The Hidden Number Problem with Small Unknown Multipliers

Cryptanalyzing MEGA in Six Queries and Other Applications

Nadia Heninger **Keegan Ryan**

UC San Diego

MEGA encrypted cloud storage

MEGA encrypted cloud storage

MEGA

MEGA: Malleable Encryption Goes Awry

Matilda Backendal, Miro Haller and Kenneth G. Paterson
Department of Computer Science, ETH Zurich, Zurich, Switzerland
Email: {mbackendal, kenny.paterson}@inf.ethz.ch, miro.haller@alumni.ethz.ch

***Abstract*—MEGA is a leading cloud storage platform with more than 250 million users and 1000 Petabytes of stored data. MEGA claims to offer user-controlled, end-to-end security. This is achieved by having all data encryption and decryption operations done on MEGA clients, under the control of keys that are only available to those clients. This is intended to protect MEGA users from attacks by MEGA itself, or by adversaries who have taken control of MEGA's infrastructure.**

We provide a detailed analysis of MEGA's use of cryptography in such a malicious server setting. We present five distinct attacks against MEGA, which together allow for a full compromise of the confidentiality of user files. Additionally, the integrity of user data is damaged to the extent that an attacker can insert malicious files of their choice which pass all authenticity checks of the client. We built proof-of-concept versions of all the attacks. Four of the five attacks are eminently practical. They have all been responsibly disclosed to MEGA and remediation is underway.

Taken together, our attacks highlight significant shortcomings in MEGA's cryptographic architecture. We present immediately deployable countermeasures, as well as longer-term recommendations. We also provide a broader discussion of the challenges of cryptographic deployment at massive scale under strong threat models.

I. Introduction

The cloud – for outsourcing of both computation and data storage – has become a very popular approach to address scaling and management problems in IT. This applies to both enterprise and consumer domains. In the latter case, the market offers a myriad of different cloud services, with products having different combinations of storage, computation and collaboration features, and making a range of security and privacy claims. The consumer storage market alone was valued at USD 13.6 billion in 2021.[1]

As a prominent example, MEGA[2] is a cloud storage and collaboration platform founded in 2013 offering "secure storage and communication" services. With over 250 million registered users, 10 million daily active users [1] and 1000 PB of stored data [2], MEGA is a significant player in the consumer domain. What sets them apart from their competitors such as DropBox, Google Drive, iCloud and Microsoft OneDrive is the claimed security guarantees: MEGA advertise themselves as "the privacy company" and promise *user-controlled end-to-end encryption* (UCE).

UCE refers to the fact that data uploaded to the MEGA cloud is encrypted, and that only the user who owns the data has access to the key (derived from the user's password) needed to decrypt. Thus, MEGA's main selling point is confidentiality of user data even against MEGA themselves, as showcased in the following quote from their website [3]:

> "MEGA does not have access to your password or your data. Using a strong and unique password will ensure that your data is protected from being hacked and gives you total confidence that your information will remain just that – yours."

This implies a threat model in which the service provider itself should be considered potentially adversarial, and yet the service should remain secure. All the service is then trusted for is availability. This adversarial model provides an interesting setting for cryptanalysis: not only does the adversary have access to encrypted user keys and data, it can also interact with users through legitimate channels during steps like user authentication and file access.

This may seem a very strong adversarial model. However, we stress that it is consistent with the security claims made by MEGA themselves. Moreover, we must consider the possibility that even if MEGA is not adversarial, their systems may have been compromised by malicious third parties, for example nation state security agencies or hacking groups, who wish to gain access to users' data and files. Indeed, the sheer size of MEGA – and the likelihood of it attracting users who wish to protect highly sensitive data precisely because of the security the service claims to offer – surely make MEGA an attractive target. Additionally, UCE should ensure that MEGA cannot be coerced into revealing user data, e.g. through subpoenas, since they are technically unable to do so.

In this work, we review the security of MEGA in this threat model and find significant issues in how it uses cryptography. These lead to devastating attacks on the confidentiality and integrity of user data in the MEGA cloud.

A. *The MEGA Key Hierarchy*

MEGA's approach to UCE begins with the user password, PW, which acts as the root of the key hierarchy depicted in Figure 1. The MEGA client derives an authentication key and an encryption key from the password. The authentication key is used to identify users to MEGA. The encryption key encrypts a randomly generated master key, which in turn encrypts other key material of the user. Every account has a set of asymmetric keys: an RSA key pair for sharing data, a `Curve25519` key pair for exchanging chat keys for MEGA's chat functionality, and an `Ed25519` key pair for signing the

MEGA encrypted cloud storage

MEGA

ENGINEERING

MEGA Security Update

Mathias Ortmann
Chief Architect

Published on 21 Jun 2022

MEGA: Malleable Encryption Goes Awry

Matilda Backendal, Miro Haller and Kenneth G. Paterson
Department of Computer Science, ETH Zurich, Zurich, Switzerland
Email: {mbackendal, kenny.paterson}@inf.ethz.ch, miro.haller@alumni.ethz.ch

Abstract—MEGA is a leading cloud storage platform with more than 250 million users and 1000 Petabytes of stored data. MEGA claims to offer user-controlled, end-to-end security. This is achieved by having all data encryption and decryption operations done on MEGA clients, under the control of keys that are only available to those clients. This is intended to protect MEGA users from attacks by MEGA itself, or by adversaries who have taken control of MEGA's infrastructure.

We provide a detailed analysis of MEGA's use of cryptography ...icious server setting. We present five distinct attacks ...A, which together allow for a full compromise of the ...y of user files. Additionally, the integrity of user data ...o the extent that an attacker can insert malicious ... choice which pass all authenticity checks of the ...ilt proof-of-concept versions of all the attacks. Four ...tacks are eminently practical. They have all been ...isclosed to MEGA and remediation is underway. ...ther, our attacks highlight significant shortcomings ...ryptographic architecture. We present immediately ...untermeasures, as well as longer-term recommen... ...lso provide a broader discussion of the challenges of ... deployment at massive scale under strong threat

I. Introduction

... – for outsourcing of both computation and data ...s become a very popular approach to address ...management problems in IT. This applies to both ...d consumer domains. In the latter case, the market ...riad of different cloud services, with products ...rent combinations of storage, computation and ... features, and making a range of security and ...s. The consumer storage market alone was valued ... billion in 2021.[1]

As a prominent example, MEGA[2] is a cloud storage and collaboration platform founded in 2013 offering "secure storage and communication" services. With over 250 million registered users, 10 million daily active users [1] and 1000 PB of stored data [2], MEGA is a significant player in the consumer domain. What sets them apart from their competitors such as DropBox, Google Drive, iCloud and Microsoft OneDrive is the claimed security guarantees: MEGA advertise themselves as "the privacy company" and promise *user-controlled end-to-end encryption* (UCE).

UCE refers to the fact that data uploaded to the MEGA cloud is encrypted, and that only the user who owns the data has access to the key (derived from the user's password) needed to decrypt. Thus, MEGA's main selling point is confidentiality of user data even against MEGA themselves, as showcased in the following quote from their website [3]:

> "MEGA does not have access to your password or your data. Using a strong and unique password will ensure that your data is protected from being hacked and gives you total confidence that your information will remain just that – yours."

This implies a threat model in which the service provider itself should be considered potentially adversarial, and yet the service should remain secure. All the service is then trusted for is availability. This adversarial model provides an interesting setting for cryptanalysis: not only does the adversary have access to encrypted user keys and data, it can also interact with users through legitimate channels during steps like user authentication and file access.

This may seem a very strong adversarial model. However, we stress that it is consistent with the security claims made by MEGA themselves. Moreover, we must consider the possibility that even if MEGA is not adversarial, their systems may have been compromised by malicious third parties, for example nation state security agencies or hacking groups, who wish to gain access to users' data and files. Indeed, the sheer size of MEGA – and the likelihood of it attracting users who wish to protect highly sensitive data precisely because of the security the service claims to offer – surely make MEGA an attractive target. Additionally, UCE should ensure that MEGA cannot be coerced into revealing user data, e.g. through subpoenas, since they are technically unable to do so.

In this work, we review the security of MEGA in this threat model and find significant issues in how it uses cryptography. These lead to devastating attacks on the confidentiality and integrity of user data in the MEGA cloud.

A. The MEGA Key Hierarchy

MEGA's approach to UCE begins with the user password, PW, which acts as the root of the key hierarchy depicted in Figure 1. The MEGA client derives an authentication key and an encryption key from the password. The authentication key is used to identify users to MEGA. The encryption key encrypts a randomly generated master key, which in turn encrypts other key material of the user. Every account has a set of asymmetric keys: an RSA key pair for sharing data, a Curve25519 key pair for exchanging chat keys for MEGA's chat functionality, and an Ed25519 key pair for signing the

MEGA encrypted cloud storage

MEGA

MEGA: Malleable Encryption Goes Awry

Matilda Backendal, Miro Haller and Kenneth G. Paterson
Department of Computer Science, ETH Zurich, Zurich, Switzerland
Email: {mbackendal, kenny.paterson}@inf.ethz.ch, miro.haller@alumni.ethz.ch

Who is potentially affected?

Customers who have logged into their MEGA account at least 512 times (the more, the higher the exposure). Note that resuming an existing session does not count as a login. While all MEGA client products use permanent sessions by default, some third-party clients such as Rclone do not, so their users may be exposed.

ENG

M

Mathias Ortmann
Chief Architect

Published on 21 Jun 2022

deployment at massive scale under strong threat

I. Introduction

– for outsourcing of both computation and data
s become a very popular approach to address
management problems in IT. This applies to both
d consumer domains. In the latter case, the market
riad of different cloud services, with products
rent combinations of storage, computation and
features, and making a range of security and
s. The consumer storage market alone was valued
billion in 2021.[1]

As a prominent example, MEGA[2] is a cloud storage and collaboration platform founded in 2013 offering "secure storage and communication" services. With over 250 million registered users, 10 million daily active users [1] and 1000 PB of stored data [2], MEGA is a significant player in the consumer domain. What sets them apart from their competitors such as DropBox, Google Drive, iCloud and Microsoft OneDrive is the claimed security guarantees: MEGA advertise themselves as "the privacy company" and promise *user-controlled end-to-end encryption* (UCE).

UCE refers to the fact that data uploaded to the MEGA cloud is encrypted, and that only the user who owns the data has access to the key (derived from the user's password) needed to decrypt. Thus, MEGA's main selling point is

by MEGA themselves. Moreover, we must consider the possibility that even if MEGA is not adversarial, their systems may have been compromised by malicious third parties, for example nation state security agencies or hacking groups, who wish to gain access to users' data and files. Indeed, the sheer size of MEGA – and the likelihood of it attracting users who wish to protect highly sensitive data precisely because of the security the service claims to offer – surely make MEGA an attractive target. Additionally, UCE should ensure that MEGA cannot be coerced into revealing user data, e.g. through subpoenas, since they are technically unable to do so.

In this work, we review the security of MEGA in this threat model and find significant issues in how it uses cryptography. These lead to devastating attacks on the confidentiality and integrity of user data in the MEGA cloud.

A. The MEGA Key Hierarchy

MEGA's approach to UCE begins with the user password, PW, which acts as the root of the key hierarchy depicted in Figure 1. The MEGA client derives an authentication key and an encryption key from the password. The authentication key is used to identify users to MEGA. The encryption key encrypts a randomly generated master key, which in turn encrypts other key material of the user. Every account has a set of asymmetric keys: an RSA key pair for sharing data, a Curve25519 key pair for exchanging chat keys for MEGA's chat functionality, and an Ed25519 key pair for signing the

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
	AES(RSA private key)

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
AES(RSA private key)	AES(RSA private key)

1. Server sends the client's password-encrypted RSA private key.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)

1. Server sends the client's password-encrypted RSA private key.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
	RSA challenge plaintext

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
Decrypted RSA plaintext	

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
Decrypted RSA plaintext	Client's challenge

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
Decrypted RSA plaintext	Client's challenge

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.
3. Authenticate if the challenge value matches.

MEGA encrypted cloud storage login (simplified)

Client	Server
Password-derived AES key	Client's RSA public key
RSA private key	AES(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
Decrypted RSA plaintext	Client's challenge

1. Server sends the client's password-encrypted RSA private key.
2. Send an RSA-encrypted challenge to verify decryption.
3. Authenticate if the challenge value matches.
4. In reality, only 43 bytes are sent as verification.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
	AES-ECB(RSA private key)

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA private key	AES-ECB(RSA private key)

1. Attacker modifies 128-bit blocks of encrypted private key.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA "private key"	AES-ECB(RSA private key)

1. Attacker modifies 128-bit blocks of encrypted private key.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA "private key"	AES-ECB(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext

1. Attacker modifies 128-bit blocks of encrypted private key.
2. Modified values used to "decrypt" the challenge.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA "private key"	AES-ECB(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
"Decrypted" RSA ciphertext	

1. Attacker modifies 128-bit blocks of encrypted private key.
2. Modified values used to "decrypt" the challenge.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA "private key"	AES-ECB(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
"Decrypted" RSA ciphertext	

1. Attacker modifies 128-bit blocks of encrypted private key.
2. Modified values used to "decrypt" the challenge.
3. Custom padding scheme was *not* checked.

Malicious MEGA login process (simplified)

Client	Attacker
Password-derived AES key	Client's RSA public key
(Modified) RSA "private key"	AES-ECB(RSA private key)
RSA challenge ciphertext	RSA challenge plaintext
"Decrypted" RSA ciphertext	43 bytes from "decryption"

1. Attacker modifies 128-bit blocks of encrypted private key.
2. Modified values used to "decrypt" the challenge.
3. Custom padding scheme was *not* checked.
4. The 43 bytes leak information about the RSA private key.

Cryptanalysis of (Backendal, Haller, Paterson, Oakland 23)

Private key (q, p, d, u) is encrypted with AES-ECB.

Cryptanalysis of (Backendal, Haller, Paterson, Oakland 23)

Private key (q, p, d, u) is encrypted with AES-ECB.

Modify 128 bits of RSA-CRT coefficient $u \mapsto \tilde{u}$.

Cryptanalysis of (Backendal, Haller, Paterson, Oakland 23)

Private key (q, p, d, u) is encrypted with AES-ECB.

Modify 128 bits of RSA-CRT coefficient $u \mapsto \tilde{u}$.

Do binary search for q based on 43-byte response.

Cryptanalysis of (Backendal, Haller, Paterson, Oakland 23)

Private key (q, p, d, u) is encrypted with AES-ECB.

Modify 128 bits of RSA-CRT coefficient $u \mapsto \tilde{u}$.

Do binary search for q based on 43-byte response.

One login attempt reveals *one bit* of information.

Cryptanalysis of (Backendal, Haller, Paterson, Oakland 23)

Private key (q, p, d, u) is encrypted with AES-ECB.

Modify 128 bits of RSA-CRT coefficient $u \mapsto \tilde{u}$.

Do binary search for q based on 43-byte response.

One login attempt reveals *one bit* of information.

512 login attempts are required.

Representing the client response

$$\text{Client response} = \mathrm{MSB}((\tilde{u}_i(m_p - m_q)q + m_q) \mod N)$$

- Each login attempt returns 43 bytes from Garner's formula.

Representing the client response

$$\text{Client response} = \mathrm{MSB}((\tilde{u}_i x + m_q) \mod N)$$

- Each login attempt returns 43 bytes from Garner's formula.
- There is a large hidden number x shared between samples.

Representing the client response

$$a_i = \tilde{u}_i x + e_i \pmod{N}$$

- Each login attempt returns 43 bytes from Garner's formula.
- There is a large hidden number x shared between samples.
- There is a bounded error e_i in the samples.

Representing the client response

$$a_i = \tilde{u}_i x + e_i \pmod{N}$$

- Each login attempt returns 43 bytes from Garner's formula.
- There is a large hidden number x shared between samples.
- There is a bounded error e_i in the samples.
- The unknown multiplier is decrypted 128-bit AES blocks.
- Goal: recover the unknown multipliers.

Hidden Number Problem with Small Unknown Multipliers

$$a_i = t_i x + e_i \pmod{N}$$

- The attacker is given two or more samples.
- There is a large hidden number x shared between samples.
- There is a bounded error e_i in the samples.
- The unknown multiplier is bounded.
- Goal: recover the unknown multipliers.

Challenges to solving HNP-SUM

We know a_i, N, and bounds for $|t_i| \leq T$, $|e_i| \leq E$. Recover t_i.

$$a_i = t_i x + e_i \pmod N$$

Challenges to solving HNP-SUM

We know a_i, N, and bounds for $|t_i| \leq T$, $|e_i| \leq E$. Recover t_i.

$$a_i = t_i x + e_i \pmod{N}$$

- Two new unknowns for every new sample.

Challenges to solving HNP-SUM

We know a_i, N, and bounds for $|t_i| \leq T$, $|e_i| \leq E$. Recover t_i.

$$a_i = t_i x + e_i \pmod{N}$$

- Two new unknowns for every new sample.
- Resembles the Hidden Number Problem of Boneh and Venkatesan (Crypto 1996), except multipliers t_i are unknown.

Challenges to solving HNP-SUM

We know a_i, N, and bounds for $|t_i| \leq T$, $|e_i| \leq E$. Recover t_i.

$$a_i = t_i x + e_i \pmod{N}$$

- Two new unknowns for every new sample.
- Resembles the Hidden Number Problem of Boneh and Venkatesan (Crypto 1996), except multipliers t_i are unknown.
- Linearization fails because $(t_i x \mod N)$ is unbounded.

Challenges to solving HNP-SUM

We know a_i, N, and bounds for $|t_i| \leq T$, $|e_i| \leq E$. Recover t_i.

$$a_i = t_i x + e_i \pmod{N}$$

- Two new unknowns for every new sample.
- Resembles the Hidden Number Problem of Boneh and Venkatesan (Crypto 1996), except multipliers t_i are unknown.
- Linearization fails because $(t_i x \mod N)$ is unbounded.

HNP-SUM lets us cryptanalyze MEGA with 6 samples.

Solving HNP-SUM

Solving HNP-SUM with two samples

$$\begin{aligned} a_1 &= t_1 x + e_1 \pmod{N} \\ a_2 &= t_2 x + e_2 \pmod{N} \end{aligned}$$

Solving HNP-SUM with two samples

$$
\begin{array}{rlcll}
t_2(& a_1 & = & t_1 x + e_1 &) \pmod N \\
-\,t_1(& a_2 & = & t_2 x + e_2 &) \pmod N \\
\hline
& t_2 a_1 + -t_1 a_2 & = & t_2 e_1 - t_1 e_2 & \pmod N
\end{array}
$$

We know there exists a small linear combination of known values with small coefficients.

Solving HNP-SUM with two samples

$$
\begin{array}{rcccl}
t_2(& a_1 & = & t_1 x + e_1 &) \pmod{N} \\
-\, t_1(& a_2 & = & t_2 x + e_2 &) \pmod{N} \\
\hline
 & t_2 a_1 + -t_1 a_2 & = & t_2 e_1 - t_1 e_2 & \pmod{N}
\end{array}
$$

We know there exists a small linear combination of known values with small coefficients. We want to find these coefficients.

Solving HNP-SUM with two samples

Consider the lattice spanned by the rows of

$$\begin{bmatrix} 1 & 0 & a_1 \\ 0 & 1 & a_2 \\ 0 & 0 & N \end{bmatrix}$$

This lattice contains the short vector

$$\begin{bmatrix} t_2 & -t_1 & t_2 e_1 - t_1 e_2 \end{bmatrix}.$$

Solving HNP-SUM with two samples

Consider the lattice spanned by the rows of

$$\begin{bmatrix} 1 & 0 & a_1 \\ 0 & 1 & a_2 \\ 0 & 0 & N \end{bmatrix}$$

This lattice contains the short vector

$$\begin{bmatrix} t_2 & -t_1 & t_2 e_1 - t_1 e_2 \end{bmatrix}.$$

Lattice reduction finds this target vector.

Trying to solve HNP-SUM with three samples

Consider the lattice spanned by the rows of

$$\begin{bmatrix} 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & N \end{bmatrix}$$

This lattice contains the short vectors

$$\begin{bmatrix} t_2 & -t_1 & 0 & t_2 e_1 - t_1 e_2 \end{bmatrix}$$

$$\begin{bmatrix} t_3 & 0 & -t_1 & t_3 e_1 - t_1 e_3 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -t_3 & t_2 & t_3 e_2 - t_2 e_3 \end{bmatrix}$$

Trying to solve HNP-SUM with three samples

Consider the lattice spanned by the rows of

$$\begin{bmatrix} 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & N \end{bmatrix}$$

This lattice contains the short vectors

$$\begin{bmatrix} t_2 & -t_1 & 0 & t_2 e_1 - t_1 e_2 \end{bmatrix}$$
$$\begin{bmatrix} t_3 & 0 & -t_1 & t_3 e_1 - t_1 e_3 \end{bmatrix}$$
$$\begin{bmatrix} 0 & -t_3 & t_2 & t_3 e_2 - t_2 e_3 \end{bmatrix}$$

But these are *not* found by lattice reduction.

Solving HNP-SUM with three samples

HNP-SUM instance with multipliers (-292, 264, 185):

$$\begin{bmatrix} 1 & 0 & 0 & 16434376644250 \\ 0 & 1 & 0 & 18067839662587 \\ 0 & 0 & 1 & 6420926526082 \\ 0 & 0 & 0 & 27006979257190 \end{bmatrix}$$

Solving HNP-SUM with three samples

HNP-SUM instance with multipliers (-292, 264, 185):

$$\begin{bmatrix} 1 & 0 & 0 & 16434376644250 \\ 0 & 1 & 0 & 18067839662587 \\ 0 & 0 & 1 & 6420926526082 \\ 0 & 0 & 0 & 27006979257190 \end{bmatrix} \rightarrow \begin{bmatrix} 15 & 25 & -12 & 71 \\ -47 & -66 & 20 & 68 \\ -36967 & 16082 & -25946 & -2238 \\ 12565 & -30656 & -63041 & -2494 \end{bmatrix}$$

Lattice reduction finds a dense sublattice, but none of the target vectors. We want to find a vector with 0 in the correct spot.

Solving HNP-SUM with three samples

HNP-SUM instance with multipliers (-292, 264, 185):

$$\begin{bmatrix} 1 & 0 & 0 & 16434376644250 \\ 0 & 1 & 0 & 18067839662587 \\ 0 & 0 & 1 & 6420926526082 \\ 0 & 0 & 0 & 27006979257190 \end{bmatrix} \rightarrow \begin{bmatrix} 15 & 25 & -12 & 71 \\ -47 & -66 & 20 & 68 \\ -36967 & 16082 & -25946 & -2238 \\ 12565 & -30656 & -63041 & -2494 \end{bmatrix}$$

Lattice reduction finds a dense sublattice, but none of the target vectors. We want to find a vector with 0 in the correct spot.

$$\begin{bmatrix} 15 & 25 & -12 & 71 \\ -47 & -66 & 20 & 68 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 88 & -124 & 2038 \\ 0 & 185 & -264 & 4357 \end{bmatrix}$$

Hermite Normal Form finds one, reveals multipliers 264 and 185

Solving HNP-SUM

Observation #1:

We build a lattice that *cancels out* terms involving x.

Solving HNP-SUM

Observation #1:
We build a lattice that *cancels out* terms involving x.

Observation #2:
Lattice reduction finds a *dense sublattice*, not just a short vector.

Application:
The RSA Implicit Factoring Problem

Implicit factoring problem (May and Ritzenhofen, PKC 2009)

Given multiple RSA moduli whose unbalanced factors share the same most significant bits, recover the factorization.

$$
\begin{aligned}
3709606718119160021 &= 6372799\,72190201 \times 5821 \\
4244922075900467567 &= 6372799\,99384547 \times 6661 \\
3078699429183112997 &= 6372799\,48081787 \times 4831
\end{aligned}
$$

We can also consider cases where the least significant bits are shared, some of each, or bits in the middle.

Solving Implicit factoring using HNP-SUM

$$N_i = (p_{\text{msb}} + p_{\text{lsb},i})q_i = q_i p_{\text{msb}} + (q_i p_{\text{lsb},i})$$

N_i	is the HNP-SUM sample.
p_{msb}	is the hidden number shared between samples.
q_i	is the small unknown multiplier.
$q_i p_{\text{lsb},i}$	is the bounded error.

Similar modular equations exist for other types of shared bits.

Implicit factoring results

We compare our HNP-SUM approach to prior lattice constructions.

Bits shared	Comparison

Implicit factoring results

We compare our HNP-SUM approach to prior lattice constructions.

Bits shared	Comparison
LSBs	Improve upon (May and Ritzenhofen, PKC 2009)

Implicit factoring results

We compare our HNP-SUM approach to prior lattice constructions.

Bits shared	Comparison
LSBs	Improve upon (May and Ritzenhofen, PKC 2009)
MSBs	Close to (Faugère, Marinier, Renault, PKC 2010)

Implicit factoring results

We compare our HNP-SUM approach to prior lattice constructions.

Bits shared	Comparison
LSBs	Improve upon (May and Ritzenhofen, PKC 2009)
MSBs	Close to (Faugère, Marinier, Renault, PKC 2010)
Middle	Polynomially smaller lattices than in [FMR10]

Implicit factoring results

We compare our HNP-SUM approach to prior lattice constructions.

Bits shared	Comparison
LSBs	Improve upon (May and Ritzenhofen, PKC 2009)
MSBs	Close to (Faugère, Marinier, Renault, PKC 2010)
Middle	Polynomially smaller lattices than in [FMR10]
LSB+MSBs	No direct lattice construction

Cryptanalyzing MEGA in Six Queries

Cryptanalyzing MEGA: The aftermath

June 21, 2022 - Backendal, Haller, and Paterson publish 512-login attack on MEGA (Oakland 2023).

Who is potentially affected?

Customers who have logged into their MEGA account at least 512 times (the more, the higher the exposure). Note that resuming an existing session does not count as a login. While all MEGA client products use permanent sessions by default, some third-party clients such as Rclone do not, so their users may be exposed.

Cryptanalyzing MEGA: The aftermath

June 21, 2022 - Backendal, Haller, and Paterson publish 512-login attack on MEGA (Oakland 2023).

July 13, 2022 - We disclose improved 6-login attack on unpatched systems (PKC 2023).

Summary

The original ETHZ results put a relatively small subset of MEGA users at risk. The UCSD research lowers the threshold from 512 to just six logins, broadening the potential exposure to the vast majority of users. This confirms the importance of using MEGA client software released on 22 June 2022 or later, which is immune to both attacks.

Cryptanalyzing MEGA: The aftermath

June 21, 2022 - Backendal, Haller, and Paterson publish 512-login attack on MEGA (Oakland 2023).

July 13, 2022 - We disclose improved 6-login attack on unpatched systems (PKC 2023).

Sep. 29, 2022 - Albrecht, Haller, Mareková, Paterson disclose 2-login attack (Eurocrypt 2023).

Cryptanalyzing MEGA

Analyzing MEGA's real-world design has been productive.

Cryptanalyzing MEGA

Analyzing MEGA's real-world design has been productive.

- Three new papers (Oakland, PKC, Eurocrypt)

Cryptanalyzing MEGA

Analyzing MEGA's real-world design has been productive.

- Three new papers (Oakland, PKC, Eurocrypt)
- HNP-SUM involves unusual lattice techniques

Cryptanalyzing MEGA

Analyzing MEGA's real-world design has been productive.

- Three new papers (Oakland, PKC, Eurocrypt)
- HNP-SUM involves unusual lattice techniques
- Surprising applications to implicit factoring

Thank You

The Hidden Number Problem with Small Unknown Multipliers: Cryptanalyzing MEGA in Six Queries and Other Applications

`https://ia.cr/2022/914`