Can security be broken by software-defined radio leakage?

Séminaire mensuel du département Information, Communications, Electronique d'IP Paris

Giovanni Camurati
ETH Zurich

05/05/2022

About me

Giovanni Camurati

Postdoc @ETH Zurich with Pr. Srdjan Capkun

PhD @EURECOM with Pr. Aurélien Francillon and Pr. Ludovic Apvrille
MS @PoliTO + Télécom-ParisTech/EURECOM

giovanni.camurati@inf.ethz.ch. https://giocamurati.github.io, @giocamurati

Research interests

Security of Software + Hardware + Radios, e.g.,

- **Screaming Channels**: radio side channels, ACM CCS 2018, IACR TCHES 2020
- **Noise-SDR**: electromagnetic noise modulation, IEEE SP 2022
- **Ghost Peak**: distance reduction attacks, USENIX Security 2022

Firmware analysis, SoC security, hardware design, etc.

About me

Giovanni Camurati

Postdoc @ETH Zurich with Pr. Srdjan Capkun

PhD @EURECOM with Pr. Aurélien Francillon and Pr. Ludovic Apvrille
MS @PoliTO + Télécom-ParisTech/EURECOM

giovanni.camurati@inf.ethz.ch. https://giocamurati.github.io, @giocamurati

Research interests

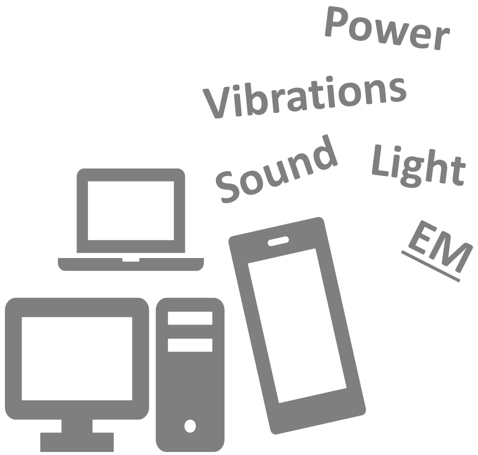
Security of Software + Hardware + Radios, e.g.,

- **Screaming Channels**: radio side channels, ACM CCS 2018, IACR TCHES 2020
- **Noise-SDR**: electromagnetic noise modulation, IEEE SP 2022
- **Ghost Peak**: distance reduction attacks, USENIX Security 2022

Firmware analysis, SoC security, hardware design, etc.

Let’s start with some context

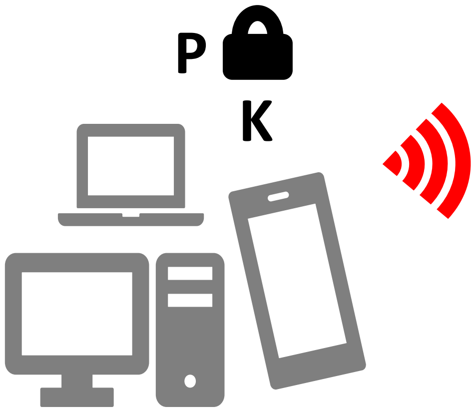
Emission Security

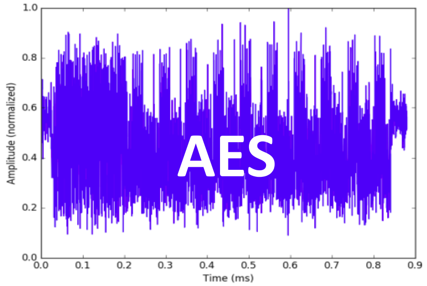


Compromising emanations related to digital activity

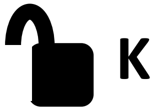
R. J. Anderson, “Security Engineering - a Guide to Building Dependable Distributed Systems” (2. Ed.) (Wiley, 2008).

What attacks are possible? Side Channels





Statistical analysis
Key recover



D. Agrawal et al., “The EM Side-Channel(s),” in CHES 2002.

What attacks are possible? TEMPEST

Recovery of the plaintext signal (e.g., video on the screen)

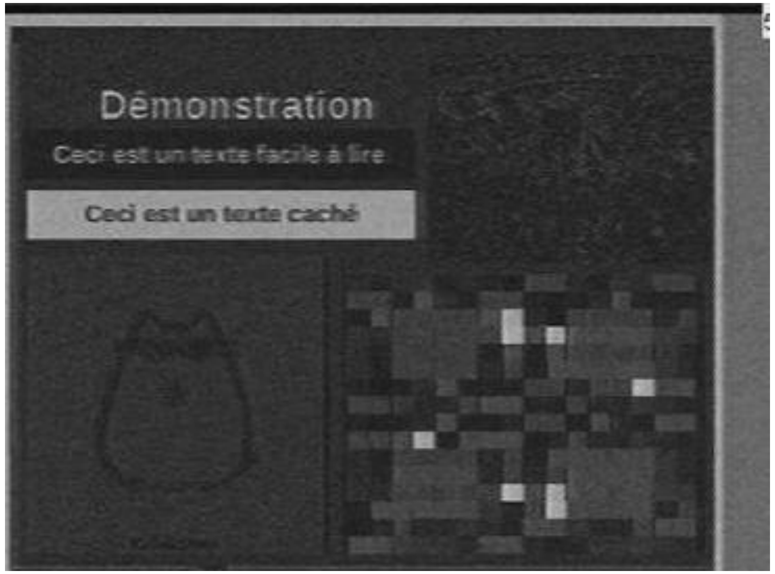
P

"TEMPEST: A Signal Problem" (NSA, 1972).
W. van Eck, "Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk?," Comput. Secur. 4, no. 4 (1985).

What attacks are possible? TEMPEST

Laptop **Cable** **Image on screen**

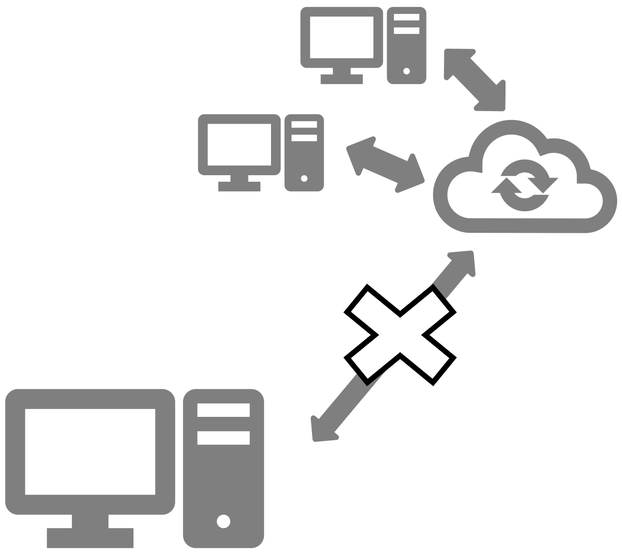
Image recovered by the attacker 10m away



Screenshots from https://static.sstic.org/videos2018/SSTIC_2018-06-13_P05.mp4
Nice demo at minute 3.41 .

Context: Soft-TEMPEST

In theory…
Fully disconnected
Even an attacker able to execute code cannot exfiltrate data

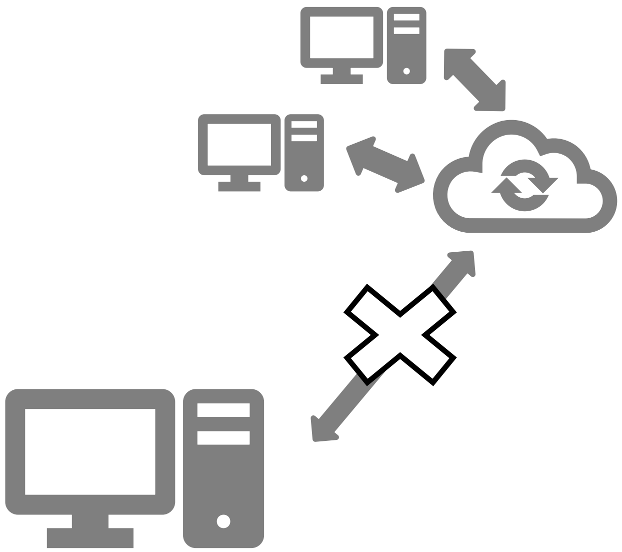


Air-gapped device

Context: Soft-TEMPEST

In theory…
Fully disconnected
Even an attacker able to execute code cannot exfiltrate data

Physical leakage…
Software execution triggers and modulates EM radiation



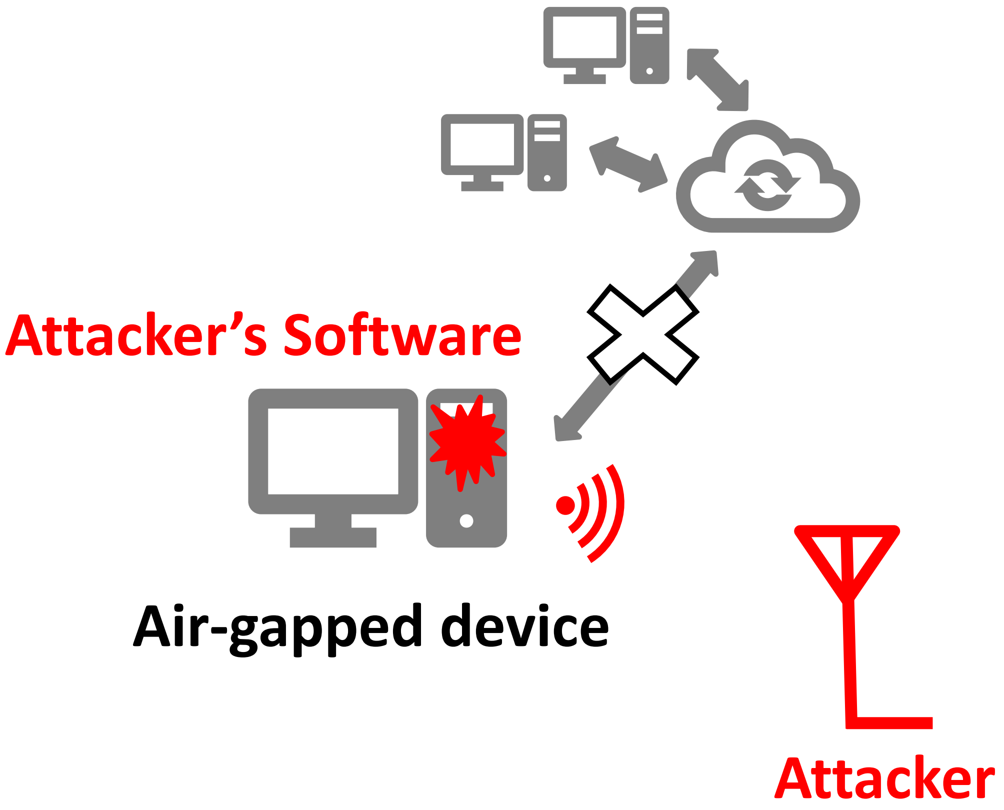
Air-gapped device

Context: Soft-TEMPEST

In theory…
Fully disconnected
Even an attacker able to execute code cannot exfiltrate data

Physical leakage…
Software execution triggers and modulates EM radiation

In practice…
Exfiltrate data via EM radiation
Communication is possible!

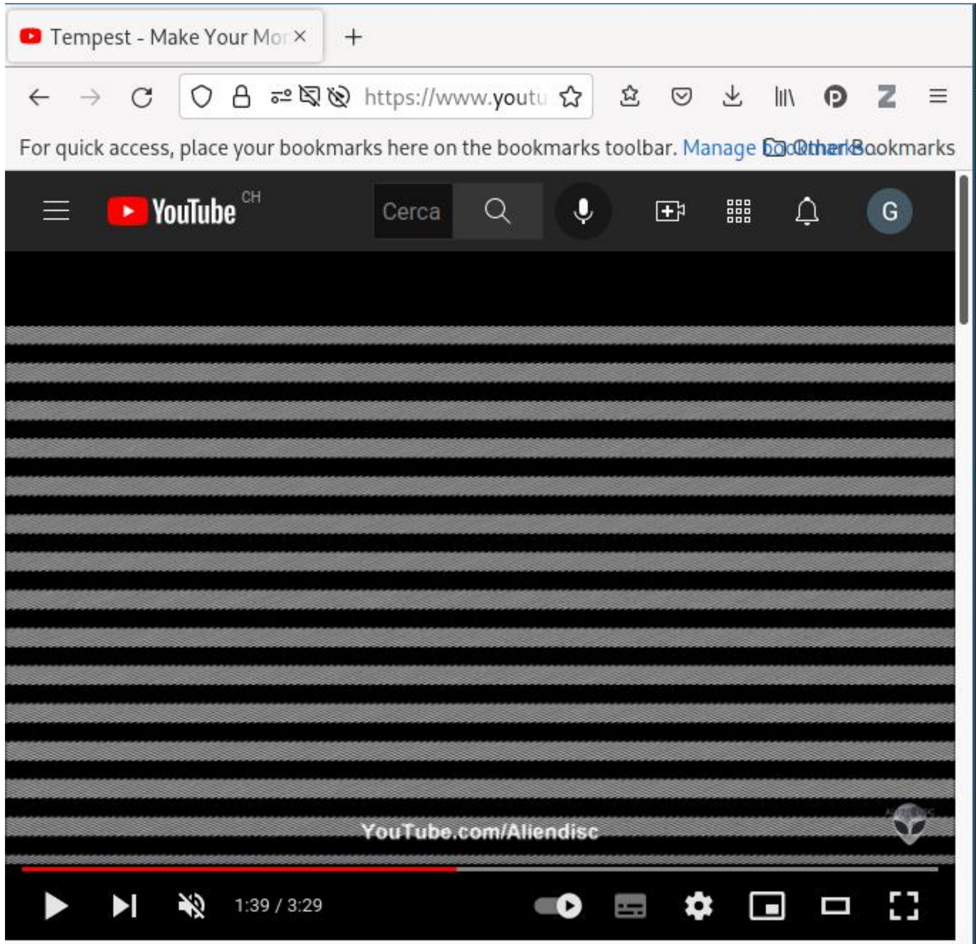


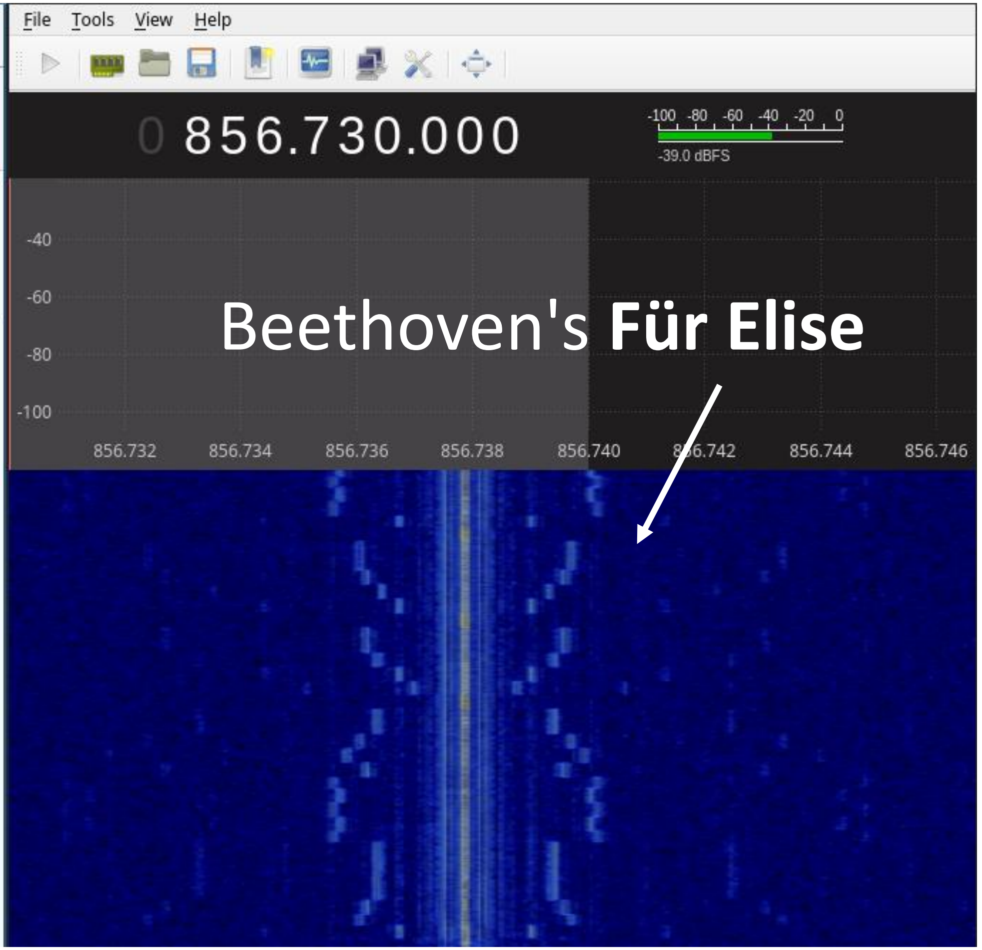
M. G. Kuhn and R. J. Anderson, "Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations," in Information Hiding (1998).

Let’s focus on Soft-TEMPEST

i.e. transmissions using software-controlled leakage

Background: the old classic "Tempest for Elise" example





https://www.youtube.com/watch?v=DlVM9xqGKx8

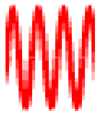
My laptop + HackRF radio

Background: soft-TEMPEST communications 101

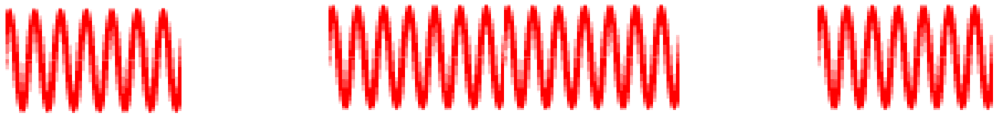
Goal
Modulate a carrier
to transmit data

101101

Find something that
produces EM leakage
e.g., DRAM access

doSomething() 

Modulate it
based on data to transmit



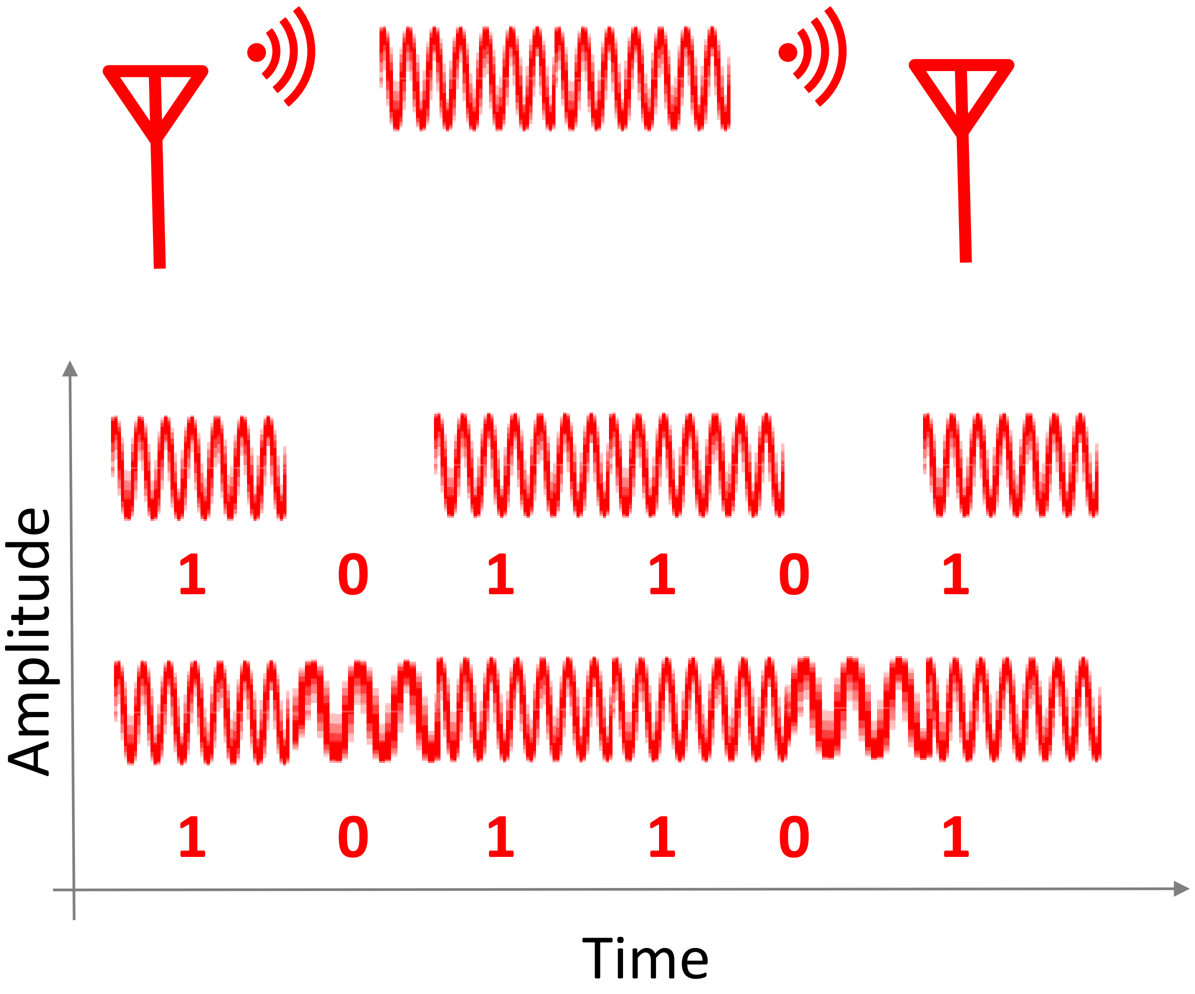
1 0 1 1 0 1

Background: modulation 101

Carrier
Sinusoidal wave at radio frequency

OOK
On-Off Keying

FSK
Frequency-Shift Keying



Background: general primitive in related work

```
start = now()
while( now() – start < T/2 )
        doSomething()
while( now() – start < T )
        doNothing()
```

*M. Guri et al., “GSMem: Data Exfiltration from Air-Gapped Computers over GSM Frequencies,” in USENIX Security 2015.

*Z. Zhan, Z. Zhang, and X. Koutsoukos, “BitJabber: The World’s Fastest Electromagnetic Covert Channel,” in IEEE ITC 2010

**C. Shen et al., “When LoRa Meets EMR: Electromagnetic Covert Channels Can Be Super Resilient”, IEEE S&P 2021

**W. Entriken, System Bus Radio, 2013, https://github.com/fulldecent/system-bus-radio.

Background: general primitive in related work

```
start = now()
while( now() – start < T/2 )
        doSomething()
while( now() – start < T )
        doNothing()
```

Trigger leakage @$F_{leakage}$ from SW
E.g., with memory accesses*

*M. Guri et al., "GSMem: Data Exfiltration from Air-Gapped Computers over GSM Frequencies," in USENIX Security 2015.

*Z. Zhan, Z. Zhang, and X. Koutsoukos, "BitJabber: The World's Fastest Electromagnetic Covert Channel," in IEEE ITC 2010

**C. Shen et al., "When LoRa Meets EMR: Electromagnetic Covert Channels Can Be Super Resilient", IEEE S&P 2021
**W. Entriken, System Bus Radio, 2013, https://github.com/fulldecent/system-bus-radio.

Background: general primitive in related work

```
start = now()
while( now() – start < T/2 )
        doSomething()
while( now() – start < T )
        doNothing()
```

"Square wave"@f=1/T
E.g., sys-bus-radio**

Trigger leakage @$F_{leakage}$ from SW
E.g., with memory accesses*

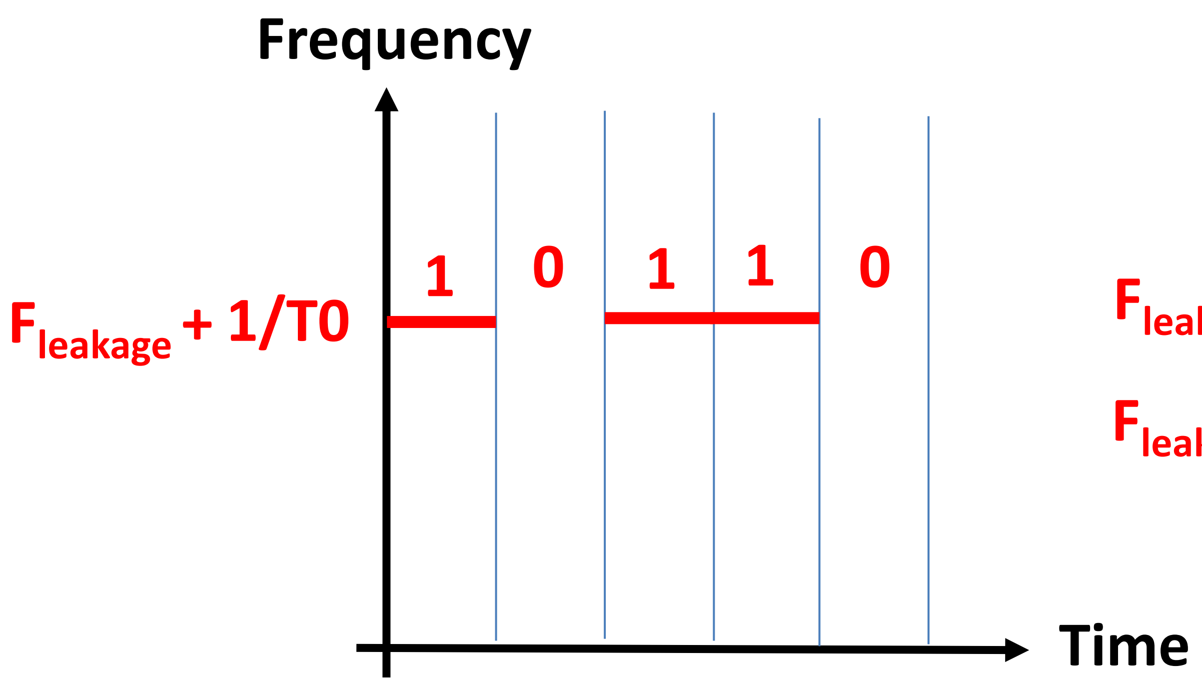
*M. Guri et al., "GSMem: Data Exfiltration from Air-Gapped Computers over GSM Frequencies," in USENIX Security 2015.

*Z. Zhan, Z. Zhang, and X. Koutsoukos, "BitJabber: The World's Fastest Electromagnetic Covert Channel," in IEEE ITC 2010

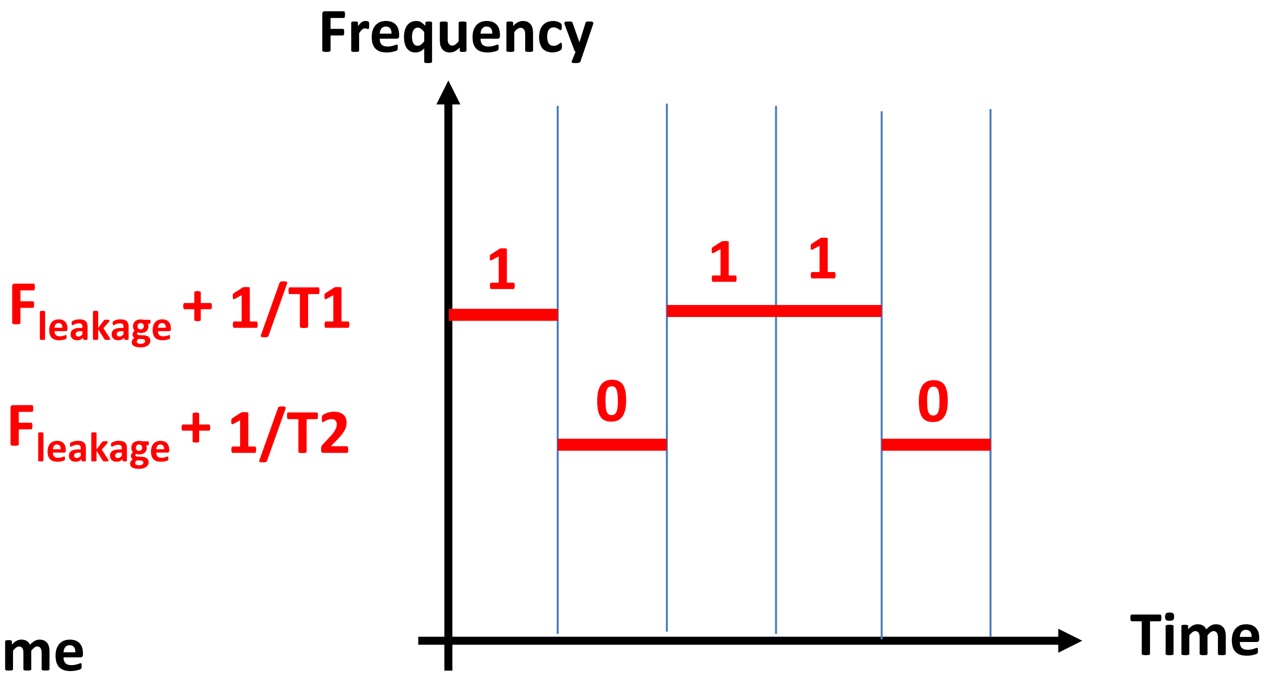
**C. Shen et al., "When LoRa Meets EMR: Electromagnetic Covert Channels Can Be Super Resilient", IEEE S&P 2021

**W. Entriken, System Bus Radio, 2013, https://github.com/fulldecent/system-bus-radio.

Background: general primitive in related work



OOK



2-FSK (or M-FSK)

Related work (EM)

Simple custom modulation/protocol

Name	Leakage Type	Modulation Type	Publication Venue
Soft-TEMPEST	Electromagnetic	AM, FSK	Information Hiding 1998
AirHopper	Elecromagnetic	FSK	MALWARE 2014
USBee	Elecromagnetic	FSK	PST 2016
GSMem	Elecromagnetic	OOK	USENIX Security 2015
BitJabber	Elecromagnetic	OOK, FSK	IEEE ITC 2020
MAGNETO	Magnetic	OOK, FSK	ArXiv 2018
ODINI	Magnetic	OOK-(many cores), FSK	IEEE Trans. Inf. Forensics Secur. 2020
Matyunin et. al	Magnetic	OOK, FSK	ASP-DAC 2016
EMLora	Electromagnetic	CSS	IEEE S&P 2021

A first step towards more advanced modulation

Limitations of previous work

Simple custom modulation
Mostly OOK or FSK, often requires custom receivers

Simple custom protocol
No error correction, etc.

Not flexible
Only one fixed modulation

Single application
Exfiltration from air-gapped devices

Meanwhile the real radios...

Software-defined

Signals entirely defined in software

Minimal hardware to create the actual waves

Arbitrary modulation

Can shape generic signals

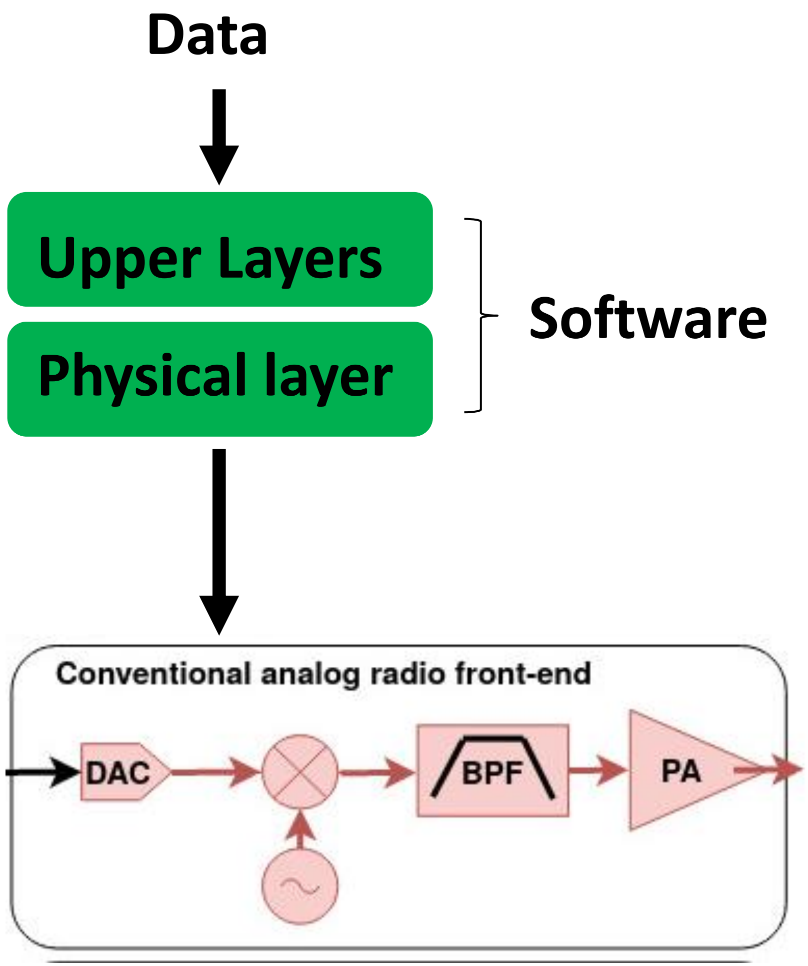
Advanced modulation techniques possible

Advanced protocols

E.g., error correction

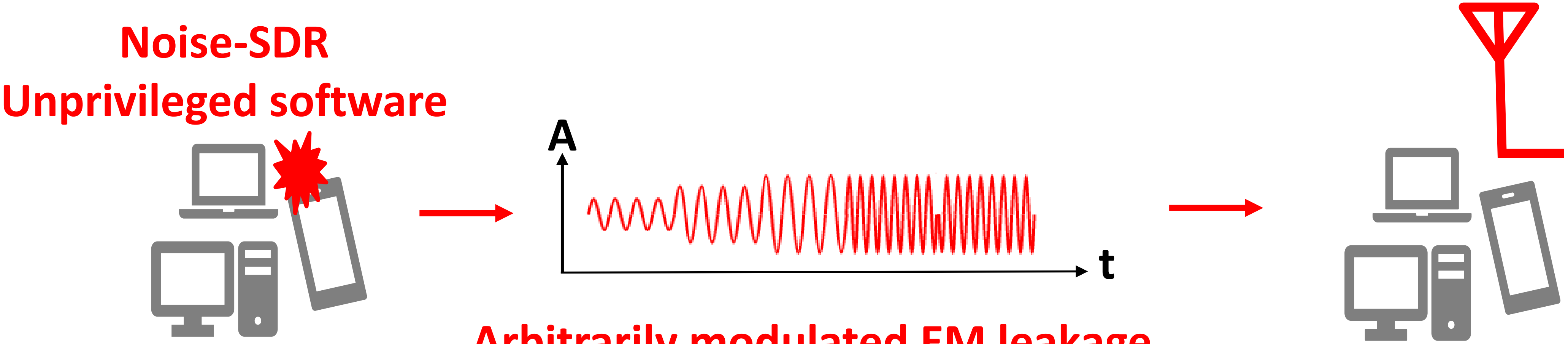
Flexible

Can handle virtually any protocol / application



Can we make Soft-TEMPEST more similar to a real software-defined radio and give way more performance and flexibility to the attacker?

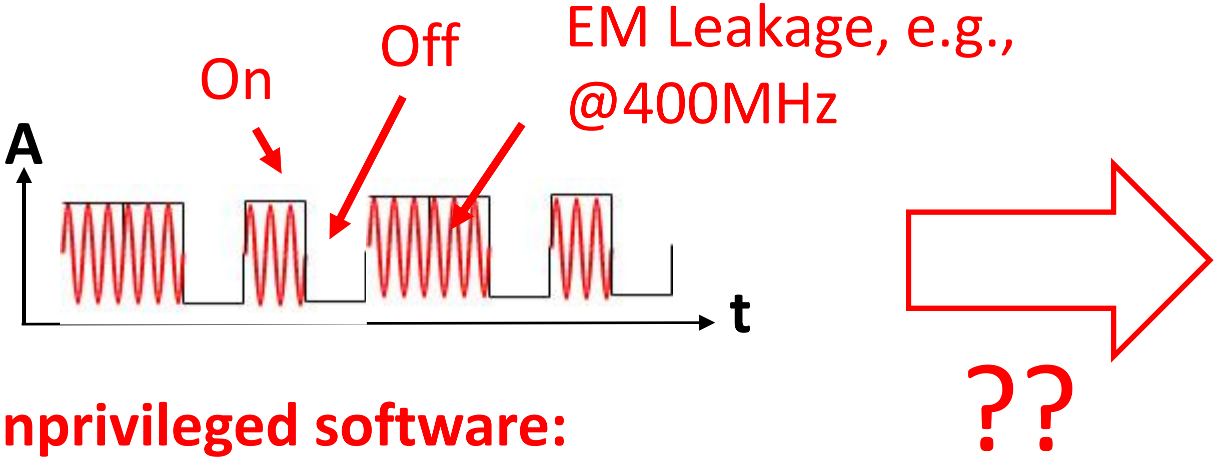
Goal: Can we do more?

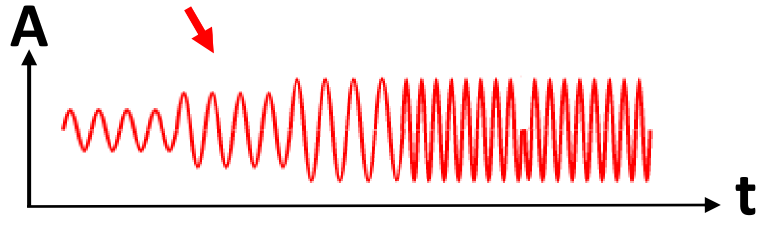


Arbitrarily modulated EM leakage

+ Software-defined: flexibility, existing protocols
+ Advanced PHY layer: performance
+ More applications: exfiltration, tracking, injection, …

Challenge: from square wave to generic passband signal



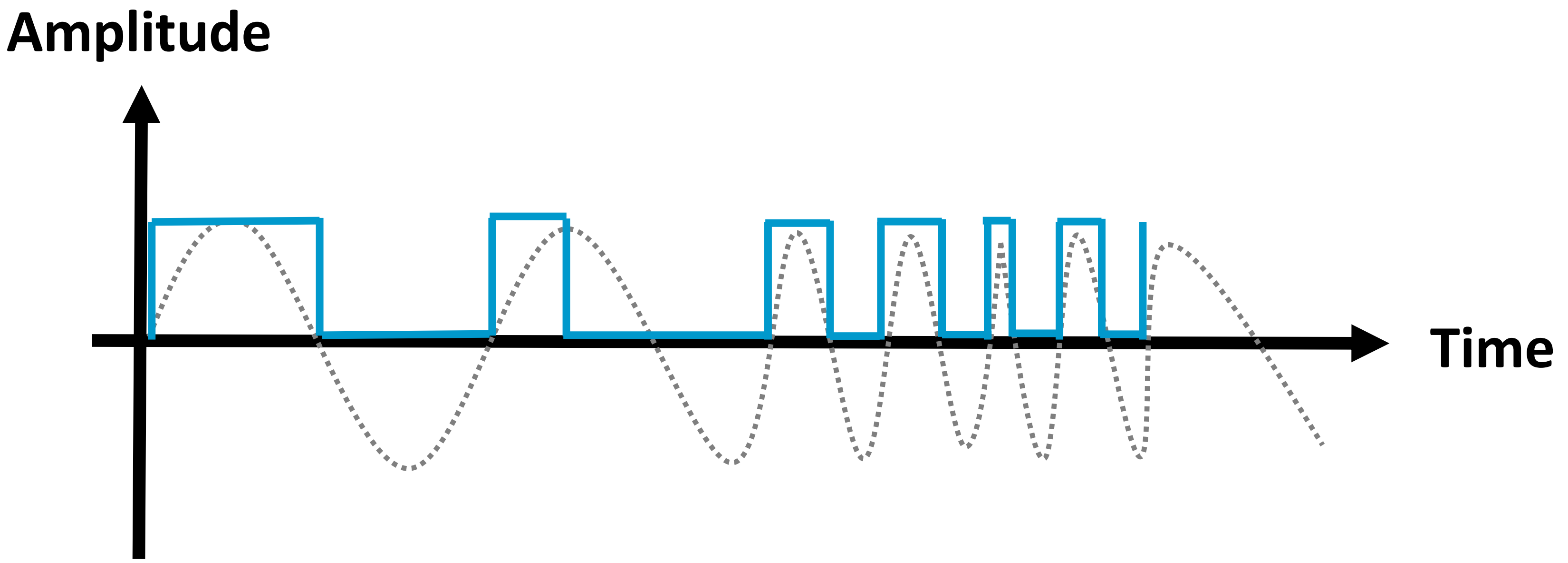


Unprivileged software:

1. **DRAM access:** "EM leakage ON"
2. **Do nothing:** "EM leakage OFF"

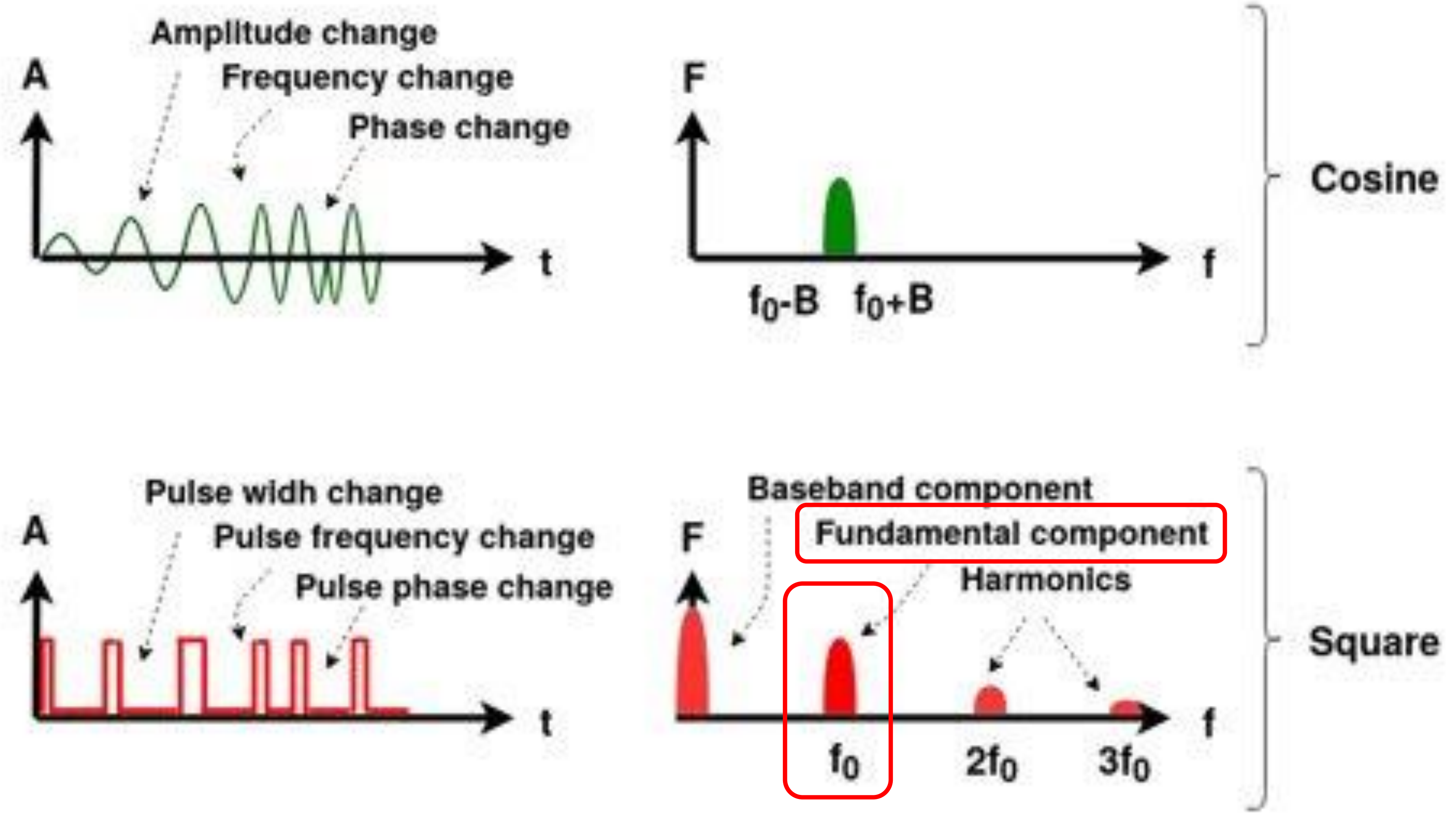
Solution: leverage pass-band one-bit coding (RF-PWM)

Long story short: approximate a modulated sine-wave with a square wave

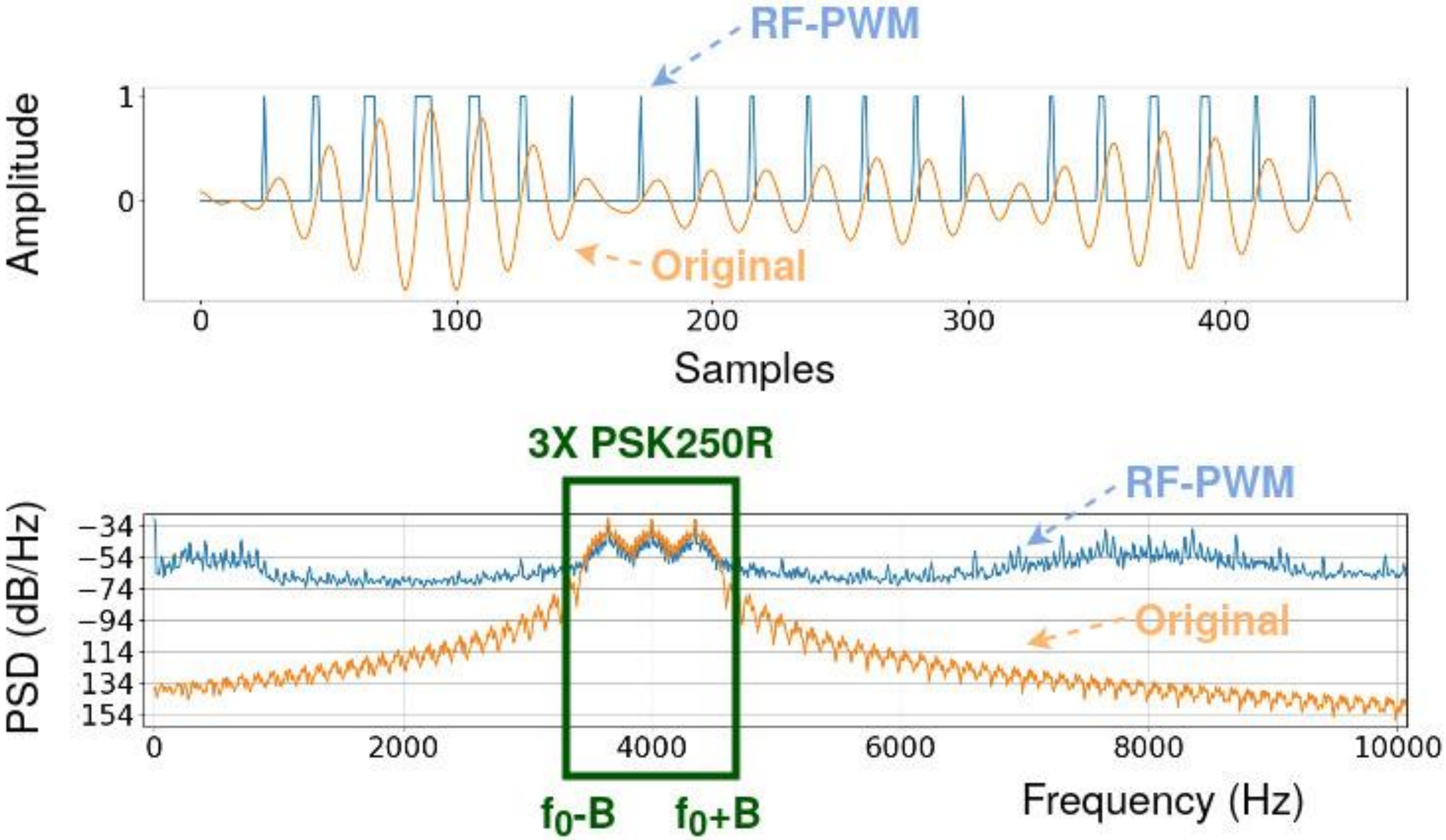


P. AJ Nuyts, P. Reynaert, and W. Dehaene, “Continuous-Time Digital Front-Ends for Multistandard Wireless Transmission” (Springer, 2014).

Background: fundamental of a modulated square wave



Example: good approximation in the band of interest



Noise-SDR: Arbitrary Modulation of EM noise

Data

f

DRAM

Digital Hardware (DRAM)

$F_{leakage}$

Noise-SDR: Arbitrary Modulation of EM noise

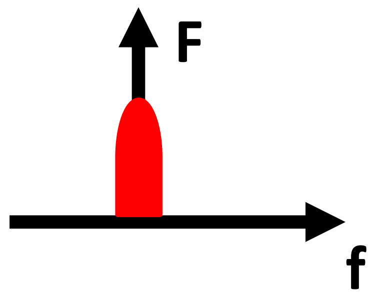
Data

Upper Layers

Physical layer

Software (existing SDR tools)

$\mathbf{a}(\mathbf{t}), \theta(t)$



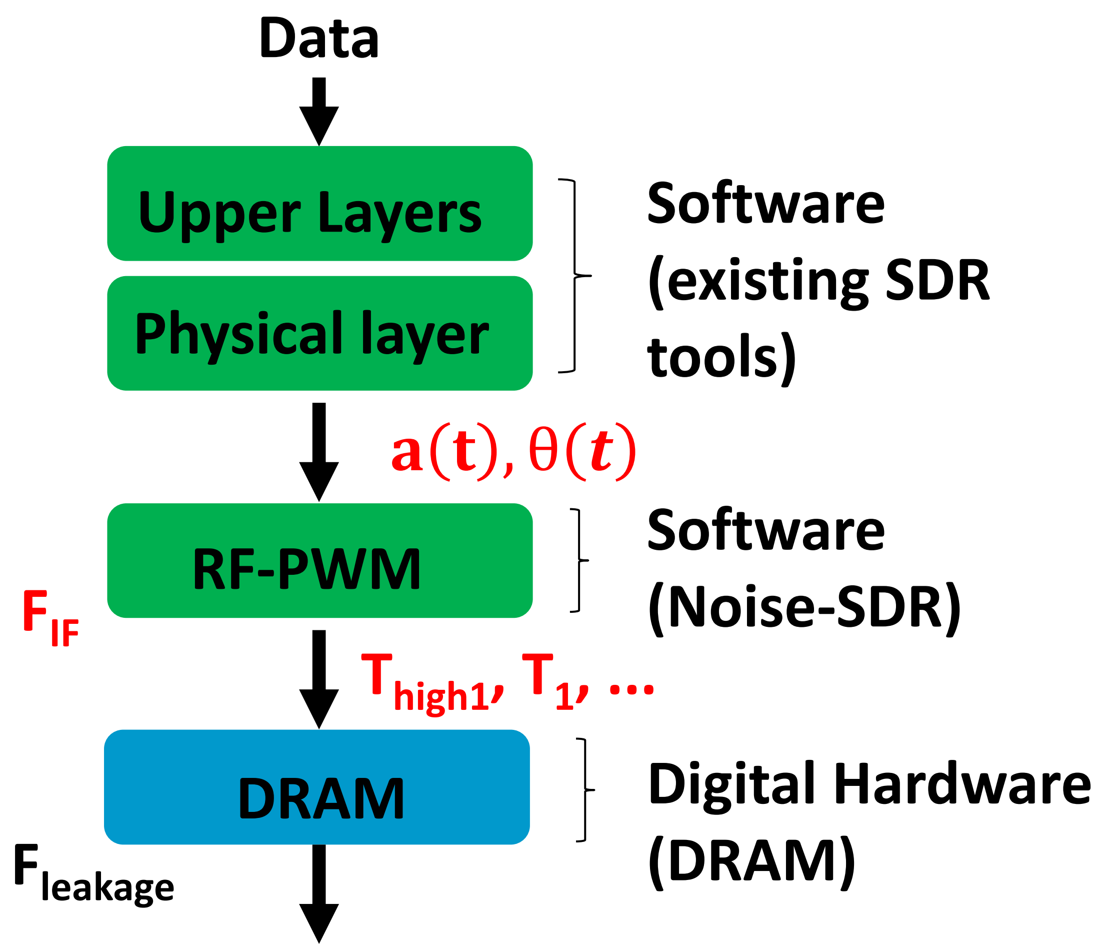
f

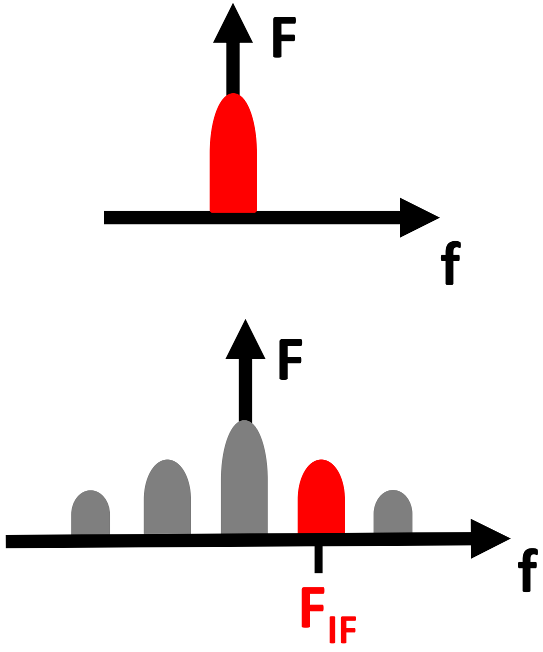
DRAM

Digital Hardware (DRAM)

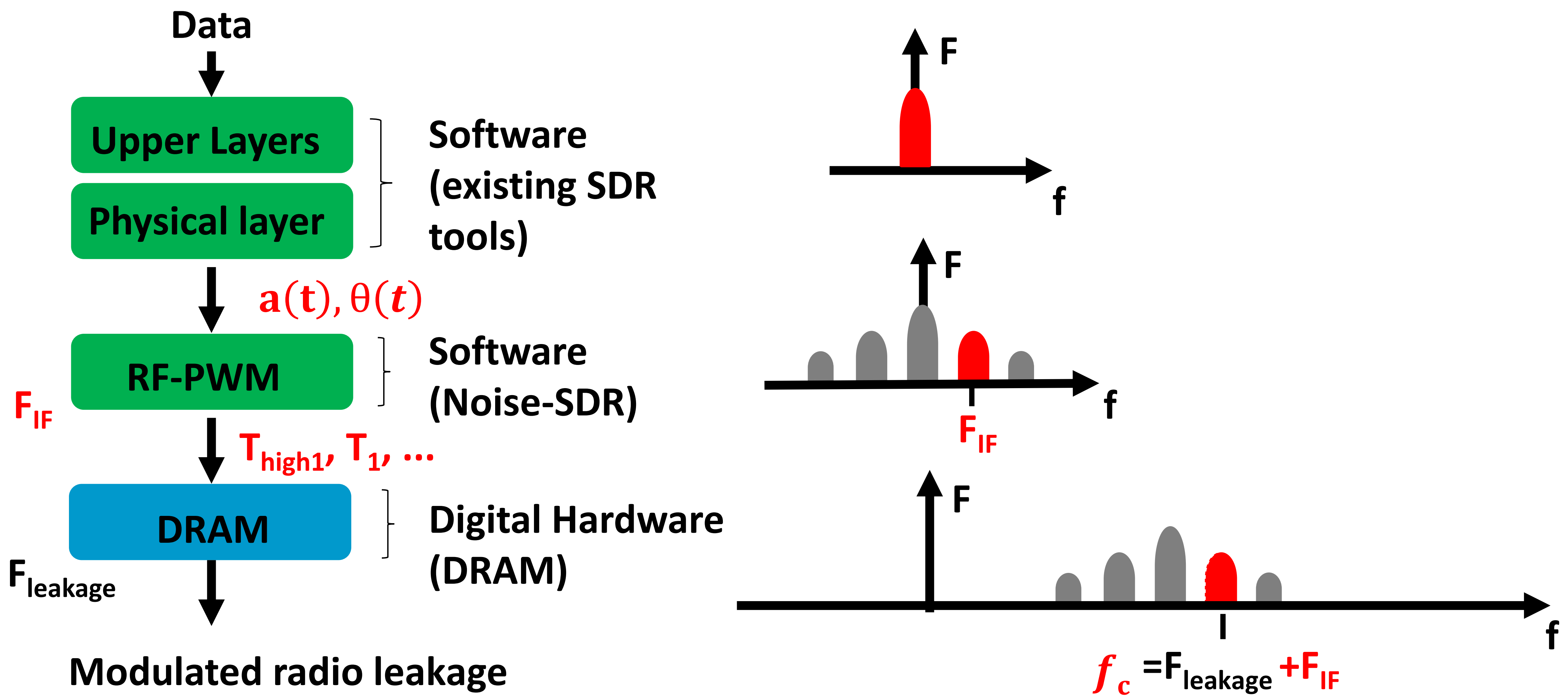
$\mathbf{F}_{\mathbf{leakage}}$

Noise-SDR: Arbitrary Modulation of EM noise

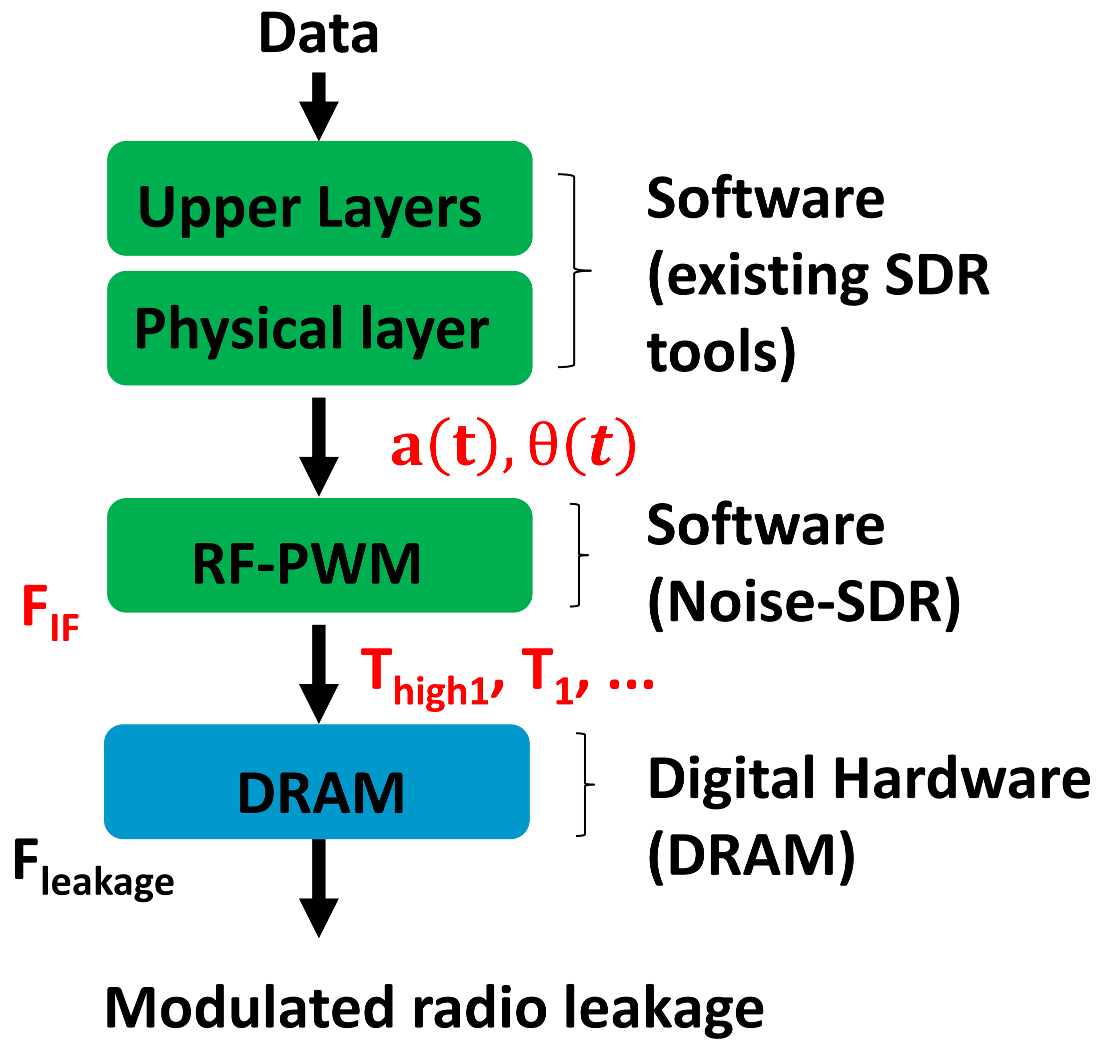


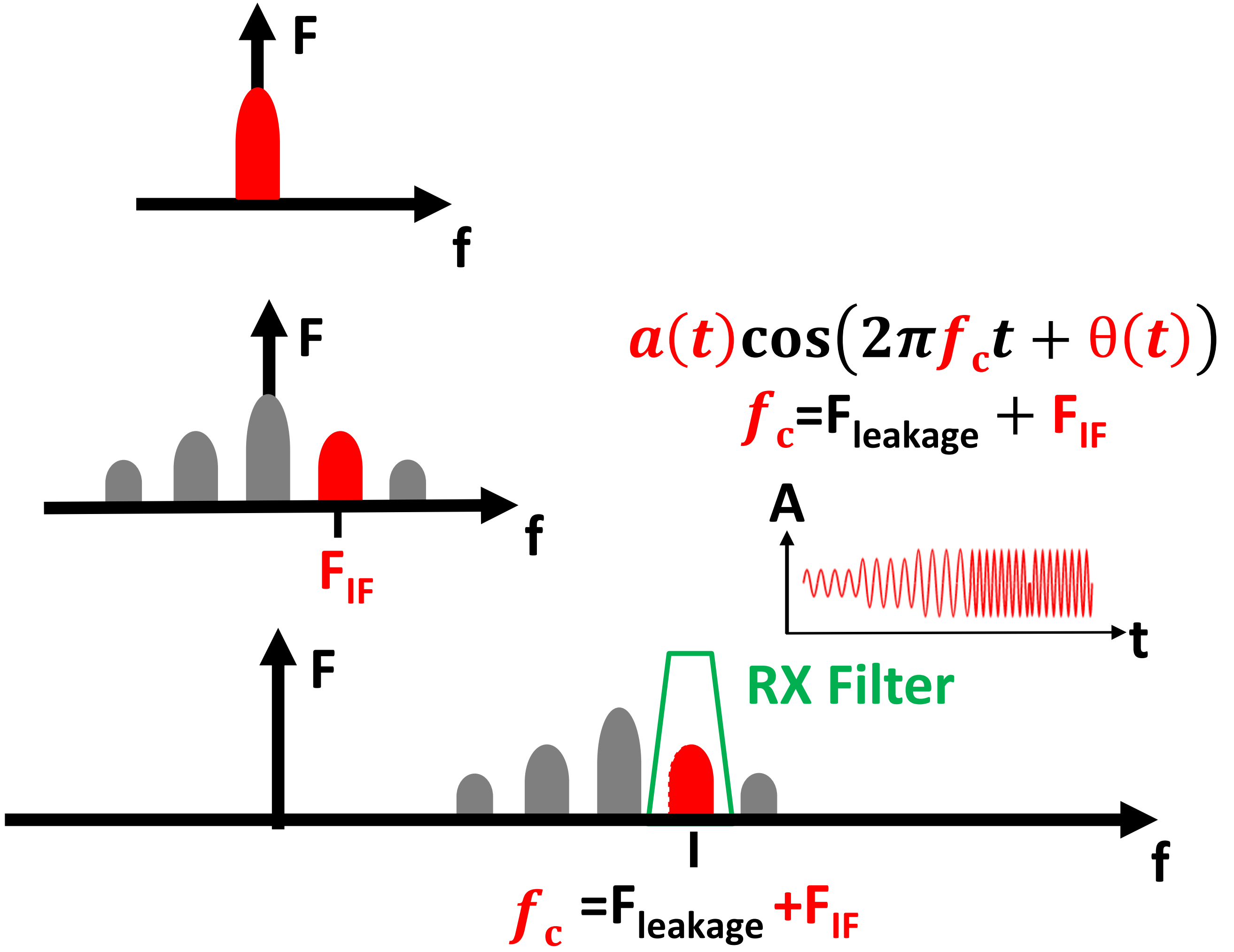


Noise-SDR: Arbitrary Modulation of EM noise



Noise-SDR: Arbitrary Modulation of EM noise



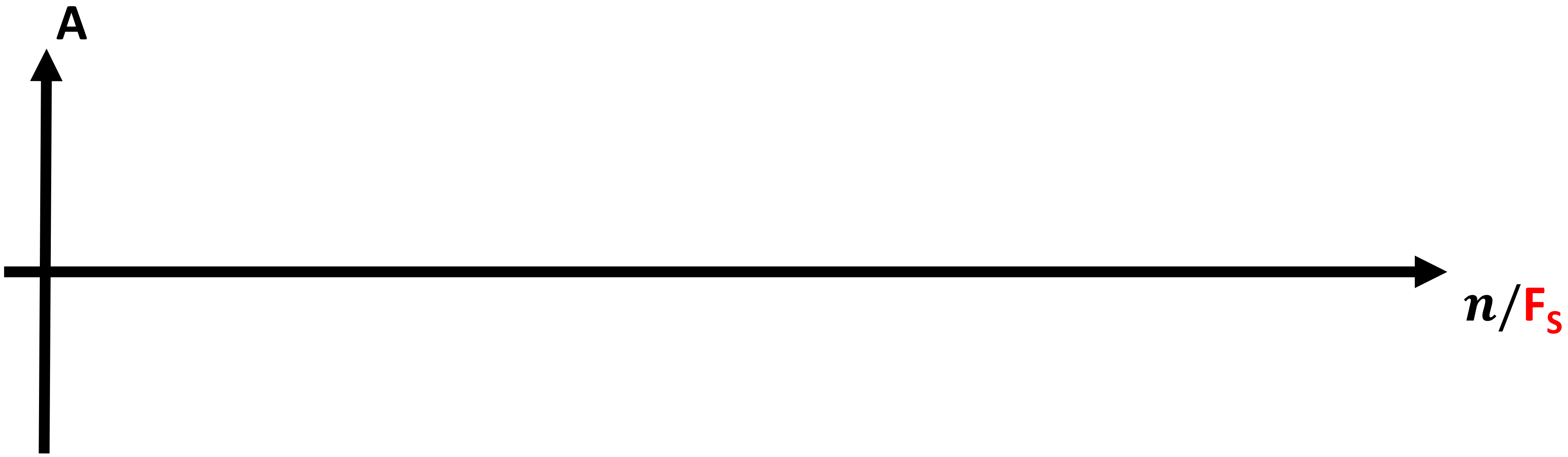


How do we implement it in practice?

Implementation: discrete-time RF-PWM

Simplified explanation

Input: F_S, $a(n/F_S)$, $\theta(n/F_S)$, F_{IF}

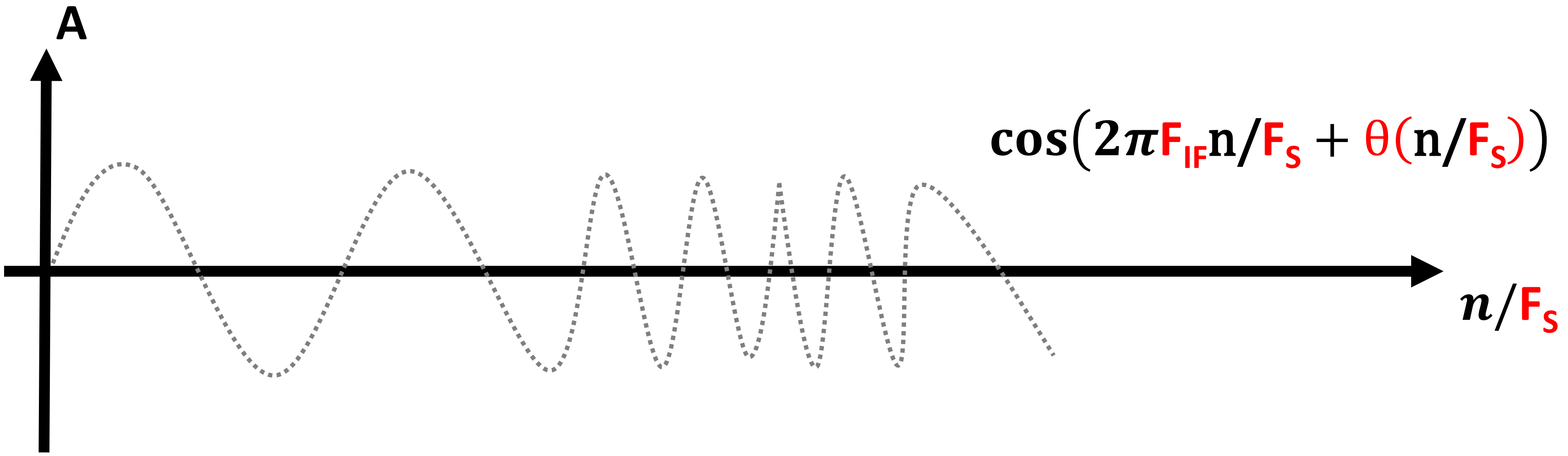


Output:

Implementation: discrete-time RF-PWM

Simplified explanation

Input: F_S, $a(n/F_S)$, $\theta(n/F_S)$, F_{IF}



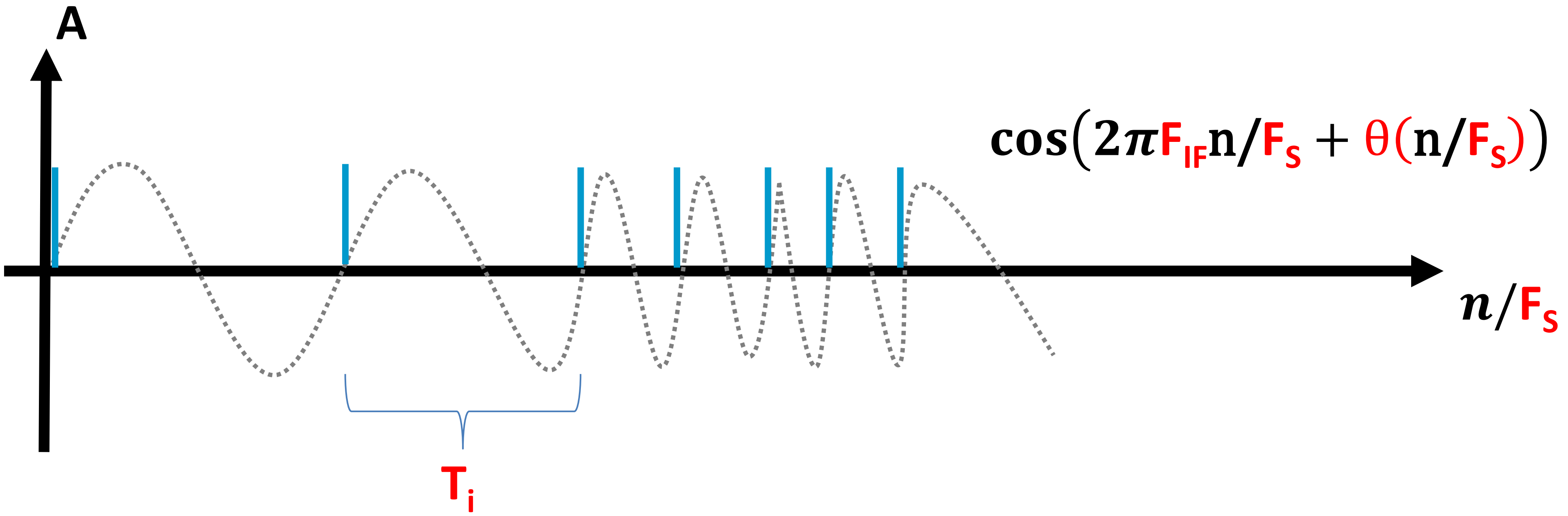
$$\cos\left(2\pi F_{IF} n/F_S + \theta(n/F_S)\right)$$

Output:

Implementation: discrete-time RF-PWM

Simplified explanation

Input: F_S, $a(n/F_S)$, $\theta(n/F_S)$, F_{IF}

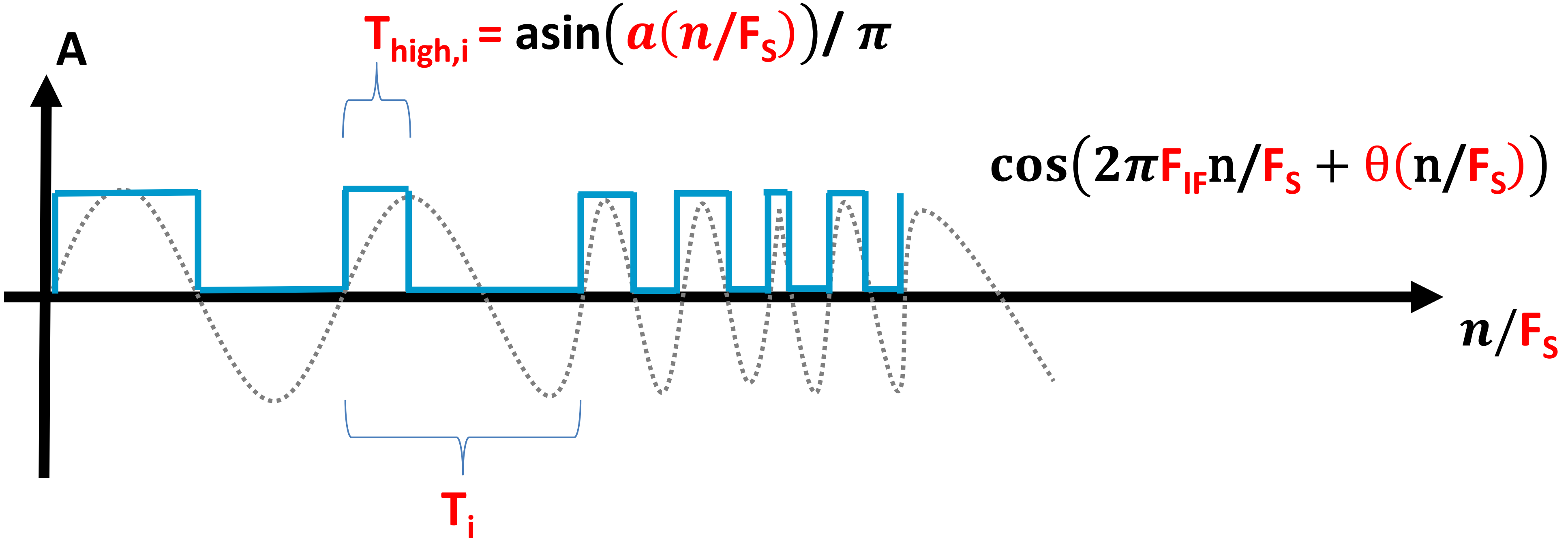


Output: T_1, T_2, T_3, …

Implementation: discrete-time RF-PWM

Simplified explanation

Input: F_S, $a(n/F_S)$, $\theta(n/F_S)$, F_{IF}



Output: $T_{high,1}$ T_1, $T_{high,2}$ T_2, $T_{high,3}$ T_3, …

Implementation: software-control

-: Time accuracy is fundamental!**
(Bandwidth, am/fm/pm quantization)

```
start = now()
while( now() – start < T_high,i )
        leakyOperation()
while( now() – start < T_i )
        doNothing()
```

*M. Schwarz et al., "Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript," in FC 2017.
**Z. Zhang et al., "Leveraging EM Side-Channel Information to Detect Rowhammer Attacks," in IEEE S&P 2020
***Z. Zhang et al., "Triggering Rowhammer Hardware Faults on ARM: A Revisit," ASHES@CCS 2018.

Implementation: software-control

[*-*]: Time accuracy is fundamental!**
(Bandwidth, am/fm/pm quantization)

```
start = now()
while( now() – start < T_high,i )
        leakyOperation()
while( now() – start < T_i )
        doNothing()
```

Leaky[**], fast[***]

Accurate[*], stable

*M. Schwarz et al., “Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript,” in FC 2017.
**Z. Zhang et al., “Leveraging EM Side-Channel Information to Detect Rowhammer Attacks,” in IEEE S&P 2020
***Z. Zhang et al., “Triggering Rowhammer Hardware Faults on ARM: A Revisit,” ASHES@CCS 2018.

Implementation: software-control

-: Time accuracy is fundamental!**
(Bandwidth, am/fm/pm quantization)

```
start = now()
while( now() – start < T_high,i )
        leakyOperation()
while( now() – start < T_i )
        doNothing()
```

Leaky, fast*****

Accurate*, stable
clock_gettime()
(or µ-arch attacks literature)

*M. Schwarz et al., “Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript,” in FC 2017.
**Z. Zhang et al., “Leveraging EM Side-Channel Information to Detect Rowhammer Attacks,” in IEEE S&P 2020
***Z. Zhang et al., “Triggering Rowhammer Hardware Faults on ARM: A Revisit,” ASHES@CCS 2018.

Implementation: software-control

```
start = now()
while( now() – start < T_high,i )
        leakyOperation()
while( now() – start < T_i )
        doNothing()
```

-: Time accuracy is fundamental!**
(Bandwidth, am/fm/pm quantization)

Leaky, fast*****
Many in the paper and in general

Accurate*, stable
clock_gettime()
(or µ-arch attacks literature)

*M. Schwarz et al., "Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript," in FC 2017.
**Z. Zhang et al., "Leveraging EM Side-Channel Information to Detect Rowhammer Attacks," in IEEE S&P 2020
***Z. Zhang et al., "Triggering Rowhammer Hardware Faults on ARM: A Revisit," ASHES@CCS 2018.

Implementation: software-control

start = now()
while(now() – start < $T_{high,i}$ *)*
leakyOperation()
while(now() – start < T_i *)*
doNothing()

Accurate*, stable
clock_gettime()
(or µ-arch attacks literature)

-: Time accuracy is fundamental!**
(Bandwidth, am/fm/pm quantization)

Leaky, fast*****
Many in the paper and in general
E.g., on Arm-v8 (re)use ROWHAMMER

```
__attribute__((naked)) \
void hammer_civac(uint64_t *addr) {
  __asm volatile("LDR X9, [X0]");
  __asm volatile("DC CIVAC, X0");
  __asm volatile("DSB 0xB");
  __asm volatile("RET");
}
```

*M. Schwarz et al., “Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript,” in FC 2017.
**Z. Zhang et al., “Leveraging EM Side-Channel Information to Detect Rowhammer Attacks,” in IEEE S&P 2020
***Z. Zhang et al., “Triggering Rowhammer Hardware Faults on ARM: A Revisit,” ASHES@CCS 2018.

Implementation: software-control, several architectures

```
__attribute__((naked)) \
void hammer_civac(uint64_t *addr) {
  __asm volatile("LDR X9, [X0]");
  __asm volatile("DC CIVAC, X0");
  __asm volatile("DSB 0xB");
  __asm volatile("RET");
}
```

Listing 4. *leakyOperation* for *ARMv8-A* native code (inspired from [67]).

```
cnt++; // Followed by sleep during the low period
```

Listing 5. *leakyOperation* for *MIPS32* native code (inspired from [44]).

```
void stream(void) {
  _mm_stream_si128(&reg, reg_one);
  _mm_stream_si128(&reg, reg_zero);
}
```

Listing 2. *leakyOperation* for *x86-64* native code (inspired from [8], [44]).

```
static inline void ion_leak(void) {
  ion_user_handle_t ion_handle;
  ion_alloc(ion_fd
      , len, 0, (0x1 << chipset), 0, &ion_handle);
  ion_free(ion_fd, ion_handle);
}
```

Listing 3. *leakyOperation* for *ARMv7-A/* (or *ARMv8-A*) native code (inspired from [66]).

Implementation: combine Noise-SDR with popular SDR tools

Fldigi-Noise-SDR

```
> ./fldigi-noise
    -sdr -i secret.txt -m MODE_3X_PSK250R -c 4000
```

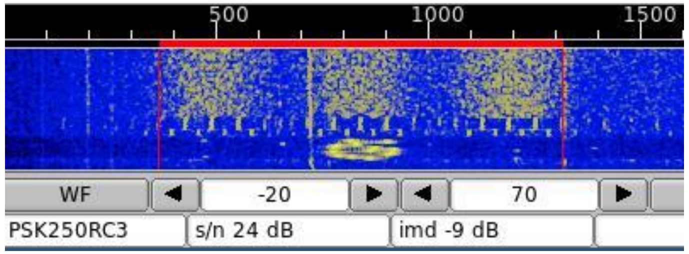
Or **Gnuradio+Offline-Noise-SDR**

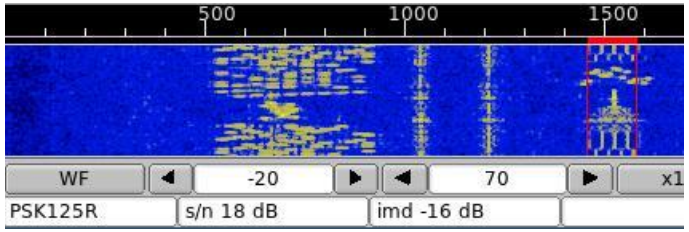
```
> ./offline-noise-sdr ft4.iq
```

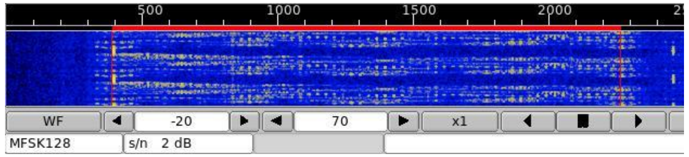
Evaluation

Noise-SDR in action: a few examples

More videos: https://github.com/eurecom-s3/noise-sdr

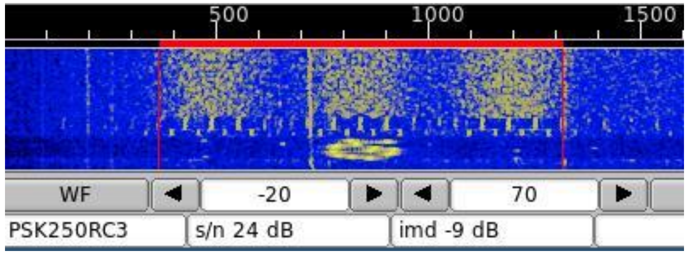


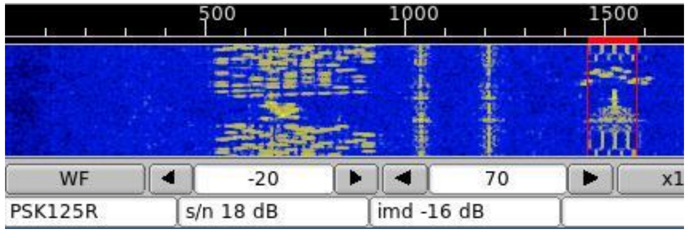


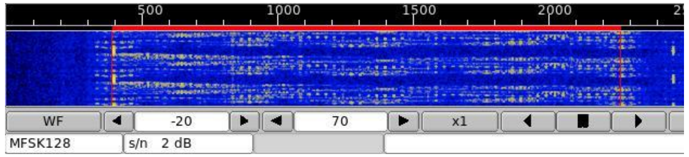


Noise-SDR in action: a few examples

More videos: https://github.com/eurecom-s3/noise-sdr





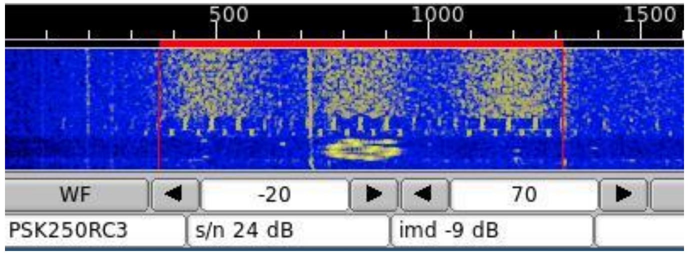


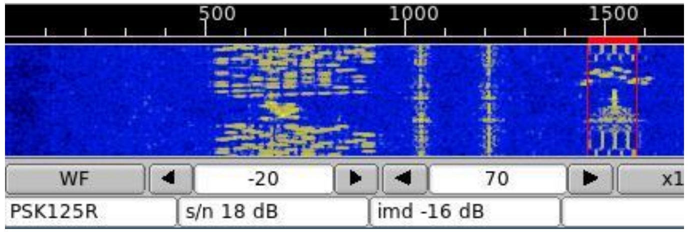
Pros

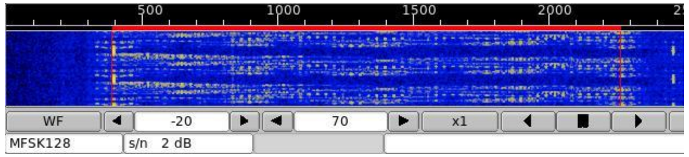
1. Software-defined, flexible
2. AM, FM, PSK, RTTY, THOR, SSTV, LoRa, GLONASS C/A, etc.
3. ArmV7A, ArmV8A, x86, MIPS
4. Mobile/desktop/laptop/IoT

Noise-SDR in action: a few examples

More videos: https://github.com/eurecom-s3/noise-sdr







Pros

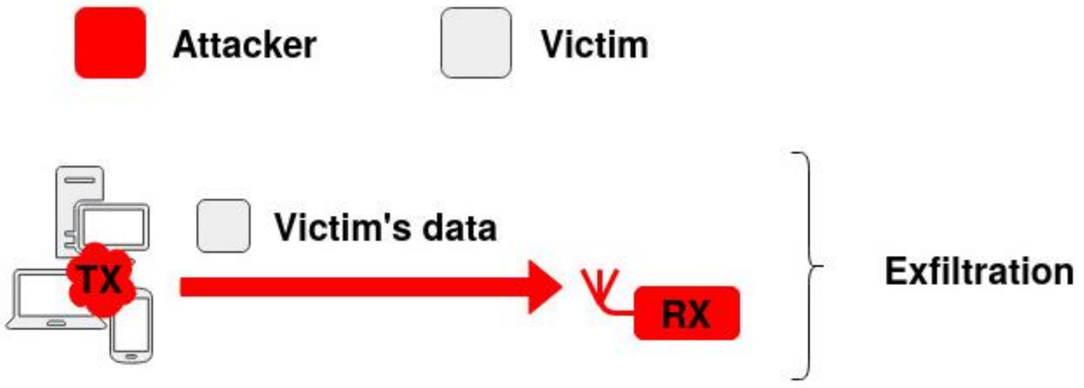
1. Software-defined, flexible
2. AM, FM, PSK, RTTY, THOR, SSTV, LoRa, GLONASS C/A, etc.
3. ArmV7A, ArmV8A, x86, MIPS
4. Mobile/desktop/laptop/IoT

Limitations

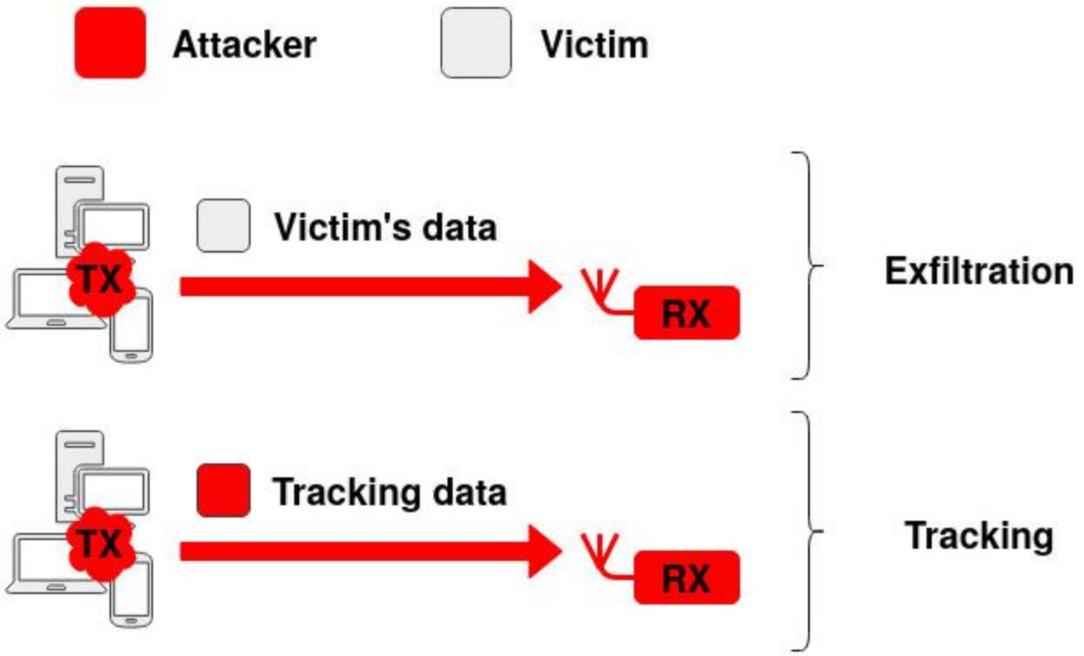
1. Limited bandwidth
2. Limited choice of carrier frequency

Security Impact

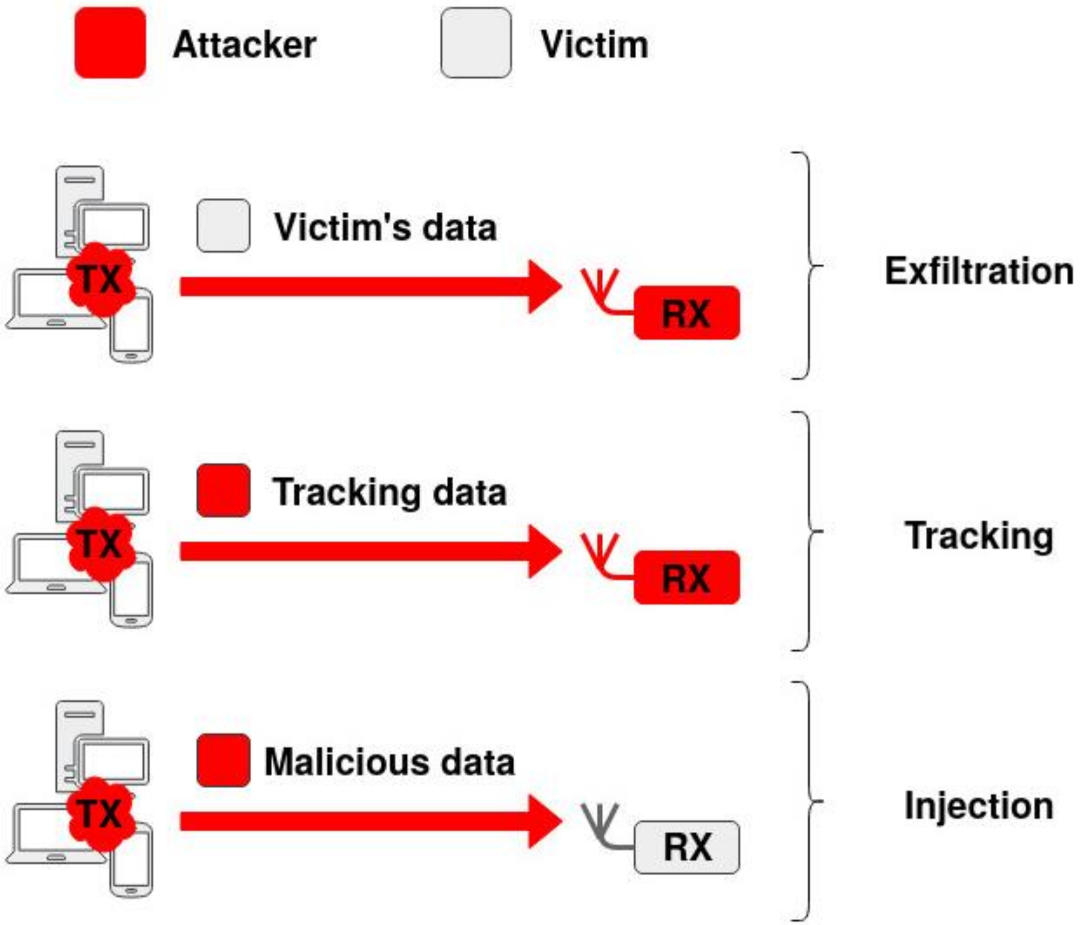
Security impact: exfiltration, tracking, injection



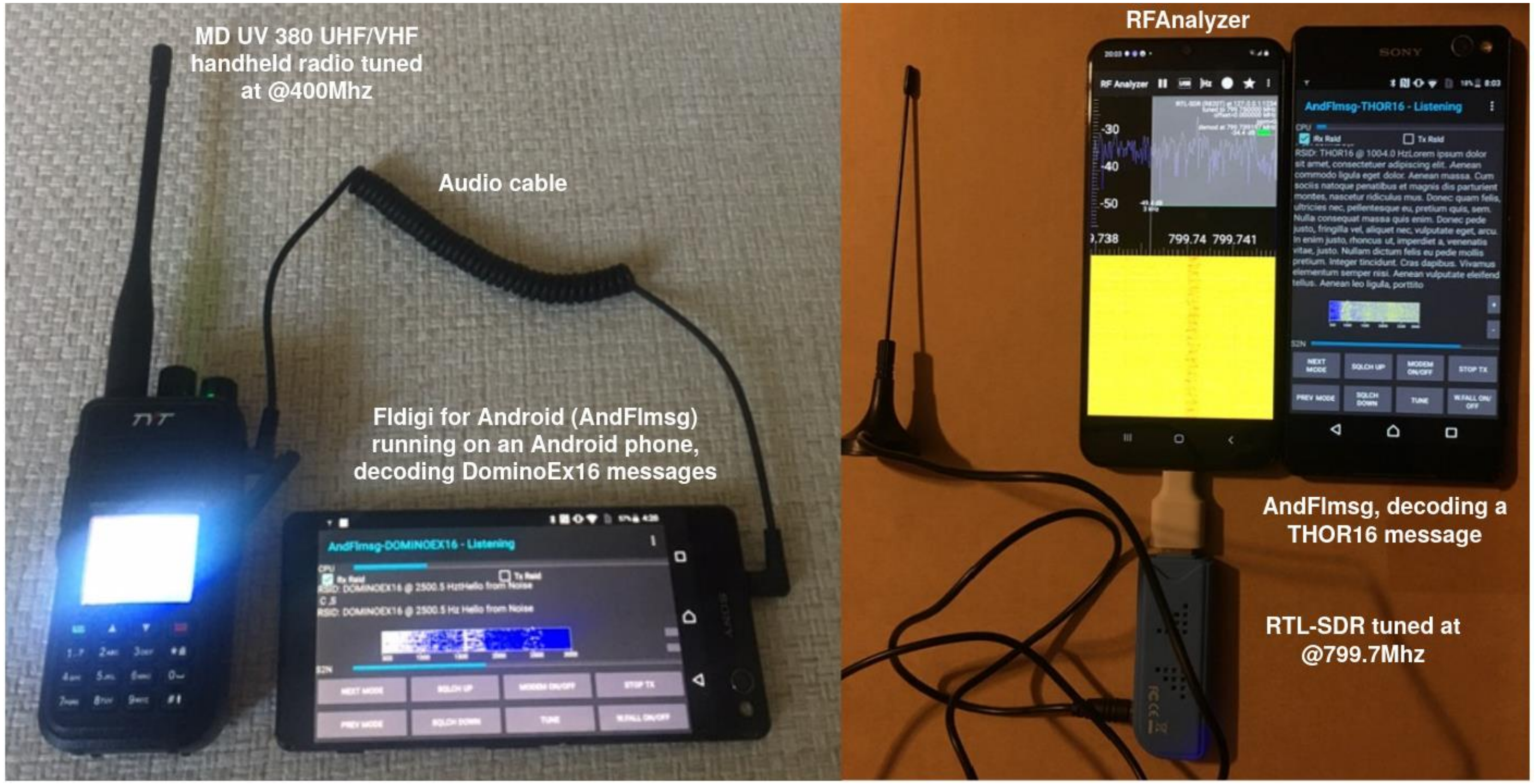
Security impact: exfiltration, tracking, injection

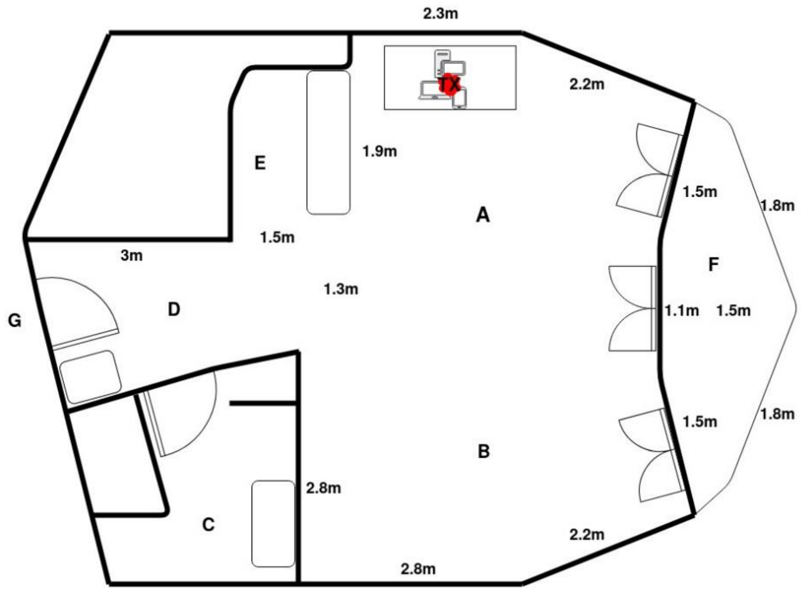


Security impact: exfiltration, tracking, injection



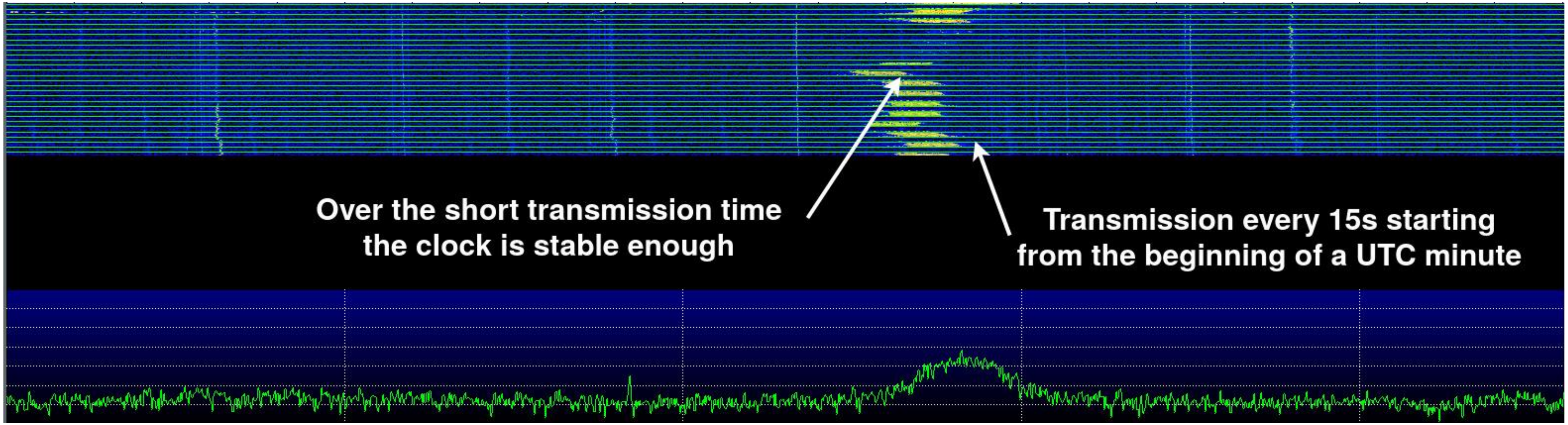
Examples of Exfiltration





Good results in particular with THOR and RSIDs (e.g., MIPS32 >5m beind wall)

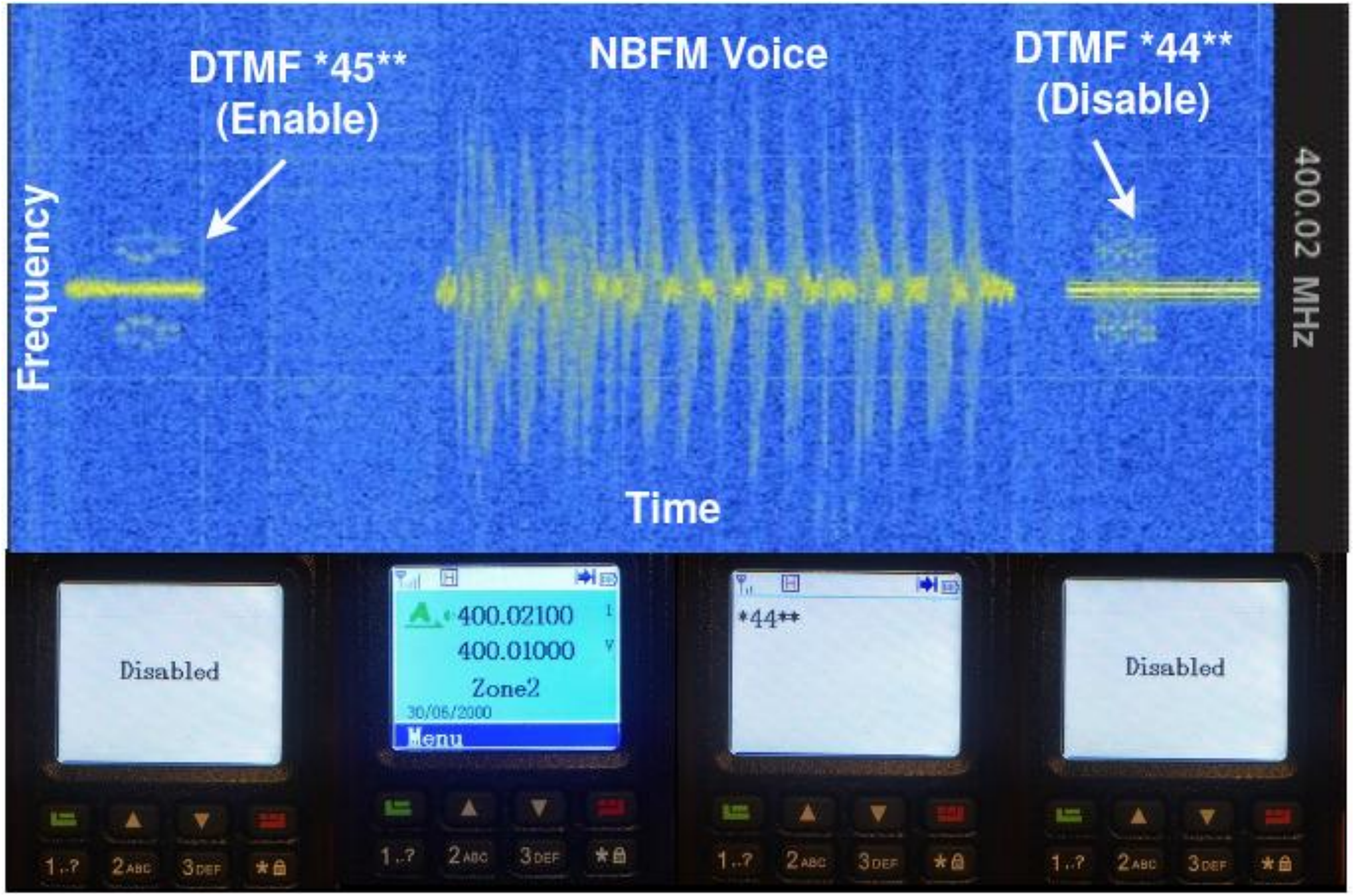
Examples of Tracking



Tracking using FT4 beacons, up to 5m on Galaxy S5 Mini
Using existing reception tools

J. Taylor, FT4, https://physics.princeton.edu/pulsar/k1jt/FT4_Protocol.pdf.
J. Taylor, WSJT, https://physics.princeton.edu/pulsar/K1JT/.

Examples of Injection



IoT to UHF radio injection, a few meters

Countermeasures

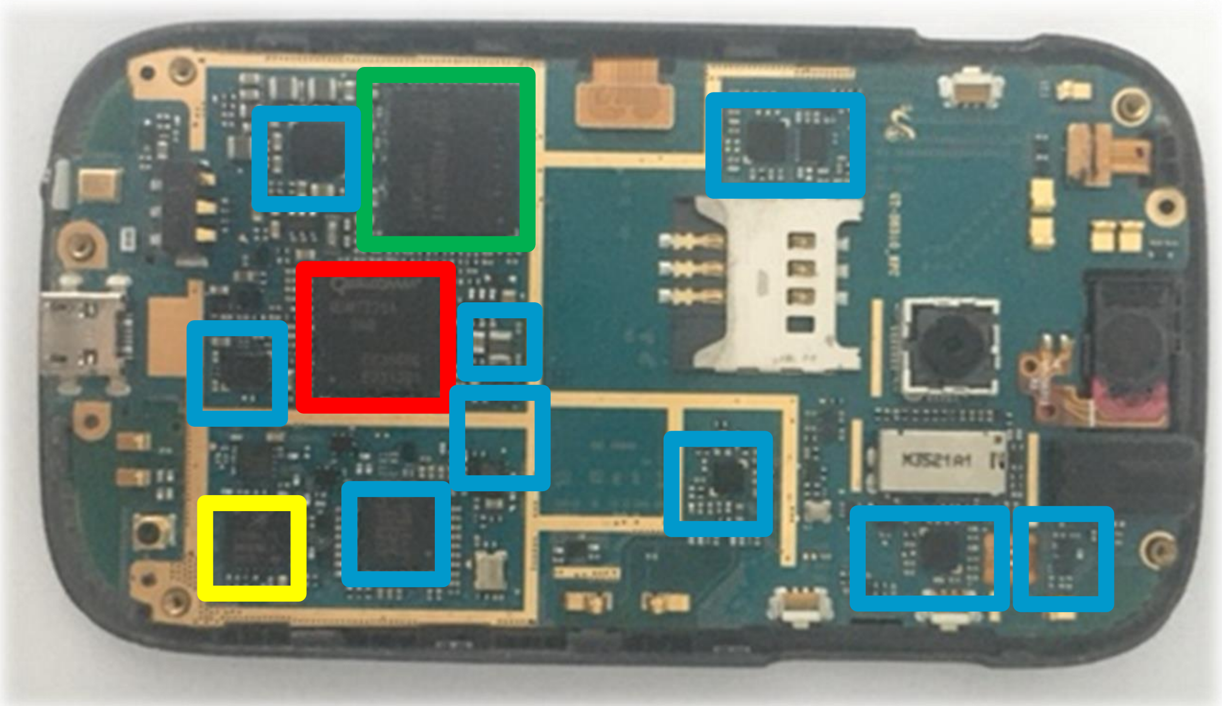
Soft-TEMPEST-specific (HW)
Reduce leakages and coupling

Soft-TEMPEST-specific (SW)
Reduce timing resolution and software control on hardware

Applications specific (SW/HW):
Shield smartphone, spoofing detection, …

Results in perspective

Security threats due to integration of digital and radio*

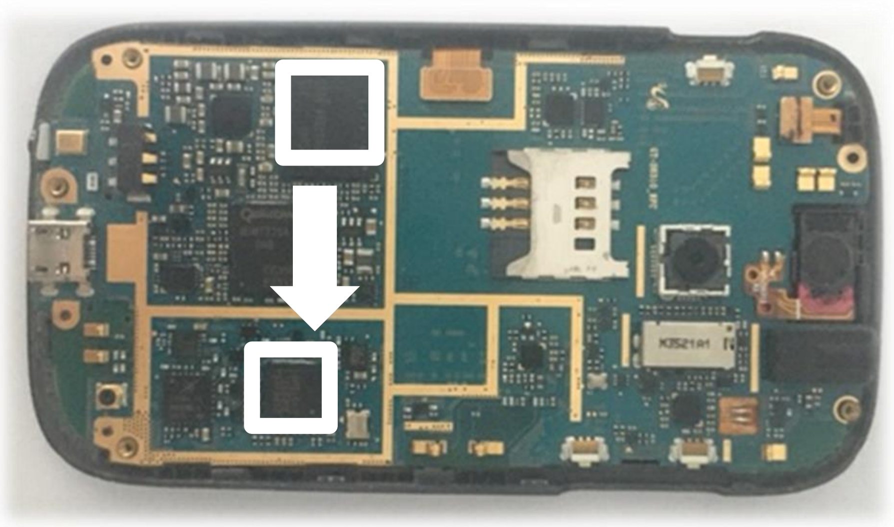


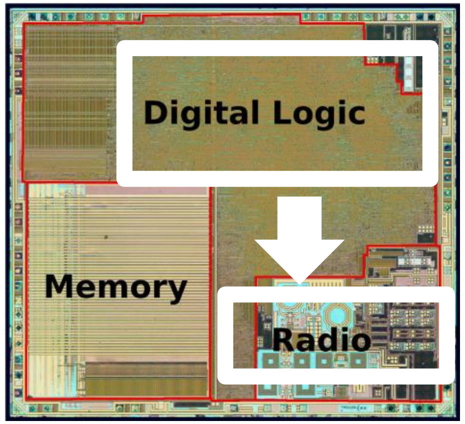
A complex platform
(an old one, easy to open …)

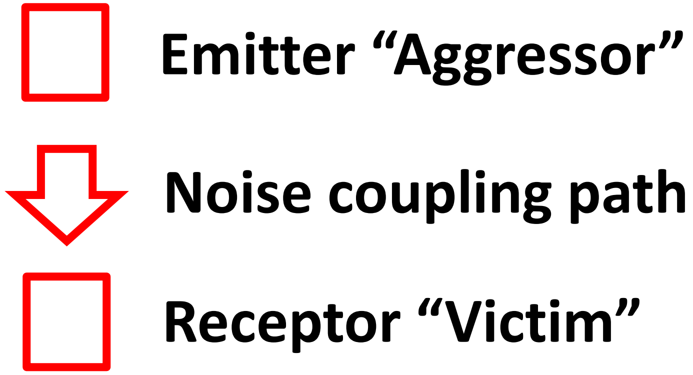
- Red: **CPU, GPU, GSM, …**
- Green: **eMCP (eMMC + LPDDR)**
- Yellow: **GSM + GPRS**
- Blue: **Much more (GPS, FM, WiFi, …)**

*G. Camurati., “Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers” doctoral thesis 2020.

EM Interference between digital and radio components







K. Slattery and H. Skinner, “Platform Interference in Wireless Systems: Models, Measurement, and Mitigation” (Newnes, 2011).

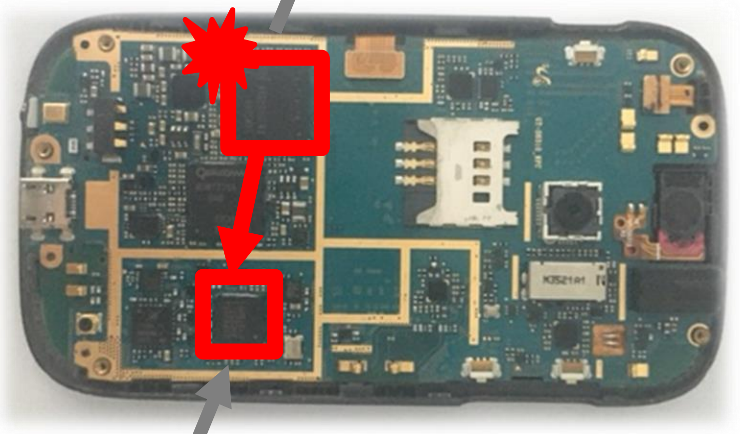
S. Bronckers et al., “Substrate Noise Coupling in Analog/RF Circuits” (Norwood, MA, USA: ARTECH HOUSE, 2009).

K. Slattery and H. Skinner, “Platform Interference in Wireless Systems: Models, Measurement, and Mitigation” (Newnes, 2011).

A. Afzali-Kusha et al., “Substrate Noise Coupling in SoC Design: Modeling, Avoidance, and Validation,” Proceedings of the IEEE (December 2006).

*G. Camurati., “Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers” doctoral thesis 2020.

Can we inject valid packets using noise?



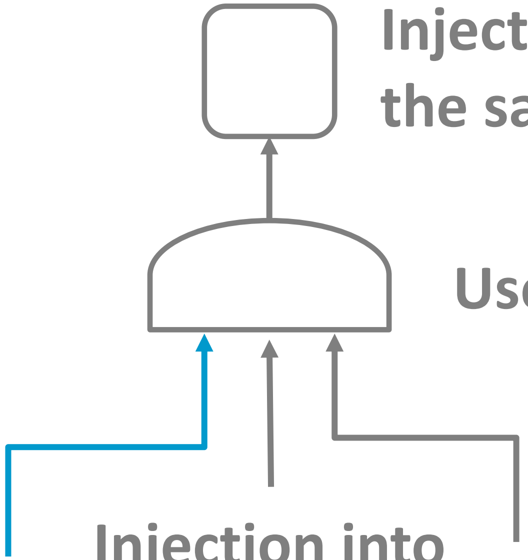
- Natural question in this context
- We need Arbitrary Modulation
- Hence the importance of Noise-SDR
- Though Noise-SDR is more general

*G. Camurati., "Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers" doctoral thesis 2020.

The grand vision: GPS spoofing on the same device

Main results

Future work



Injecting fake position on the same device

Use platform noise modulation

Arbitrary noise modulation

Focus on Arm smartphones

"Noise-SDR"

Injection into radio receiver

RFI, noise coupling, …

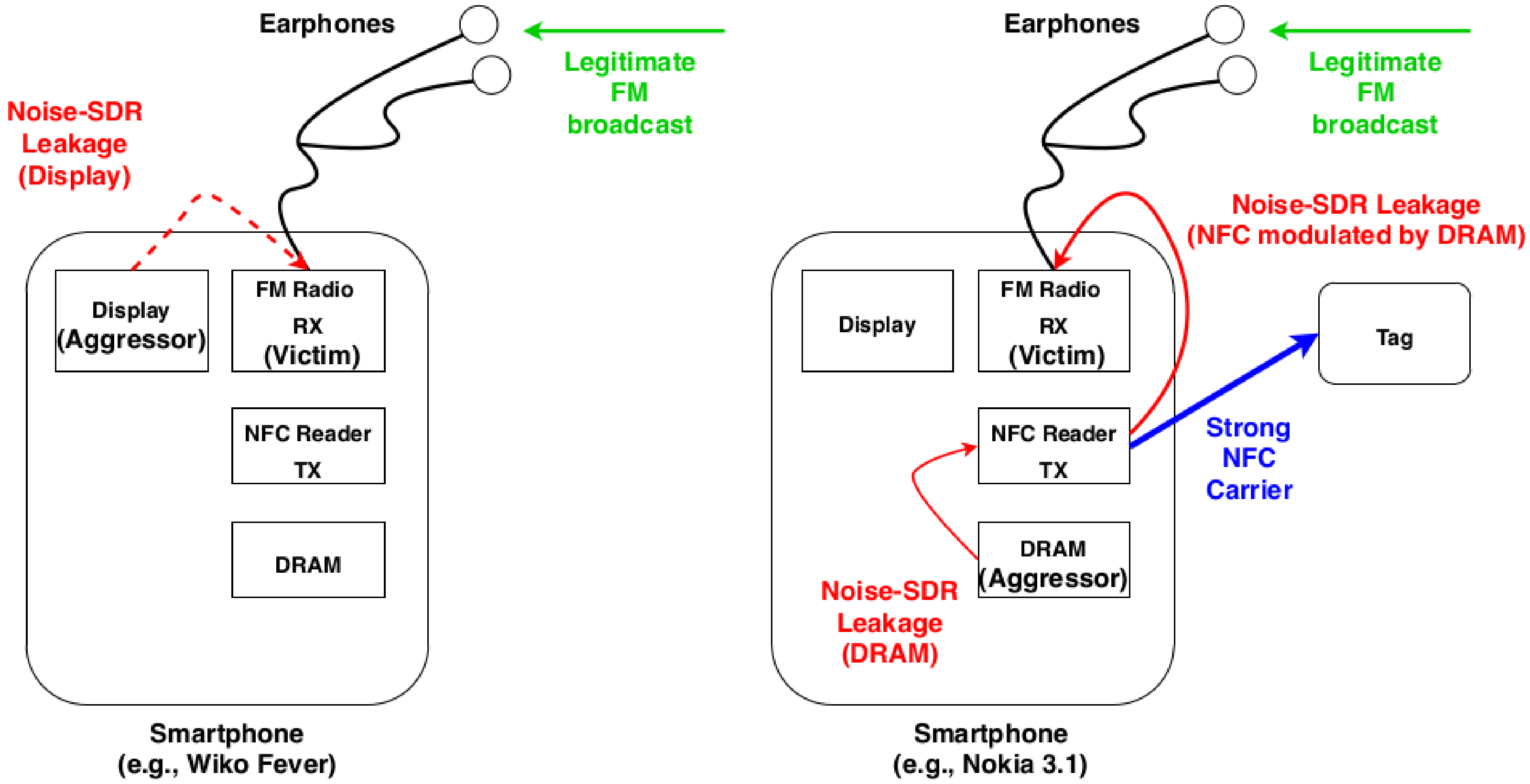
e.g., GPS jamming, FM injection, NFC modulation

Simplified spoofing

e.g. GLONASS C/A Codes

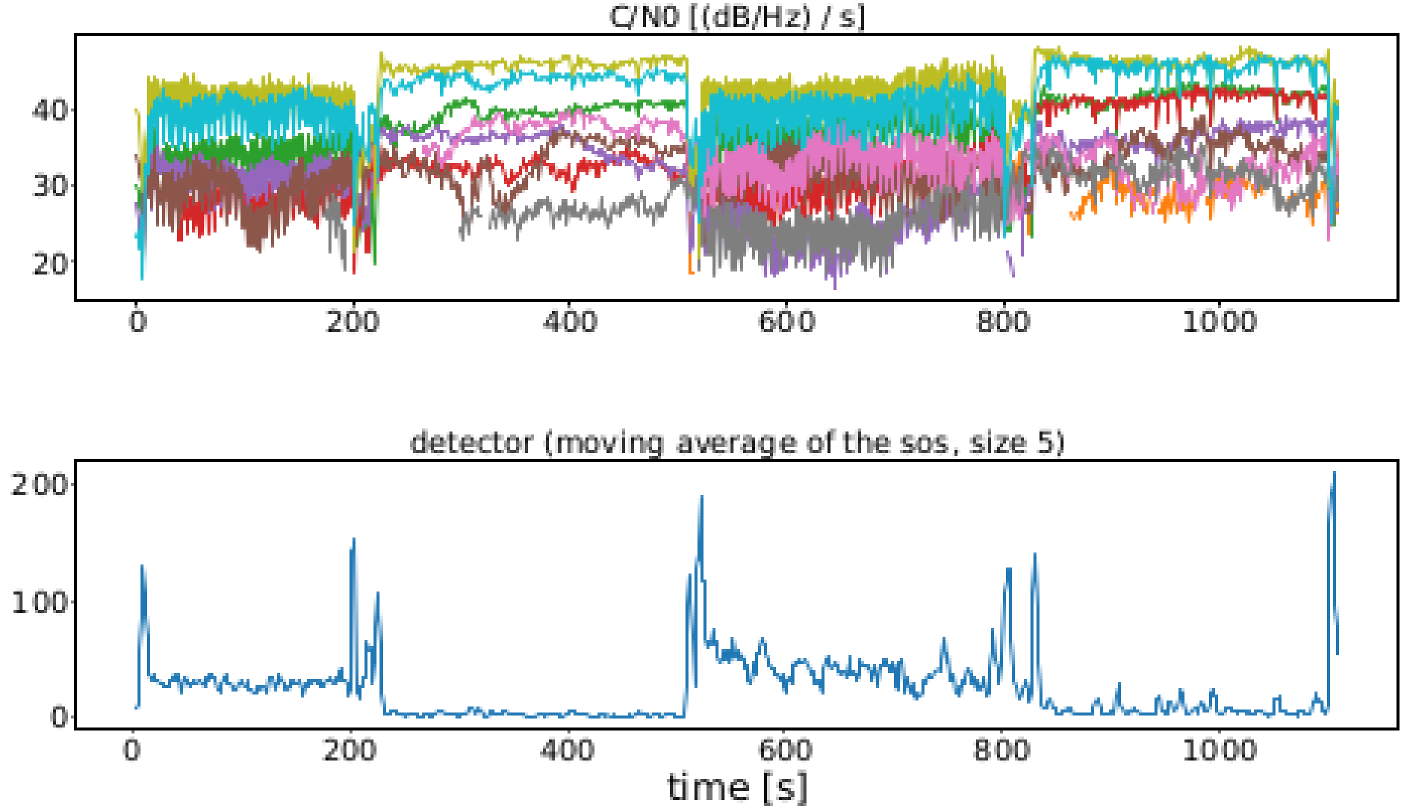
*G. Camurati., "Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers" doctoral thesis 2020.

Preliminary results on injection



*G. Camurati., "Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers" doctoral thesis 2020.

Preliminary results on jamming



*G. Camurati., "Security Threats Emerging From The Interaction Between Digital Activity and Radio Transceivers" doctoral thesis 2020.

Future work and conclusion

Future work

Optimizations

Time resolution, other types of one-bit coding, ...

Other sources / languages

JavaScript, WebAssembly, GPU, … (some preliminary results)

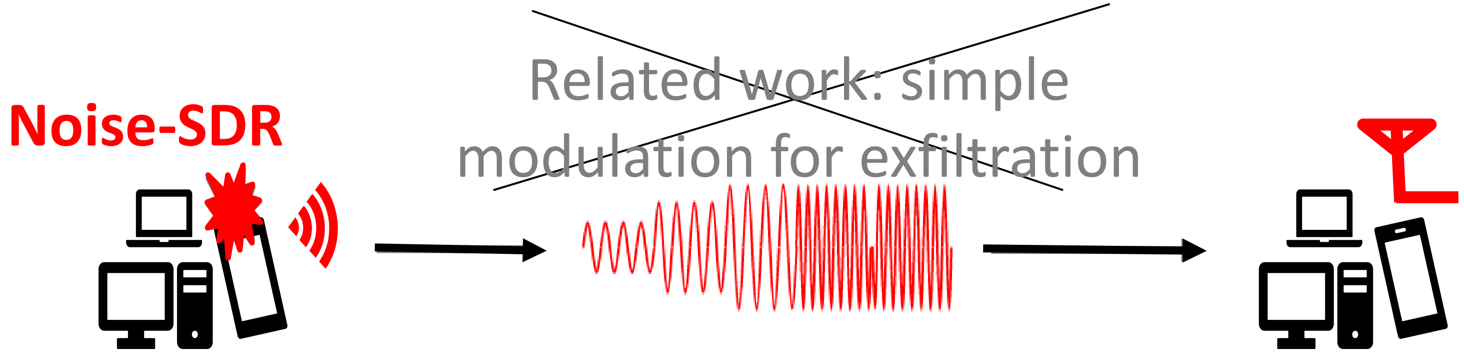
Spoofing and jamming

Radios, sensors, ...

Open source

https://github.com/eurecom-s3/noise-sdr

Noise-SDR: Arbitrary Modulation of Electromagnetic Noise from Unprivileged Software and Its Impact on Emission Security



Arbitrarily modulated EM leakage

How: DRAM accesses, pass-band one-bit coding, software-defined

Pros: flexibility, performance, reuse of existing protocols

Cons: limited bandwidth, center frequency

Implementation: ArmV8A, ArmV7A, x86, MIPS

Security impact: exfiltration, tracking, injection, …

EURECOM Sophia Antipolis

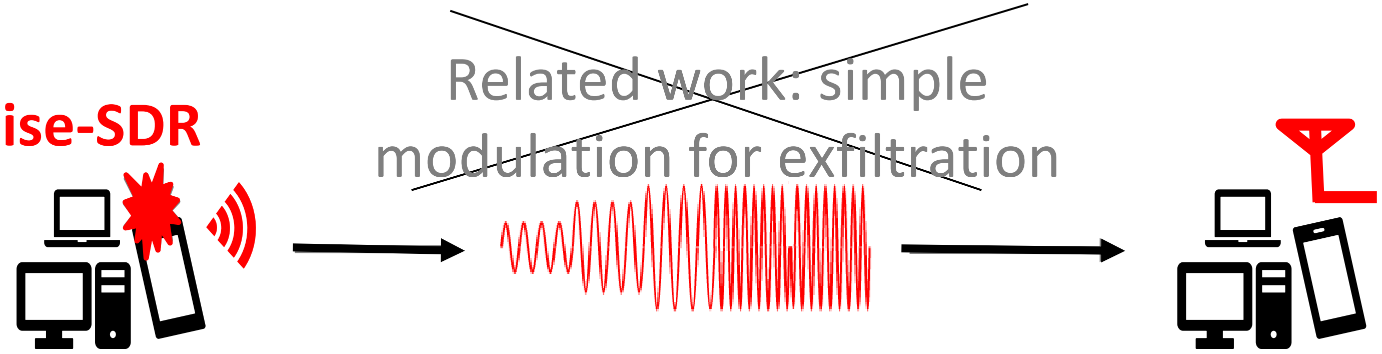
Open source

https://github.com/eurecom-s3/noise-sdr

Noise-SDR: Arbitrary Modulation of Electromagnetic Noise from Unprivileged Software and Its Impact on Emission Security

Thank you!

Questions?



Arbitrarily modulated EM leakage

How: DRAM accesses, pass-band one-bit coding, software-defined

Pros: flexibility, performance, reuse of existing protocols

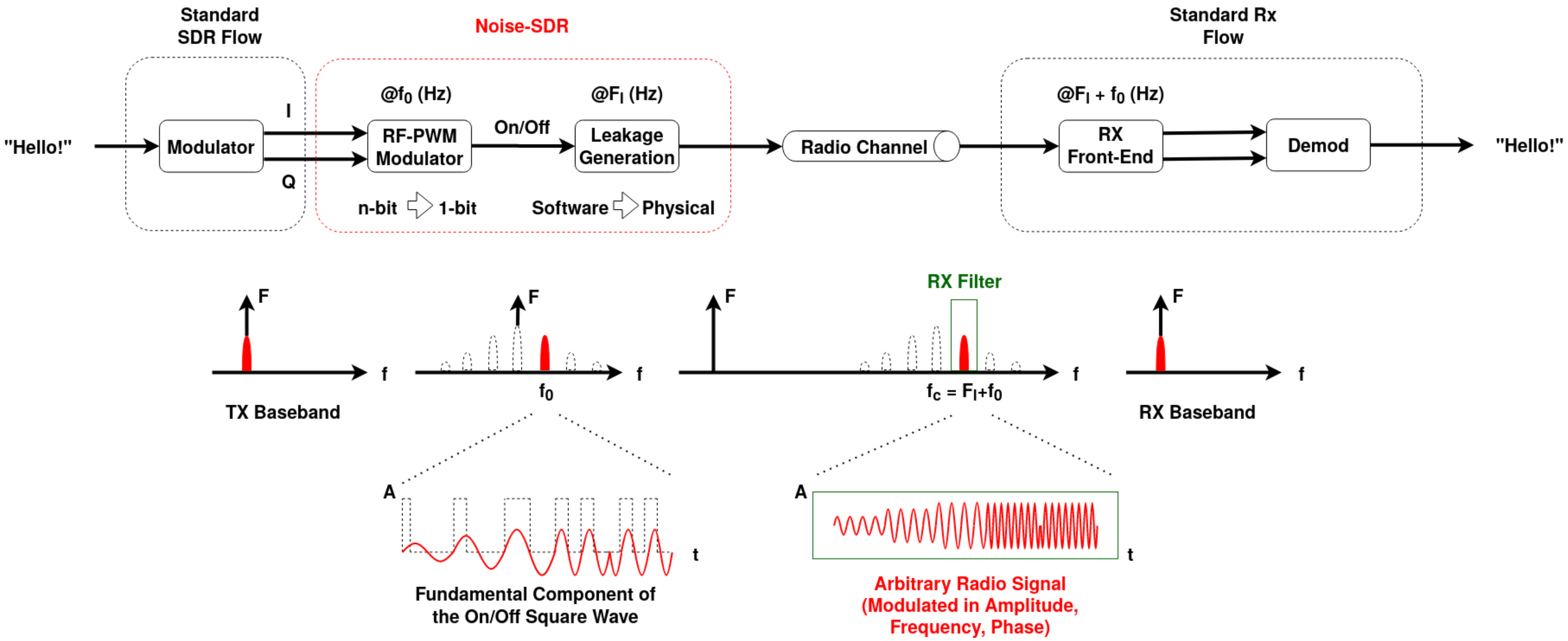
Cons: limited bandwidth, center frequency

Implementation: ArmV8A, ArmV7A, x86, MIPS

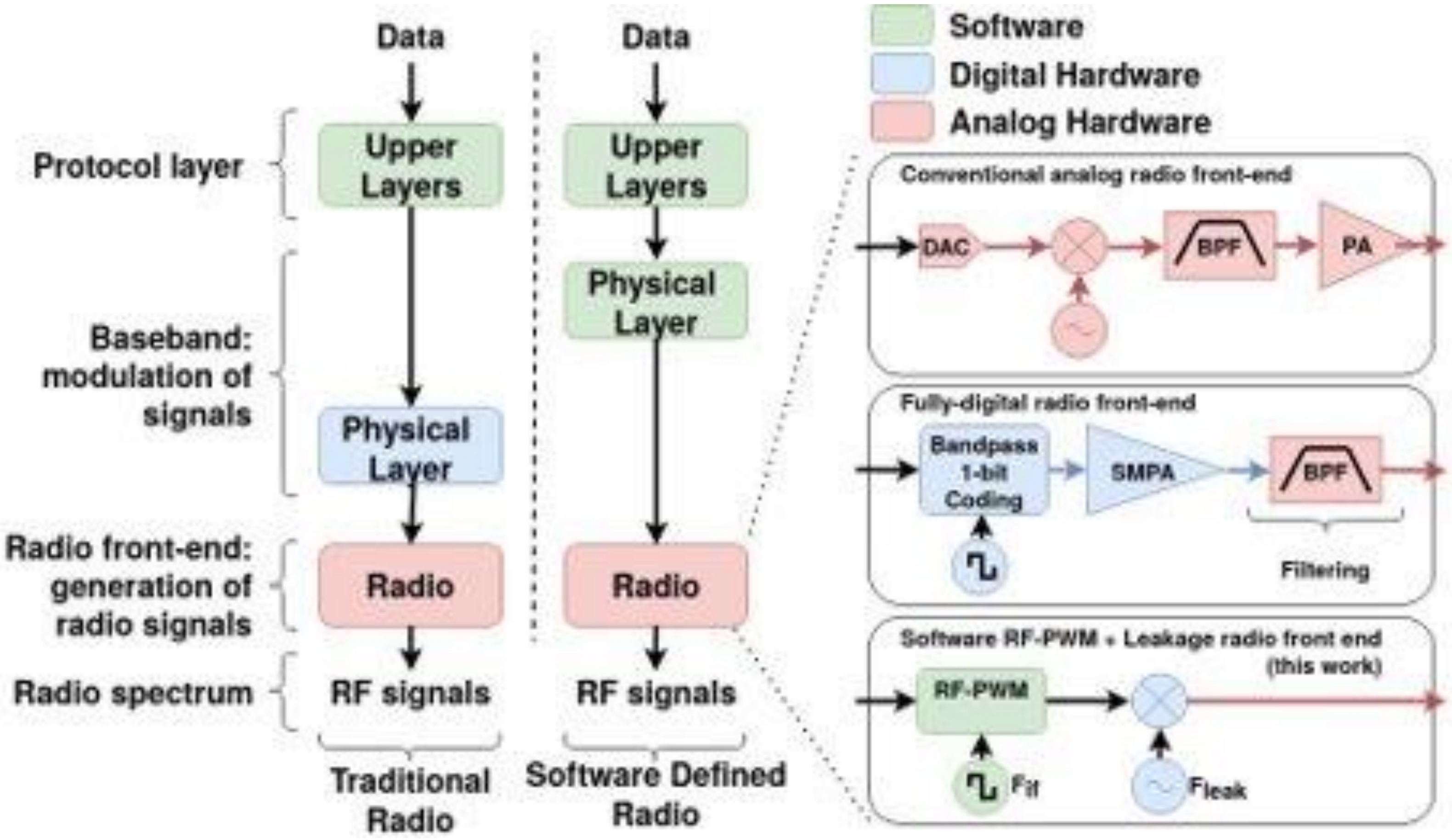
Security impact: exfiltration, tracking, injection, …

Backup Slides

The full chain

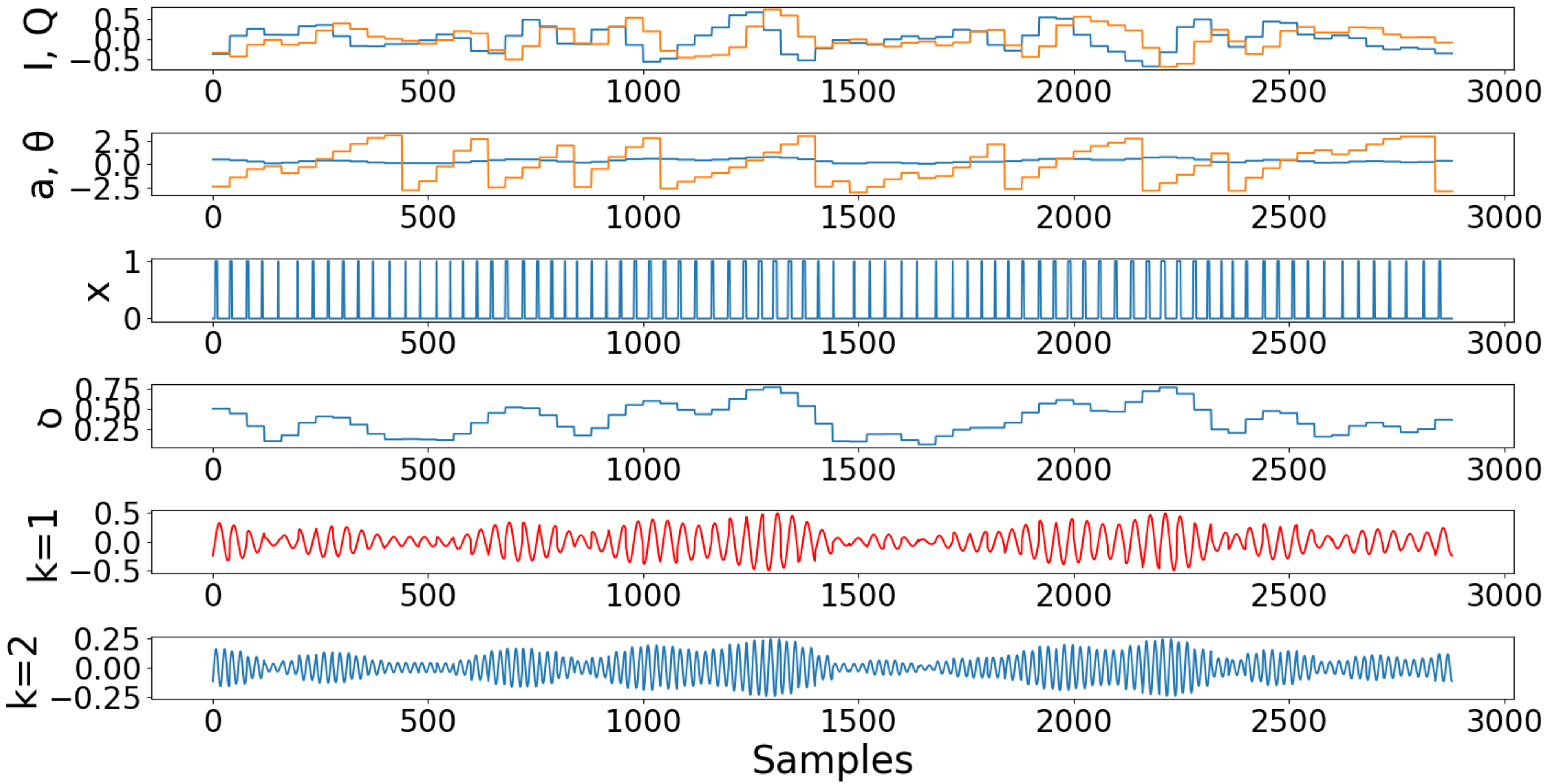


Comparison with other radios



G. Camurati, A. Francillon

Example of HamDRM RF-PWM



Implementation of discrete-time RF-PWM

```
while (i
      < outputBufferIndex && outputBuffer[i++] < 0) {
  }
  while (i < outputBufferIndex && len < EDGESIZE) {
    uint64_t j = 0;
    double a = outputBuffer[i];
    while (i + j < outputBufferIndex
         and outputBuffer[i + j] >= 0) {
      j++;
      if (outputBuffer[i + j] > a)
        a = outputBuffer[i + j];
    }
    while (i + j < outputBufferIndex
         and outputBuffer[i + j] < 0) {
      j++;
    }
    if (len < EDGESIZE - 1) {
      edges[len++] = (asin(a) /
           M_PI) * (uint64_t)(1e9 * j / samplerate);
      edges[
          len++] = (uint64_t)(1e9 * j / samplerate);
    }
    i += j;
  }
}
```

Listing 1. From a sinusoidal IF carrier (*outputBuffer*) modulated in amplitude/frequency/phase to the corresponding RF-PWM square wave timings (*edges*).

$$f_0 = \frac{F_{res}}{q}, q \geq 2$$

$$\theta_k = 2k\pi f_0 \frac{q}{F_{res}}, q \in \left[-\left\lfloor \frac{F_{res}}{2kf_0} \right\rfloor, \left\lfloor \frac{F_{res}}{2kf_0} \right\rfloor\right)$$

$$a_k = \sin(k\pi q \frac{f_0}{F_{res}}), q \in \left[0, \frac{1}{2k} \frac{F_{res}}{f_0}\right)$$

Future Work

Model the spectrum in detail

Effect of the edges

Effect of interpolation

Effect of jitter

Etc.

HamDRM to GLONASS, chose the best trade-off!

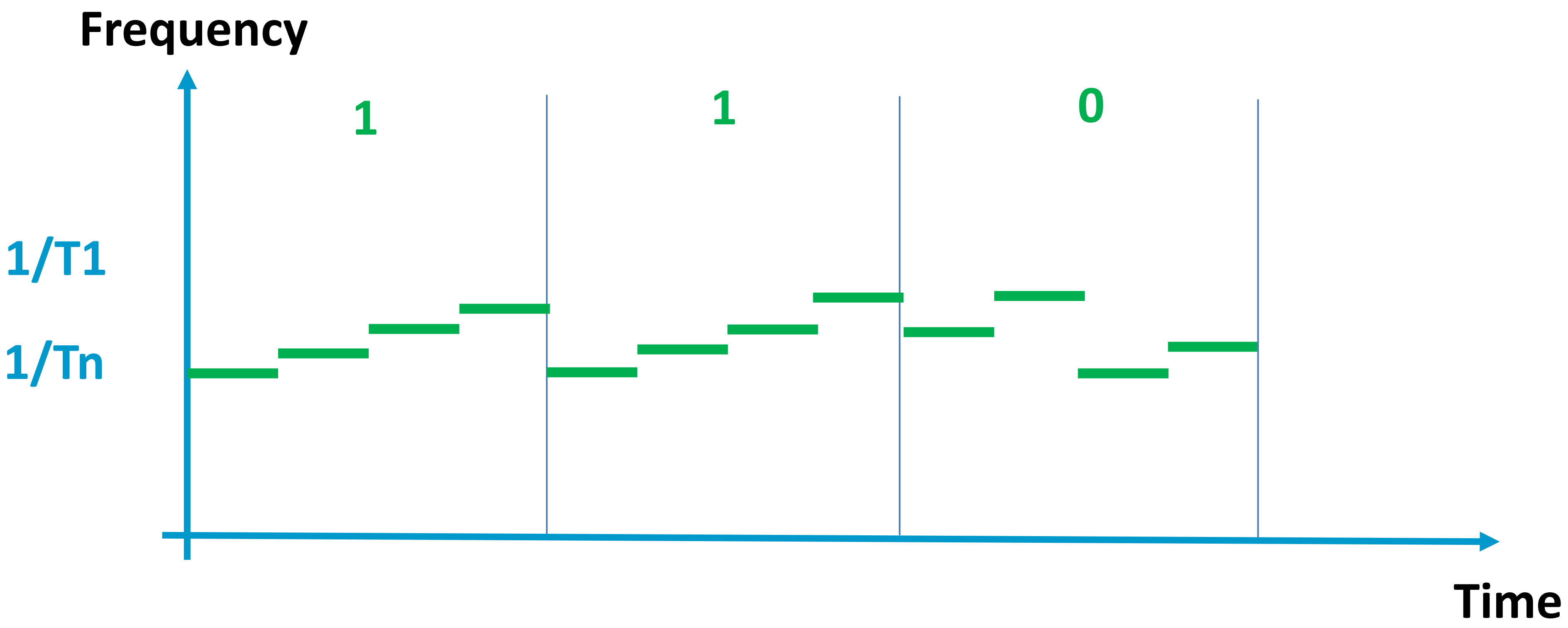
Name	Modulation	Bandwidth
Voice AM	AM	10 kHz
Voice FM	NBFM	12.5 kHz
PSK31	2-PSK, USB	31 Hz
2xPSK500	2 2-PSK subcarriers, USB	1.2 kHz
RTTY45.45	2-FSK, USB	170 Hz
MFSK128	M-FSK, USB	1.928 kHz
Olivia 64/2000	M-FSK USB	2 kHz
SSTV	FM, USB	2.5 kHz
HamDRM	QAM, OFDM, USB	2.4 kHz
FT4	4-GFSK, USB	90 Hz
LoRa	CSS	8 kHz (customizable)
GLONASS C/A	DSSS	0.511 MHz

Evaluation

	Device	Type	Arch.	OS Family	DRAM	F_{leak}	$(F_{IF}+B)_{max}$	SSC	Harmonics n
A	HP ENVY	Laptop	*x86-64*	Ubuntu	DDR3	800 MHz	15.062 kHz	yes	1
B	PC	Desktop	*x86-64*	Windows	DDR3	800 MHz	35.062 kHz	yes	1
C	Samsung Galaxy S5 Mini	Phone	*ARMv7-A*	Android	n.a.	400 MHz	15.062 kHz	no	1-11, 13-19, 26
D	Innos D6000	Phone	*ARMv8-A*	Android	LPDDR3	800 MHz	1.130 MHz	no	1-4
E	8Devices Carambola2	IoT	MIPS	OpenWRT	DDR2	400 MHz	35.062 kHz	no	1-6

	Protocol	Speed	A (cm)	B (cm)	C (cm)	D (cm)	E (cm)
IV.1	Simple CW20	20 wpm	-	200	2	-	300
IV.2	Simple CW100	100 wpm	-	2	-	-	60
IV.3	Simple RTTY50	66 wpm	-	1	3	0	30
IV.4	Simple RTTY75	100 wpm	-	0	2	-	25
IV.5	LoRa-like 8 kHz, SF=8	16 bytes, 1.128 s	-	75	8	0	210
IV.6	LoRa 8 kHz, SF=8	16 bytes, 1.928 s	-	120	9	3	300
IV.7	MFSK32	120 wpm	0	20	15	1	300
IV.8	MFSK128	480 wpm	-	9	8	0	84
IV.9	THOR4	14 wpm	8	250	110	10	>500
IV.10	THOR16	58 wpm	0	105	65	4	>500
IV.11	THOR100	352 wpm	-	30	5	2	65
IV.12	PSK125	200 wpm	0	100	4	0	40
IV.13	PSK125R	110 wpm	0	250	15	1	75
IV.14	3xPSK250R	660 wpm	-	2	1	-	50
IV.15	2xPSK500	3200 wpm	-	-	0 (Unreliable)	-	1 (Unreliable)
IV.16	2xPSK500R	1760 wpm	-	-	1	-	10
IV.17	HamDRM A QAM4	1140x960RGB, 45 s	-	-	0 (Needs multiple runs)	-	5
IV.18	GLONASS C/A	511 chips per 1 ms	-	-	-	0	-
IV.19	GLONASS /10	511 chips per 10 ms; 5 bps	-	-	-	0	-
IV.20	GPS C/A /100 (2 codes)	1023 chips per 100 ms	-	-	-	0	-
IV.21	FT4	77 bits, 4.48 s	0	100	500 (If detected, see Figure 12)	1	500
IV.22	AM	16-bit 44.1 kHz audio	-	4	5	0	50
IV.23	NBFM	16-bit 44.1 kHz audio	-	10	10	0	>400
IV.24	SSTV Martin1	320x256RGB, 114 s	-	2	5	0	30

Concurrent work: LoRa-like spread spectrum



CCS (Simplified)

C, Shen et al., “When LoRa Meets EMR: Electromagnetic Covert Channels Can Be Super Resilient”, IEEE S&P 2021

Acknowledgements

We would like to thank:

- Google, Elie Bursztein, and Jean-Michel Picod, for the Faculty Research Award assigned to Aurélien Francillon.
- Andrea Possemato, Giulia Clerici, Matteo Guarrera, the anonymous reviewers and our shepherd.