption

lutra security

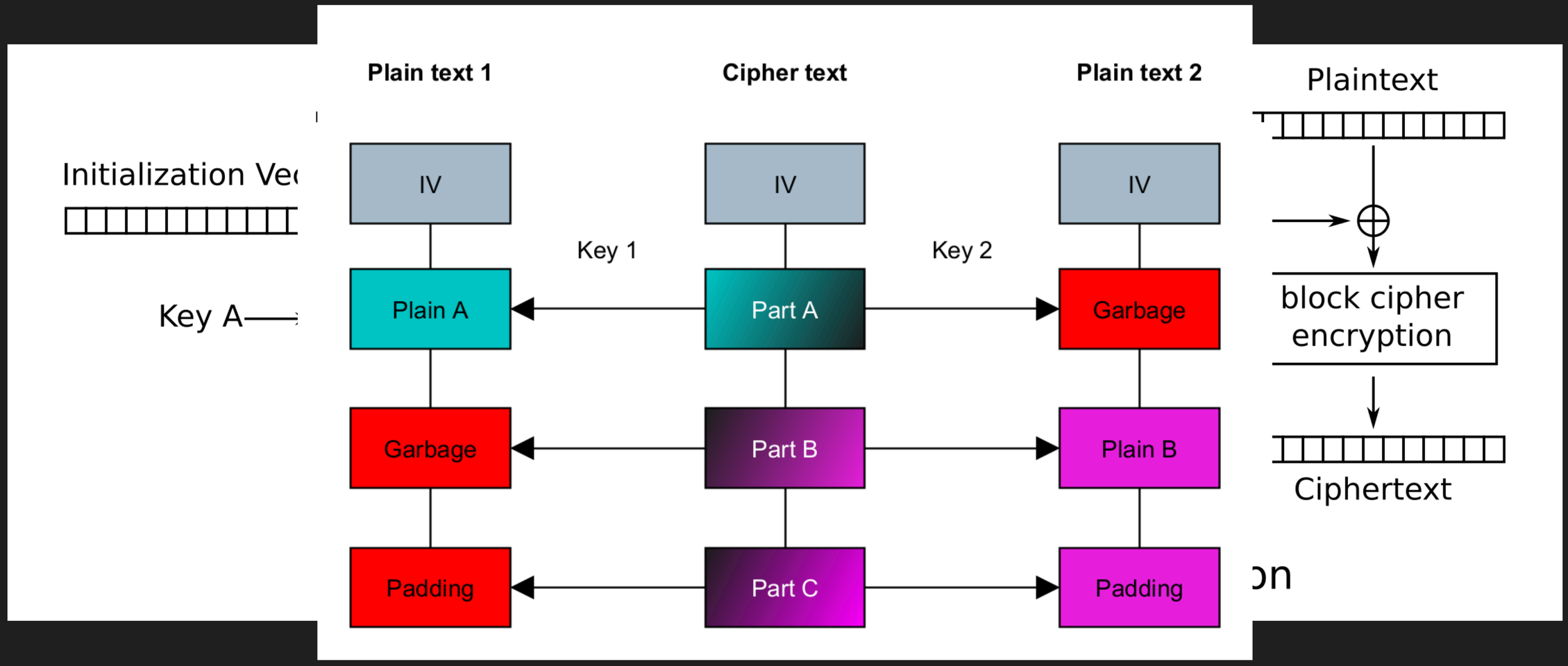
ADOPTION FOR S/MIME



Cipher Block Chaining (CBC) mode encryption

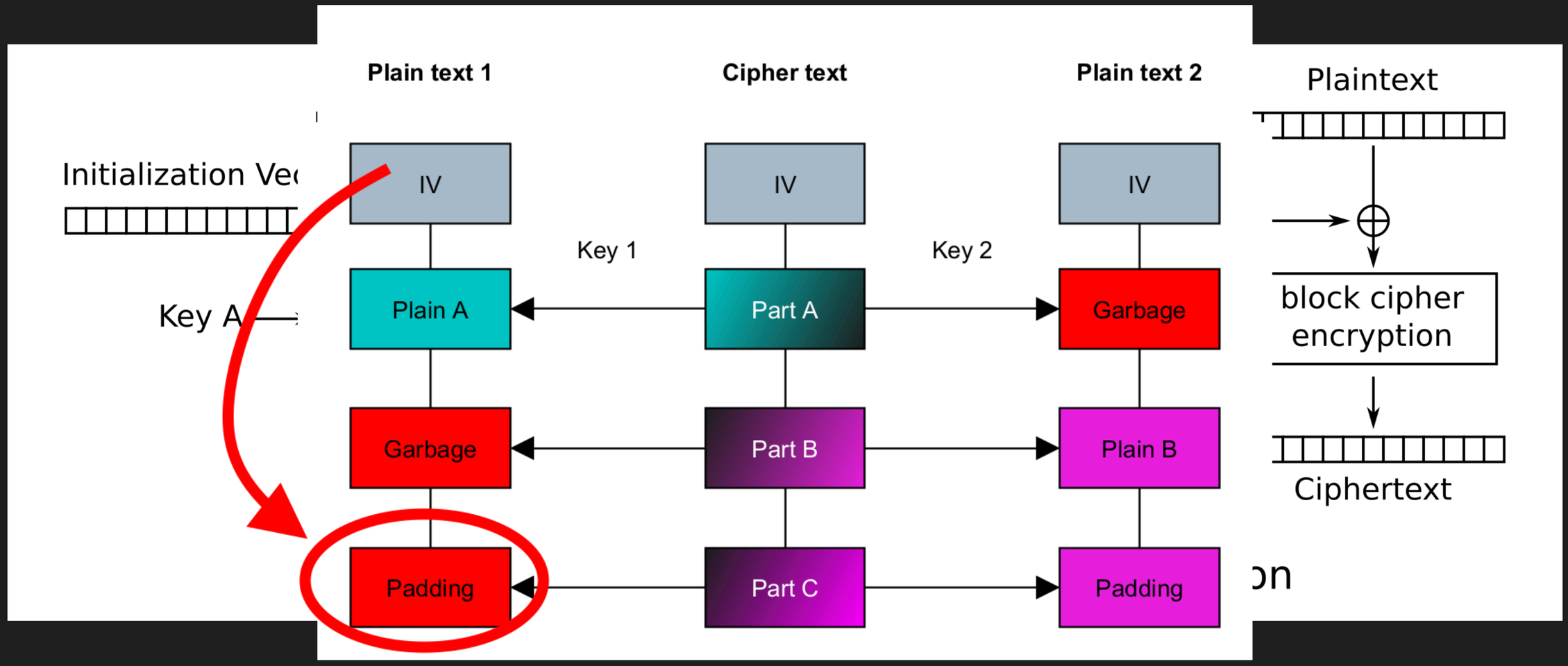
lutra security

ADOPTION FOR S/MIME



lutra security

ADOPTION FOR S/MIME



lutra security

THE EMAIL

Message 1

```
Content-Type: multipart/alternative; boundary="salamander-mime"

--salamander-mime
Content-Type: text/plain; charset="us-ascii"

Axolotls are the best salamanders.

--salamander-mime
Content-Type: text/foo
```

Message 2

```
Content-Type: text/plain; charset="us-ascii"

Fire salamanders are the best salamanders.
```

lutra security

THE ENVELOPE

- Create S/MIME message according to S/MIME 4.0 (RFC 8551)
- The enveloped data (RFC 5652) includes
  - Encrypted content
  - Encrypted content-encryption keys per recipient

lutra security

AFFECTED CLIENTS

- Thunderbird
- Evolution
- Outlook
- ⇨ Every S/MIME client is affected

lutra security

AFFECTED CLIENTS

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\alice.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter pass phrase for PKCS12 import pass phrase:

Content-Type: multipart/alternative; boundary="salamander-mime"

--salamander-mime
Content-Type: text/plain; charset="us-ascii"

Axolotls are the best salamanders.

--salamander-mime
Content-Type: text/foo

◆◆q~3◆]◆93◆◆◆]◆◆
◆Q◆◆`B◆>V
.◆i◆U◆`◆◆◆e◆2◆
◆W◆◆o◆HauS◆◆l◆◆j◆P
 > |
```

lutra security

AFFECTED CLIENTS

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\alice.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter pass phrase for PKCS12 import pass phrase:

Content-Type: multipart/alternative; boundary="salamander-mime"

--salamander-mime
Content-Type: text/plain; charset="us-ascii"

Axolo
```

From [redacted]

To alice@example.com, bob@example.com 07.10.2024, 23:46

Subject **Salamander MIME** S/MIME

Axolotls are the best salamanders.

```
--sal
Conte

◆◆q~3◆]◆93◆◆◆]◆◆
◆Q◆◆`B◆>V
.◆i◆U◆`◆◆◆e◆2◆
◆W◆◆o◆HauS◆◆l◆◆j◆P
 >
```

lutra security

AFFECTED CLIENTS

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\alice.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter pass phrase for PKCS12 import pass phrase:

Content-Type: multipart/alternative; boundary="salamander-mime"
```

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\bob.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter pass phrase for PKCS12 import pass phrase:

◆ħ7p◆◆◆◆ʰ◆E◆웜◆◆◆、◆zMS◆◆◆(9zw◆◆ #◆◆◆e◆p◆◆◆◆◆    ◆◆e◆◆a◆-j◆      ◆;
K◆i◆c◆~◆f◆◆j_9◆◆e◆◆N◆bb◆◆/[◆cgD◆ʕ?◆◆◆Z◆.◆◆◆◆◆q◆◆◆◆BnJË}◆◆◆◆N◆*◆!U◆
oĠı◆◆1>◆◆◆I◆◆◆◆Si◆9◆◆◆4<◆◆◆0P◆(◆◆◆#◆◆
◆wI◆w◆◆=T◆b◆(S◆y◆/
Content-Type: text/plain; charset="us-ascii"

Fire salamanders are the best salamanders.

 >
```

```
◆Q◆◆`B◆>V
.◆i◆U◆`◆◆◆e、◆2◆
◆W◆◆o◆HauS◆◆l◆◆、j◆P
 >
```

lutra security

AFFECTED CLIENTS

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\alice.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter pass phrase for PKCS12 import pass phrase:

Content-Type: multipart/alternative; boundary="salamander-mime"
```

```
 > openssl.exe smime -decrypt -in .\test.eml -recip .\bob.p12
Enter pass phrase for PKCS12 import pass phrase:

Enter
```

From [redacted]

To alice@example.com, bob@example.com 07.10.2024, 23:46

Subject **Salamander MIME** S/MIME

Fire salamanders are the best salamanders.

```
 >
```

```
◆Q◆◆`B◆>V
.◆i◆U◆`◆◆◆e◆2◆
◆W◆◆o◆HauS◆◆l◆◆j◆P
 >
```

lutra security

WHAT CAN WE DO AGAINST IT?

- Detect garbage bytes (unreliable)
- Expect encrypted emails to be signed
- Introduce key commitment in future S/MIME version

lutra security

Konstantin Weddige
kw@lutrasecurity.com
@weddige@gruene.social
lutrasecurity.com