Chip Multi Processors, System On a Chip, and Multi-socket Protections

Chapter 7

[1] J. Szefer, “Principles of secure processor architecture design,” *Synth. Lect. Comput. Archit.*, vol. 13, no. 3, pp. 1–173, 2018.

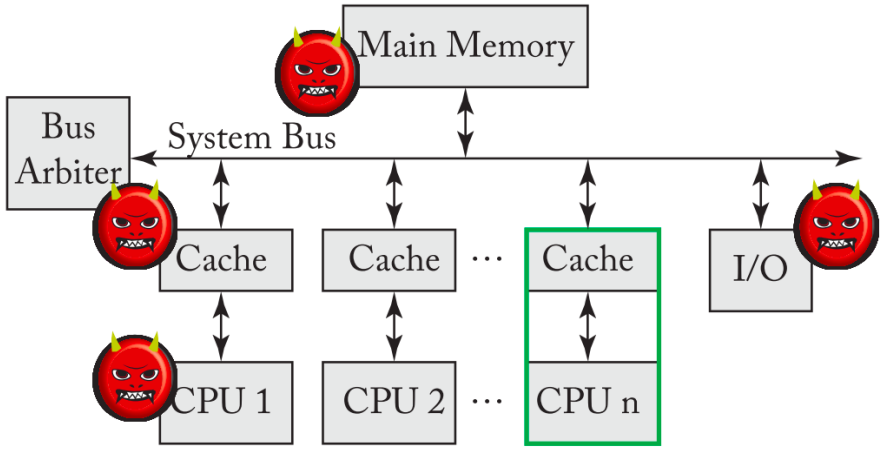
Security Challenges on Multiprocessors

- Multi-socket Multiprocessors required off-chip **communication** between cores
  - Susceptible to similar attacks of memory (e.g., probing, physical interchange with malicious processors chips, …)
  - Inter-processor communication is the main challenge (not passive, as in the memory)
  - Require new solutions (in **communications** and **memory**) to guarantee system **confidentiality**, **integrity** and **authenticity**

- Chip Multiprocessors (CMP) / Many-core Processors / Systems-on-a-chip, additional threats **from inside the chip**
  - Less susceptible than off-chip
  - Might include many IPs (accelerators, ASICs,..)
  - Certain IP can be malicious

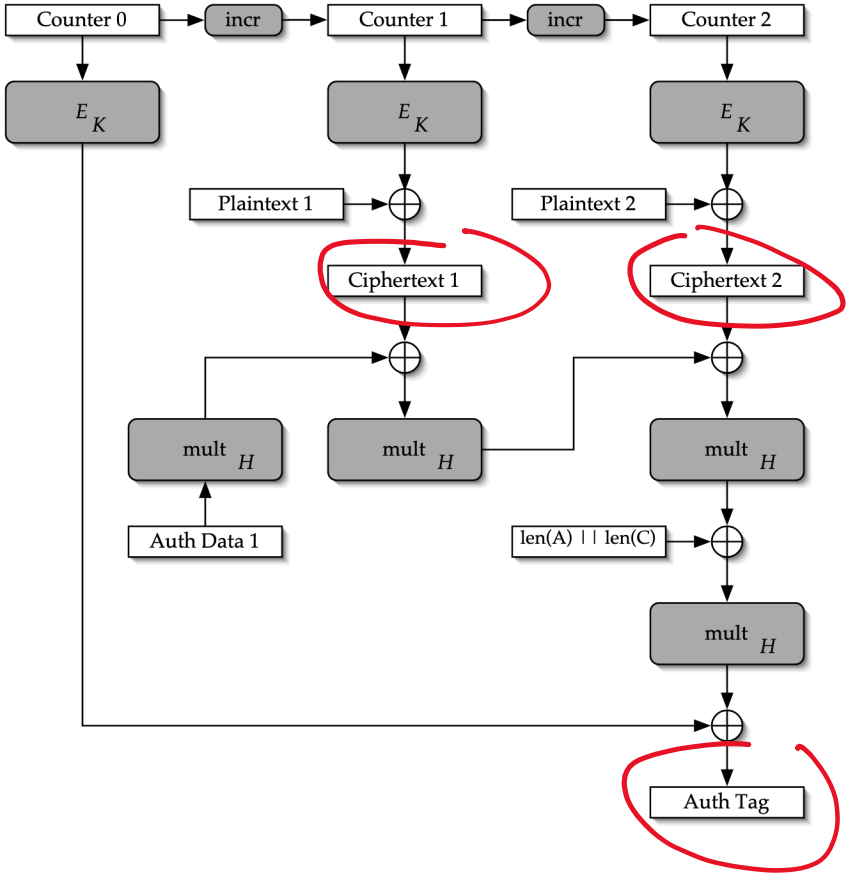
UMA (off-chip) Threat Model: **Communications**

- Confidentiality and Integrity of Communications
  - For confidentiality **AES CTR** for performance reasons (like memory)
    - Challenge to track the counters:
      - pair origin destiny → higher storage requirements
      - shared counters → increases (coherence) protocol complexity
  - For integrity use **MAC** (Message Authentication Codes)
    - Combined with encryption: AES Galois/Counter Mode (**AES GCM**)

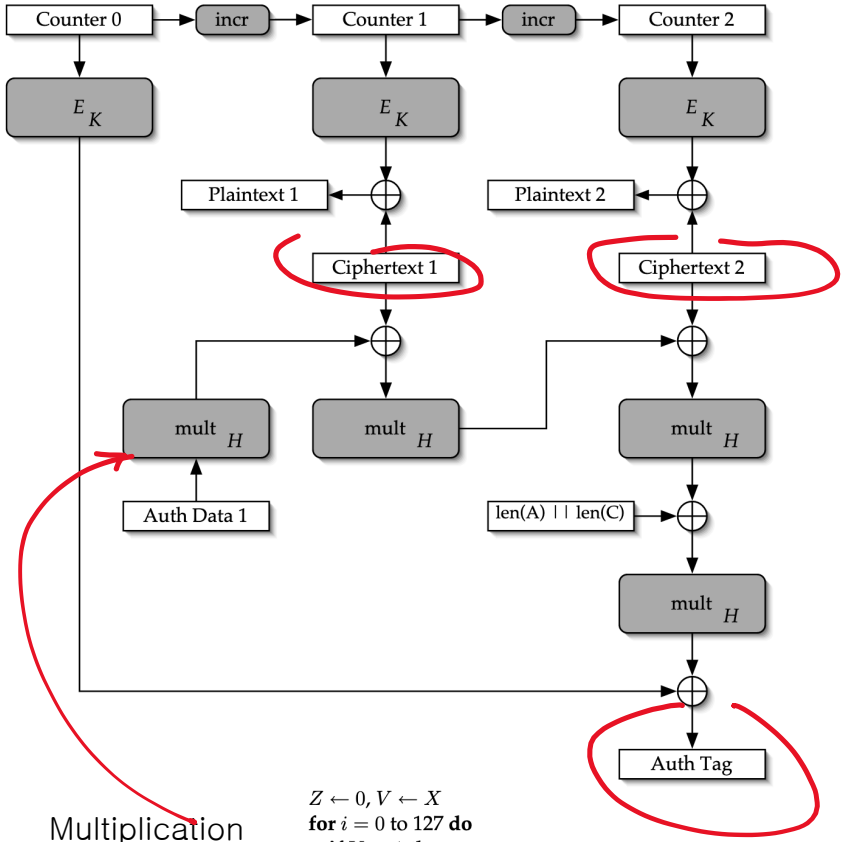


Aside: AES GCM

- Encrypt.



Decrypt



Multiplication in Galois Field $GF(2^{128})$

$Z \leftarrow 0, V \leftarrow X$
for $i = 0$ to 127 **do**
  if $Y_i = 1$ **then**
    $Z \leftarrow Z \oplus V$
  end if
  if $V_{127} = 0$ **then**
    $V \leftarrow \text{rightshift}(V)$
  else
    $V \leftarrow \text{rightshift}(V) \oplus R$
  end if
end for
return Z

UMA Threat Model: **Memory**

- Integrity
  - Use (Bonsai) Merkle Trees for integrity
  - Single Tree
    - Processor's "share" the memory: they has to reach a consensus about the root tree value (which is in some processor)
      - **Bus simplifies the problem**: snooping can allow to authenticate messages and update the root accordingly
  - Multiple Tree
    - Each processor handles a tree of the data is working with (data privately handled by other processors appears as null leaves)
    - Exclude shared memory from integrity checks (not in any tree)
      - Copy it in private regions where integrity is performed
- Pattern Access protection
  - Pattern attacks are much easier to do with SMT (if assumed malicious). Not needed physical probing
  - Independent ORAM per hardware context (more memory operations)

UMA Thread Model: **Key Management**

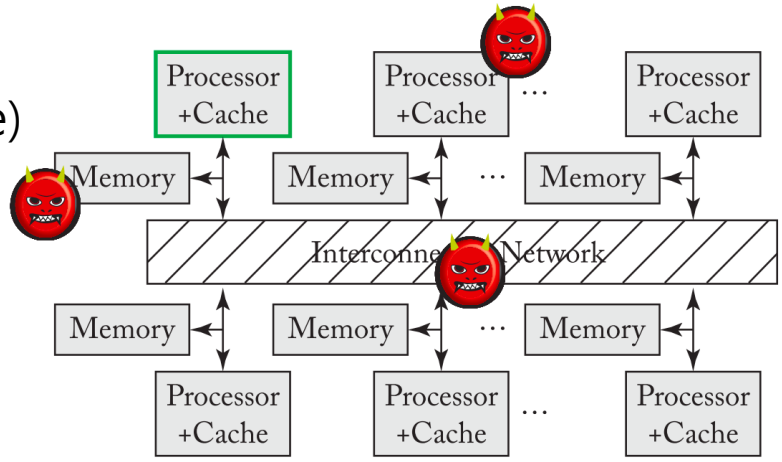
- Each processor has to have its **own K_r**
  - Other processors should be informed of the legitimacy of such keys
  - But they have to use a common key encrypt comms. and memory (**it's an open problem**)

- **Solution 1:** Generate a key at boot time a shared key? (pre-install)
  - Susceptible of being attacked at boot time

- **Solution 2:** Public key cryptography
  - Can be too expensive (in hardware terms) just for sharing keys in a system
  - Doing it at system first boot or when a processor is added to the system?
  - A malicious attacker can insert a rogue processor just to steal the key

NUMA (off-chip) **Threat Model**

- None of the processors have a global vision of what is happening in the system
- **Confidentiality and Integrity**
  - Processor-to-processor comms AES (conf.) or AES GCM (conf.+integr.)
  - Need to be accommodate within the coherency protocol (prohibitive to cypher and decipher all messages)
    - Coherence protocol can check if messages "follows" a normal path (e.g., detect if a rogue processor is trying to stole a cache block)
  - AES might be too Slow ?(0.25B per cycle) in a 25 GB/s network (QPI) 50GB/s (IF)
    - Try to encrypt only "important" things?
    - Faster ciphers?

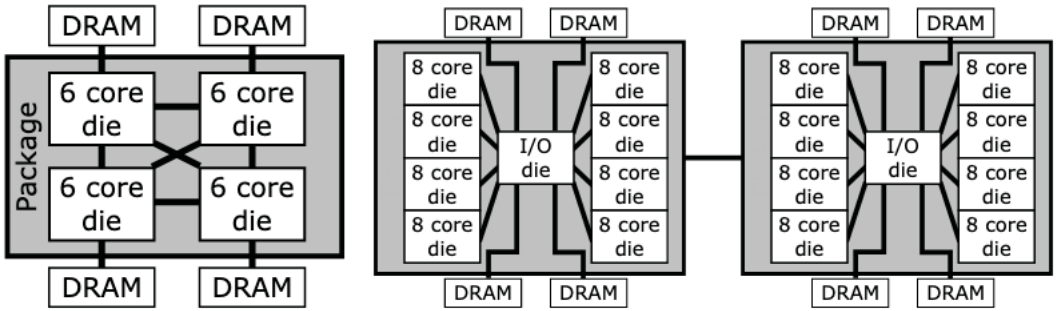


NUMA Threat Model (cont…)

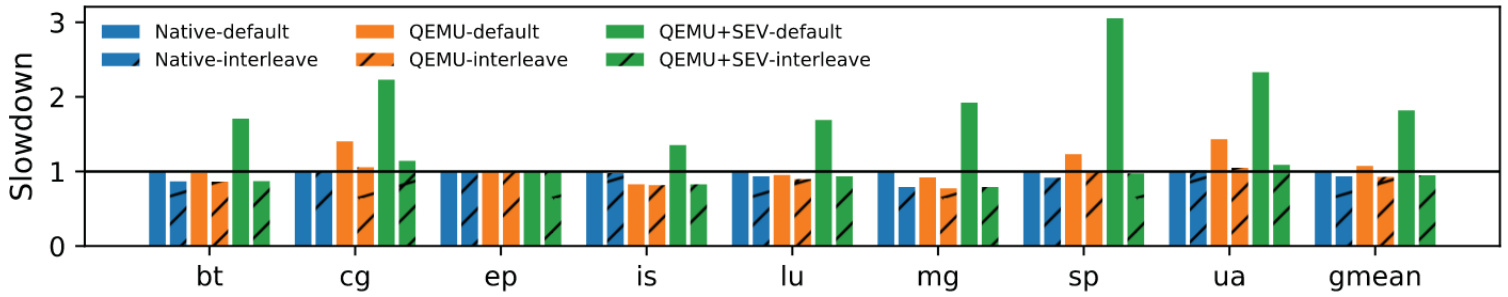
- Access Pattern Protection
  - Easier to obfuscate than UMA (not all processors see all memory requests)
- Key Management
  - Preinstall or use PKI to authenticate others.
  - Pair-wise keys can be used (safer with point-to-point links)
  - Still an attacker can extract the keys using a malicious processor
- Memory encryption
  - Ephemeral key at memory controllers
  - Once a page is allocated, **can't migrate to other memory controller**

Example: Case of AMD-V

- Once a page is created, can't move to another memory controller
- Initial memory emplacement is critical
  - Linux kernel NUMA memory police API
    - `set_mempolicy, get_mempolicy, mbind, sys_set_mempolicy_home_node`

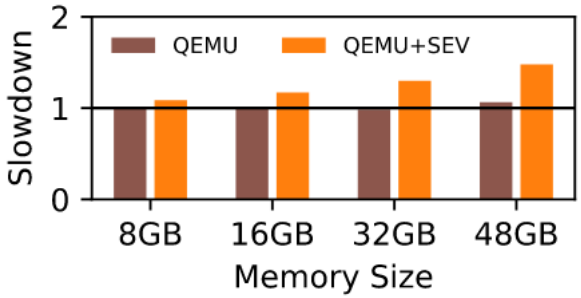


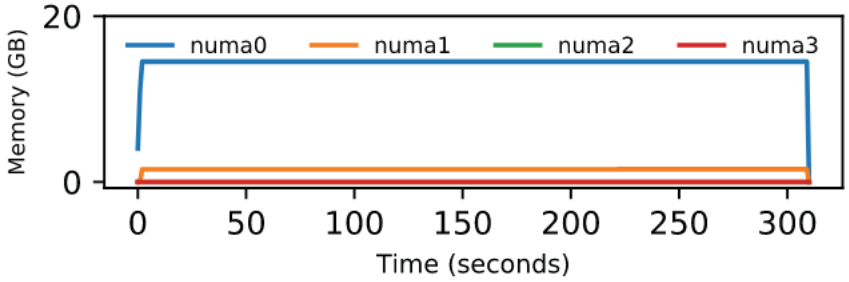
(a) AMD EPYC 7401P (Naples) (b) AMD EPYC 7702 (Rome)



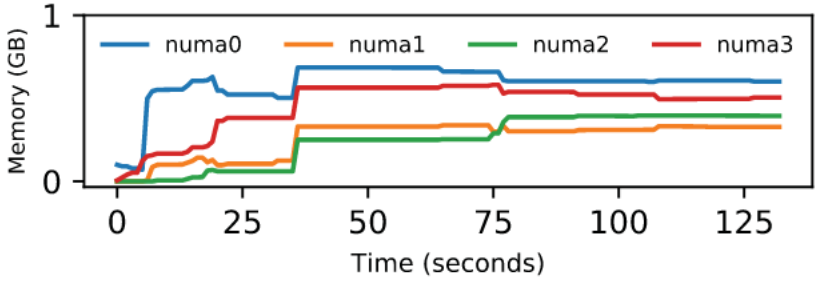
Example: AMD-V (cont.)

- Memory is allocated at boot time (VM)
  - No fancy things with memory (e.g., ballooning, page migration, etc…)





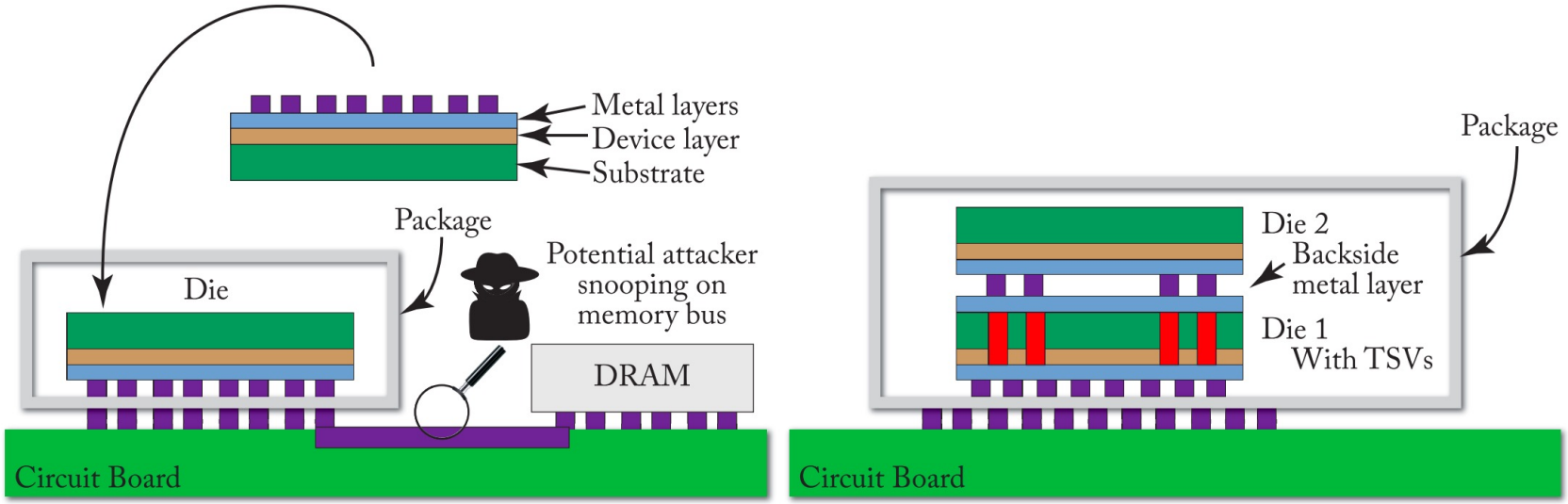
(a) SEV Default Allocation



(b) No SEV Default Allocation

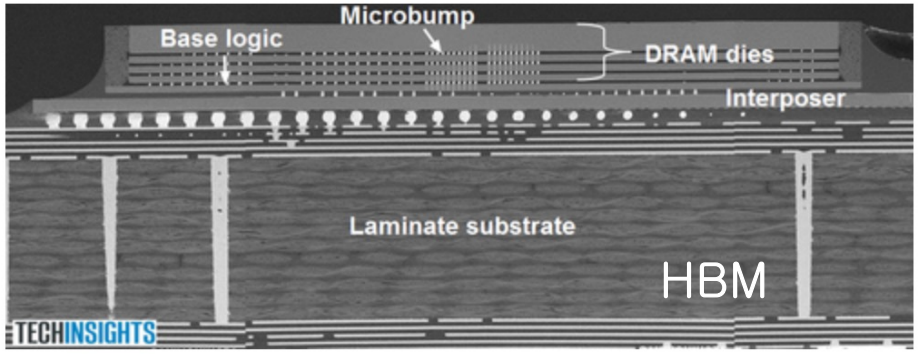
3D Integration Considerations

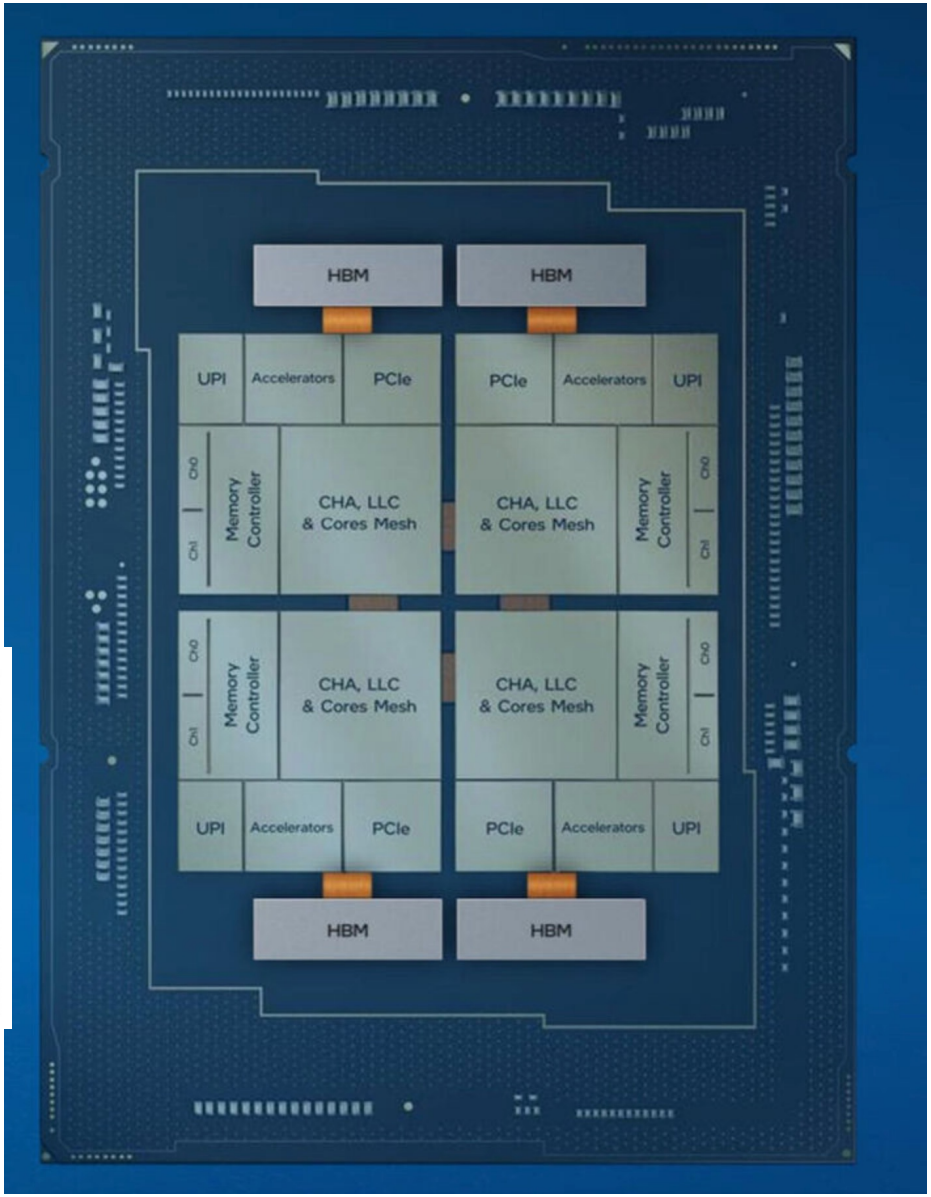
- Higher integration make less easy to access to the wires
  - Might need to change the threat model



Alder Lake SP (12th gen Core / 4th gen Xeon Max)

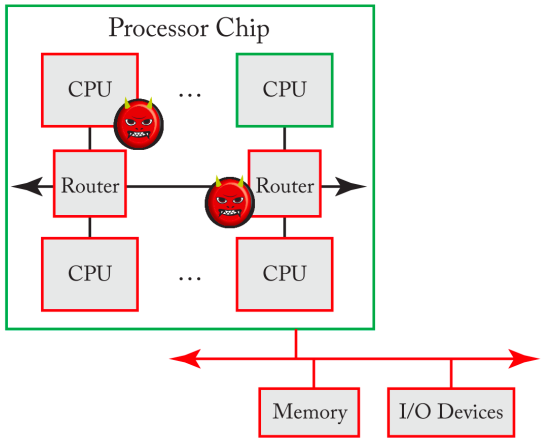
- Naming
  - S → Desktop
  - P → Laptop
  - PS→ IoT
  - SP → Server
  - Sapphire Rapids





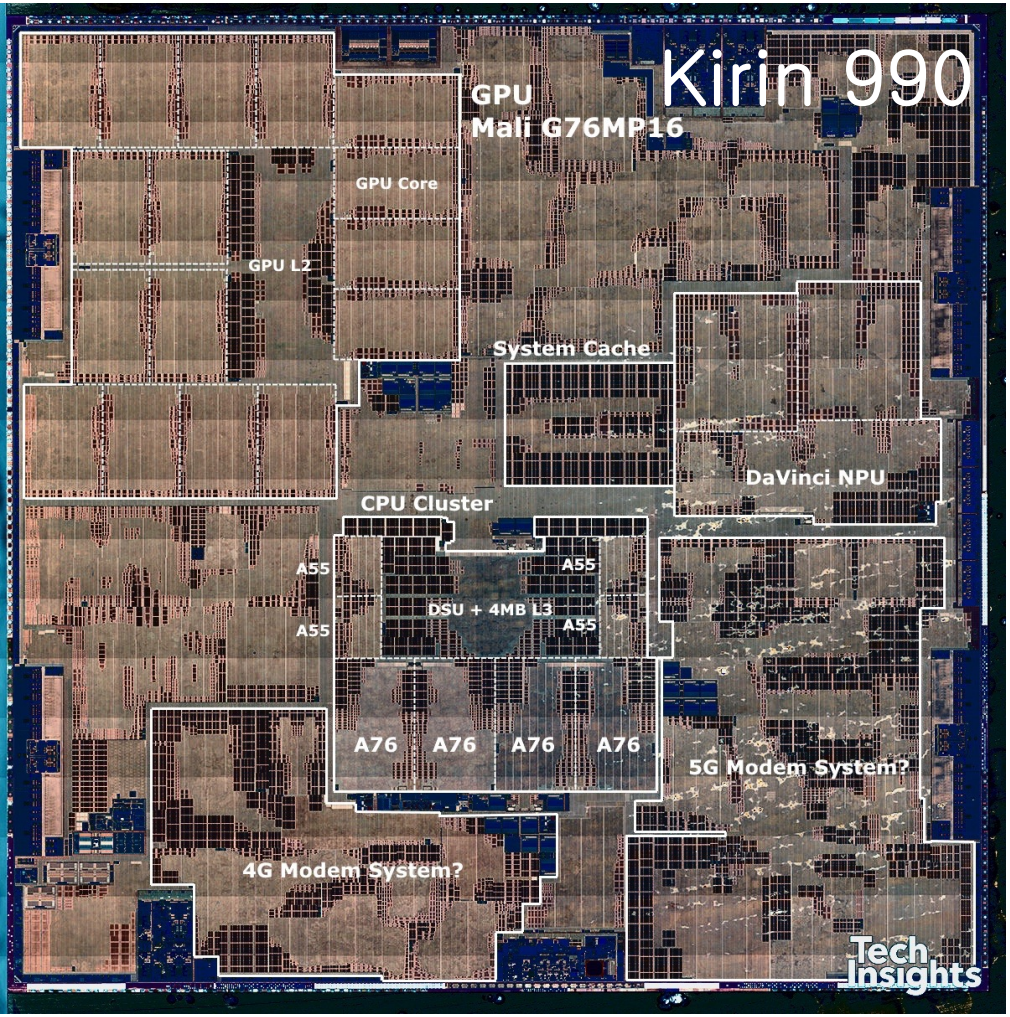
Threat Model for CMP + SoC (MPSoCs)

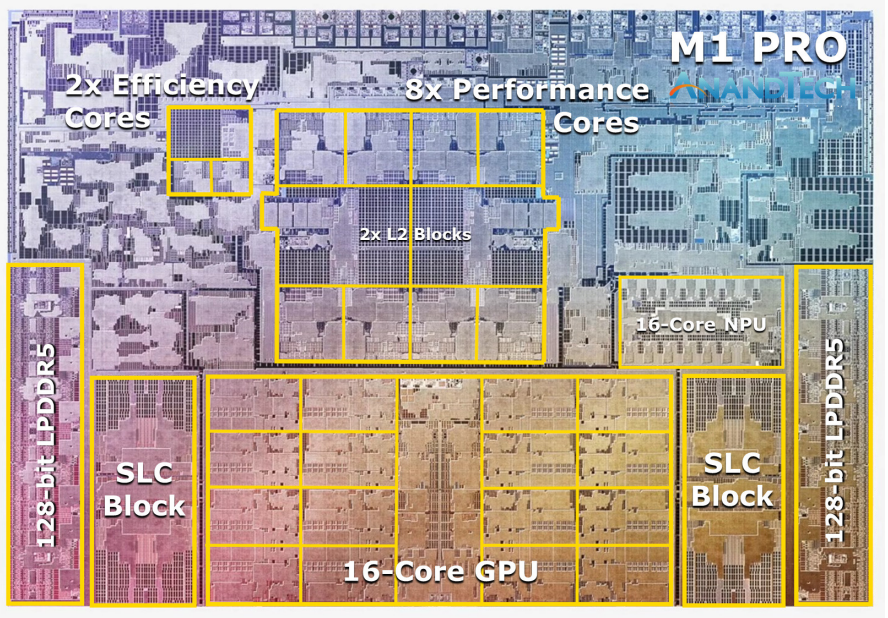
- Usually, many IP inside a single chip: increased risk of having a malicious component (both in the supply chain and manufacture process)
- Processors or Accelerators can tamper with memory
- Routers can tamper packets (i.e., tamper memory)
- NoC wires are more easily detectable to external probing (i.e., packets can be accessed or modified)



Aside: SoC

- Solution to Dark silicon: use specialized hardware for key operations





Aside: CMP or SOC?

- Core i9-12900K

Communication Protection Mechanisms

- Packets are broken down into flits->phits
- Phits should move fast ("1-cycle" from router to router): no room for encryption
  - Optimized (big) AES 0.25 bytes/cycle, lightweight cyphers 1 byte/cycle while phits are 4bytes
- Solutions
  - Combine network coding into the cyphering strategy
  - Injection time
- Memory encryption integrity might be easier in single socket systems