

Dragonblood is Still Leaking: Practical Cache-based Side-Channel in the Wild

Daniel De Almeida Braga

Pierre-Alain Fouque

Mohamed Sabt

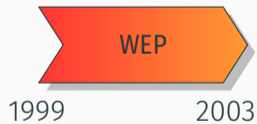
CSAW 2020 - November, 6th 2020



Context and Motivations



Context and Motivations



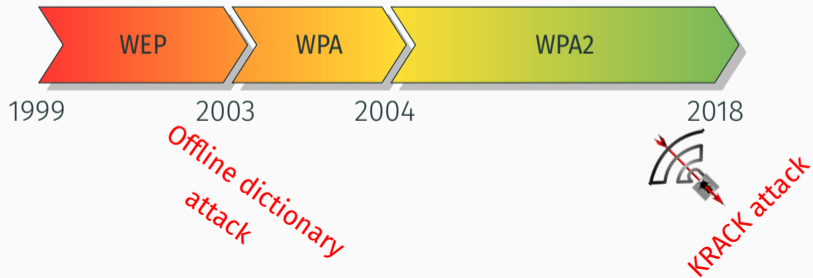
Context and Motivations



Context and Motivations



Context and Motivations



Context and Motivations

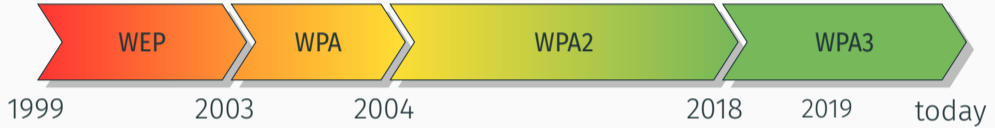


- + More secure
- + Based on a PAKE (Dragonfly)

PAKE: Password Authenticated Key Exchange

- PAKE protocols aim to combine the Key Exchange and authentication parts
- Password is used to:
 - Authenticate the user
 - Derive strong cryptographic material
- No offline dictionary attack

Context and Motivations



- + More secure
- + Based on a PAKE (Dragonfly)



Dragonblood attacks

Side Channel Attacks

```
def processPassword(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)  
    else:  
        res = short_processing(pwd)  
    return res
```

Side Channel Attacks

```
def processPassword(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)  
    else:  
        res = short_processing(pwd)  
    return res
```

Gain information through timing:



0.5 seconds \Rightarrow no *a*



10 seconds \Rightarrow *a*

Side Channel Attacks

```
def processPassword(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)  
    else:  
        res = short_processing(pwd)  
    return res
```

```
def processPassword2(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)  
    else:  
        res = long_processing2(pwd)  
    return res
```

Gain information through timing:



0.5 seconds \Rightarrow no *a*



10 seconds \Rightarrow *a*

Side Channel Attacks

```
def processPassword(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)  
    else:  
        res = short_processing(pwd)  
    return res
```


Gain information through timing:



0.5 seconds \Rightarrow no *a*



10 seconds \Rightarrow *a*

```
def processPassword2(pwd):  
    if "a" in pwd:  
        res = long_processing(pwd)   
    else:  
        res = long_processing2(pwd)  
    return res
```

Gain information execution flow:

- Execute `long_processing` \Rightarrow *a*
- Else, no *a* in *pwd*

1. Show that current countermeasures are not sufficient for cache-based side-channel

1. Show that current countermeasures are not sufficient for cache-based side-channel
2. Mount an offline dictionary attack to recover the password

1. Show that current countermeasures are not sufficient for cache-based side-channel
2. Mount an offline dictionary attack to recover the password
3. Provide a PoC on Real-World-like scenarios (IWD and FreeRadius)

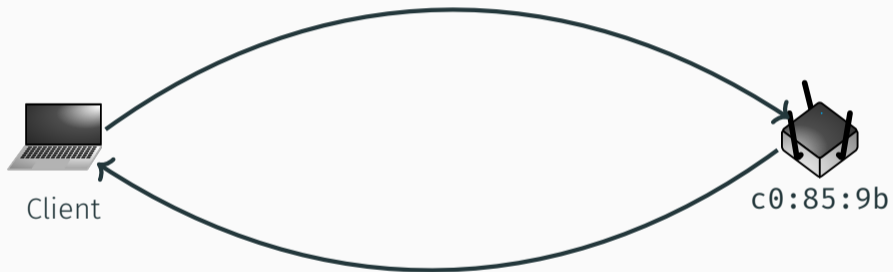


1. Show that current countermeasures are not sufficient for cache-based side-channel
2. Mount an offline dictionary attack to recover the password
3. Provide a PoC on Real-World-like scenarios (IWD and FreeRadius)



4. Raise awareness on how practical these attacks are

Attacker Model



Attacker Model



Client



c0:85:9b

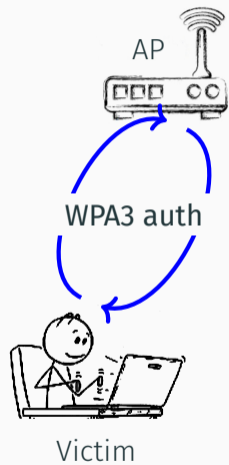
Attacker Model



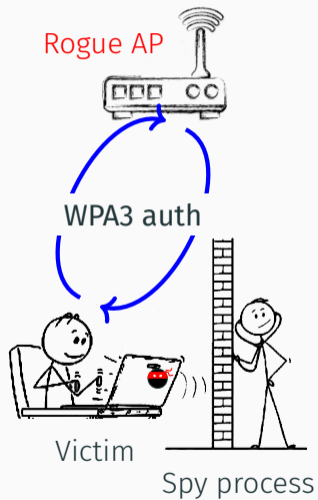
Attacker Model



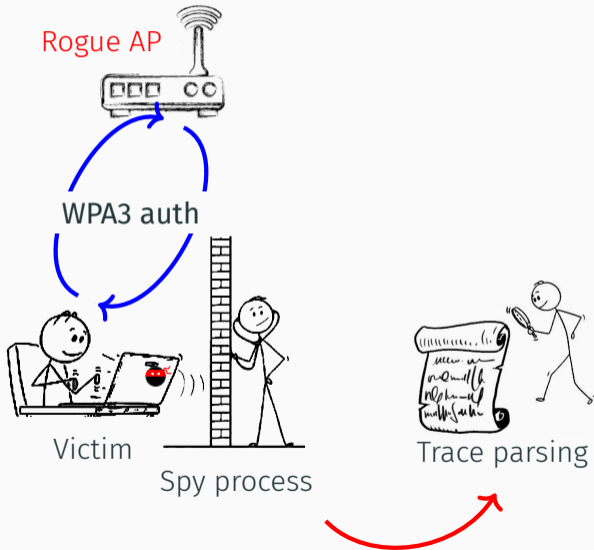
Core Idea



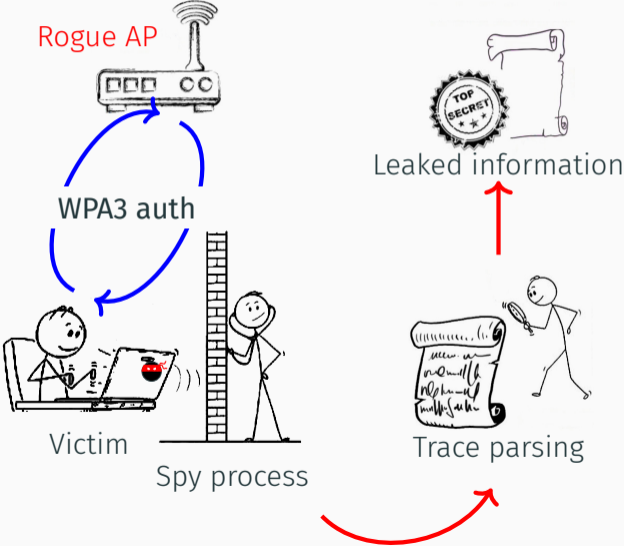
Core Idea



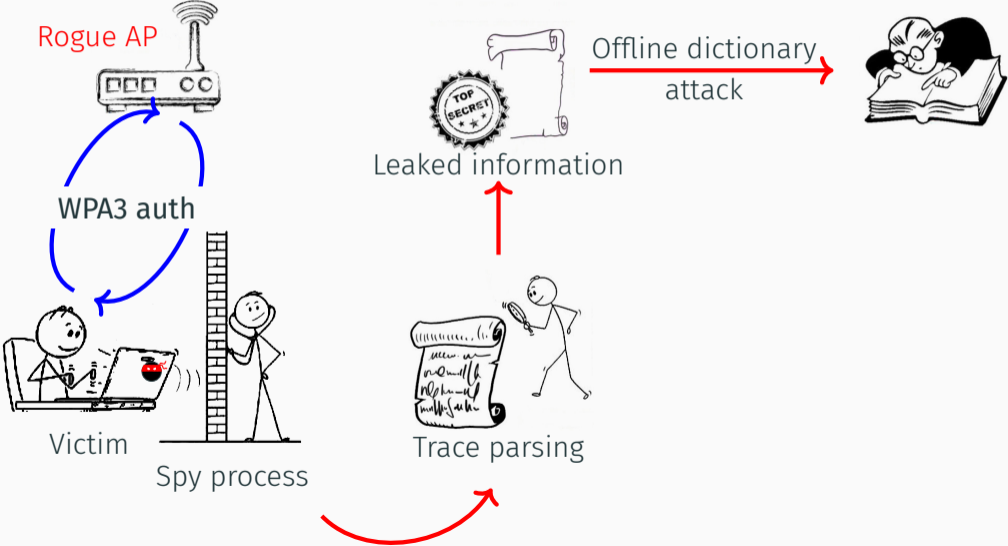
Core Idea



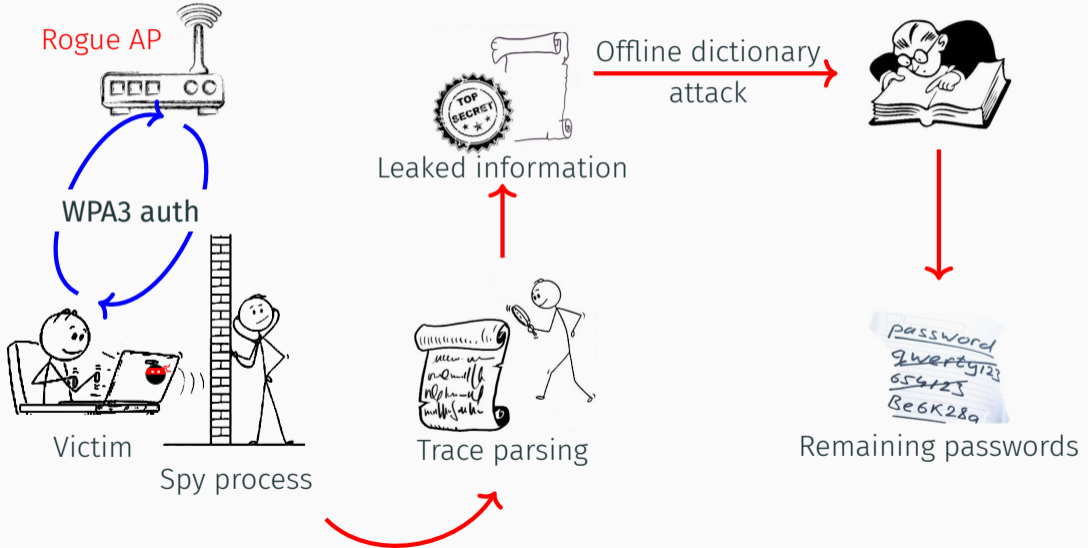
Core Idea



Core Idea

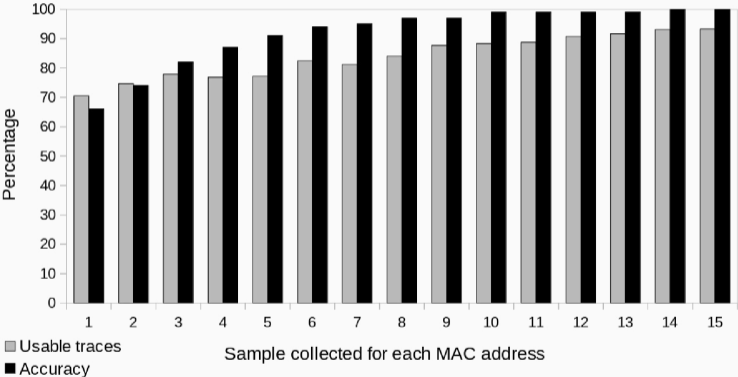


Core Idea



Practical Results

Achieve very reliable results with only 10 measurements per MAC address



	Dict. size	Cost on AWS	Avg traces for full reduction
Rockyou	$1.4 \cdot 10^7$	0,00037 €	16
CrackStation	$3.5 \cdot 10^7$	0,0011 €	17
HavelBeenPwned	$5.5 \cdot 10^8$	0,014 €	20
8 characters	$4.6 \cdot 10^{14}$	11848,2 €	32

Number of the Required Traces / Cost to Prune all Wrong Passwords

Practical Results

	Dict. size	Cost on AWS	Avg traces for full reduction
Rockyou	$1.4 \cdot 10^7$	0,00037 €	16
CrackStation	$3.5 \cdot 10^7$	0,0011 €	17
HaveIBeenPwned	$5.5 \cdot 10^8$	0,014 €	20
8 characters	$4.6 \cdot 10^{14}$	11848,2 €	32

Number of the Required Traces / Cost to Prune all Wrong Passwords

Practical Results

	Dict. size	Cost on AWS	Avg traces for full reduction
Rockyou	$1.4 \cdot 10^7$	0,00037 €	16
CrackStation	$3.5 \cdot 10^7$	0,0011 €	17
HaveIBeenPwned	$5.5 \cdot 10^8$	0,014 €	20
8 characters	$4.6 \cdot 10^{14}$	11848,2 €	32

Number of the Required Traces / Cost to Prune all Wrong Passwords

IWD v1.9 ✓

2020-08-03 sae: Fix a side channel leak on the password  Daniel DE ALMEIDA BRAGA 2 -40/+135

FreeRadius to be fixed in 3.0.22

merge constant time fixes from "master" ...

Based on a patch from Daniel De Almeida Braga.

The code is now largely the same between master and v3.0.x, which makes it easier to see that it's correct

Thank you for your attention!



`https://gitlab.inria.fr/ddealmei/poc-iwd-acsac2020`



`daniel.de-almeida-braga@irisa.fr`

DragonflyProtocol overview

A and B agree on a prime order group $E(\mathbb{F}_p)$, of order q

Dragonfly

Alice (A)

$P \leftarrow \text{p2g}(\text{pwd}, A, B)$

Bob (B)

$P \leftarrow \text{p2g}(\text{pwd}, A, B)$

Commit



Key derivation

Key derivation

Confirmation



DragonflyProtocol overview

A and B agree on a prime order group $E(\mathbb{F}_p)$, of order q

Dragonfly

Alice (A)

$P \leftarrow p2g(pwd, A, B)$

Bob (B)

$P \leftarrow p2g(pwd, A, B)$

Commit



Key derivation

Key derivation

Confirmation



Dragonfly - Secret derivation (EC)

HuntingAndPecking(*pwd*, *A*, *B*)

n, *found*, *i* \leftarrow *len*(*p*) + 64, **false**, 1

while not *found* :

base \leftarrow *H*(*max*(*A*, *B*) | *min*(*A*, *B*) | *pwd* | *i*)

s \leftarrow (*KDF*_{*n*}(*base*, *label*) mod (*p* - 1)) + 1

if *s*³ + *as* + *b* is a quadratic residue mod *p*

found, *x* \leftarrow **true**, *s*

i = *i* + 1

y \leftarrow $\sqrt{x^3 + ax + b}$

if *lsb*(*y*) == *lsb*(*base*) :

return (*x*, *y*)

return (*x*, *p* - *y*)

Dragonfly - Secret derivation (EC)

HuntingAndPecking(*pwd*, *A*, *B*, *k*)

n, *found*, *i* \leftarrow *len*(*p*) + 64, **false**, 1

while not *found* **and** *i* < *k* :

base \leftarrow *H*(*max*(*A*, *B*) | *min*(*A*, *B*) | *pwd* | *i*)

s \leftarrow (*KDF*_{*n*}(*base*, *label*) mod (*p* - 1)) + 1

if *s*³ + *as* + *b* is a quadratic residue mod *p*

if not *found* :

found, *x*, *base*_{*saved*} \leftarrow **true**, *s*, *base*

i = *i* + 1

y \leftarrow $\sqrt{x^3 + ax + b}$

if *lsb*(*y*) == *lsb*(*base*_{*saved*}) :

return (*x*, *y*)

return (*x*, *p* - *y*)

Dragonfly - Secret derivation (EC)

HuntingAndPecking(*pwd*, *A*, *B*, *k*)

n, *found*, *i* \leftarrow *len*(*p*) + 64, **false**, 1

while not found and *i* < *k* :

base \leftarrow *H*(*max*(*A*, *B*) | *min*(*A*, *B*) | *pwd* | *i*)

s \leftarrow (*KDF*_{*n*}(*base*, *label*) mod (*p* - 1)) + 1

if *s*³ + *as* + *b* is a quadratic residue mod *p*

if not found :

found, *x*, *base*_{*saved*} \leftarrow **true**, *s*, *base*

i = *i* + 1

y \leftarrow $\sqrt{x^3 + ax + b}$

if *lsb*(*y*) == *lsb*(*base*_{*saved*}) :

return (*x*, *y*)

return (*x*, *p* - *y*)

Dragonfly - Secret derivation (EC)

HuntingAndPecking(pwd, A, B, k)

$n, found, i \leftarrow len(p) + 64, \text{false}, 1$

while not found and $i < k$:

$base \leftarrow H(\max(A, B) \mid \min(A, B) \mid pwd \mid i)$

$s \leftarrow (KDF_n(base, label) \bmod (p - 1)) + 1$

if $s^3 + as + b$ is a quadratic residue mod p

if not found :

$found, x, base_{saved} \leftarrow \text{true}, s, base$

$i = i + 1$

$y \leftarrow \sqrt{x^3 + ax + b}$

if $lsb(y) == lsb(base_{saved})$:

return (x, y)

return $(x, p - y)$

\leftarrow : new iteration

Dragonfly - Secret derivation (EC)

HuntingAndPecking(*pwd*, *A*, *B*, *k*)

n, *found*, *i* \leftarrow *len*(*p*) + 64, **false**, 1

while not found and *i* < *k* :

base \leftarrow *H*(*max*(*A*, *B*) | *min*(*A*, *B*) | *pwd* | *i*)

s \leftarrow (*KDF*_{*n*}(*base*, *label*) mod (*p* - 1)) + 1

if *s*³ + *as* + *b* is a quadratic residue mod *p*

if not found :

found, *x*, *base*_{*saved*} \leftarrow **true**, *s*, *base*

i = *i* + 1

y \leftarrow $\sqrt{x^3 + ax + b}$

if *lsb*(*y*) == *lsb*(*base*_{*saved*}) :

return (*x*, *y*)

return (*x*, *p* - *y*)

\leftarrow : new iteration

\leftarrow : successful conversion