DVFSspy: Using Dynamic Voltage and Frequency Scaling As A Covert Channel for Multiple Procedures

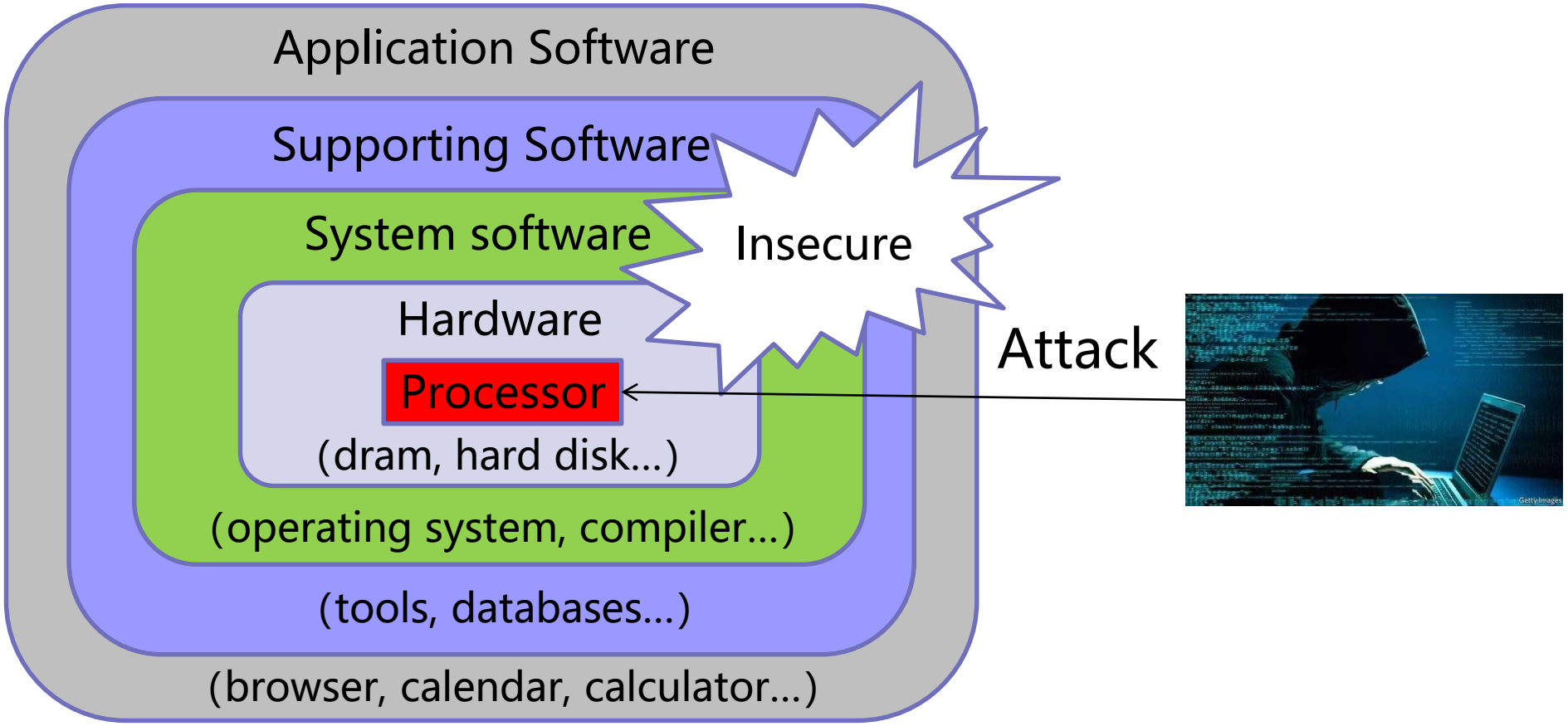
Pengfei Qiu[1], Dongsheng Wang[1], Yongqiang Lyu[1], Gang Qu[2]
[1]Tsinghua University
[2]University of Maryland

qpf15@tsinghua.edu.cn

Self-Introduction

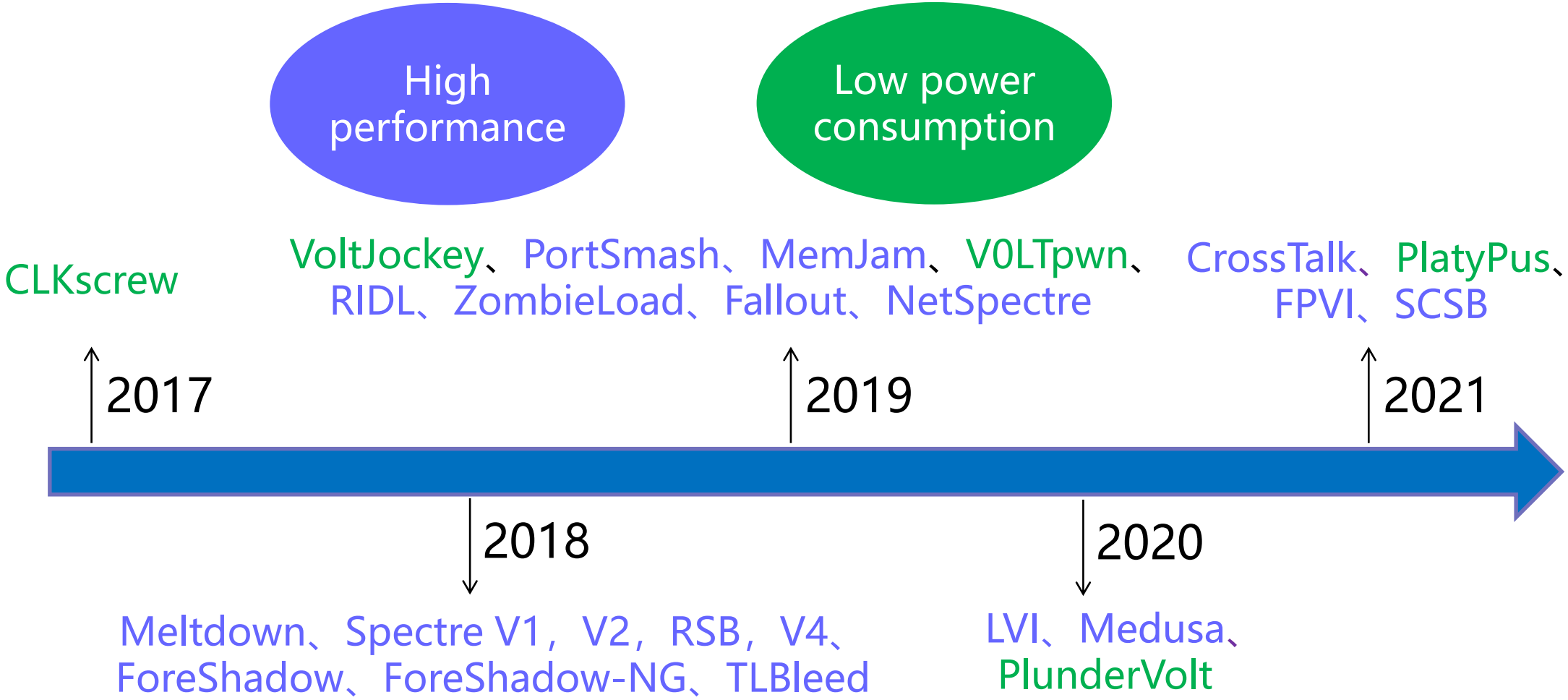
- Basic information
  - Ph.D. degree from the Tsinghua University
  - Computer science and technology
  - Post doctor
- Research field——Processor security
  - Processor vulnerability mining
  - Secure processor design

The Keystone of the Computer Security



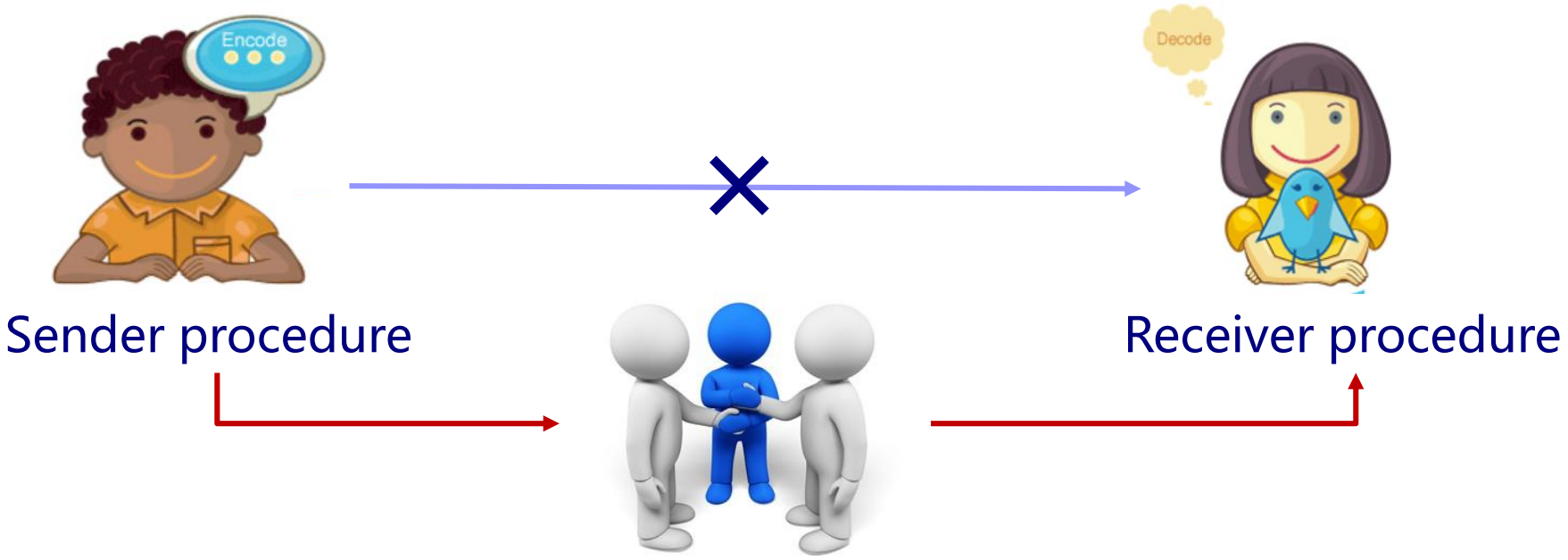
Processor Vulnerabilities

- Main optimization objectives in traditional processor design



Our Work

- Security issue caused by dynamic voltage and frequency scaling (DVFS)
- DVFSspy
  - A covert channel attack
  - Enable different procedures transmit messages secretly
  - Dynamic voltage and frequency scaling technology (DVFS) is a middleman

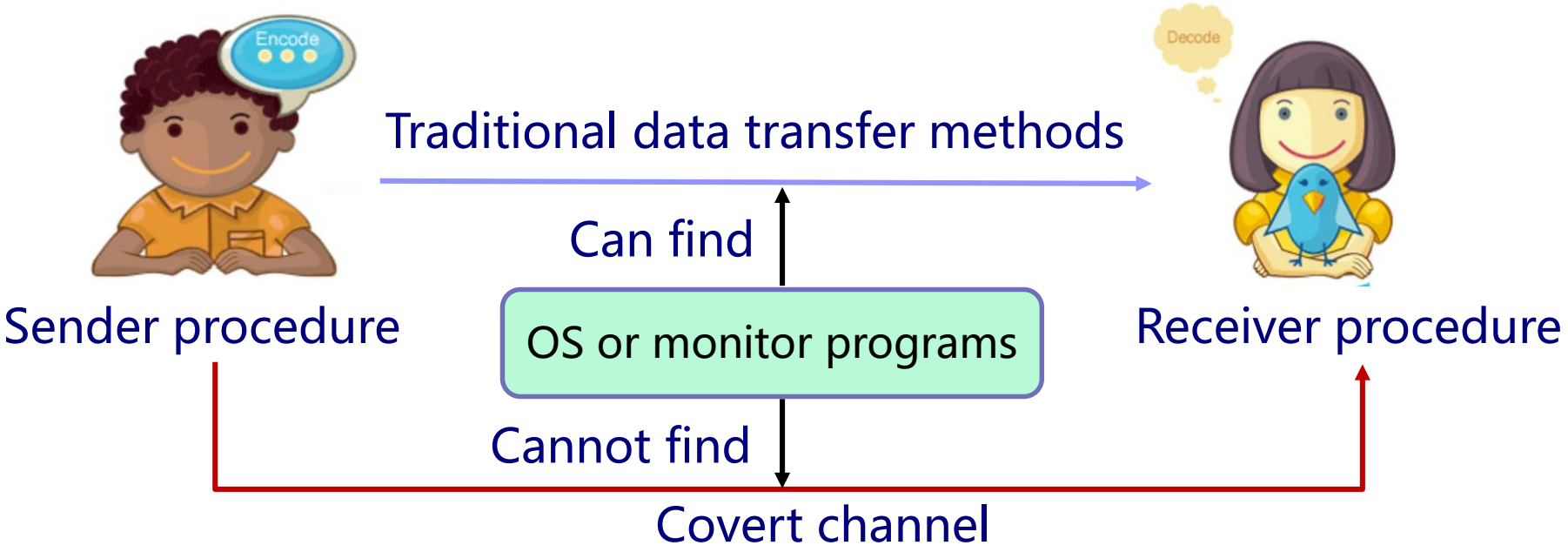


Outline

Outline

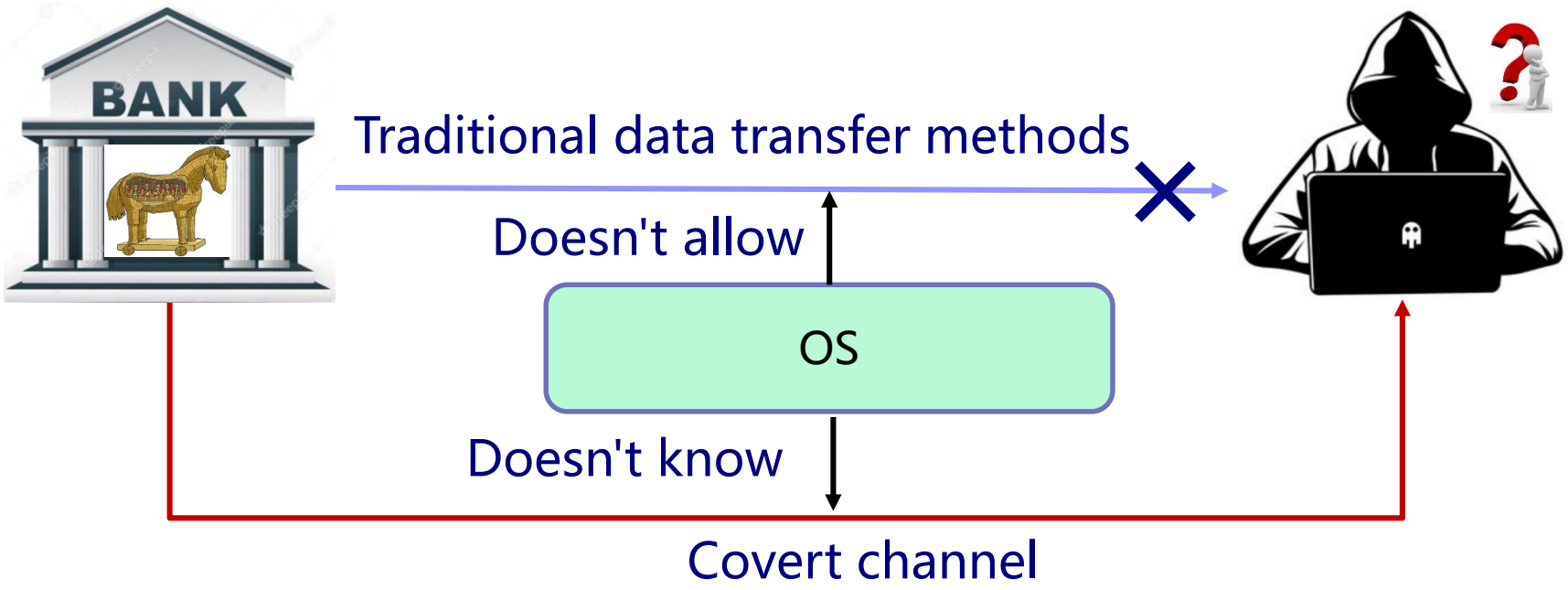
Background——Covert Channel Attack

- Traditional data transfer methods
  - Shared memory, signal, and socket et al.
  - Leave physical evidences to the Operating System (OS) or other monitor programs
- Covert Channel Attack
  - A channel to transfer data that are not allowed to do so by the security policy
  - Hidden from access control s enforced by OS or other monitor programs

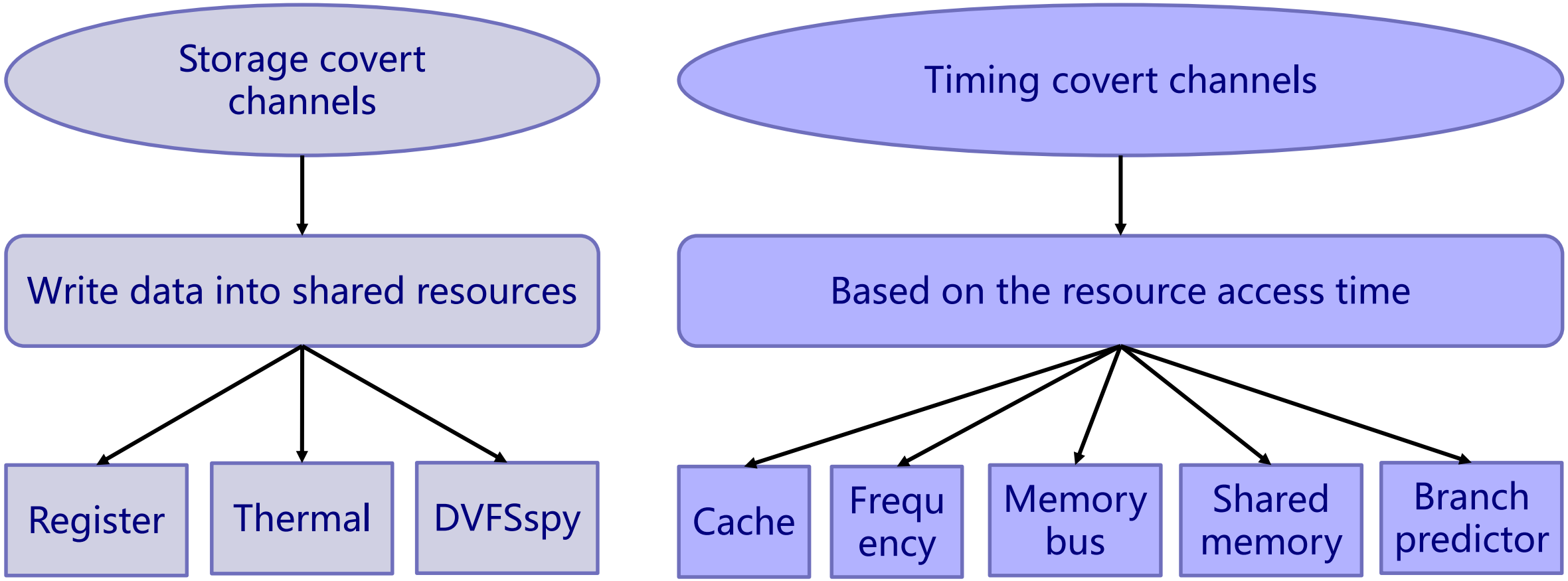


Background——Covert Channel Attack

- An example——Banking application
  - Possesses important information such as account, password, identification
  - OS enforces security policies to ensure that it is isolated from other applications
  - Data is only allowed to be transferred to the authenticated bank servers
  - The attacker gains the secret information by a Trojan horse or a backdoor, but cannot send the data to anyone other than the trusted bank servers



Background——Covert Channel Attack



- Partition and isolation can are proposed to address covert channels

Background——Frequency Covert Channel Attack

- Sender procedure:
  - Change the frequency by the privileged interface
- Receiver procedure
  - Measure the time spent by a loop

- Limitations.
  - The sender procedure has a high privilege
  - The frequency is fixed when measuring

Background——Dynamic Voltage and Frequency Scaling

- Energy consumption is the integral of instantaneous power over time

$$E_T \int_0^T (DP_t + SP_t) d_t$$

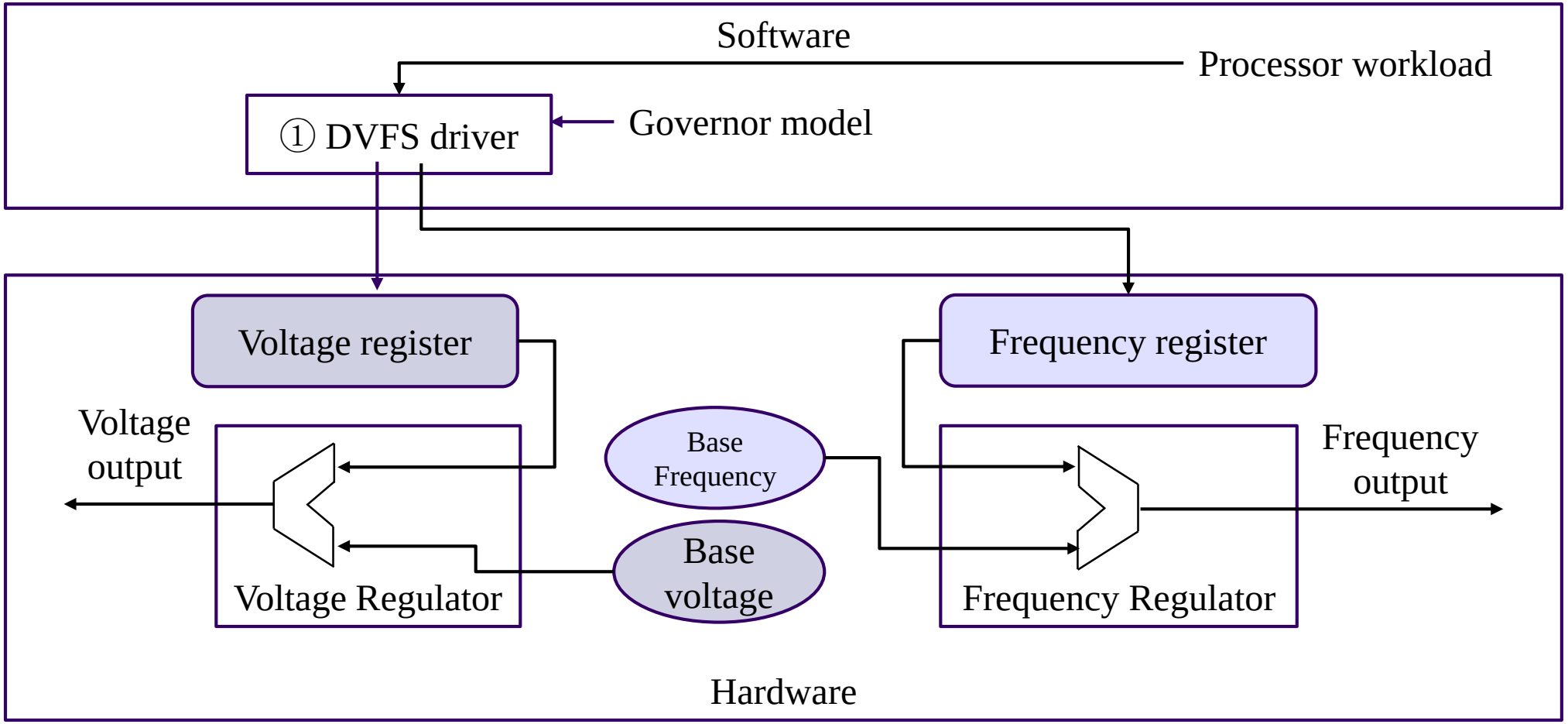
- Instantaneous power consists of static and dynamic power
- Dynamic power

$$DP_t \propto V_t^2 * F_t * C$$

- Frequency and voltage are two key factors for energy dissipation
- Dynamic Voltage and Frequency Scaling (DVFS)
  - Update the frequency and voltage based on the workload

Background——Dynamic Voltage and Frequency Scaling

- DVFS driver selects the proper frequency and voltage
- The registers control the times of base frequency and voltage



Outline

Motivation

- Methods to Measure the Processor’s Frequency
  - Time spent by an loop (Utilized by prior frequency covert channels)
    - Loop will raise the workload and make the frequency inaccurate
  - The registers that control the frequency (high privilege)
  - System monitor modules (high privilege)
  - The frequency logged by the DVFS (low privilege, a vulnerability)

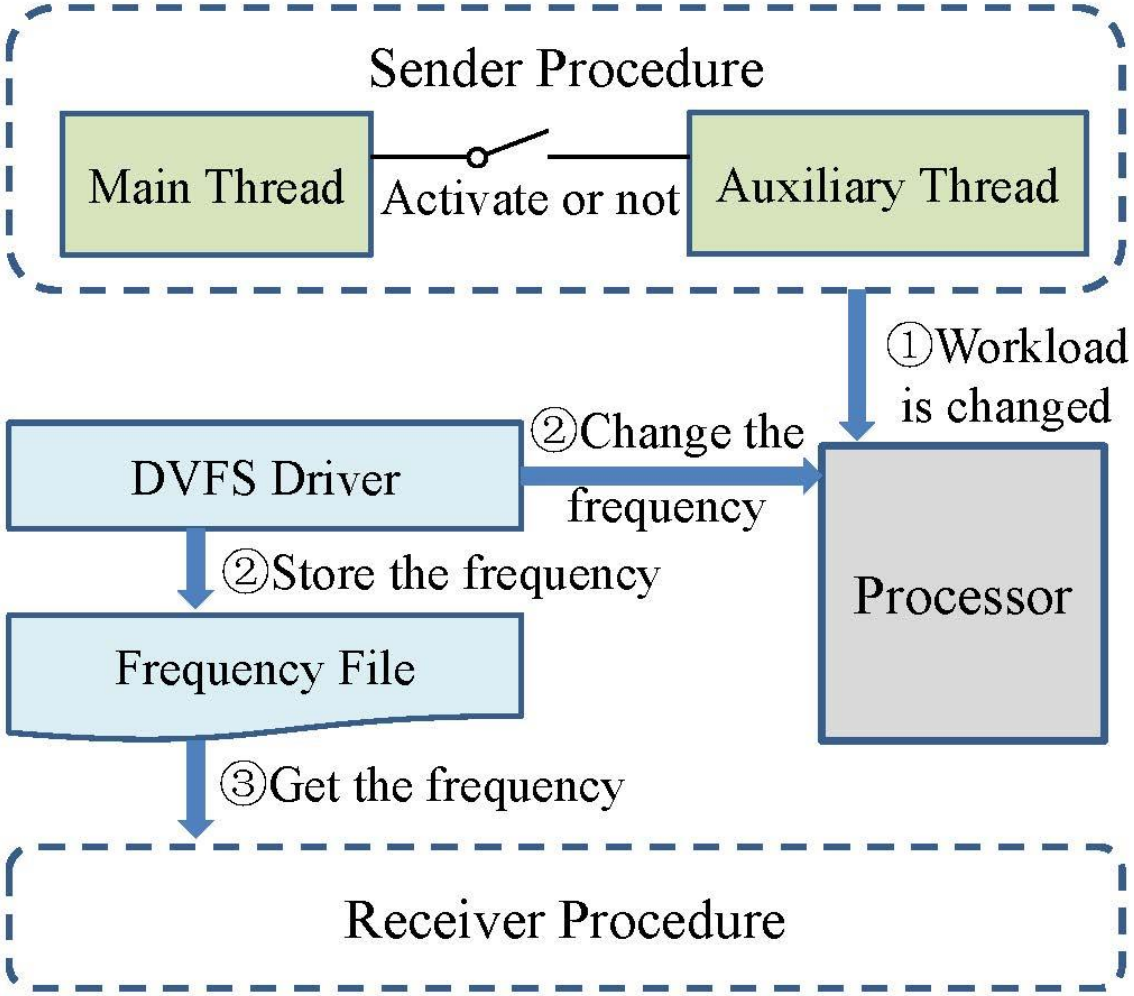
- Create an efficient frequency covert channel
  - Does not require a high privilege for the sender and receiver procedures

Assumption and Threat Model

- Assumption
  - Processor is a multi-core processor equipped with DVFS
  - Processor’ s requency can be obtained in the userspace

- Threat Model
  - Sender and receiver procedure communicate with each other

Overview of DVFSspy



- DVFS is a middleman
- Auxiliary thread
  - performs an intensive computing job

- Raise the frequency
  - Activate the auxiliary thread
- Reduce the frequency
  - Doesn’ t activate the auxiliary thread
  - Kill the running auxiliary thread

Three features

- Does not directly manipulate the frequency
- Sender and receiver procedures can deliver data from both sides
- DVFSspy exposes the security risks caused by hardware information that is unintentionally leaked by the privileged software.

Some Challenges

- Idle Waiting
  - Busy waiting or no waiting will make the workload high
  - Idle waiting functions: sleep, msleep, and usleep
- Synchronization
  - Cannot use semaphore, shared memory, and socket package
  - Utilize special byte sequences: #### for the start and $$$$ for the end
- Error Detection and Correction
  - Noises from the environment and other applications
  - Hamming codes for error detect and correct

Outline

Experiment setup

- DELL XPS
  - CPU: i7-8550U
  - Memory: 8G
  - Cores: 4 physical, 8 logical cores
  - OS: Ubuntu 16.04

- Sender procedure is bound to the core 1
- Receiver procedure is bound to the core 2
- Auxiliary thread is in the detached status and is bound to core 3

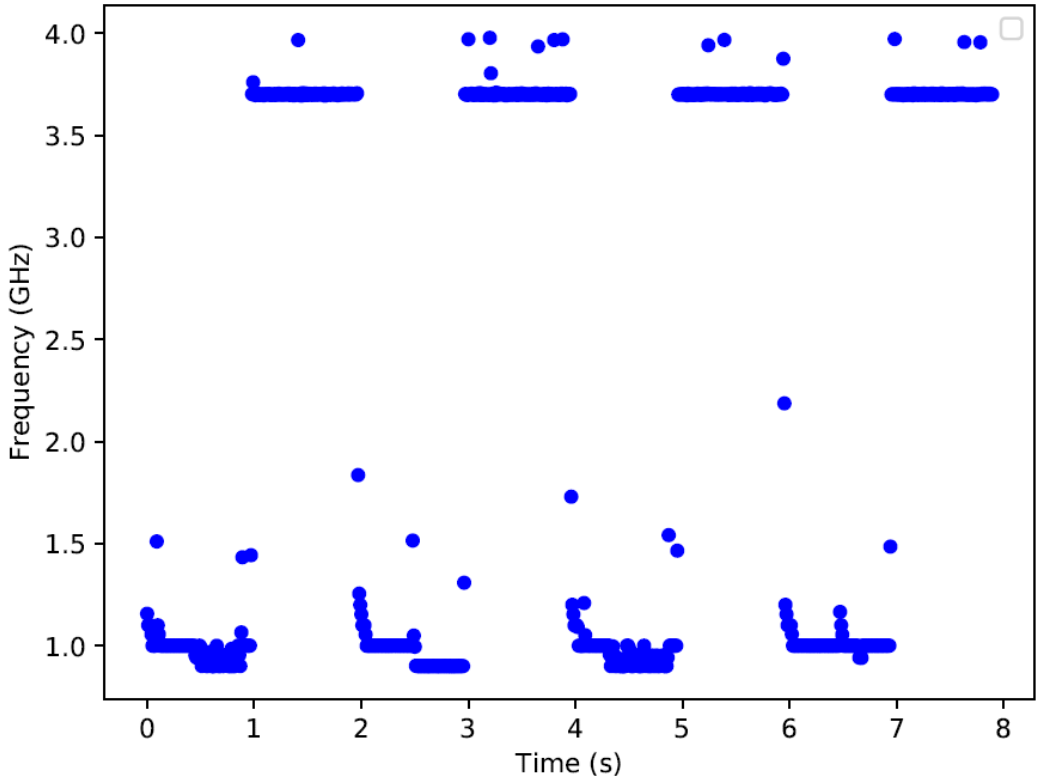
Induce and Monitor Window

- Induce window
  - Time for the sender procedure to update the workload
  - T_{i}
- Monitor window
  - Time between two monitor processes as the monitor window,
  - T_m
- Ensure that the data transmission process is reliable

$$T_{\mathrm{i}} > T_m$$

Encoding and Decoding Protocols

- Repeat the following two steps
  - Auxiliary thread is not invoked
  - Auxiliary thread is invoked
- T_{i} is 1s
- T_m is 10ms

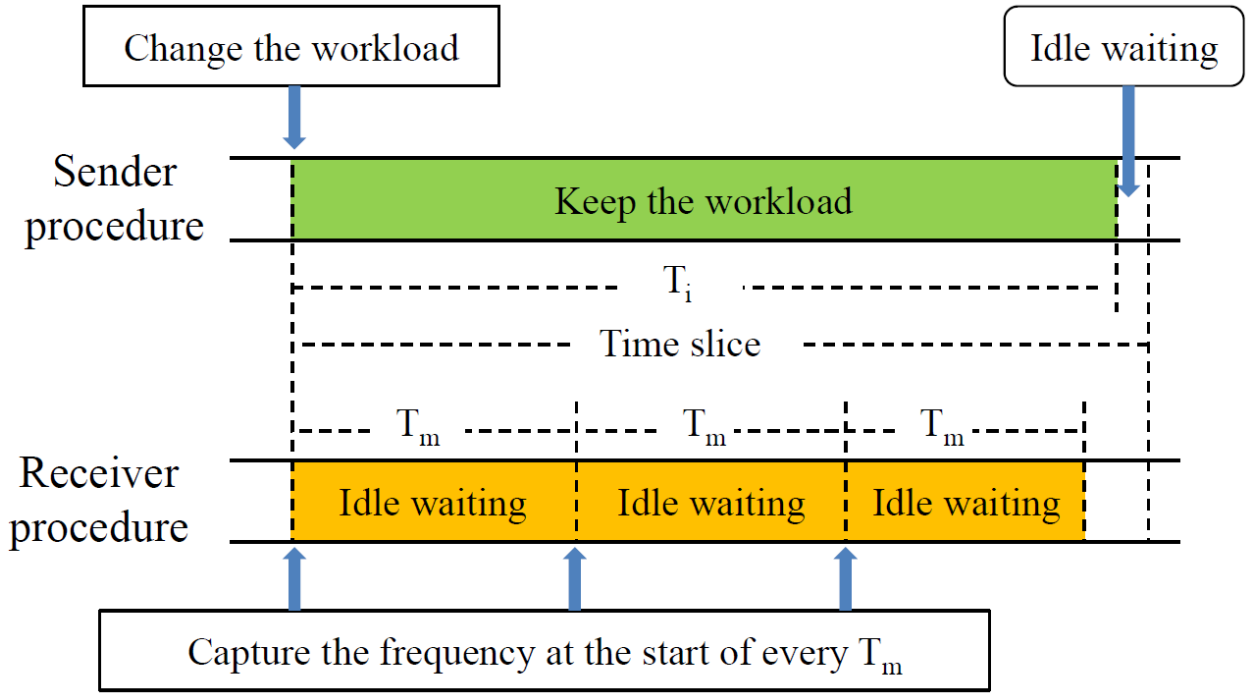


- Encoding
  - Activates auxiliary thread to send 1
  - Doesn’ t activate auxiliary thread to send 0

Noise Mitigating

- Create time slices and insert the majority voting mechanism
  - Measure frequency for K times
  - The decoding data for this slice is decided by the K frequency measurements

$$\text{time slice} > T_{\mathrm{i}} > \mathrm{K}T_m$$



- Create time slices
  - Utilize the instruction of rdtsc
  - Utilize other time functions in the C language is also feasible such as *clock(), gmtime(), timeGetTime(), gettimeofday()*

T_i

- DVFS periodically samples workload and adjusts frequency
- T_{i} should be larger than the sampling interval of the DVFS

- Send a bit string of "010101..." with different T_{i}
  - Measure whether the frequency is updated expectantly.
  - When $T_{\mathrm{i}} \leq$1.35ms, we cannot observe the desired frequency variations
  - Because of the majority voting mechanism, T_{i} is 25ms

K

- Total amount of votes, K will affect he error rate
  - K is low, the error rate may be high
  - K is high, the error rate is low but may affect the workload.

- Time slice is 30ms, T_i is 25 ms, T_m is 1ms
  - Measure the error rate with different K in leaking 10000 random bits

K	1	3	5	7	9	11	13
#Error bits	3856	1834	175	64	57	54	53
Error rate (%)	38.56	18.34	1.75	0.60	0.57	0.54	0.53

- Error rate is very low when K≥7
  - We choose K = 9 to identify the received data

T_m

- A large T_m may make the sampling cross transmission processes
  - T_m should be at most 2.78ms because K = 9 and T_i = 25ms
- A small T_m may make the workload high

- Measure error rate with different T_m in leaking 10000 random bits

$T_m(ms)$	2.7	2.5	2.0	1.5	1.3	1.0	0.8	0.3	0.08	0.05
#Error bits	6586	3428	654	171	62	57	56	56	55	123
Error rate (%)	65.86	34.28	6.54	1.71	0.62	0.57	0.56	0.56	0.55	1.23

- The error rate is 0.57% on average when 0.08ms ≤ T_m ≤ 1.3ms.
- Theerror rate is stable as reading frequency cost very small time.

Performance

- T_i = 25ms, T_m = 1ms, K = 9, threshold = 2.5GHz, time slice = 30ms
  - Transfer 10000 random bits along with their hamming codes
  - They can be fully delivered in about 352s with the error rate is 0.53
  - The throughput is about 28.41bps.

Covert channel	Cache [4]	Shared memory [6]	Memory bus [5]	Branch predictor [7]	Register [8]	Frequency [3]	Frequency [16]	Frequency (DVFSspy)
Throughput (bps)	1291	174.98	1168	66.53k	534	20	1	28.41
Error rate (%)	3.1	2	11	-	-	-	1	0.53

- Throughput is lower than some of prior covert channels
  - DVFS costs some time to decide which frequency should be set.
- Throughput is higher than the prior frequency covert channels

Countermeasures

- Hardware
  - Provide a fixed frequency to the processor
    - Destroy the normal functions of DVFS
  - A watchdog circuit to continually monitor the frequency
    - Need to define the pattern of unexpected frequency changes

- Software
  - Delete the frequency-related interfaces
  - Make the interfaces only can be invoked by privileged applications
    - May damage some system monitor applications

Outline

Conclusion

- Find that processor’ s frequency can be read in the userspace
- Propose DVFSspy, a covert channel in which DVFS is the middleman
- Frequency is the covert signal

- Achieve the covert channel on the mercantile Intel platform
- The error rate as well as the throughput are studied.

Thanks

Pengfei Qiu
qpf15@tsinghua.org.cn
Tsinghua University