
Kevin D'Souza

A Practical Cryptanalysis of the Algebraic Eraser™

Adi Ben-Zvi[1] **Simon R. Blackburn**[2] Boaz Tsaban[1]

[1]Bar Ilan University, Israel

[2]Royal Holloway University of London, UK

15th August 2016

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:
  - Jan 2008: Myasnikov and Ushakov break proposed parameters.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:
  - Jan 2008: Myasnikov and Ushakov break proposed parameters.
  - May 2011: Gunnells recommends increasing parameter sizes.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:
  - Jan 2008: Myasnikov and Ushakov break proposed parameters.
  - May 2011: Gunnells recommends increasing parameter sizes.
  - Jan 2008: Kalka, Teicher and Tsaban break for generic parameters.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:
  - Jan 2008: Myasnikov and Ushakov break proposed parameters.
  - May 2011: Gunnells recommends increasing parameter sizes.
  - Jan 2008: Kalka, Teicher and Tsaban break for generic parameters.
  - Feb 2012: Goldfeld and Gunnells avoid attack by careful choice of system parameters.

Overview

- Anshel, Anshel, Goldfeld, Lemieux announce the Algebraic Eraser (AE) in 2002.
- A key exchange primitive, based on matrix groups, permutation groups and braid groups.
- SecureRF (trademark owners) marketing it for IoT.
- Nov 2015: AE-based RFID tag authentication proposal under ISO/IEC SC31: posted on SecureRF website.
- Previous attacks on underlying AE primitive:
  - Jan 2008: Myasnikov and Ushakov break proposed parameters.
  - May 2011: Gunnells recommends increasing parameter sizes.
  - Jan 2008: Kalka, Teicher and Tsaban break for generic parameters.
  - Feb 2012: Goldfeld and Gunnells avoid attack by careful choice of system parameters.
- This work: an attack that recovers the key in just 8 hours on a single core in Magma, for 128-bit parameters.

AE Diffie Hellman

Set $n = 16$, $q = 256$.

AE Diffie Hellman

Set $n = 16$, $q = 256$.

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

AE Diffie Hellman

Set $n = 16$, $q = 256$.

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

AE Diffie Hellman

$$\text{Set } n = 16,\ q = 256.$$

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.

AE Diffie Hellman

$$\text{Set } n = 16,\ q = 256.$$

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.
2. Alice computes public key $(M_A, \sigma_A) \in \Omega$ and sends to Bob.

AE Diffie Hellman

Set $n = 16$, $q = 256$.

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.
2. Alice computes public key $(M_A, \sigma_A) \in \Omega$ and sends to Bob.
3. Bob generates private info.

AE Diffie Hellman

Set $n = 16$, $q = 256$.

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.
2. Alice computes public key $(M_A, \sigma_A) \in \Omega$ and sends to Bob.
3. Bob generates private info.
4. Bob computes public key $(M_B, \sigma_B) \in \Omega$ and sends to Alice.

AE Diffie Hellman

$$\text{Set } n = 16,\ q = 256.$$

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.
2. Alice computes public key $(M_A, \sigma_A) \in \Omega$ and sends to Bob.
3. Bob generates private info.
4. Bob computes public key $(M_B, \sigma_B) \in \Omega$ and sends to Alice.
5. Parties compute shared value $(M, \sigma) \in \Omega$ from private info and public keys.

AE Diffie Hellman

$$\text{Set } n = 16,\ q = 256.$$

$$\Omega = \{(M, \sigma) : M \in \mathrm{GL}_n(q) \text{ and } \sigma \in \mathrm{Sym}(n)\}.$$

Diffie–Hellman-style protocol:

1. Alice generates private info.
2. Alice computes public key $(M_A, \sigma_A) \in \Omega$ and sends to Bob.
3. Bob generates private info.
4. Bob computes public key $(M_B, \sigma_B) \in \Omega$ and sends to Alice.
5. Parties compute shared value $(M, \sigma) \in \Omega$ from private info and public keys.
6. M is the shared key.

The platform group

The platform group

- Based on the coloured Burau group $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$.

The platform group

- Based on the coloured Burau group $\mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n)) \rtimes \mathrm{Sym}(n)$.
- Elements:

$$(M,\sigma) \text{ where } M \in \mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n)) \text{ and } \sigma \in \mathrm{Sym}(n).$$

The platform group

- Based on the coloured Burau group $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$.
- Elements:

$$(M, \sigma) \text{ where } M \in \mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \text{ and } \sigma \in \mathrm{Sym}(n).$$

- To multiply:

$$(M, \sigma)(M', \sigma') = (M(M')^\sigma, \sigma\sigma').$$

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S, \pi) \in \Omega$ and $(M, \sigma) \in \mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S, \pi) * (M, \sigma) = (S\,\varphi(M^{\pi}), \pi\sigma) \in \Omega.$$

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S,\pi)\in\Omega$ and $(M,\sigma)\in \mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S,\pi)*(M,\sigma) = (S\,\varphi(M^{\pi}),\pi\sigma)\in\Omega.$$

- Choose commuting subgroups A and B of coloured Burau group in some way.

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S, \pi) \in \Omega$ and $(M, \sigma) \in \mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S, \pi) * (M, \sigma) = (S\,\varphi(M^\pi), \pi\sigma) \in \Omega.$$

- Choose commuting subgroups A and B of coloured Burau group in some way.
- Choose commuting subgroups C and D of $\mathrm{GL}_n(q)$ in some way.

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S, \pi) \in \Omega$ and $(M, \sigma) \in \mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S, \pi) * (M, \sigma) = (S\,\varphi(M^\pi), \pi\sigma) \in \Omega.$$

- Choose commuting subgroups A and B of coloured Burau group in some way.
- Choose commuting subgroups C and D of $\mathrm{GL}_n(q)$ in some way.
- Alice picks $c \in C$, $a \in A$ and sends $c(I, e) * a$ to Bob.

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S,\pi) \in \Omega$ and $(M,\sigma) \in \mathrm{GL}_n(\mathbb{F}_q(t_1,\ldots,t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S,\pi) * (M,\sigma) = (S\,\varphi(M^\pi), \pi\sigma) \in \Omega.$$

- Choose commuting subgroups A and B of coloured Burau group in some way.
- Choose commuting subgroups C and D of $\mathrm{GL}_n(q)$ in some way.
- Alice picks $c \in C$, $a \in A$ and sends $c(I,e) * a$ to Bob.
- Bob picks $d \in D$, $b \in B$ and sends $d(I,e) * b$ to Alice.

Details of key exchange

- Define a map φ from (a subgroup of) $\mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n))$ to $\mathrm{GL}_n(q)$: replace each t_i by some non-zero element $\tau_i \in \mathbb{F}_q$.
- For $(S, \pi) \in \Omega$ and $(M, \sigma) \in \mathrm{GL}_n(\mathbb{F}_q(t_1, \ldots, t_n)) \rtimes \mathrm{Sym}(n)$, define E-multiplication by

$$(S, \pi) * (M, \sigma) = (S\,\varphi(M^\pi), \pi\sigma) \in \Omega.$$

- Choose commuting subgroups A and B of coloured Burau group in some way.
- Choose commuting subgroups C and D of $\mathrm{GL}_n(q)$ in some way.
- Alice picks $c \in C$, $a \in A$ and sends $c(I, e) * a$ to Bob.
- Bob picks $d \in D$, $b \in B$ and sends $d(I, e) * b$ to Alice.
- Common key is

$$d\big(c(I, e) * a\big) * b = c\big(d(I, e) * b\big) * a.$$

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 1: Generate lots of elements from A. Find linear information about d and the matrix part of b. Find d up to a scalar.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 1: Generate lots of elements from A. Find linear information about d and the matrix part of b. Find d up to a scalar.
- Phase 2: Use an algorithm from permutation group theory to find $a' \in A$ with same permutation as $c(I, e) * a$. Derive the shared key.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 1: Generate lots of elements from A. Find linear information about d and the matrix part of b. Find d up to a scalar.
- Phase 2: Use an algorithm from permutation group theory to find $a' \in A$ with same permutation as $c(I, e) * a$. Derive the shared key.

Both phases heuristic but practical for random system parameters.

The Kalka–Teicher–Tsaban approach

- System parameters: matrix size n, field size q, elements $\tau_i \in \mathbb{F}_q$.
- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 1: Generate lots of elements from A. Find linear information about d and the matrix part of b. Find d up to a scalar.
- Phase 2: Use an algorithm from permutation group theory to find $a' \in A$ with same permutation as $c(I, e) * a$. Derive the shared key.

Both phases heuristic but practical for random system parameters. Gunnells and Goldfeld avoid attack by choosing C carefully.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.
- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.
- Phase 1: Find $\tilde{a} \in A$ whose permutation agrees with a.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.
- Phase 1: Find $\tilde{a} \in A$ whose permutation agrees with a.
- Phase 2: Find $\tilde{c} \in C$ equivalent to $c \in C$. Recover remaining parameters, and the shared key.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.
- Phase 1: Find $\tilde{a} \in A$ whose permutation agrees with a.
- Phase 2: Find $\tilde{c} \in C$ equivalent to $c \in C$. Recover remaining parameters, and the shared key.

All phases are heuristic and practical.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.
- Phase 1: Find $\tilde{a} \in A$ whose permutation agrees with a.
- Phase 2: Find $\tilde{c} \in C$ equivalent to $c \in C$. Recover remaining parameters, and the shared key.

All phases are heuristic and practical. They do not depend on the choice of C.

The new approach

- Alice's public information: Generators for the group C of matrices and the subgroup A of the coloured Burau group.
- Eve also gets $c(I, e) * a$ and $d(I, e) * b$.

- Phase 0: Generate lots of words in the generators A whose associated permutation is trivial.
- Phase 1: Find $\tilde{a} \in A$ whose permutation agrees with a.
- Phase 2: Find $\tilde{c} \in C$ equivalent to $c \in C$. Recover remaining parameters, and the shared key.

All phases are heuristic and practical. They do not depend on the choice of C. Phases 0 and 1 use the KTT permutation group algorithm.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.
- Attack takes 8 hours. Half of this is precomputation.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.
- Attack takes 8 hours. Half of this is precomputation.
- Proposed ISO tag authentication protocol is vulnerable.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.
- Attack takes 8 hours. Half of this is precomputation.
- Proposed ISO tag authentication protocol is vulnerable.
- Defensive response from SecureRF (Anshel, Atkins, Gunnells, Goldfeld).

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.
- Attack takes 8 hours. Half of this is precomputation.
- Proposed ISO tag authentication protocol is vulnerable.
- Defensive response from SecureRF (Anshel, Atkins, Gunnells, Goldfeld).
- I would currently not recommend using the Algebraic Eraser primitive in any applications.

Consequences of the attack

- SecureRF provided five 128-bit parameter sets.
- Non-optimized implementation in Magma on one 2GHz Core.
- Attack takes 8 hours. Half of this is precomputation.
- Proposed ISO tag authentication protocol is vulnerable.
- Defensive response from SecureRF (Anshel, Atkins, Gunnells, Goldfeld).
- I would currently not recommend using the Algebraic Eraser primitive in any applications.
- Independent security analysis is vital.

Further discussion

- "Why Algebraic Eraser may be the riskiest cryptosystem you've never heard of", Dan Goodin, Ars Technica.

Further discussion

- "Why Algebraic Eraser may be the riskiest cryptosystem you've never heard of", Dan Goodin, Ars Technica.
- There is a thread on Cryptography Stack Exchange.

Further discussion

- “Why Algebraic Eraser may be the riskiest cryptosystem you’ve never heard of”, Dan Goodin, Ars Technica.
- There is a thread on Cryptography Stack Exchange.
- SRB, Robshaw ACNS 2016 give a real-time cryptanalysis of proposed ISO protocol.

Further discussion

- "Why Algebraic Eraser may be the riskiest cryptosystem you've never heard of", Dan Goodin, Ars Technica.
- There is a thread on Cryptography Stack Exchange.
- SRB, Robshaw ACNS 2016 give a real-time cryptanalysis of proposed ISO protocol.
- There is a new proposed ISO protocol.

Further discussion

- “Why Algebraic Eraser may be the riskiest cryptosystem you’ve never heard of”, Dan Goodin, Ars Technica.
- There is a thread on Cryptography Stack Exchange.
- SRB, Robshaw ACNS 2016 give a real-time cryptanalysis of proposed ISO protocol.
- There is a new proposed ISO protocol.
- Techniques from BR and this paper will apply.

Thanks!