QoP-ML processing algorithms - Semantics

Table 1. The parameters and variables for the QoP-ML processing algorithms

SET	- make a choice indication
$READ$	- reading indication
$DROP$	- operator dropping the communication operation
$RETURN$	- processing functions will be ended
$PASS$	- processing function will go on (used to show particular cases)
$BREAK$	- processing loop will be interrupted
$CALL$	- algorithm execution
num	- the number of the **version** of the protocol
rep	- the number of multiples hosts defined in the particular of the protocol
H	- the number of high hierarchy processes (**host**) taking part in the specific version of the protocol (including host replication)
h	- the index of current high hierarchy process
p	- the index of current process in current host
o	- the index of current operation in current process p and current host h
H_{max}	- the number of all hosts
P_{max}	- the number of all processes in current host **h**
H_{now}	- temporary variable which holds the index of current host when checking FINISHED status
P_{now}	- temporary variable which holds the index of current process when checking FINISHED status of hosts
$P[h]$	- the process **P** of the host **h**
$O[h][P[h]]$	- index of the operation **Op** for the host **h** and for the process **P** in the host **h**
$Op[h][p][o]$	- the currently processed operation
$O_{\#}[h]$	- the set of operations which were executed before **host** process starts - (# operator)

$CH[h]$	- the set of possible communication channels for **host** process
$ALG[h]$	- the schedule algorithm for **host** process
p_{last}	- the index to the last process in current host **h**
o_{last}	- the index to the last operationin current process **p** and current host **h**
$Status[h]$	- the status for the host indexed by **h**; **RUNNING** - host **h** has operations to execute; **FINISH** - this status means that the host indexed by**h** have no more operation to execute
O^{loop}	- the operation which refers to any loop type operator
O^{loop}_{do}	- the operation which refers to **do** operator
O^{loop}_{while}	- the operation which refers to **while** operator
O^{loop}_{break}	- the operation which refers to **break** operator
$PUSH$	- push the index to the stack
$STACK[h][p]$	- the stack of indexes of the **loop** operations of process **p** in host **h**
POP	- pop the operation index from the stack and return it
$CHECK$	- check indication
o_{while}	- the index of nearest **while** operator in current process **p** and host **h**
O^{comm}	- the operation which refers to operator **in** or **out** (communication action)
O^{comm}_{out}	- the operation which refers to **out** operator (communication action)
O^{comm}_{in}	- the operation which refers to **in** operator (communication action)
$CH[h][p][o]$	- the possible channel **CH** for specific communication operation indexed by **o** and process indexed **p** and host indexed by **h**
$CH[h][p][o][buf]$	- the current size of the buffer **buf** of the specific **CH** for specific communication operation indexed by **o** and process indexed **p** and host indexed by **h**
$CH^{def}[h][p][o][buf]$	- the defined size of the buffer **buf** of the specific **CH** for specific communication operation indexed by **o** and process indexed **p** and host indexed by **h**

$FIND$ - operator seeking first operation connected to operation from argument; connected operation is an operation that has different communication action (in, out) and is connected with the same channel; if no linked operation is found, **False** is returned

$EXISTS$ - operator checking if operation exists in table (has been added and not yet removed)

ADD - adds value to a table

$REMOVE$ - removes value from a table

$Waiting^{in}[ch]$ - the table of the operations which refer to the **in** communication action buffered on the channel **ch**

$Waiting^{out}[ch]$ - the table of the operations which refer to the **out** communication action buffered on the channel **ch**

$LINK(Op[h_1][p_1][o_1], Op[h_2][p_2][o_2])$ - the communication connection is established between the operations of different communication action type and indexed by different triples (**h**,**p**,**o**); these operations must refer to the same channel; connection is bidirectional

$GETLINK(Op[h][p][o])$ - returns the operation linked to operation from argument or **False** if no operation is linked

Op^{linked} - variable containing the result of **FIND** and **GETLINK** operators

O^{qop} - the operation which refers to any QOP operation

QOP^{label} - the **qop label** for the current QOP operation

QOP^{fun} - the **qop function** used in the current QOP operation

QOP^{opts} - **qop options** used in the current QOP operation

$QOP^{metrics}[h][fun][opts][label]$ - metrics for function **fun** read from configuration of host **h** specified by options **opts** and label **label**

ω - the parameter specifying multiple metrics used

$QOP^{metrics}[h][fun][opts][label](sec)(\omega)$ - the set of the security metrics for function **fun**, specified for the concrete security attribute **sec** and with multiplication parameter ω

$T^{global}[h]$ - absolute execution time of the protocol measured relative to the host **h**

$HOST$ - the operator returns index of the host of the operation from parameter

$\overline{T}$ - the minimum value of the units of time set for a given protocol

$T^{global}_{in}[h]$ - the absolute execution time of the protocol measured relative to the host **h**, which set the communication connection (**LINK**), containing **in** operation

$\Delta(T^{global}_{out,in})$ - the difference between the execution time of the protocol between the host which proceed operation **out** and host performing operation **in**

end - the operator in QoP-ML which finish the protocol with success

T_{Total} - the total execution time of the protocol

$max(T^{global}[h{=}1], ..., T^{global}[h{=}H])$ - the maximum of the $T^{global}[h]$ for all host taking part in the protocol

Algorithm 1 Init

Require: definition of: functions, equations, channels, security metrics; protocol modelled by QoP-ML; process instantiation.

SET num
READ rep
SET H

for $h = 1$ to H **do**
  READ $O_{\#}[h]$
  READ $CH[h]$
  READ $ALG[h]$
  $h \leftarrow h + 1$
end for

for $h = 1$ to H_{max} **do**
  P[h] $\leftarrow$ 1
  Status[h] $\leftarrow$ RUNNING
  for $p = 1$ to P_{max} **do**
    O[h][p] $\leftarrow$ 1
  end for
end for

h $\leftarrow$ 1
p $\leftarrow$ 1
o $\leftarrow$ 1
CALL Main

Algorithm 2 Main

if $Op[h][p][o] = Op^{comm}$ **then**
  if $GETLINK(Op[h][p][o])! = False$ **then**
    O[h][p] $\leftarrow$ O[h][p] + 1
  else if $Op[h][p][o] = Op_{in}^{comm}$ **then**
    if EXISTS $Op[h][p][o]$ in $Waiting^{in}[CH[h][p][o]]$ **then**
      CALL Next operation
      CALL Main
      RETURN
    end if
  end if
end if
CALL Check protocol state
CALL Execute operation

Algorithm 3 Check protocol state

$H_{now} \leftarrow h$
repeat
  if Status[h] = RUNNING **then**
    BREAK
  end if
  if $h = H^{max}$ **then**
    h ← 1
  else
    h ← h + 1
  end if
until $h = H_{now}$
if $(Status[h] = FINISHED)$ && $(h = H_{now})$ **then**
  CALL Final QoP evaluation
  end
end if
p ← P[h]
$P_{now} \leftarrow$ p
repeat
  if $O[h][p] \leq o_{last}$ **then**
    BREAK
  end if
  if p = P^{max} **then**
    p ← 1
  else
    p ← p + 1
  end if
until p = P_{now}
P[h] ← p
h ← 1
if $(O[h][p] > o_{last})$ && $(p = P_{now})$ **then**
  Status[h] = FINISHED
  if $h = H^{max}$ **then**
    h ← 1
  else
    h ← h + 1
  end if
  CALL Main
  RETURN
end if
o ← O[h][p]

Algorithm 4 Execute operation

```
if Op[h][p][o] = O^comm then
    CALL Communication
end if

if Op[h][p][o] = O^qop then
    for each security attribute do
        CALL QoP evaluation
    end for
end if
CALL Next operation
CALL Main
```

Algorithm 5 Next operation

```
if Op[h][p][o] = Op^loop then
    CALL Loop next operation
else if Op[h][p][o] = Op^comm then
    CALL Communication next operation
else
    o ← o + 1
end if

O[h][p] ← o
if ALG[h] = RR then
    if p < P_max then
        p ← p + 1
    else
        p ← 1
    end if
else if ALG[h] = FIFO then
    PASS
end if
P[h] ← p
if h < H_max then
    h ← h + 1
else
    h ← 1
end if
```

Algorithm 6 Communication

```
if CH[h][p][o]  CH[h] then
    DROP Op[h][p][o]
    RETURN
end if

CALL Link
```

Algorithm 7 Link

```
1: if Op[h][p][o] = O_out^comm then
2:    if (CH[h][p][o][buf] = 0 && CH^def[h][p][o][buf] > 0) then
3:       DROP Op[h][p][o]
4:       RETURN
5:    end if
6:    if CH^def[h][p][o][buf] = 0 then
7:       Op^linked ← FIND for Op[h][p][o] in Waiting^in[CH[h][p][o]]
8:       if Op^linked = False then
9:          DROP Op[h][p][o]
10:      else
11:         REMOVE Op^linked from Waiting^in[CH[h][p][o]]
12:         LINK(Op[h][p][o], Op^linked)
13:      end if
14:   else
15:      Op^linked ← FIND for Op[h][p][o] in Waiting^in[CH[h][p][o]]
16:      if Op^linked = False then
17:         ADD Op[h][p][o] to Waiting^out[CH[h][p][o]]
18:         CH[h][p][o][buf] ← CH[h][p][o][buf] − 1
19:      else
20:         REMOVE Op^linked from Waiting^in[CH[h][p][o]]
21:         LINK(Op[h][p][o], Op^linked)
22:      end if
23:   end if
24: else if Op[h][p][o] = O_in^comm then
25:   if CH^def[h][p][o][buf] = 0 then
26:      ADD Op[h][p][o] to Waiting^in[CH[h][p][o]]
27:   else
28:      Op^linked ← FIND for Op[h][p][o] in Waiting^out[CH[h][p][o]]
29:      if Op^linked = False then
30:         ADD Op[h][p][o] to Waiting^in[CH[h][p][o]]
31:      else
32:         REMOVE Op^linked from Waiting^out[CH[h][p][o]]
33:         CH[h][p][o][buf] ← CH[h][p][o][buf] + 1
34:         LINK(Op[h][p][o], Op^linked)
35:      end if
36:   end if
37: end if
```

Algorithm 8 QoP evaluation (availability)

$T^{global}[h] = T^{global}[h] + QOP^{metrics}[h][QOP^{fun}][QOP^{opts}][QOP^{label}](availability)(\omega)$

$Op^{linked} = GETLINK(Op[h][p][o])$
if (Op^{linked} != False) && ($Op[h][p][o] = O^{comm}_{out}$) **then**
    $\overline{T}^{global}_{in}[h] = \frac{T^{global}_{in}[HOST\ Op^{linked}]}{\overline{T}}$
    $\overline{T}^{global}_{out}[h][= \frac{T^{global}_{out}[h]}{\overline{T}}$
    $\Delta(T^{global}_{out,in}) = \overline{T}^{global}_{out}[h] - \overline{T}^{global}_{in}[HOST\ Op^{linked}]$

    if $\Delta(T^{global}_{out,in}) > 0$ **then**
        $T^{global}[HOST\ Op^{linked}] = T^{global}_{in}[h] + \Delta(T^{global}_{out,in}) * \overline{T}$
    else if $\Delta(T^{global}_{out,in}) < 0$ **then**
        $T^{global}[h] = T^{global}_{out}[h] + |\Delta(T^{global}_{out,in})| * \overline{T}$
    else if $\Delta(T^{global}_{out,in}) = 0$ **then**
        PASS
    end if
end if

Algorithm 9 Final QoP evaluation (availability)

$T_{Total} = max(T^{global}[h = 1], ..., T^{global}[h = H])$

Algorithm 10 Loop next operation

if $Op[h][p][o] = O^{loop}_{do}$ **then**
    PUSH $o + 1$ to **STACK**[h][p]
    $o \leftarrow o + 1$
end if
if $Op[h][p][o] = O^{loop}_{while}$ **then**
    if $Op[h][p][o]$ = True **then**
        o ← **POP STACK**[h][p]
        PUSH o to **STACK**[h][p]
    else if $Op[h][p][o]$ = False **then**
        POP STACK[h][p]
        $o \leftarrow o + 1$
    end if
end if
if $Op[h][p][o] = O^{loop}_{break}$ **then**
    POP STACK[h][p]
    $o \leftarrow o_{while} + 1$
end if

Algorithm 11 Communication next operation

1: **if** $Op[h][p][o] = O_{out}^{comm}$ **then**
2: $o \leftarrow o + 1$
3: **end if**
4: **if** $Op[h][p][o] = O_{in}^{comm}$ **then**
5: $Op^{linked} = GETLINK(Op[h][p][o])$
6: **if** Op^{linked} != False **then**
7: $o \leftarrow o + 1$
8: **end if**
9: **end if**