Privacy-Preserving AIS for Network Security

Karthik S
Chandan Yeshwanth

March 22, 2016

Recap

- Defined secret sharing scheme to be used (Bitwise Additive secret sharing over Z_2)
- Explored privacy-preserving random detector set generation

LISYS Overview

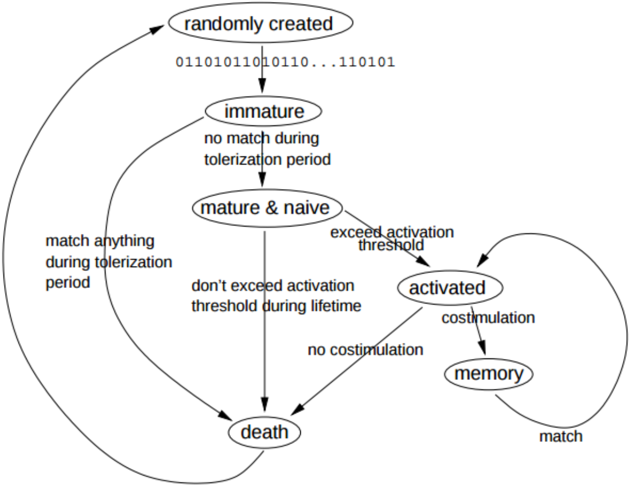

Figure : Detector lifecycle[2]

Outline

- Affinity function : privacy-preserving r contiguous bits (R-CONT)
- Tolerization/Maturation of detectors using R-CONT

Notation

- Detector indexed by i denoted by d_i
  - Bit representation of $d_i \rightarrow d_{i1} d_{i2} \ldots d_{il}$
  - Party k's share of $d_{ij} \rightarrow [d_{ij}]^k$
- Connection c
  - Bit representation of $c \rightarrow c_1 c_2 \ldots c_l$
  - Party k's share of $c_j \rightarrow [c_j]^k$

Affinity Function

- Given a secret shared detector $[d]$ and a secret shared connection $[c]$, the affinity function works as follows:

$$A_f([c],[d]) = [z]$$

  where z is the affinity value between d, c
- Affinity function must also be privacy preserving.

- Two bit strings d, c (both of equal length) match if a and c have atleast r contiguous bits in common (in the same positions)
- Example:

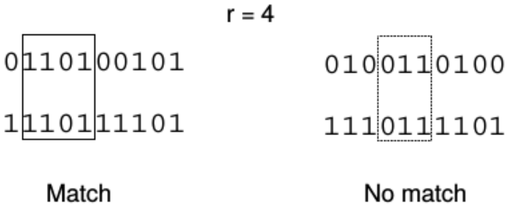

Figure : Matching under the contiguous bit match rule[2]

PP Invert Bit

- For a party P_i

Algorithm 1 INVERT-BIT($[b]^i$)

```
if i = n ∧ n is even then
    return [b]^i
else
    return ¬[b]^i
end if
```

INVERT-BIT

PP r-Contiguous Bits

Algorithm 2 R-CONT($[c], [d_i]$)

```
A ← Array of size l − r + 1
for f = r to l do
    B ← Array of size r
    for w = f − r + 1 to f do
        [z_w] = [c_w] − [d_iw]
        B[w − f + r] = INVERT-BIT([z_w])
    end for
    [v_f] = ∧_{h=1}^{r} B[h]
    A[f − r + 1] = INVERT-BIT([v_f])
end for
return ∧_{h=1}^{l−r+1} A[h]
```

R-CONT

AND (product) of shares using multiplication protocol described in [1]

Tolerization

- Detectors which match with safe connections must be removed
- In PP version, parties remove their share of the matching detector
- Match determined through R-CONT

Tolerization : Algorithm

Algorithm 3 TOLERIZE($[D]$)

On receiving connection c
    Share c in bitwise additive fashion
for $i = 1$ to q **do**
    $[u_i] = R - CONT([c], [d_i])$
    $u_i =$ RECONSTRUCT($[u_i]$)
    if u_i **then**
        remove ($[d_i]$)
    end if
end for

Next Steps

- Intrusion detection using trained detector set
- Correctness, security and performance analysis

Ronald Cramer, Ivan Damgaard, and Ueli Maurer.
General secure multi-party computation from any linear secret-sharing scheme.
In *Advances in Cryptology-EUROCRYPT 2000*, pages 316–334. Springer, 2000.

Steven A Hofmeyr and Stephanie Forrest.
Architecture for an artificial immune system.
Evolutionary computation, 8(4):443–473, 2000.