We have formalized NDN access control using CSP [1]. Due to the space limit, we only present four models in [1], and the improved models are illustrated in this appendix. In addition, auxiliary definitions are also listed here.

[1] Fei, Y., Zhu, H.: Modeling and Verifying NDN Access Control Using CSP (submitted to ICFEM2018)

Appendix

In order to describe the improved models, we update the definition in message MSG_{pro} and MSG_{dat2} in [1] as below.

$$
\begin{aligned}
MSG_{dat2} = \; & \{msg_{dat}.a.b.n.H(k).E(k, E(k_1, c_1)), msg_{dat}.a.b.n.H(k).E(k, E(k_1, c_1)).E(k_2, c_2) \mid \\
& a, b \in \mathit{Entity}, n \in \mathit{Name}, k, k_1, k_2 \in \mathit{Key}, c_1, c_2 \in \mathit{Content}\} \\
MSG_{pro} = \; & \{msg_{pro}.E(k, E(k_1, c_1)).k_2.k_3, msg_{pro}.E(k, E(k_1, c_1)).E(k_2, c_2).k_3.k_4.k_5 \mid \\
& k, k_1, k_2, k_3, k_4, k_5 \in \mathit{Key}, c_1, c_2 \in \mathit{Content}\}
\end{aligned}
$$

Table 1 represents the new involved constants and variables.

Table 1. New involved constants and variables

Constants	K_I(public key of intruder), K_M(public key of ACM), K_A(public key of CA), K_I^{-1}(private key of intruder), K_M^{-1}(private key of ACM), K_A^{-1}(private key of CA)
Variables	k_m,k_{m1},k_{m2},k_{m3}(public key of ACM), k_m^{-1},k_{m1}^{-1},k_{m2}^{-1}(private key of ACM), k_a,k_{a1},k_{a2}(public key of CA), k_a^{-1},k_{a1}^{-1}(private key of CA),

Here we give sixteen improved models as below.

$$
\begin{aligned}
\mathit{SystemR\_Sig} =_{df} \; & \mathit{READER\_Sig}(R, M, K, K_M, NN, ND, NDK)[|\mathit{PROCESS\_PATH}|]\mathit{PROCESS} \\
& [|\mathit{COM\_PATH}|]\mathit{ACM\_R\_Sig}(R, M, K_M, DK, DATA, NK\_e, NK\_d, HL) \\
\mathit{SystemW\_Sig} =_{df} \; & \mathit{WRITER\_Sig}(W, M, K, K_M, NN, ND, DK, DATA)[|\mathit{PROCESS\_PATH}|]\mathit{PROCESS} \\
& [|\mathit{COM\_PATH}|]\mathit{ACM\_W\_Sig}(W, M, HL, NK\_e, NK\_d, NDK) \\
\mathit{SystemR\_Sig}_I =_{df} \; & \mathit{SystemR\_Sig}[|\mathit{INTRUDER\_PATH}|]\mathit{INTUDER\_Sig} \\
\mathit{SystemW\_Sig}_I =_{df} \; & \mathit{SystemW\_Sig}[|\mathit{INTRUDER\_PATH}|]\mathit{INTUDER\_Sig} \\
\mathit{SystemR\_Sig\_C} =_{df} \; & \mathit{READER\_Sig}(R, M, K, K_I, NN, ND, NDK)[|\mathit{PROCESS\_PATH}|]\mathit{PROCESS} \\
& [|\mathit{COM\_PATH}|]\mathit{ACM\_R\_Sig}(R, M, K_M, DK, DATA, NK\_e, NK\_d, HL) \\
\mathit{SystemW\_Sig\_C} =_{df} \; & \mathit{WRITER\_Sig}(W, M, K, K_I, NN, ND, DK, DATA)[|\mathit{PROCESS\_PATH}|]\mathit{PROCESS} \\
& [|\mathit{COM\_PATH}|]\mathit{ACM\_W\_Sig}(W, M, HL, NK\_e, NK\_d, NDK) \\
\mathit{SystemR\_Sig\_C}_I =_{df} \; & \mathit{SystemR\_Sig\_C}[|\mathit{INTRUDER\_PATH}|]\mathit{INTUDER\_Sig} \\
\mathit{SystemW\_Sig\_C}_I =_{df} \; & \mathit{SystemW\_Sig\_C}[|\mathit{INTRUDER\_PATH}|]\mathit{INTUDER\_Sig}
\end{aligned}
$$

$$\begin{aligned}
SystemR_Dig =_{df}\ & READER_Dig(R, M, K, K_M, NN, ND, NDK)[|PROCESS_PATH|]PROCESS \\
& [|COM_PATH|]ACM_R_Dig(R, M, K_M, K_A, DK, DATA, NK_e, NK_d, HL) \\
SystemW_Dig =_{df}\ & WRITER_Dig(W, M, K, K_M, K_A, NN, ND, DK, DATA)[|PROCESS_PATH|]PROCESS \\
& [|COM_PATH|]ACM_W_Dig(W, M, HL, NK_e, NK_d, NDK) \\
SystemR_Dig_I =_{df}\ & SystemR_Dig[|INTRUDER_PATH|]INTUDER_Dig \\
SystemW_Dig_I =_{df}\ & SystemW_Dig[|INTRUDER_PATH|]INTUDER_Dig \\
SystemR_Dig_C =_{df}\ & READER_Dig(R, M, K, K_I, NN, ND, NDK)[|PROCESS_PATH|]PROCESS \\
& [|COM_PATH|]ACM_R_Dig(R, M, K_M, K_A, DK, DATA, NK_e, NK_d, HL) \\
SystemW_Dig_C =_{df}\ & WRITER_Dig(W, M, K, K_I, K_A, NN, ND, DK, DATA)[|PROCESS_PATH|]PROCESS \\
& [|COM_PATH|]ACM_W_Dig(W, M, HL, NK_e, NK_d, NDK) \\
SystemR_Dig_C_I =_{df}\ & SystemR_Dig_C[|INTRUDER_PATH|]INTUDER_Dig \\
SystemW_Dig_C_I =_{df}\ & SystemW_Dig_C[|INTRUDER_PATH|]INTUDER_Dig
\end{aligned}$$

Here gives the definition of subprocesses $READER\_Sig$ and $READER\_Dig$. They simulate the read operations.

$$\begin{aligned}
& READER_1(r, m, k, k_m, nn, nd, ndk) =_{df} \\
&\quad Initialization\{n = false; d = false\} \to \\
&\quad ComRM!msg_{int}.r.m.nd \to ComRM?msg_{dat}.m.r.nd.E(dk, data) \to \\
&\quad ComRM!msg_{int}.r.m.ndk \to ComRM?msg_{dat}.m.r.ndk.E(nk_e, dk1) \to \\
&\quad ComRM!msg_{int}.r.m.nn \to ComRM?msg_{dat}.m.r.nn.hl \to \\
&\quad ComRM!msg_{int}.r.m.nn.H(k) \to ComRM?msg_{dat}.m.r.nn.H(k1).E(k1, E((k_m^{-1}), nk_d)) \to \\
&\quad CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), nk_d)).k^{-1}.k_m \to CheckNK?msg_{ack}.ack \to \\
&\quad \begin{pmatrix} (NKFakingSuccess\{n = true\} \to SKIP) \\ \lhd(ack == YES) \rhd (NKFakingError\{n = false\} \to SKIP) \end{pmatrix}; \\
&\quad GetData!msg_{pro}.E(dk, data).E(nk_e, dk1).nk_d \to GetData?msg_{ack}.ack1 \to \\
&\quad \begin{pmatrix} (DataAcquisitionSuccess\{d = true\} \to SKIP) \\ \lhd(ack1 == YES) \rhd (DataAcquisitionError\{d = false\} \to SKIP) \end{pmatrix}; \\
&\quad READER_1(r, m, k, k_m, nn, nd, ndk) \\
& READER_2(r, m, k, k_m, k_a, nn, nd, ndk) =_{df} \\
&\quad READER_1(r, m, k, k_m, nn, nd, ndk)[[\\
&\quad ComRM?msg_{dat}.m.r.nn.H(k1).E(k1, E((k_m^{-1}), nk_d)) \\
&\quad \leftarrow ComRM?msg_{dat}.m.r.nn.H(k1).E(k1, E((k_m^{-1}), nk_d)).E((k_a^{-1}), k_m), \\
&\quad CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), nk_d)).k^{-1}.k_m \\
&\quad \leftarrow CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), nk_d)).E((k_a^{-1}), k_m).k^{-1}.k_m.k_a]] \\
& READER_Sig(r, m, k, k_m, nn, nd, ndk) =_{df} \\
&\quad READER_1(r, m, k, k_m, nn, nd, ndk)[[\\
&\quad ComRM?\{|ComRM|\} \leftarrow ComRM?\{|ComRM|\}, ComRM?\{|ComRM|\} \leftarrow FakeRM2?\{|ComRM|\}, \\
&\quad ComRM!\{|ComRM|\} \leftarrow ComRM!\{|ComRM|\}, ComRM!\{|ComRM|\} \leftarrow FakeRM2!\{|ComRM|\}]] \\
& READER_Dig(r, m, k, k_m, nn, nd, ndk) =_{df} \\
&\quad READER_2(r, m, k, k_m, nn, nd, ndk)[[\\
&\quad ComRM?\{|ComRM|\} \leftarrow ComRM?\{|ComRM|\}, ComRM?\{|ComRM|\} \leftarrow FakeRM2?\{|ComRM|\}, \\
&\quad ComRM!\{|ComRM|\} \leftarrow ComRM!\{|ComRM|\}, ComRM!\{|ComRM|\} \leftarrow FakeRM2!\{|ComRM|\}]]
\end{aligned}$$

Here lists the definition of subprocesses $WRITER\_Sig$ and $WRITER\_Dig$. They simulate the write operations.

$$
\begin{aligned}
&WRITER_1(w, m, k, k_m, nn, nd, dk, data) =_{df}\\
&\quad Initialization\{n = false\} \rightarrow\\
&\quad ComWM!msg_{int}.w.m.nn \rightarrow ComWM?msg_{dat}.m.w.nn.hl \rightarrow\\
&\quad ComWM!msg_{int}.w.m.nn.H(k) \rightarrow\\
&\quad ComWM?msg_{dat}.m.w.nn.H(k1).E(k1, E((k_m^{-1}), (nk\_e, nk\_d))) \rightarrow\\
&\quad ComWM!msg_{dat}.w.m.nd.E(dk, data) \rightarrow ComWM?msg_{int}.m.w.ndk \rightarrow\\
&\quad ComWM!msg_{dat}.w.m.ndk.E(nk\_e, dk) \rightarrow\\
&\quad CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), (nk\_e, nk\_d))).k^{-1}.k_m \rightarrow CheckNK?msg_{ack}.ack \rightarrow\\
&\quad \begin{pmatrix} (NKFakingSuccess\{n = true\} \rightarrow SKIP)\\ \lhd (ack == YES) \rhd (NKFakingError\{n = false\} \rightarrow SKIP) \end{pmatrix};\\
&\quad WRITER_1(w, m, k, k_m, nn, nd, dk, data)\\
&WRITER_2(w, m, k, k_m, k_a, nn, nd, dk, data) =_{df}\\
&\quad WRITER_1(w, m, k, k_m, nn, nd, dk, data)[[\\
&\quad ComWM?msg_{dat}.m.w.nn.H(k1).E(k1, E((k_m^{-1}), (nk\_e, nk\_d)))\\
&\quad \leftarrow ComWM?msg_{dat}.m.w.nn.H(k1).E(k1, E((k_m^{-1}), (nk\_e, nk\_d))).E(k_a^{-1}, k_m),\\
&\quad CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), (nk\_e, nk\_d))).k^{-1}.k_m\\
&\quad \leftarrow CheckNK!msg_{pro}.E(k1, E((k_m^{-1}), (nk\_e, nk\_d))).E((k_a^{-1}), k_m).k^{-1}.k_m.k_a]]\\
&WRITER\_Sig(w, m, k, k_m, nn, nd, dk, data) =_{df}\\
&\quad WRITER_1(w, m, k, k_m, nn, nd, dk, data)[[\\
&\quad ComWM?\{|ComWM|\} \leftarrow ComWM?\{|ComWM|\}, ComWM?\{|ComWM|\} \leftarrow FakeWM2?\{|ComWM|\},\\
&\quad ComWM!\{|ComWM|\} \leftarrow ComWM!\{|ComWM|\}, ComWM!\{|ComWM|\} \leftarrow FakeWM2!\{|ComWM|\}]]\\
&WRITER\_Dig(w, m, k, k_m, k_a, nn, nd, dk, data) =_{df}\\
&\quad WRITER_2(w, m, k, k_m, k_a, nn, nd, dk, data)[[\\
&\quad ComWM?\{|ComWM|\} \leftarrow ComWM?\{|ComWM|\}, ComWM?\{|ComWM|\} \leftarrow FakeWM2?\{|ComWM|\},\\
&\quad ComWM!\{|ComWM|\} \leftarrow ComWM!\{|ComWM|\}, ComWM!\{|ComWM|\} \leftarrow FakeWM2!\{|ComWM|\}]]
\end{aligned}
$$

Here shows the definition of subprocesses ACM_R_Sig and ACM_R_Dig. They simulate the behavior of ACM when it communicates with the readers.

$$
\begin{aligned}
&ACM\_R_1(r, m, k_m, dk, data, nk\_e, nk\_d, hl) =_{df}\\
&\quad ComRM?msg_{int}.r.m.nd \rightarrow ComRM!msg_{dat}.m.r.nd.E(dk, data) \rightarrow\\
&\quad ComRM?msg_{int}.r.m.ndk \rightarrow ComRM!msg_{dat}.m.r.ndk.E(nk\_e, dk) \rightarrow\\
&\quad ComRM?msg_{int}.r.m.nn \rightarrow ComRM!msg_{dat}.m.r.nn.hl \rightarrow\\
&\quad ComRM?msg_{int}.r.m.nn.H(k) \rightarrow ComRM!msg_{dat}.m.r.nn.H(k).E(k, E(k_m^{-1}, nk\_d)) \rightarrow\\
&\quad ACM\_R_1(r, m, k_m, dk, data, nk\_e, nk\_d, hl)\\
&ACM\_R_2(r, m, k_m, k_a, dk, data, nk\_e, nk\_d, hl) =_{df}\\
&\quad ACM\_R_1(r, m, k_m, dk, data, nk\_e, nk\_d, hl)[[\\
&\quad ComRM!msg_{dat}.m.r.nn.H(k).E(k, E(k_m^{-1}, nk\_d))\\
&\quad \leftarrow ComRM!msg_{dat}.m.r.nn.H(k).E(k, E(k_m^{-1}, nk\_d)).E(k_a^{-1}, k_m)]]\\
&ACM\_R\_Sig(r, m, k_m, dk, data, nk\_e, nk\_d, hl) =_{df}\\
&\quad ACM\_R_1(r, m, k_m, dk, data, nk\_e, nk\_d, hl)[[\\
&\quad ComRM?\{|ComRM|\} \leftarrow ComRM?\{|ComRM|\}, ComRM?\{|ComRM|\} \leftarrow FakeRM1?\{|ComRM|\},\\
&\quad ComRM!\{|ComRM|\} \leftarrow ComRM!\{|ComRM|\}, ComRM!\{|ComRM|\} \leftarrow FakeRM1!\{|ComRM|\}]]
\end{aligned}
$$

$$ACM\_R\_Dig(r, m, k_m, k_a, dk, data, nk\_e, nk\_d, hl) =_{df}$$
$$ACM\_R_2(r, m, k_m, k_a, dk, data, nk\_e, nk\_d, hl)[[$$
$$ComRM?\{|ComRM|\} \leftarrow ComRM?\{|ComRM|\}, ComRM?\{|ComRM|\} \leftarrow FakeRM1?\{|ComRM|\},$$
$$ComRM!\{|ComRM|\} \leftarrow ComRM!\{|ComRM|\}, ComRM!\{|ComRM|\} \leftarrow FakeRM1!\{|ComRM|\}]]$$

We illustrate the definition of subprocesses ACM_W_Sig and ACM_W_Dig. They simulate the behavior of ACM when it communicates with the writers.

$$ACM\_W_1(w, m, k_m, hl, nk\_e, nk\_d, ndk) =_{df}$$
$$Initialization\{d = false\} \rightarrow$$
$$ComWM?msg_{int}.w.m.nn \rightarrow ComWM!msg_{dat1}.m.w.nn.hl \rightarrow$$
$$ComWM?msg_{int}.w.m.nn.H(k) \rightarrow$$
$$ComWM!msg_{dat}.m.w.nn.H(k).E(k, E(k_m^{-1}, (nk\_e, nk\_d))) \rightarrow$$
$$ComWM?msg_{dat}.w.m.nd.E(dk, data) \rightarrow ComWM!msg_{int}.w.m.ndk \rightarrow$$
$$ComWM?msg_{dat}.m.w.ndk.E(nk\_e1, dk1) \rightarrow$$
$$GetData!msg_{pro}.E(dk, data).E(nk\_e1, dk1).nk\_d \rightarrow$$
$$GetData?msg_{ack}.ack \rightarrow$$
$$\left(\begin{array}{l}(DataAcquisitionSuccess\{d = true\} \rightarrow SKIP)\\ \lhd(ack == YES) \rhd (DataAcquisitionError\{d = false\} \rightarrow SKIP)\end{array}\right);$$
$$ACM\_W_1(w, m, hl, nk\_e, nk\_d, ndk)$$

$$ACM\_W_2(w, m, k_m, k_a, hl, nk\_e, nk\_d, ndk) =_{df}$$
$$ACM\_W_1(w, m, k_m, hl, nk\_e, nk\_d, ndk)[[$$
$$ComWM!msg_{dat}.m.w.nn.H(k).E(k, E(k_m^{-1}, (nk\_e, nk\_d)))$$
$$\leftarrow ComWM!msg_{dat}.m.w.nn.H(k).E(k, E(k_m^{-1}, (nk\_e, nk\_d)).E(K_a^{-1}, k_m)]]$$

$$ACM\_W\_Sig(w, m, hl, nk\_e, nk\_d, ndk) =_{df}$$
$$ACM\_W_1(w, m, hl, nk\_e, nk\_d, ndk)[[$$
$$ComWM?\{|ComWM|\} \leftarrow ComWM?\{|ComWM|\}, ComWM?\{|ComWM|\} \leftarrow FakeWM1?\{|ComWM|\},$$
$$ComWM!\{|ComWM|\} \leftarrow ComWM!\{|ComWM|\}, ComWM!\{|ComWM|\} \leftarrow FakeWM1!\{|ComWM|\}]]$$

$$ACM\_W\_Dig(w, m, hl, nk\_e, nk\_d, ndk) =_{df}$$
$$ACM\_W_2(w, m, hl, nk\_e, nk\_d, ndk)[[$$
$$ComWM?\{|ComWM|\} \leftarrow ComWM?\{|ComWM|\}, ComWM?\{|ComWM|\} \leftarrow FakeWM1?\{|ComWM|\},$$
$$ComWM!\{|ComWM|\} \leftarrow ComWM!\{|ComWM|\}, ComWM!\{|ComWM|\} \leftarrow FakeWM1!\{|ComWM|\}]]$$

We also update the definition of $PROCESS$.

$$PROCESS() =_{df}$$
$$CheckNK?msg_{pro}.E(k1, E((k_{m1}^{-1}), nk\_d)).k2^{-1}.k_{m2}$$
$$\left(\begin{array}{l}(checkNK!msg_{ack}.YES \rightarrow PROCESS())\\ \lhd((k1 == k2)\&\&(k_{m1} == k_{m2})\&\&(nk\_d == NK\_d\_f)) \rhd (CheckNK!msg_{ack}.NO \rightarrow PROCESS())\end{array}\right)$$
$$\Box CheckNK?msg_{pro}.E(k1, E((k_{m1}^{-1}), nk\_d)).E((k_{a1}^{-1}), k_{m2}).k2^{-1}.k_{m3}.k_{a2} \rightarrow$$
$$\left(\begin{array}{l}(CheckNK!msg_{ack}.YES \rightarrow PROCESS())\\ \lhd((k1 == k2)\&\&(k_{a1} == k_{a2})\&\&(k_{m1} == k_{m3})\&\&(k_{m2} == k_{m3})\&\&(nk\_d == NK\_d\_f))\rhd\\ (CheckNK!msg_{ack}.NO \rightarrow PROCESS())\end{array}\right)$$
$$\Box CheckNK?msg_{pro}.E(k1, E((k_{m1}^{-1}), (nk\_e, nk\_d))).k2^{-1}.k_{m2} \rightarrow$$
$$\left(\begin{array}{l}(CheckNK!msg_{ack}.YES \rightarrow PROCESS())\\ \lhd((k1 == k2)\&\&(k_{m1} == k_{m2})\&\&(nk\_e == NK\_e\_f)\&\&(nk\_d == NK\_d\_f))\rhd\\ (CheckNK!msg_{ack}.NO \rightarrow PROCESS())\end{array}\right)$$

$\Box CheckNK?msg_{pro}.E(k1, E((k_{m1}^{-1}), (nk\_e, nk\_d))).E((k_{a1}^{-1}), k_{m2}).k2^{-1}.k_{m3}.k_{a2} \rightarrow$

$$\left(\begin{array}{l}(CheckNK!msg_{ack}.YES \rightarrow PROCESS())\\ \lhd((k1 == k2)\&\&(k_{a1} == k_{a2})\&\&(k_{m1} == k_{m3})\&\&(k_{m2} == k_{m3})\&\&(nk\_e == NK\_e\_f)\&\&(nk\_d == NK\_d\_f))\rhd\\ (CheckNK!msg_{ack}.NO \rightarrow PROCESS())\end{array}\right)$$

$\Box GetData?msg_{ack}.E(dk1, data).E(nk\_e, dk2).nk\_d \rightarrow$

$$\left(\begin{array}{l}(GetData!msg_{ack}.YES \rightarrow PROCESS())\\ \lhd((((nk\_e == NK\_e)\&\&(nk\_d == NK\_d))||((nk\_e == NK\_e\_f)\&\&(nk\_d == NK\_d\_f)))\\ \quad \&\&(dk1 == dk2)) \rhd (GetData!msg_{ack}.NO \rightarrow PROCESS())\end{array}\right)$$

We will add new element to *Fact* in [1], which is the set of facts which intruders might learn.

$$Fact_1 =_{df} \quad Fact \cup \{E(K, E(K_M^{-1}, content)) \mid content \in \{NK\_d, (NK\_e, NK\_d), NK\_d\_f, (NK\_e\_f, NK\_d\_f)\}\}$$

And we also need to define a new deducing rule for the new element.

$$\{K^{-1}, E(K, E(K_M^{-1}, content)\} \mapsto E(K_M^{-1}, content)$$

We also add new definitions of how the intruders get new facts form messages:

$$Info(msg_{dat}.a.b.n.H(k).E(k, E(k_1, c_1))) =_{df} \{a, b, n, H(k), E(k, E(k_1, c_1))\}$$
$$Info(msg_{dat}.a.b.n.H(k).E(k, E(k_1, c_1)).E(k_2, c_2)) =_{df} \{a, b, n, H(k), E(k, E(k_1, c_1)), E(k_2, c_2)\}$$

where $a,b \in Entity$, $n \in Name$, $k,k1,k2 \in Key$, $c1,c2 \in Content$.

Finally, we give the definition of *INTUDER_Sig* and *INTUDER_Dig*. They simulate the behavior of the intruders.

$INTUDER_1(F) =_{df}$

$\Box\Box_{m \in MSG_{out}} FakeRM1?m \rightarrow FakeRM2!m \rightarrow INTUDER_1(F \cup Info(m))$

$\Box\Box_{m \in (MSG_{out} \setminus MSG_{dat2})} FakeRM2?m \rightarrow FakeRM1!m \rightarrow INTUDER_1(F \cup Info(m))$

$\Box\Box_{m \in MSG_{dat2}} FakeRM2?m \rightarrow FakeRM1!m[[k_m^{-1} \leftarrow K_I^{-1}, nk\_d \leftarrow NK\_d\_f]] \rightarrow$
$INTUDER_1(F \cup Info(m))$

$\Box\Box_{m \in MSG_{out}} FakeWM1?m \rightarrow FakeWM2!m \rightarrow INTUDER_1(F \cup Info(m))$

$\Box\Box_{m \in (MSG_{out} \setminus MSG_{dat2})} FakeWM2?m \rightarrow FakeWM1!m \rightarrow INTUDER_1(F \cup Info(m))$

$\Box\Box_{m \in MSG_{dat2}} FakeWM2?m \rightarrow FakeWM1!m[[k_m^{-1} \leftarrow K_I^{-1}, (nk\_e, nk\_d) \leftarrow (NK\_e\_f, NK\_d\_f)]] \rightarrow$
$INTUDER_1(F \cup Info(m))$

$\Box\Box_{f \in Fact, f \notin F, F \mapsto f} Initialization\{l = false\} \rightarrow Deduce.f.F \rightarrow$

$$\left(\begin{array}{l}(DataLeakageSuccess\{l = true\} \rightarrow INTRUDER_1(F \cup \{f\}))\\ \lhd(f == Data) \rhd (DataLeakageError\{l = false\} \rightarrow INTRUDER_1(F \cup \{f\}))\end{array}\right)$$

$INTUDER\_Sig =_{df}$

$INTUDER_1(IK \cup \{K_I, K_I^{-1}, K_M\})$

$INTUDER\_Dig =_{df}$

$INTUDER_1(IK \cup \{K_I, K_I^{-1}, K_M, K_A\})$