Return of ECC dummy point additions: SPA on efficient P-256 implementation

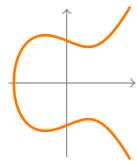
Andy Russon

3 June 2021

orange™

Introduction

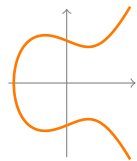
- Secure cryptography: avoid secret dependent behavior in execution



[1] Gueron and Krasnov, "Fast Prime Field Elliptic Curve Cryptography with 256 Bit Prime", 2013

Introduction

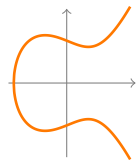
- Secure cryptography: avoid secret dependent behavior in execution
- Dummy operations might achieve this goal



[1] Gueron and Krasnov, "Fast Prime Field Elliptic Curve Cryptography with 256 Bit Prime", 2013

Introduction

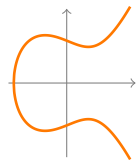
- Secure cryptography: avoid secret dependent behavior in execution
- Dummy operations might achieve this goal
- Example of elliptic curve P-256:



[1] Gueron and Krasnov, "Fast Prime Field Elliptic Curve Cryptography with 256 Bit Prime", 2013

Introduction

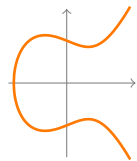
- Secure cryptography: avoid secret dependent behavior in execution
- Dummy operations might achieve this goal
- Example of elliptic curve P-256:
  - Popular curve, notably for digital signatures (with ECDSA)



[1] Gueron and Krasnov, "Fast Prime Field Elliptic Curve Cryptography with 256 Bit Prime", 2013

Introduction

- Secure cryptography: avoid secret dependent behavior in execution
- Dummy operations might achieve this goal
- Example of elliptic curve P-256:
  - Popular curve, notably for digital signatures (with ECDSA)
  - Efficient implementation in assembly available (OpenSSL and its forks)[1]



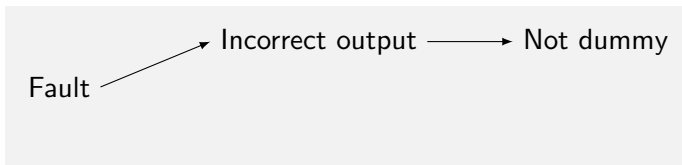
[1] Gueron and Krasnov, "Fast Prime Field Elliptic Curve Cryptography with 256 Bit Prime", 2013

Introduction

- Previously at SSTIC[2]
  - Fault injection to detect dummy operation

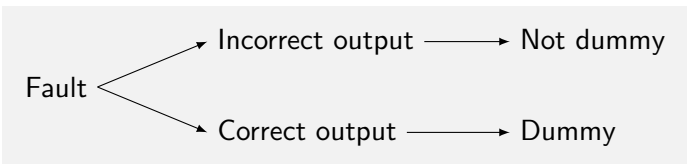
[2] "Exploiting dummy codes in Elliptic Curve Cryptography implementations", SSTIC 2020

- Previously at SSTIC[2]
  - Fault injection to detect dummy operation



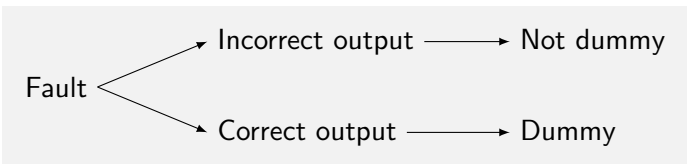
[2] "Exploiting dummy codes in Elliptic Curve Cryptography implementations", SSTIC 2020

- Previously at SSTIC[2]
  - Fault injection to detect dummy operation



[2] "Exploiting dummy codes in Elliptic Curve Cryptography implementations", SSTIC 2020

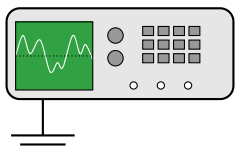
- Previously at SSTIC[2]
  - Fault injection to detect dummy operation
  - Not easy (material, timing, etc)
  - Produces many invalid outputs: hard to pass unnoticed



[2] "Exploiting dummy codes in Elliptic Curve Cryptography implementations", SSTIC 2020

In this presentation

- Goal: recover an ECDSA private key
  - On OpenSSL with the P-256 assembly optimized implementation
  - With power analysis to detect dummy operations



Plan

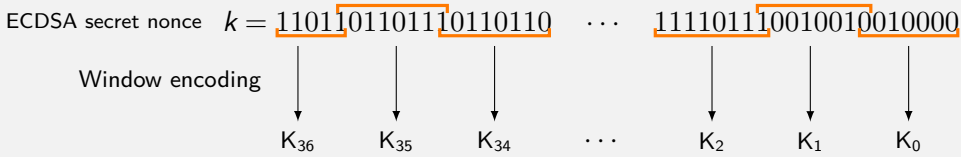
Assembly optimized P-256 implementation

ECDSA secret nonce $k = 1101101101110110110 \quad \cdots \quad 1111011100100100100000$

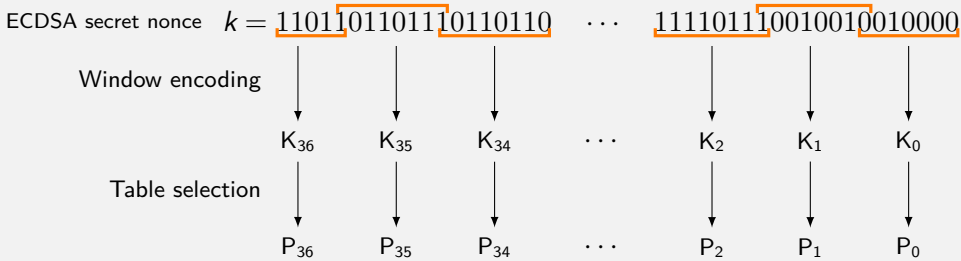
Assembly optimized P-256 implementation

ECDSA secret nonce $k = 1101101101110110110 \quad \cdots \quad 1111011100100100010000$

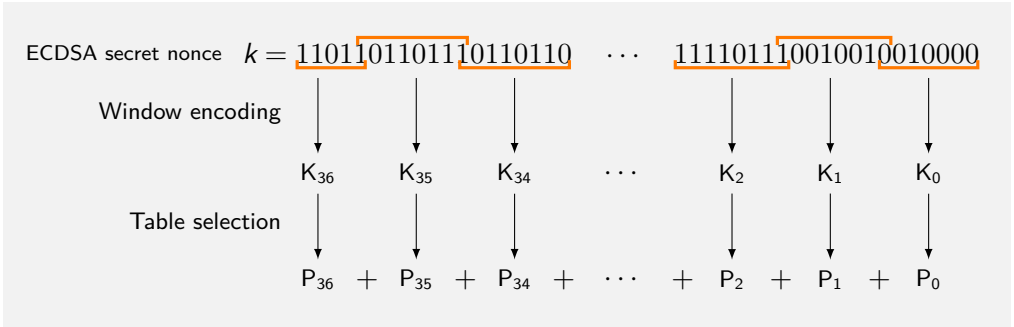
Assembly optimized P-256 implementation



Assembly optimized P-256 implementation

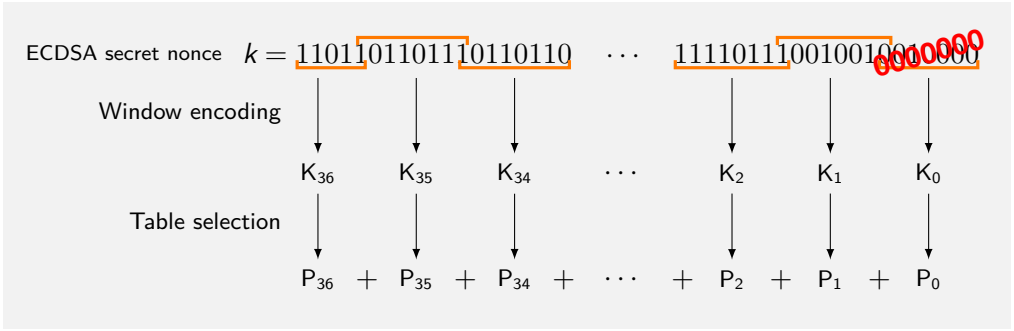


Assembly optimized P-256 implementation



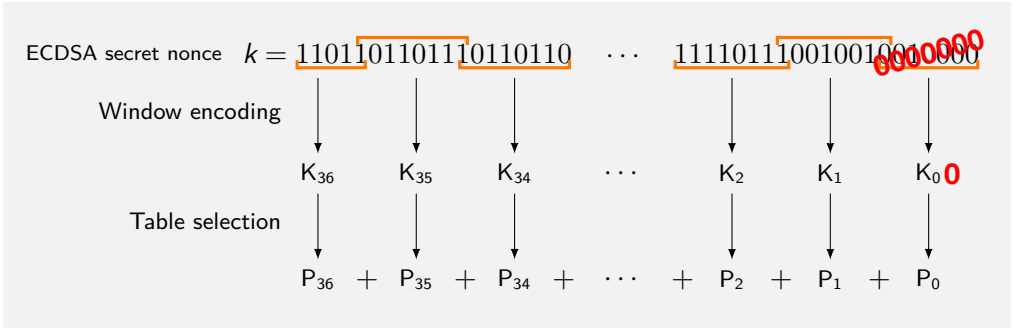
function `ecp_nistz_point_add_affine`

Assembly optimized P-256 implementation



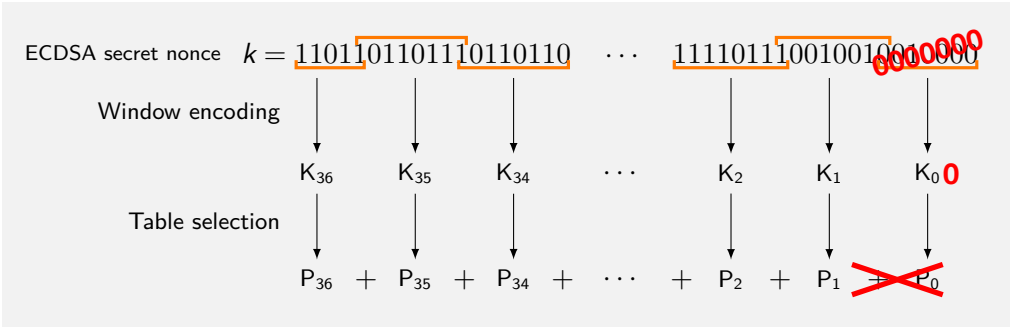
function `ecp_nistz_point_add_affine`

Assembly optimized P-256 implementation



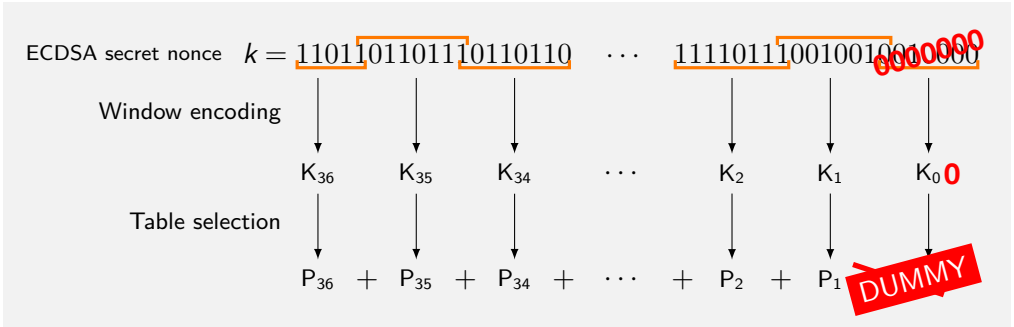
function `ecp_nistz_point_add_affine`

Assembly optimized P-256 implementation



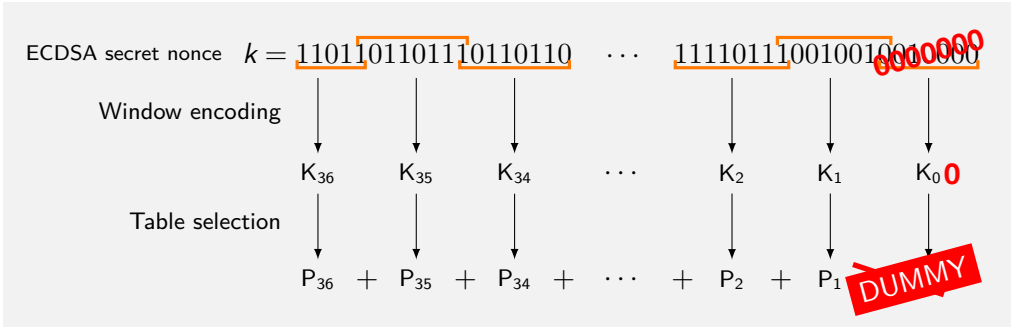
function `ecp_nistz_point_add_affine`

Assembly optimized P-256 implementation



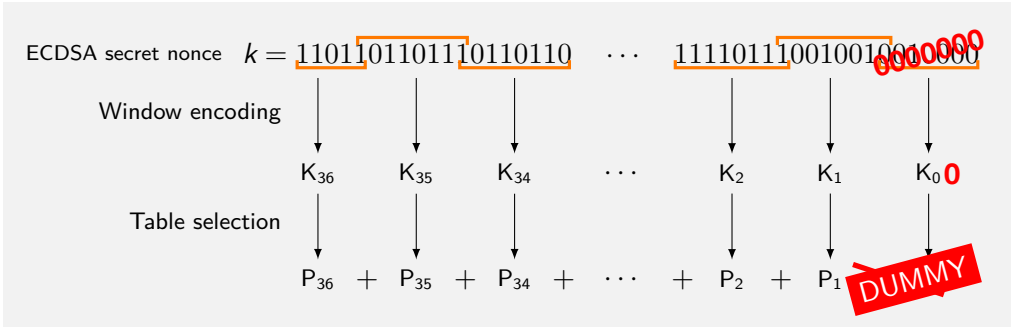
- Always 36 execution function `ecp_nistz_point_add_affine`

Assembly optimized P-256 implementation



- Always 36 execution function `ecp_nistz_point_add_affine`
- Null window $\Rightarrow$ dummy point addition

Assembly optimized P-256 implementation

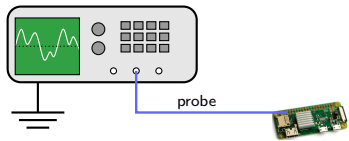


- Always 36 execution function `ecp_nistz_point_add_affine`
- Null window $\Rightarrow$ dummy point addition
- First window: K_0 is 7 lowest bits of k

Plan

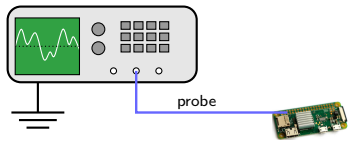
Steps of the attack

1. Capture power consumption of an ECDSA signature generation



Steps of the attack

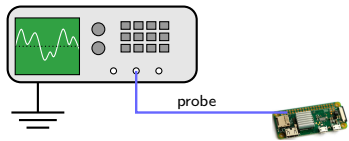
1. Capture power consumption of an ECDSA signature generation



2. Keep the signature if there is a lower power consumption during the **first** point addition

Steps of the attack

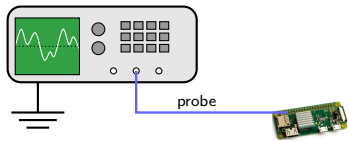
1. Capture power consumption of an ECDSA signature generation



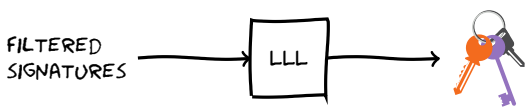
2. Keep the signature if there is a lower power consumption during the **first** point addition
3. Repeat until enough signatures are collected (around 37)

Steps of the attack

1. Capture power consumption of an ECDSA signature generation



2. Keep the signature if there is a lower power consumption during the **first** point addition
3. Repeat until enough signatures are collected (around 37)
4. Recover the private key from the signatures

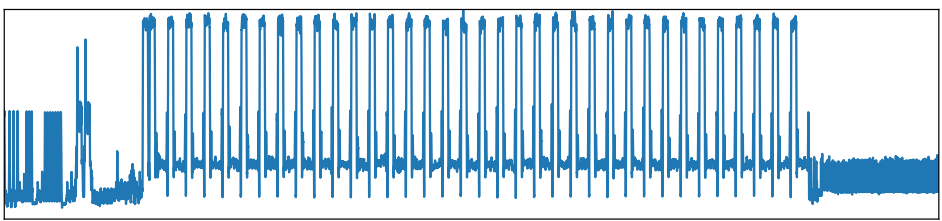


Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model

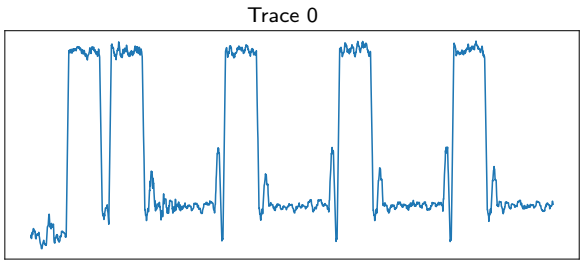
Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



Power trace simulation

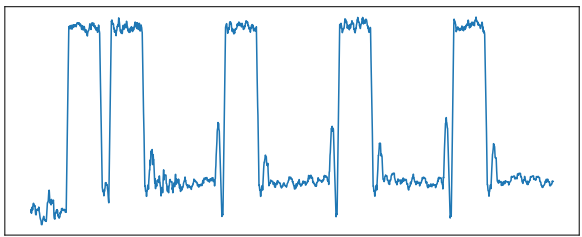
- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model

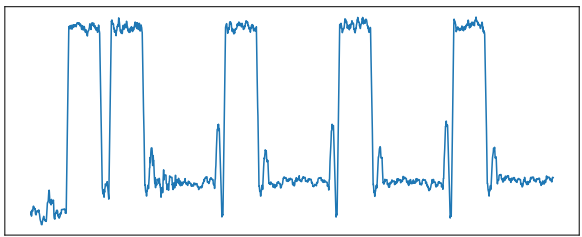
Trace 1



Power trace simulation

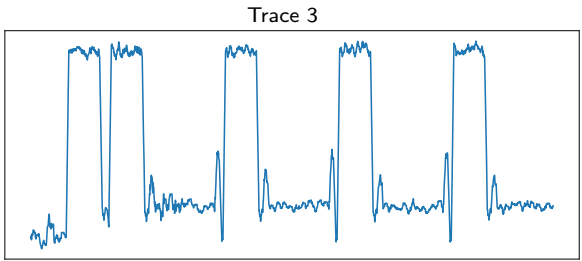
- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model

Trace 2



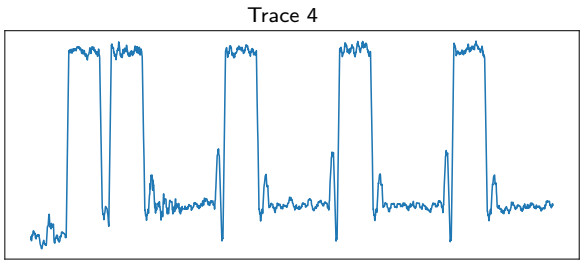
Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



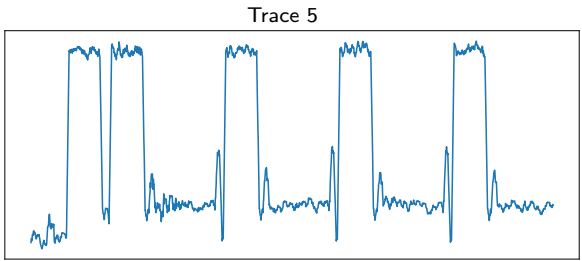
Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



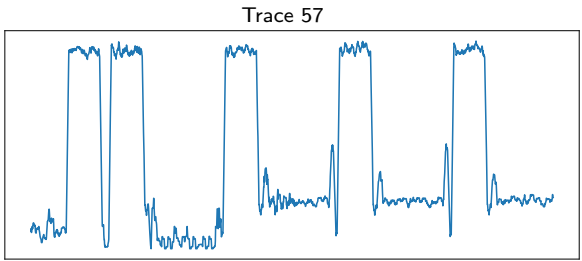
Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



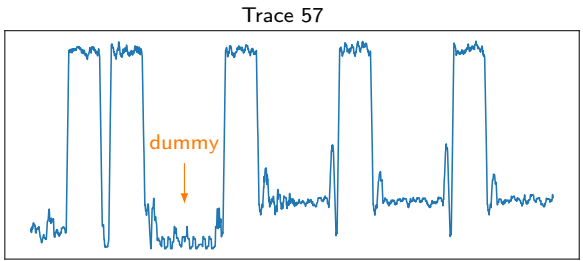
Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



Power trace simulation

- `TracerGrind`: a `Valgrind` plugin to monitor processes
- Simulation of power traces:
  - Capture memory trace of a signature generation
  - Apply Hamming weight model



How many signatures we need?

- Number of signatures:
  - 7 lowest bits of k null: 1 out of 128 signatures on average
  - Key recovery: 37 signatures (or more) if 7 bits of nonce known
  - Total: $37 \times 128 = 4738$

How many signatures we need?

- Number of signatures:
  - 7 lowest bits of k null: 1 out of 128 signatures on average
  - Key recovery: 37 signatures (or more) if 7 bits of nonce known
  - Total: $37 \times 128 = 4738$
- Clustering the traces:
  - `GaussianMixture` class in `Scikit-learn` Python library
  - Classify the traces in **two** components

```
from sklearn import mixture
gm = mixture.GaussianMixture(n_components=2)
results = gm.fit_predict(traces)
```

Example

- Trace generation:

```
python3 simulation.py -b /usr/local/bin/openssl --privkey privkey.pem
  --working-dir test -n 6000 --bounds 1844200 1844500
```

Tools available at `https://github.com/orangecertcc/ecdummyrpa`

Example

- Trace generation:

```
python3 simulation.py -b /usr/local/bin/openssl --privkey privkey.pem
  --working-dir test -n 6000 --bounds 1844200 1844500
```

- Analysis:

```
python3 analysis.py --pubkey pubkey.pem --working-dir test -n 6000
```

Tools available at `https://github.com/orangecertcc/ecdummyrpa`

Example

- Trace generation:

```
python3 simulation.py -b /usr/local/bin/openssl --privkey privkey.pem
  --working-dir test -n 6000 --bounds 1844200 1844500
```

- Analysis:

```
python3 analysis.py --pubkey pubkey.pem --working-dir test -n 6000
```

- Results:

```
Loading 6000 traces...
Clustering of the traces...
Size of clusters: 5950, 50
Lattice attack: recovering of the key with 37 signatures...
Lattice attack: recovering of the key with 38 signatures...
SUCCESS!
The private key is deb867c4cf52002d7d8cc2e440d16c8c75c7844fdb142ea8c1a1f48b411b0251
```

Tools available at `https://github.com/orangecertcc/ecdummyrpa`

Plan

Handling of the dummy point addition

`ecp_nistz_point_add_affine`:

Input: $P = (X_1, Y_1, Z_1)$, $Q = (x_2, y_2)$ **Output:** $P + Q = (X_3, Y_3, Z_3)$

$\texttt{in1infty} \leftarrow (Z_1 == 0)$	$t_5 \leftarrow 2 \cdot t_1$
$\texttt{in2infty} \leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow {Z_1}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot t_0$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow t_1 - X_1$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** $\texttt{in1infty}$ **then**
$t_4 \leftarrow t_3 - Y_1$	$\quad (X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** $\texttt{in2infty}$ **then**
$t_6 \leftarrow {t_4}^2$	$\quad (X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
$\texttt{in1infty} \leftarrow (Z_1 == 0)$	$t_5 \leftarrow 2 \cdot t_1$
$\texttt{in2infty} \leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow {Z_1}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot t_0$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow t_1 - X_1$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$ **Output:** $P + Q = (X_3, Y_3, Z_3)$

`in1infty` $\leftarrow$ **true**	$t_5 \leftarrow 2 \cdot t_1$
`in2infty` $\leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow {Z_1}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot t_0$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow t_1 - X_1$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
$\texttt{in1infty} \leftarrow \textbf{true}$	$t_5 \leftarrow 2 \cdot t_1$
$\texttt{in2infty} \leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot t_0$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow t_1 - X_1$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
`in1infty` $\leftarrow$ **true**	$t_5 \leftarrow 2 \cdot t_1$
`in2infty` $\leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow t_1 - X_1$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
$\texttt{in1infty} \leftarrow \mathbf{true}$	$t_5 \leftarrow 2 \cdot t_1$
$\texttt{in2infty} \leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow \mathbf{0} - \mathbf{0}$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow t_0 \cdot Z_1$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
`in1infty` $\leftarrow$ **true**	$t_5 \leftarrow 2 \cdot t_1$
`in2infty` $\leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow \mathbf{0} - \mathbf{0}$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow t_2 \cdot Z_1$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
$\texttt{in1infty} \leftarrow \mathbf{true}$	$t_5 \leftarrow 2 \cdot t_1$
$\texttt{in2infty} \leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow t_6 - t_5$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow X_3 - t_7$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow t_1 - X_3$
$t_2 \leftarrow \mathbf{0} - \mathbf{0}$	$t_3 \leftarrow Y_1 \cdot t_7$
$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$t_2 \leftarrow t_2 \cdot t_4$
$Z_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$Y_3 \leftarrow t_2 - t_3$
$t_3 \leftarrow t_3 \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow t_3 - Y_1$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow {t_2}^2$	**if** `in2infty` **then**
$t_6 \leftarrow {t_4}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow t_5 \cdot t_2$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow X_1 \cdot t_5$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$	**Output:** $P + Q = (X_3, Y_3, Z_3)$
`in1infty` $\leftarrow$ **true**	$t_5 \leftarrow 2 \cdot \mathbf{0}$
`in2infty` $\leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow \mathbf{0} - \mathbf{0}$
$t_2 \leftarrow \mathbf{0} - \mathbf{0}$	$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$
$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$t_2 \leftarrow \mathbf{0} \cdot \mathbf{0}$
$Z_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$Y_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_3 \leftarrow \mathbf{0} \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow \mathbf{0} - \mathbf{0}$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow \mathbf{0}^2$	**if** `in2infty` **then**
$t_6 \leftarrow \mathbf{0}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow \mathbf{0} \cdot \mathbf{0}$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow \mathbf{0} \cdot \mathbf{0}$	

Handling of the dummy point addition

- If $K_0 = 0$, then P is initiliaze with $(0, 0, 0)$
- Consequence in `ecp_nistz_point_add_affine`:

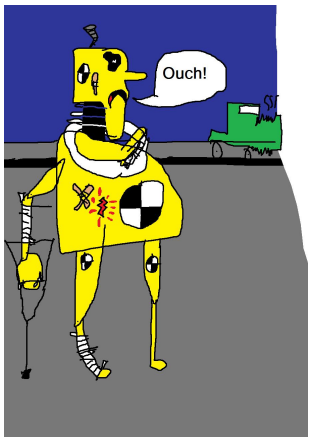
Input: $P = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q = (x_2, y_2)$ **Output:** $P + Q = (X_3, Y_3, Z_3)$

`in1infty` $\leftarrow$ **true**	$t_5 \leftarrow 2 \cdot \mathbf{0}$
`in2infty` $\leftarrow (x_2, y_2) == (0, 0)$	$X_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_0 \leftarrow \mathbf{0}^2$	$X_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_1 \leftarrow x_2 \cdot \mathbf{0}$	$t_2 \leftarrow \mathbf{0} - \mathbf{0}$
$t_2 \leftarrow \mathbf{0} - \mathbf{0}$	$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$
$t_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$t_2 \leftarrow \mathbf{0} \cdot \mathbf{0}$
$Z_3 \leftarrow \mathbf{0} \cdot \mathbf{0}$	$Y_3 \leftarrow \mathbf{0} - \mathbf{0}$
$t_3 \leftarrow \mathbf{0} \cdot y_2$	**if** `in1infty` **then**
$t_4 \leftarrow \mathbf{0} - \mathbf{0}$	$(X_3, Y_3, Z_3) \leftarrow (x_2, y_2, 1)$
$t_5 \leftarrow \mathbf{0}^2$	**if** `in2infty` **then**
$t_6 \leftarrow \mathbf{0}^2$	$(X_3, Y_3, Z_3) \leftarrow (X_1, Y_1, Z_1)$
$t_7 \leftarrow \mathbf{0} \cdot \mathbf{0}$	**return** (X_3, Y_3, Z_3)
$t_1 \leftarrow \mathbf{0} \cdot \mathbf{0}$	

ignore previous calculation

Plan

Conclusion and mitigations



- SPA on efficient implementation of elliptic curve P-256
- Less than 5000 traces to recover a private key
- Mitigations:
  - Dummy additions with random values instead of zero
  - Use regular algorithms that do not need dummy additions
- Proof of concept and tools available as a Docker:

  `https://github.com/orangecertcc/ecdummyrpa`