[37] V. Varadarajan et al. Scheduler-based defenses against cross-vm side-channels. In *Proc. USENIX Security*, 2014.
[38] B. C. Vattikonda et al. Eliminating fine grained timers in xen. In *Proc. ACM CCSW*, 2011.
[39] J. Wang et al. Hypercheck: A hardware-assisted integrity monitor. In *Recent Advances in Intrusion Detection*, 2010.
[40] Z. Wang and R. B. Lee. A novel cache architecture with enhanced performance and security. In *Proc. Micro*, 2008.
[41] Y. Yarom and K. Falkner. Flush+reload: A high resolution, low noise, l3 cache side-channel attack. In *Proc. USENIX Security*, 2014.
[42] Y. Zhang et al. Cross-vm side channels and their use to extract private keys. In *Proc. ACM CCS*, 2012.
[43] Y. Zhang et al. Incentive compatible moving target defense against vm-colocation attacks in clouds. In *Information Security and Privacy Research*, pages 388–399. Springer, 2012.
[44] Y. Zhang et al. Cross-tenant side-channel attacks in paas clouds. In *Proc. ACM CCS*, 2014.
[45] Y. Zhang and M. K. Reiter. Duppel: Retrofitting commodity operating systems to mitigate cache side channels in the cloud. In *Proc. ACM CCS*, 2013.
[46] J. Zheng et al. Workload-aware live storage migration for clouds. In *Proc. ACM VEE*, 2011.

APPENDIX

A ILP Formulation

In this section, we describe the ILP formulation for the optimization problem that the *Nomad Placement Algorithm* needs to solve every epoch. As discussed earlier, the algorithm takes as input the VM placement over the last sliding window of epochs and other parameters that determine the information leakage. The objective function is to minimize the total information leakage across all client pairs.

We use binary indicator variables, $d_{c,i,k}(\mathrm{t})$, to denote the placement assignment of a specific VM instance i of a client c at machine k in the epoch t. These are the key control variables that we need to set to determine the optimal placements. Then, it naturally follows that the inputs to the ILP formulation are these control variables for the past Δ epochs to be used in summarizing co-residency over time. The output would be the placement assignments for the next epoch.

Equation 9 in Figure 10 states that the total number of VMs assigned to a machine should not exceed machine's capacity. Furthermore, each VM should only have one machine assignment (Equation 10). Equation 11 models a per-epoch binary co-residence relationship that indicates if 2 VMs are co-resident in a machine k at epoch t. Intuitively, this is like modeling an AND operation; i.e., if both VMs have the same machine assignment k, this binary indicator is set to 1. If there exists any machine in which two VMs are co-resident, then we flag that these two VMs ($VM_{c,i}$ and $VM_{c',i'}$) are co-resident at an epoch t (Equation 12).

Having determined whether a given pair of VMs are co-resident, we can summarize the information leakage across the three dimensions, namely over time, over adversary's VMs, and over client's VMs. This models the *InfoLeakage* between a client pair c and c'. For brevity, the figure only shows the model for the $\langle R, C\rangle$ case, the summarization involves summation over adversary's and client's VMs.

Minimize $\sum_{c,c'} InfoLeak^{\langle R,C\rangle}_{c\to c'}(t,\Delta)$ such that

$$\forall k,t,\ \sum_{c}\sum_{i} d_{c,i,k}(\mathrm{t}) \le\ Cap_k \tag{9}$$

$$\forall c,i,k,t \in lifetime_{c,i},\ \sum_{k} d_{c,i,k}(\mathrm{t}) =\ 1 \tag{10}$$

$$\begin{aligned}&\forall c,i,c',i',c\neq c',k,t\\ &CoRes_{c,i,c',i',k}(\mathrm{t}) \ge d_{c,i,k}(\mathrm{t}) + d_{c',i',k}(\mathrm{t}) \ -\ 1\\ &CoRes_{c,i,c',i',k}(\mathrm{t}) \le d_{c,i,k}(\mathrm{t})\\ &CoRes_{c,i,c',i',k}(\mathrm{t}) \le\ d_{c',i',k}(\mathrm{t})\end{aligned} \tag{11}$$

$$\begin{aligned}&\forall c,i,c',i',c\neq c',t,\\ &CoRes_{c,i,c',i'}(\mathrm{t}) \le \sum_{k} CoRes_{c,i,c',i',k}(\mathrm{t})\end{aligned}$$

$$CoRes_{c,i,c',i'}(\mathrm{t}) \ge CoRes_{c,i,c',i',k}(\mathrm{t}) \tag{12}$$

$$\forall c,c',c\neq c',t,\ InfoLeak^{\langle R,C\rangle}_{c\to c'}(t,\Delta) = \sum_{i}\sum_{i'}\sum_{t'\in[\mathrm{t}\text{-}\Delta,\mathrm{t}]} CoRes_{c,i,c',i'}(\mathrm{t}') \tag{13}$$

$$\begin{aligned}&\forall c,i,k,t,\ mv_{c,i,m}(\mathrm{t}) = d_{c,i,k}(\mathrm{t}) \oplus d_{c,i,k}(\mathrm{t}-1)\\ &mv_{c,i,m}(\mathrm{t}) \le\ d_{c,i,k}(\mathrm{t}) + d_{c,i,k}(\mathrm{t}-1)\\ &mv_{c,i,m}(\mathrm{t}) \ge d_{c,i,k}(\mathrm{t}) - d_{c,i,k}(\mathrm{t}-1)\\ &mv_{c,i,m}(\mathrm{t}) \le 2 - d_{c,i,k}(\mathrm{t}) - d_{c,i,k}(\mathrm{t}-1)\end{aligned}$$

$$\forall c,i,t,\ mv_{c,i}(\mathrm{t}) = 0.5\ \sum_{k} mv_{c,i,m}(\mathrm{t}) \tag{14}$$

$$\forall c,t,\ GlobalMigCost(\mathrm{t}) = \sum_{c,i} mv_{c,i}(\mathrm{t}') \tag{15}$$

$$\forall c,t,\ GlobalMigCost(\mathrm{t}) \le PercentBudget \sum_{c,i,k} d_{c,i,k}(\mathrm{t}) \tag{16}$$

Figure 10: $\langle R, C\rangle$: ILP formulation for *Nomad Placement Algorithm*

The ILP formulation is also applicable for three other deployment models; Equation 13 needs to be changed to reflect the correct *InfoLeakage* model.

Now that we have modeled the co-residency across client pairs, the only remaining factor is the migration cost. To this end, we introduce another binary indicator that indicates whether a VM has migrated from a previous epoch $t-1$. $mv_{c,i,m}(\mathrm{t})$ is a binary variable that indicates whether $VM_{c,i}$ was either in machine k in the previous epoch and is no longer in machine k or vice versa. This gives an indicator that tells whether a VM has moved away or moved into this machine k. Intuitively, this is like modeling a XOR operation (Equation 14). Summing over this variable for each VM instance givesi the total number of migrations from the previous epoch (Equation 15). Lastly, Equation 16 models the migration budget such that the total number of migrations should not exceed *PercentBudget* of the total workloads (i.e., total number of VMs that currently have place assignments).