WPA-SEC:

The largest online WPA handshake database

Alex Stanev
alex@stanev.org
@RealEnderSec

https://wpa-sec.stanev.org

https://github.com/RealEnder/dwpa

What is wpa-sec?

- We collect and process wireless network captures – submitted by wpa-sec users
- Identify WPA/WPA2 handshakes
- Maintain set of dictionaries to check against handshakes
- Contributors use help_crack python script to download handshakes and dicts and initiate attacks
- The results are submitted back to wpa-sec DB
- Cracked dictionary available for free download, updated in realtime

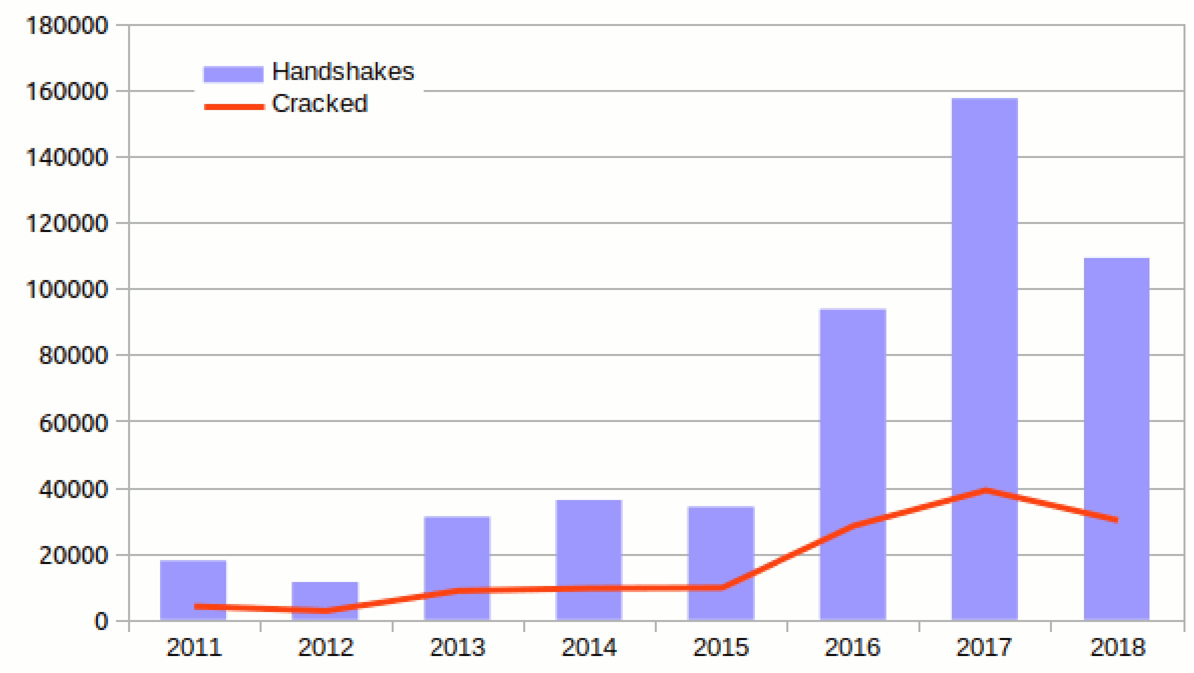
https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

wpa-sec software infrastructure

- hcxtools for handshake identification
  https://github.com/ZerBea/hcxtools
- RouterKeygenPC for known PSK algorithm generation
  https://github.com/routerkeygen/routerkeygenPC
- Hashcat & John the Ripper (bleeding) as crackers
  https://github.com/hashcat/hashcat
  https://github.com/magnumripper/JohnTheRipper
- Wigle for wireless network mapping
  https://wigle.net

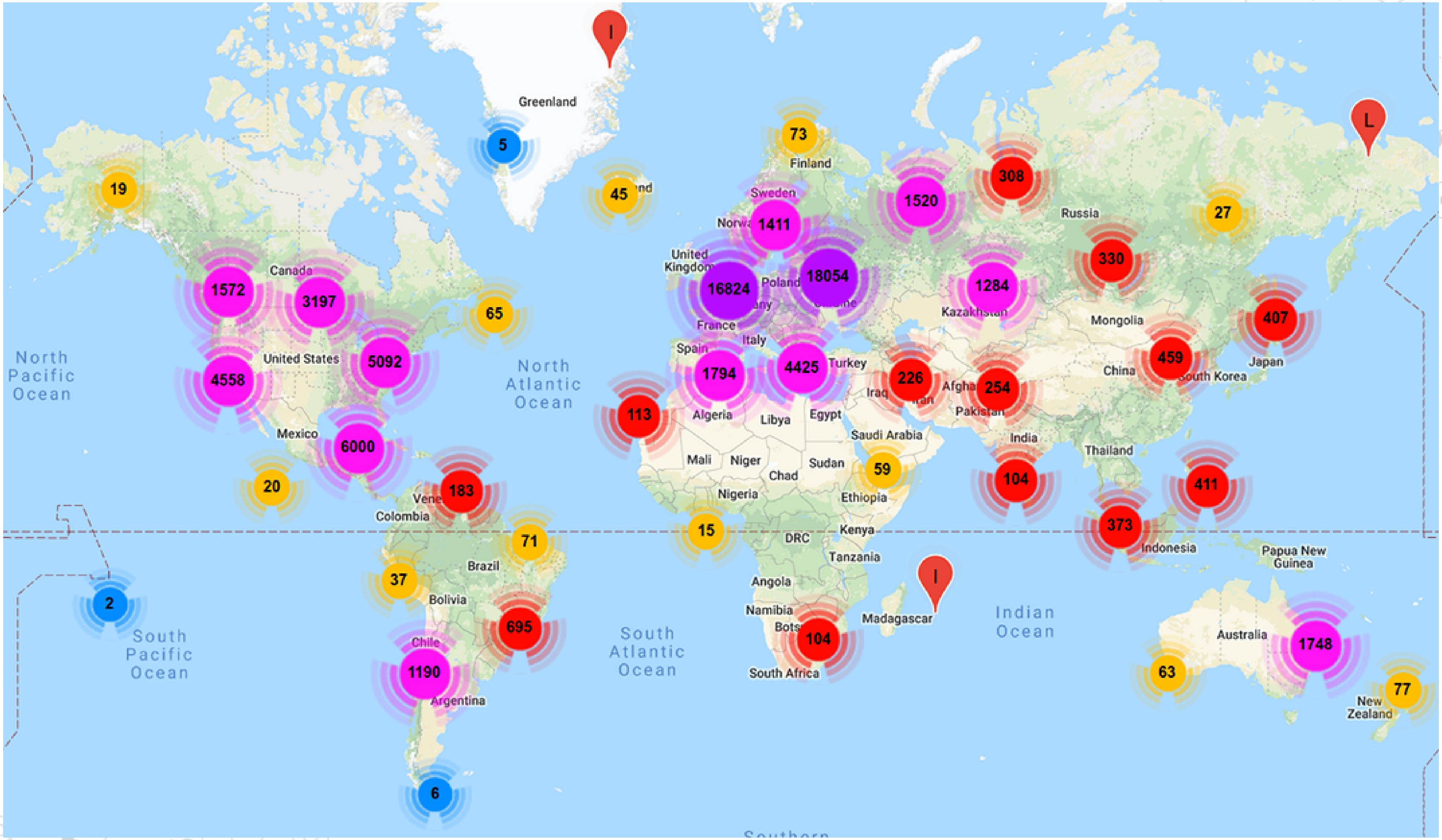
wpa-sec stats

- **0.5M+** handshakes submitted
- **140GB+** of raw capture data
- **27%+** cracked
- **<5%** hit by known algorithm
- Only **<7%** of dictionary keyspace progress



https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

AP geo distribution



https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

Get the handshake

- Oldschool **AP attack** – deauthenticate active clients

  Pros: identify exact AP bssid

  Cons: may need AP nonce correction due to retransmissions, may result in uncrackable handshakes

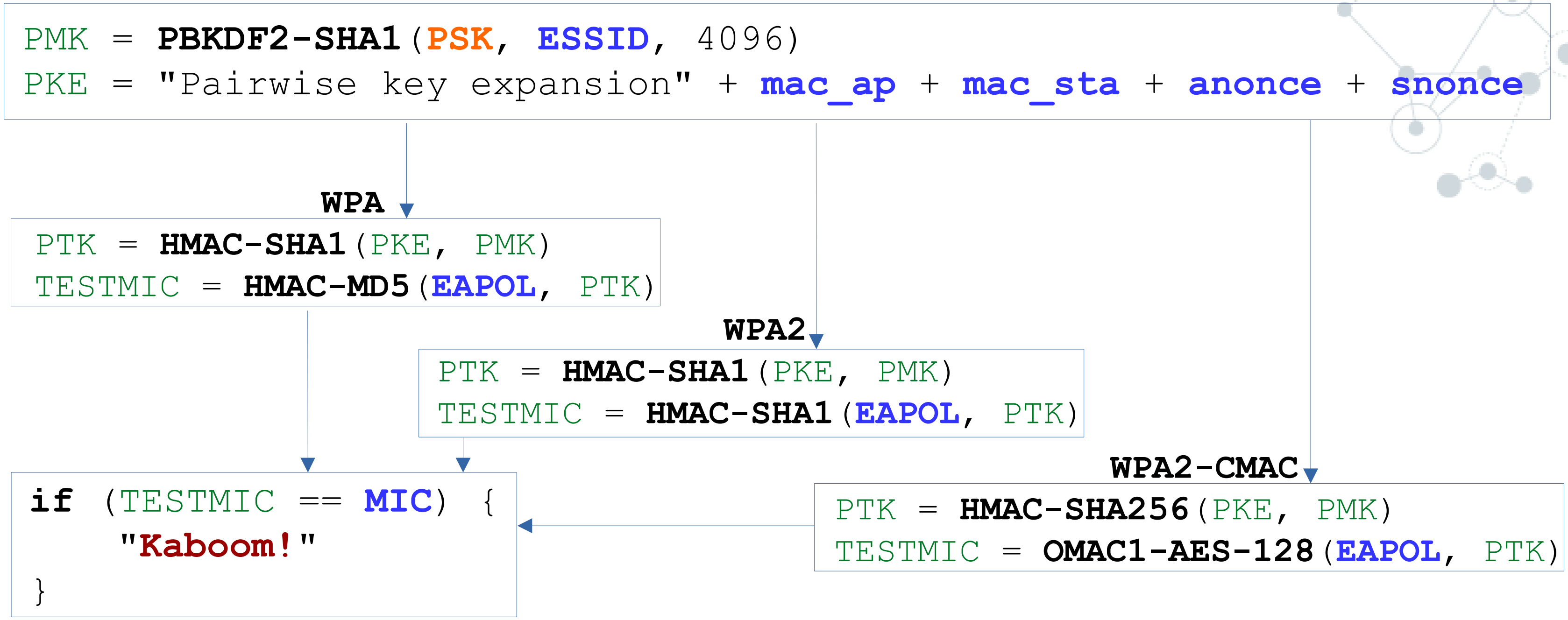
- **AP-less attack** – attack directly the client, pretending to be a known AP

  Pros: no need for AP nonce correction, no uncrackable handshakes, can continue with higher level attacks (hello WPA-Enterprise)

  Cons: we can’t extract AP bssid if the client transmits unidirected proberequest, only essid, so limited known PSK algorithm support

- Tool of choice: **hcxdumptool** – uses raw sockets https://github.com/ZerBea/hcxdumptool
- **Warning: do not postprocess/clean the captures!**

The crack – good old handshake



```
PMK = PBKDF2-SHA1(PSK, ESSID, 4096)
PKE = "Pairwise key expansion" + mac_ap + mac_sta + anonce + snonce
```

WPA

```
PTK = HMAC-SHA1(PKE, PMK)
TESTMIC = HMAC-MD5(EAPOL, PTK)
```

WPA2

```
PTK = HMAC-SHA1(PKE, PMK)
TESTMIC = HMAC-SHA1(EAPOL, PTK)
```

WPA2-CMAC

```
PTK = HMAC-SHA256(PKE, PMK)
TESTMIC = OMAC1-AES-128(EAPOL, PTK)
```

```
if (TESTMIC == MIC) {
    "Kaboom!"
}
```

Messages	EAPOL from	AP	STA	Note
M1M2	M2	M1	M2	Unauthorized handshake – typos, other nets
M1M4	M4	M1	M4	
M2M3	M2	M3	M2	Unauthorized handshake – typos, other nets
M2M3	M3	M3	M2	
M3M4	M3	M3	M4	
M3M4	M4	M3	M4	

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

What is AP nonce correction?

- Due to retransmissions in crowded areas, weak signal, aggressive deauth attacks…
- APs increment the anonce value during handshake
- In perfect world we can use Replay-counter field, but it often stays the same
- The result is uncrackable handshake – packets look good, but came from different phases of auth sequence
- The penalty nc=32 is ~3%:

  + or – the correction value

  Big endian/Little endian devices
- In **hcxtools** we detect and deal with such situations

NC needed	5%
-NC	30%
+NC	70%
BE	90%
LE	10%

```
Anonce:
7b2076cfb5c0...18eb6556d17886f38e8bd2172
```

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

The crack – welcome PMKID

- Attack against 802.11i/p/q/r – networks with roaming functions enabled
- Also works in AP-less mode
- PMK is stored in sta and ap, along with mac_sta, mac_ap, PMK lifetime and has unique identifier PMKID = PMK Security Association (PMKSA)
- PMK is computed like this:

```
PMKID=HMAC-SHA1-128(PMK,"PMK Name"+mac_ap+mac_sta)
```

- We can get those from only 2 frames:

  AssociationRequest/ReassociationRequest/ProbeResponse

  EAPOL 1/4 (M1) with included RSN IE
- Capture with *hcxdumptool*
- *Hashcat* modes 16800/16801 (since 4.2.0)

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

Capture hardware

- We must be fast, so we can respond within EAPOL-Key Timeout
- In crowded areas and octo-core devices this can be challenging

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

Captures submission

- First, issue your own wpa-sec key

On server side:

- Process capture through *hcxpcaptool*
- Hash the handshakes and look for duplicates in the DB
- For every new handshake look for already cracked handshakes with the same essid/bssid/mac_sta
- If found, try to crack new ones with PMK

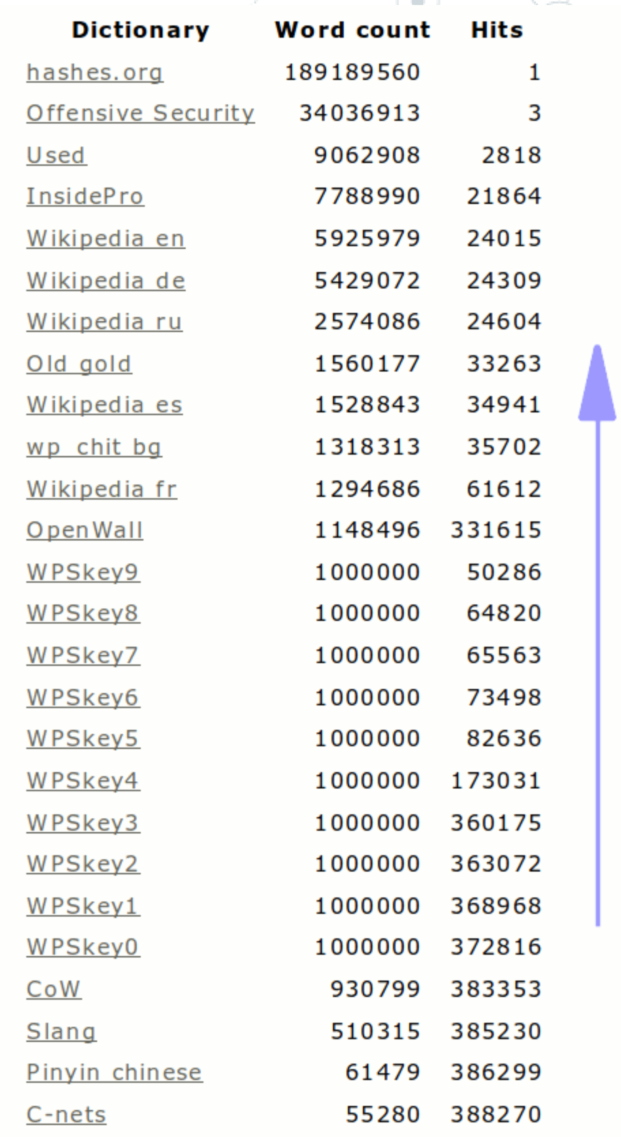
```
PMK = PBKDF2-SHA1(PSK, ESSID, 4096)
```

- Try RouterKeygenPC and some custom rules
- Query Wigle for AP geolocation
- Release the handshake for crackers

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

Cracker get_work

- Contributors run help_crack
- It downloads handshakes and dictionaries and feeds the cracker
- Start from dicts with fewer words
- Start from oldest handshakes
- **ESSID combine**: group all handshakes with same ESSIDs for current dict selection
- **Auto dict count**: download more dicts based on client performance – avoids GPU kernel initialization overhead for small dicts

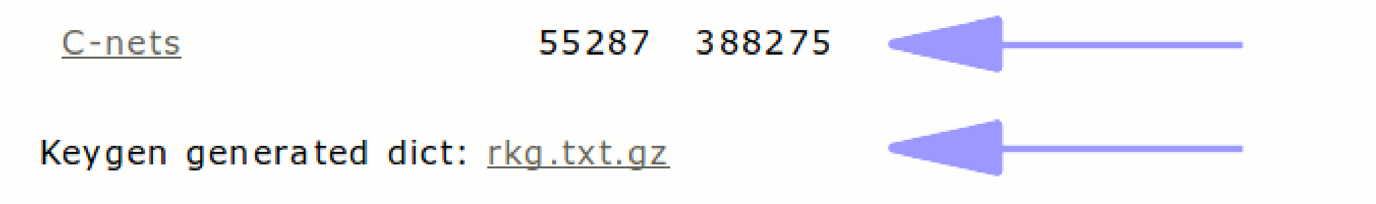


Dictionary	Word count	Hits
hashes.org	189189560	1
Offensive Security	34036913	3
Used	9062908	2818
InsidePro	7788990	21864
Wikipedia en	5925979	24015
Wikipedia de	5429072	24309
Wikipedia ru	2574086	24604
Old gold	1560177	33263
Wikipedia es	1528843	34941
wp chit bg	1318313	35702
Wikipedia fr	1294686	61612
OpenWall	1148496	331615
WPSkey9	1000000	50286
WPSkey8	1000000	64820
WPSkey7	1000000	65563
WPSkey6	1000000	73498
WPSkey5	1000000	82636
WPSkey4	1000000	173031
WPSkey3	1000000	360175
WPSkey2	1000000	363072
WPSkey1	1000000	368968
WPSkey0	1000000	372816
CoW	930799	383353
Slang	510315	385230
Pinyin chinese	61479	386299
C-nets	55280	388270

https://wpa-sec.stanev.org | https://github.com/RealEnder/dwpa

PSK submissions

- Accept one or more PSK by hash or BSSID
- Validated by custom PHP cracker on the backend
- On success, try to find other uncracked handshakes with the same essid/bssid/mac_sta and attack by PMK
- Regenerate *cracked.txt.gz* dict
- Cracked by *RouterKeygenPC* are separated in *rkg.txt.gz*



```
C-nets                55287   388275

Keygen generated dict: rkg.txt.gz
```

What we've learned

- A multitude of BSSID/ESSID based default algos
- Identified keyspace for some default router PSKs
- Confirmed results from reverse engineering efforts to extract default algos
- Hit some linux wifi adapter driver bugs
  https://bugzilla.kernel.org/show_bug.cgi?id=196715
  https://github.com/kaloz/mwlwifi/issues/107
- Identified some optimizations and possible improvements in hashcat and JtR
- wpa-sec is useful as OSINT source for penetration tests

So what’s next

- A lot more default algos are hidden in the DB
- Build online DB for default algos and keyspaces
- Refresh web interface from `90s style
- Introduce API for DB query

  For now, if you have ideas, just drop me a mail
- Prepare for WPA3 (speculations)

  Simultaneous Authentication of Equals (SAE) / Dragonfly

  Negotiates fresh PMK – forward secrecy

  Then good old 4-way handshake

  WPS -> Wi-Fi Device Provisioning Protocol (DPP)

  Mandatory Protected Management Frames (PMF) – no simple deauth

  More @ Mathy Vanhoef's blog:

https://www.mathyvanhoef.com/2018/03/wpa3-technical-details.html

- But... in 2018 we still phase out WEP (7%) and WPA (6%)

Thanks!

Any questions?

https://wpa-sec.stanev.org

https://github.com/RealEnder/dwpa

Alex Stanev
alex@stanev.org
@RealEnderSec

Greetings to
ZeroBeat, atom, magnumripper, Rui Araujo, Bobzilla, Diego and all wpa-sec contrbutors and users

This page intentionally left blank.