1 Notations about Eviction Set

Let $ES = \{ES(0), ..., ES(m-1)\}$ to denote an eviction set of m consecutive cache sets. Under our assumption that the attacker wants to monitor an entire page, $m = \frac{page\_size}{cache\_line\_size}$. Each cache set in ES is composed of n cache lines: $ES(s) = \{ES(s,0), ..., ES(s,n-1)\}$. For a given l, all $ES(s,l)$s are located in the same page. The warmup section is composed of the extra $m \times (n-a)$ cache lines, these accesses are used to manipulate the replacement state of the cache, making the final cache state after priming more predictable. For simplicity, we use the higher a lines $(ES(i,n-a), ..., ES(i,n-1))$ to denote the occupation section and use upper lines $(ES(i,0), ..., ES(i,n-a-1))$ to denote warmup section.

2 Pseudo-code Algorithms

Algorithm 1 Occupancy Profiling for cache set s

$ES \leftarrow$ Eviction set of size $m \times n$
for all Cache line $(s', l') \in ES$ **do**
  loop
    Flush ES from CPU cache.
    Prime set s by accessing all cache lines in $ES(s)$
    Access cache line $ES(s', l')$. Determine hit or miss.
  end loop
  Compute hit rate of $ES(s', l')$.
end for

Algorithm 2 Replacement Policy Profiling for Cache Set s

$ES \leftarrow$ Eviction set of size $m \times n$
for all Cache line $l \in ES(s)$ **do**
  loop
    Flush ES from CPU cache.
    Prime set s by accessing all cache lines in $ES(s)$
    Access cache line $ES(s, l)$. Determine hit or miss.
    Flush ES from CPU cache.
    Prime set s by accessing all cache lines in $ES(s)$
    Access memory $m \notin ES$ that's also indexed to s'
    Access cache line $ES(s, l)$. Determine hit or miss.
  end loop
  Compute miss rate increase of $ES(s, l)$.
end for

Algorithm 3 Probe Sequence Generation

$ES \leftarrow$ Eviction set of size $m \times n$
$a \leftarrow$ Cache associativity.
$seq_{probe} \leftarrow ()$.
for all Cache set $s \in ES$ **do**
  for all Cache line $l \in \{n-a, ..., n-1\}$ **do**
    if Accessing seq_{probe} does not prefetch $ES(s, l)$ **then**
      $seq_{probe} += ES(s, l)$
      break
    end if
  end for
end for

Algorithm 4 Prime Sequence Generation

$ES \leftarrow$ Eviction set of size $m \times n$
$a \leftarrow$ Cache associativity.
$seq_{probe} \leftarrow$ Probe sequence.
$Occ \leftarrow$ Occupation matrix.
$Rpl \leftarrow$ Replacement matrix.
$seq_{prime} \leftarrow ()$
for all Cache set $s \in ES$ **do**
  for $0 <= l < n$ **do**
    if $Rpl(s, l) = 1$ **then**
      $seq_{prime} += ES(s, seq_{probe}(s))$
    else if $Occ(s, l) = 1$ **then**
      $seq_{prime} +=$ An unused cache line in $ES(s, n-a), ..., ES(s, n-1)$, except $ES(s, seq_{probe}(s))$
    else
      $seq_{prime} +=$ An unused cache line in $ES(s, 0), ..., ES(s, n-a-1)$
    end if
  end for
end for