김일성종합대학학보
정보과학

주체108(2019)년
제65권 제4호

δ계산론리에 의한 안전한 분산이동형처리의 모형화에 대한 연구

박지혜, 김용석

경애하는 최고령도자 **김정은**동지께서는 다음과 같이 말씀하시였다.

《첨단돌파전을 힘있게 벌려야 나라의 과학기술전반을 빨리 발전시키고 지식경제의 토대를 구축해나갈수 있습니다.》

오늘 우리 나라에서는 인민경제의 모든 분야를 발전시키기 위하여 공장, 기업소들마다에 업무통합체계를 확립하기 위한 사업이 힘있게 진행되고있다. 특히 나라의 경제규모가 비할바없이 커지고 정보의 류통과 처리량이 방대해지는데 따라 콤퓨터망을 통한 기업업무가 활발히 벌어지고있으며 이로 하여 업무통합체계들이 분산이동형실시간체계로 개발되여 리용되고있다.

업무통합체계들의 이러한 분산이동형실시간적특성으로 하여 세계적으로 기업업무모형화에서 업무안정성과 보안성을 담보하는 안전한 체계를 구축하는 방향으로 업무통합체계개발이 진행되고있다.[1]

현시기 발전하는 현실의 요구와 세계적추세에 맞게 기업업무통합체계를 과학적으로 확립하고 업무통합체계의 정확성을 검증하는 문제는 매우 중요한 문제로 나선다.

선행연구[2]에서는 분산이동형실시간처리들을 모형화하는데 π계산론리를 리용하였으며 모형검증도구를 리용하여 검증함으로써 설계의 정확성을 확인하였다.

그러나 π계산론리는 이동들을 값통과통신의 개념으로 표현하기때문에 처리들의 실제한 이동을 표현하는데는 적합하지 않다. 결과 이동들에 대한 현실성에서 심한 이지러짐이 있고 또한 이동들을 직접적으로 명백히 표현할수 없으므로 실지 이동의 표현에는 엄격한 제한이 있었다.

론문에서는 선행연구의 제한성을 극복하고 처리들을 직접적으로 명백히 표현할수 있을뿐아니라 처리들을 본문적으로, 그라프적으로도 다 표시할수 있는 δ계산론리를 제안하였다. 또한 체계보안의 실례로서 CryptoLocker체계[3]를 모형화함으로써 δ계산론리의 효과성을 분석하였다.

1. δ계산론리

업무처리의 이동들을 명세화하기 위한 처리대수로 δ계산론리를 정의하기로 한다. δ계산론리에서의 이동들은 기본적으로 동기적이다. 그러므로 비동기적이동에 의하여 초래되는 교착과 같은 체계의 정상완료에 지장을 주는 문제들을 방지할수 있다. 또한 중요한것은 비동기적이동들을 동기적방식으로 조종하기 위한 제한요구들로 하여 δ계산론리에서 비동기적이동들을 표시할수 있다.

1) 문장론

δ계산론리의 문장론을 표 1에 보여주었다.

표 1. δ계산론리의 문장론

$P ::= \text{nil}$	//휴식	$\mid M.P$	//이동
$\mid P_{(n)}$	//우선권	$\mid C.P$	//처리조종
$\mid P[Q]$	//포함	$M ::= m_t^p(k)P$	//이동요청
$\mid P[\overline{Q}]$	//숨기기	$\mid \overline{m}_t^p(k)P$	//숨기기이동요청
$\mid P(r)$	//포구	$\mid Pm(k)$	//이동허가
$\mid P \setminus F$	//제한	$\mid P\,\overline{m}(k)$	//숨기기이동허가
$\mid P + Q$	//선택	$m ::= \text{in} \mid \text{out} \mid \text{get} \mid \text{put}$	//이동활동
$\mid P \parallel Q$	//병렬	$C ::= \text{new}\ P$	//처리창조
$\mid r(a).P$	//통신	$\mid \text{kill}\ P$	//처리사멸
$\mid P\Delta_t(S, T, I)$	//통신범위	$\mid \text{exit}$	//처리탈퇴
$\mid P\square_{[l,u]}(S, L, U, K, R, I)$	//이동범위		

δ계산론리의 문장론을 구체적으로 보면 다음과 같다.

① 휴식(0)

처리에 대한 정지이다.

② 우선권($P_{(n)}$)

처리 P의 우선권을 옹근수 $n \geq 0$으로 표시한다. 수가 클수록 더 높은 우선권을 나타낸다.

편리성을 위하여 기정으로 0을 가장 높은 우선권으로 정한다.

③ 포함($P[Q]$)

P는 Q를 포함한다. 안의 처리 (Q)는 밖의 처리 (P)에 의하여 조종된다. 따라서 안의 처리는 밖의 처리의 허가가 없이는 외부의 다른 처리와 통신할수 없다.

안의 처리가 밖의 처리보다 더 높은 우선권을 가지고있다 하여도 허가방식에는 변동이 없다.

그러나 더 높은 우선권은 안의 처리가 그 어떤 허가가 없이도 밖의 처리로부터 나올 수 있도록 한다.

④ 숨기기($P[\overline{Q}]$)

Q는 P에 숨겨져있으며 P의 밖에서 발견할수 없다. $\overline{Q}$는 Q의 숨겨진 상태를 가리킨다. 이때 숨겨진 처리 Q는 P와 P의 외부의 모든 처리로부터 발견되지 않게 P와 같은 우선권과 포구를 공유할수 있다.

⑤ 포구($P(r)$)

P의 포구 r를 표시하며 포구를 통하여 P가 다른 처리들과 통신할수 있다.

⑥ 제한($P\setminus F$)

P에서 처리 F의 활동이나 이동을 제한시킨다.

⑦ 선택($P + Q$)

P와 Q의 우선권들이 같다면 실행시 이것들중에서 하나만이 비결정적으로 선택된다. 우선권이 다르다면 우선권이 더 높은 처리가 선택된다.

⑧ 병렬 $(P_1 \| P_2)$

P_1과 P_2가 동시에 실행된다.

⑨ 통신 $(r(a).P)$

P가 포구 r로 련결된 다른 처리와 통신하여 통보문 a를 통과시킨다. 통과방식은 보내기는 $\overline{a}$로, 받기는 a로 지적한다.

⑩ 통신범위 $(P\Delta_t(S, T, I))$

통신에 대하여 $0 \le t \le \infty$로 지적되는 마감기한으로 P의 모든 실행가지들을 표시한다. P의 표준끝에 S가 오고 t의 위반에 T를, 외부로부터의 새치기에 I를 놓는다.

⑪ 이동범위 $(P\square_{[l,u]}(S, L, U, K, R, I))$

이동에 대하여 $0 \le l \le u \le \infty$로 지적되는 한계선으로 P의 모든 실행가지들을 표시한다. P의 마감에 S가 놓이고 아래시간한계 l의 위반에 L을, 웃한계 u의 위반에 U를, 통과암호와 같은 열쇠위반에 K를, 우선권위반에 R를, 외부로부터의 새치기에 I를 놓는다.

⑫ 이동요청 $(m_t^p(k)P;\ \overline{m}_t^p(k)P)$

목표처리에로 혹은 그것으로부터의 이동에 대한 열쇠를 가진 요청을 표시한다. 이러한 이동에는 두가지 형태가 있다. 하나는 정상적인 이동(m)이고 다른 하나는 숨겨진 이동$(\overline{m})$이다. 여기서 t, p, k는 각각 이동에 대한 시간, 우선권, 통과암호를 나타낸다.

⑬ 이동허가 $(P\,m(k);\ P\,\overline{m}(k))$

이동요청에 대한 허가를 표시한다. 요청에서와 마찬가지로 두가지 형태가 있다. 즉 하나는 정상적인 이동(m)과 숨겨진 이동$(\overline{m})$이다.

⑭ 처리창조(new P)

새로 창조되는 처리가 그것을 창조하는 처리보다 더 높은 우선권을 가질수 없다는 제한을 가지고 외부에 처리를 창조하는 동작을 표현한다.

⑮ 처리사멸(kill P)

처리가 자기보다 낮은 우선권을 가진 다른 처리를 종결시키는 동작을 표현한다. 이때 다른 처리를 종결시키는 처리는 숨겨진 상태에 있지 말아야 한다.

⑯ 처리탈퇴(exit)

자체로 종결되는 동작을 표현한다. 즉 자기의 모든 자식처리들이 자동적으로 그리고 계층적으로 종결된다. 이것은 교착으로 볼수 있는 빈(nil)조작과는 다르다.

계산론리에서 이동들은 기본적으로는 동기적으로 정의된다.

입출력조작이나 그러한 조작을 수행하는 다른 처리를 만드는 요청들은 목표처리로부터 허가를 받아야 한다.

엄격한 동기적이동은 일정한 우선권기준을 가진 허가정도에 의하여 결정되는 이동방식밑에서 모든 이동처리들을 조종하기 위하여 필요하다. 이것은 불필요하고 인증되지 않은 이동들이 허가되는것을 방지함으로써 목표체계의 안정성과 보안을 담보하게 한다. 또한 이동들의 동기성은 이동범위에서 시간속성들을 통신범위에서 정의된것과 꼭같이 규정함으로써 균형을 꼭 맞출수 있다.

2) 의미론

δ 계산론리의 통신의미론을 표 2에 보여주었다.

표 2. δ 계산론리의 통신의미론

Action	$\dfrac{-}{r(a).P \xrightarrow{r(a)} P}$	Choice$L$	$\dfrac{P_{(n)} \xrightarrow{a} P'_{(n)}}{P_{(n)} + Q_{(n)} \xrightarrow{a} P'_{(n)}}$
Choice$R$	$\dfrac{Q_{(n)} \xrightarrow{a} Q'_{(n)}}{P_{(n)} + Q_{(n)} \xrightarrow{a} Q'_{(n)}}$	Parl$L$	$\dfrac{P \xrightarrow{a} P'}{P \Vert Q \xrightarrow{a} P' \Vert Q}$
Parl$R$	$\dfrac{Q \xrightarrow{a} Q'}{P \Vert Q \xrightarrow{a} P \Vert Q'}$	ParCom	$\dfrac{P \xrightarrow{a} P',\ Q \xrightarrow{\overline{a}} Q'}{P \Vert Q \xrightarrow{\tau} P' \Vert Q'}$
CScope$C$	$\dfrac{P \xrightarrow{a} 0}{P\,\Delta_t(S,\,T,\,I) \xrightarrow{a} P'\,\Delta_t(S,\,T,\,I)}$	CScope$S$	$\dfrac{P \xrightarrow{a} 0}{P\Delta_t(S,\,T,\,I) \xrightarrow{a} S}(t>0)$
CScope$T$	$\dfrac{T \xrightarrow{a} T'}{P\Delta_t(S,\,T,\,I) \xrightarrow{a} T'}(t=0)$	CScope$I$	$\dfrac{I \xrightarrow{a} I'}{P\Delta_t(S,\,T,\,I) \xrightarrow{a} I'}(t>0)$
Hide$C$	$\dfrac{Q\langle B\rangle \xrightarrow{r(a)} Q'\langle B\rangle}{P[\overline{Q}]\langle A\rangle \xrightarrow{r(a)} P[\overline{Q}']\langle A\rangle}(r \in A,\ r \notin B)$	Choice$P$	$\dfrac{P_{(n)} \xrightarrow{a} P'_{(n)},\ Q_{(m)} \xrightarrow{b} Q'_{(m)}}{P_{(n)} + Q_{(m)} \xrightarrow{a} P'_{(n)}}(n>m)$
Restriction	$\dfrac{P \xrightarrow{a} P'}{P \setminus F \xrightarrow{a} P' \setminus F}(a,\ \overline{a} \notin F)$		

δ 계산론리의 통신의미론을 구체적으로 보면 다음과 같다.

① 활동

포구 r에서 통보문 a를 가지고 통신하는 활동 $r(a)$의 실행에 대한 이행규칙이다. 이 이행에 대하여 그 어떤 전제도 요구하지 않으며 그다음에는 P가 실행된다.

② 선택

Choice$L$과 Choice$R$이행들은 같은 전제밑에서 동등하게 선택될수 있다. Choice$P$는 우선권에 의하여 결정된다.

③ 병렬

Parl$L$과 Parl$R$에서 처리의 이행은 그것의 병렬처리에 영향을 주지 않는다. 그러나 parCom은 그 2개의 병렬처리들이 통신이행을 만들기 위하여 서로 동기적으로 호상작용할것을 요구한다.

④ 제한

F에서 정의되지 않은 활동들만이 규칙에 의하여 실행되도록 허가한다.

⑤ 통신범위

CScope$C$는 실행에서 P의 활동에 대한 시간경과를 표현한다. CScope$S$는 P의 마감기한안에서의 정상마감에서 S에로의 이행이다.

CScope$T$는 마감기한위반으로 하여 P의 비정상적인 마감에서 T에로의 이행이다. CScope$I$는 모든 새치기에서 I에로의 이행이다.

⑥ 숨기기

Hide$C$는 P에서 숨겨진 처리 Q가 P의 포구를 통하여 인증되지 않은 이행을 만들수 있다는것을 나타낸다.

δ 계산론리의 이동의미론을 표 3에 보여주었다.

표 3. δ 계산론리의 이동의미론

In	$\dfrac{P \xrightarrow{\mathrm{in}_t(k)Q} P',\ Q \xrightarrow{P\,\mathrm{in}(k)} Q'}{P \parallel Q \xrightarrow{\delta} Q'[P']}$	Get H	$\dfrac{P \xrightarrow{\overline{\mathrm{get}}_t(k)Q} P',\ Q \xrightarrow{P\,\overline{\mathrm{get}}(k)} Q'}{P \parallel Q \xrightarrow{\delta} P'[\overline{Q'}]}$
Out	$\dfrac{P \xrightarrow{\mathrm{out}_t(k)Q} P',Q \xrightarrow{P\,\mathrm{out}(k)} Q'}{Q[P] \xrightarrow{\delta} P' \parallel Q'}$	Put H	$\dfrac{P \xrightarrow{\overline{\mathrm{put}}_t(k)Q} P',\ Q \xrightarrow{P\,\overline{\mathrm{put}}(k)} Q'}{P[\overline{Q}] \xrightarrow{\delta} P' \parallel Q'}$
Get	$\dfrac{P \xrightarrow{\mathrm{get}_t(k)Q} P',\ Q \xrightarrow{P\mathrm{get}(k)} Q'}{P \parallel Q \xrightarrow{\delta} P'[Q']}$	InP	$\dfrac{P_{(n)} \xrightarrow{\mathrm{in}_t^p(k)Q_{(m)}} P'_{(n)}}{P_{(n)} \parallel Q_{(m)} \xrightarrow{\delta} Q_{(m)}[P'_{(n)}]}(n \geq m)$
Put	$\dfrac{P \xrightarrow{\mathrm{put}_t(k)Q} P',\ Q \xrightarrow{P\mathrm{put}(k)} Q'}{P[Q] \xrightarrow{\delta} P' \parallel Q'}$	Out P	$\dfrac{P_{(n)} \xrightarrow{\mathrm{out}_t^p(k)Q_{(m)}} P'_{(n)}}{Q_{(m)}[P_{(n)}] \xrightarrow{\delta} P'_{(n)} \parallel Q_{(m)}}(n \geq m)$
InH	$\dfrac{P \xrightarrow{\overline{\mathrm{in}}_t(k)Q} P',\ Q \xrightarrow{P\overline{\mathrm{in}}(k)} Q'}{P \parallel Q \xrightarrow{\delta} Q'[\overline{P'}]}$	Get P	$\dfrac{P_{(n)} \xrightarrow{\mathrm{get}_t^p(k)Q_{(m)}} P'_{(n)}}{P_{(n)} \parallel Q_{(\mathrm{m})} \xrightarrow{\delta} P'_{(n)}[Q_{(m)}]}(n \geq m)$
Out H	$\dfrac{P \xrightarrow{\overline{\mathrm{out}}_t(k)Q} P',\ Q \xrightarrow{P\overline{\mathrm{out}}(k)} Q'}{Q[\overline{P}] \xrightarrow{\delta} P' \parallel Q'}$	Put P	$\dfrac{P_{(n)} \xrightarrow{\mathrm{put}_t^p(k)Q_{(m)}} P'_{(n)}}{P_{(n)}[Q_{(m)}] \xrightarrow{\delta} P'_{(n)} \parallel Q_{(m)}}(n \geq m)$
*MS*cope *S*	$\dfrac{P \xrightarrow{a} 0}{P \square_{[l,\,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} S}(\mathrm{l} = 0,\ u > 0)$	*MS*cope *L*	$\dfrac{P \xrightarrow{a} 0}{P \square_{[l,\,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} L}(\mathrm{l} > 0)$
*MS*cope *U*	$\dfrac{U \xrightarrow{a} U'}{P \square_{[l,\,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} U'}(u = 0)$	*MS*cope *K*	$\dfrac{P \xrightarrow{m_t(a)Q} P',\ Q \xrightarrow{P\,m(b)} Q'}{P \square_{[l,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} K}(u > 0,\ a \neq b)$
*MS*cope *C*	$\dfrac{P \xrightarrow{a} P'}{P \square_{[l,\,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} P' \square_{[l-t,\,u-t]}(S,\ L,\ U,\ K,\ R,\ I)}$		
*MS*cope *R*	$\dfrac{P_{(n)} \xrightarrow{m_t^p(a)Q_{(m)}} P'_{(n)}}{P_{(n)} \square_{[\mathrm{l,u}]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{m_t^p(a)Q_{(m)}} R}(u > 0,\ n < m)$		
*MS*cope *I*	$\dfrac{I \xrightarrow{a} I'}{P \square_{[l,\,u]}(S,\ L,\ U,\ K,\ R,\ I) \xrightarrow{a} I'}(u > 0)$		

δ 계산론리의 이동의미론을 구체적으로 보면 다음과 같다.

① 이동

In, Out, Get, Put이행들은 일반적인 동기적이동들을 표시한다. 매 이동은 같은 열쇠를 가진 대응하는 호상이동을 반드시 가진다. InH, Out$H$, Get$H$, Put$H$들은 앞에서의 규칙들과 같지만 숨겨진 조건에서의 이동들이다. 마찬가지로 InP, Out$P$, Get$P$, Put$P$들은 우선권을 가진 이동들에 대한 규칙들이다. 그러나 이 이동들은 우선권에 의하여 비동기적인것으로 될

수 있다. 이것은 보다 높은 우선권을 가진 이 이동들이 호상이동에 의한 허가를 요구하지 않는다는것을 의미한다.

② 이동범위

*M*scope*C*는 *P*의 이동에 대한 시간경과를 표시한다. *M*Scope*S*는 마감기한안에서의 *P*의 정상마감에서 *S*에로의 이행이다. *M*Scope*L*은 아래한계위반에서의 *P*의 비정상적인 마감에서 *L*에로의 이행이다. 마찬가지로 *M*Scope*U*는 웃한계위반에서의 *P*의 비정상마감에서 *U*에로의 이행이다. *M*Scope*K*는 열쇠가 정합되지 않았을 때 *K*에로의 이행이다. *M*Scope*R*는 우선권위반에서 *R*에로의 이행이다. *M*Scope*I*는 모든 새치기에서 *I*에로의 이행이다.

2. δ 계산론리의 응용

CryptoLocker실례를 가지고 보안을 위한 업무처리령역에로의 δ 계산론리의 적용가능성을 론의하기로 한다. CryptoLocker는 Microsoft Windows로 실행되는 콤퓨터들을 목표로 하는 ransomware trojan이다.[3] 이것은 체계에 침입하여 특수한 화일들을 감염시키고 회복시키는데 보상금을 요구한다.

CryptoLocker의 동작을 다음과 같이 δ 계산론리로 서술할수 있다.

$$\text{System} ::= S_{(0)}\left[E_{(0)} \parallel P_{1(m)} \parallel P_{2(m)} \parallel P_{3(m)} \parallel \cdots \parallel P_{n(m)}\right](\text{SYS, NET}) \parallel (F\ \text{in})^{\infty}$$

$$F ::= P_1 \parallel P_2 \parallel P_3 \parallel \cdots \parallel CL \parallel CP \parallel \cdots$$

$$CL ::= \text{in}_{t_1} S.\text{new}\ I.\text{new}\ D.\text{nil}$$

$$D ::= \text{kill}\, CL.\text{exit}$$

$$I ::= \overline{\text{in}}^{p}_{t_2} E.W$$

$$W ::= \text{nil}\ \Delta_{\infty}(\text{nil, nil, SYS(trigger).NET}(\overline{\text{download}}).W)$$

$$CP ::= \text{NET(download).in}_{t_3} S.\text{new}\ K.\overline{\text{get}}^{p}_{t_4} K.(\text{get}^{p}_{tij}\ P_i.\text{new}\ P_i'.\text{kill}\ P_i)^n.$$

$$(\text{SYS.(ReqM).SYS(sendM)})\Delta_{t_5}(\text{OP, kill}\, K.\text{exit, nil})$$

$$OP ::= \left(\text{get}^{p}_{t_k}\ P_i'.\text{new}\ P_i.\text{kill}\ P_i'\right)^n.\text{exit}$$

System은 우에서 보여주는것처럼 두 부분 즉 *S*와 그것의 내부부분품들과 외부처리가 *S*에 들어갈수 있도록 허가를 주기 위한 처리로 구성된다.

*S*는 체계에서 화일들을 검색하기 위하여 *E*를 포함하며 다른 각이한 처리들도 포함한다. *E*를 제외한 다른 처리들은 같은 우선권 *m*을 가진다.

CryptoLocker는 다음의 단계로 동작한다.

① *CL*이 *S*에 들어간 후 *CL*은 *I*와 *D*처리들을 창조한다.

② *D*는 *CL*을 종결시키고 자체로 끝난다.

③ *I*는 우선권에 기초한 비동기적이동에 의하여 강제로 *E*로 들어가서 *E*의 숨겨진 처리로 되며 *W*상태에서 대기한다.

④ *S*의 특수한 동작에 의하여 Trigger를 만나면 *W*는 *S*의 포구 NET를 통하여 *CP*를 내리적재한다.

⑤ *CP*가 내리적재되여 *S*로 들어갈 때 *S*의 외부에서 열쇠를 창조하고 그 열쇠를 가지고 들어간다.

⑥ *CP*는 다른 처리들을 가지고 들어와 그 열쇠로 처리의 암호변경된 판본을 창조하고 원래판본의 처리들을 강제로 종결시킨다.

⑦ 암호바꾸기가 일단 수행되면 마감기한을 가진 체계 *S*를 통하여 보상금을 요구한다.

⑧ 보상금이 제때에 지불되면 암호바꾸기가 종결되고 암호변경되였던 모든 처리들이 복귀된다.

⑨ 보상금이 제때에 지불되지 않으면 *CP*는 열쇠를 삭제하고 자체로 종결된다.

CryptoLocker의 동작을 서술하기 위하여 *CL*의 이동과 *E*에로 들어가 숨은 처리로 되는 *I*의 주입, 내리적재와 *CP*의 암호바꾸기들은 이동활동으로 표시된다. 또한 *W*가 대기상태에서 Trigger에 응답하도록 하기 위하여 이동범위를 Trigger에 의한 새치기에 대하여 정의한다.

실례의 실행전의 ITL(In The Large)보기와 ITS(In The Small)보기를 그림 1과 2에 보여주었다.

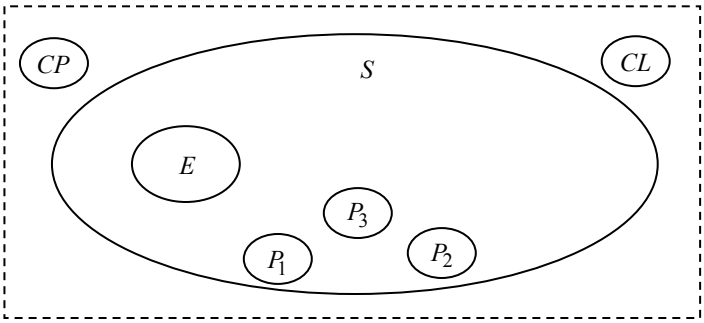

그림 1. 실행전의 ITL보기

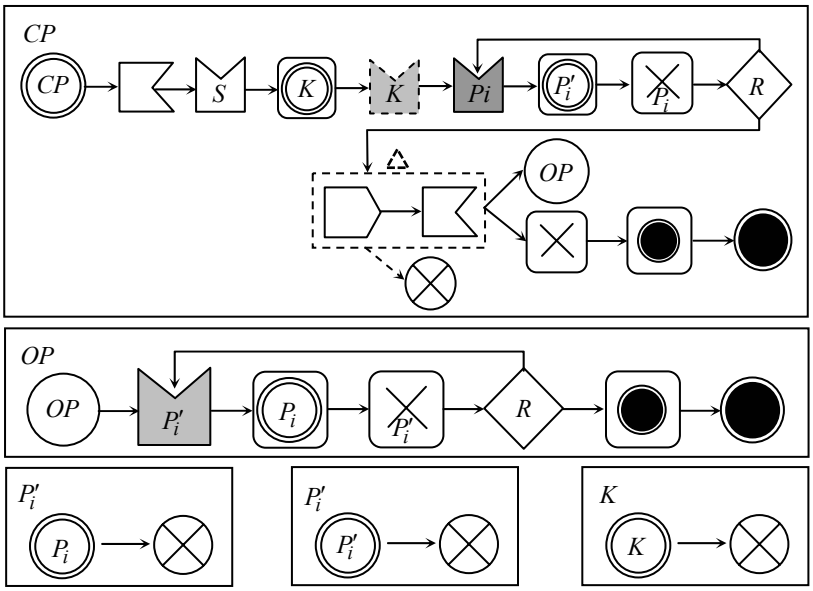

그림 2. 실행전의 ITS보기

CryptoLocker실례의 실행은 다음의 통신 및 이동활동들에 의하여 결정된다.
동기적 및 비동기적활동들을 표 4에 보여주었다.

표 4. 동기적 및 비동기적활동

동기적통신과 이동	비동기적통신과 이동
$CL :: \mathrm{in}_{t_1} S \sim S :: F\mathrm{in}$	$I :: \overline{\mathrm{in}}^{p}_{t_2} E$
	$CP :: \overline{\mathrm{get}}^{p}_{t_4} K$
$W :: \mathrm{NET}(\overline{\mathrm{download}}) \sim CP :: \mathrm{NET}(\mathrm{download})$	$CP :: \overline{\mathrm{get}}^{p}_{t_2} P_i$
$CL :: \mathrm{in}_{t_3} S \sim S :: F\mathrm{in}$	$OP :: \overline{\mathrm{get}}^{p}_{t_7} P_i'$

매 통신과 이동들에 대하여 아래첨수로 서술된 마감기한들이 있다. 이 통신들과 이동들에 기초하여 실례는 다음의 단계로 실행된다. 여기서 Time은 실행에 대한 가상시간이다.

① $\mathrm{Time} = t_1$: 처리 CL이 체계 혹은 처리 S에 들어간다.

② $\mathrm{Time} = t_1 + 1$: 처리 I가 창조된다.

③ $\mathrm{Time} = t_1 + 2$: 처리 D가 창조된다.

④ $\mathrm{Time} = t_1 + 3$: CL이 종결된다.

⑤ $\mathrm{Time} = t_1 + 4$: 처리 D가 종결된다.

⑥ $\mathrm{Time} = t_1 + t_2 + 1$: 처리 I가 E로 들어가 E의 숨은 처리 W로 된다.

⑦ $\mathrm{Time} = t_1 + t_2 + 1 + a$: 처리 W가 trigger를 얻고 CP를 적재한다. 여기서 a는 trigger를 받기 위하여 기다리는 시간이다. 이때 a에 대한 조건은 없다.

⑧ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3$: CP가 적재되고 S에 들어간다.

⑨ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1$: 처리 K가 창조된다.

⑩ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4$: 처리 K가 CP에 들어온다.

⑪ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n$: CP가 P_1로부터 P_n까지의 처리들의 암호를 변경시킨다.

다음단계는 보상금지불에 따라 두가지 가능한 선택이 있다.

보상금이 지불된 경우에는 다음과 같다.

⑫ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + b$: 처리 CP가 지불을 검사한다. 여기서 b는 보상금이 지불되는 시간이다. 이때 $b < t_6$이다.

⑬ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + b + (t_7 + 2) \times n$: CP는 P_1'부터 P_n'의 암호변경된 처리들을 복귀시킨다.

⑭ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + b + (t_7 + 2) \times n + 1$: CP는 자체로 종결된다.

보상금이 지불되지 않은 경우에는 다음과 같다.

⑫ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + t_6$: 마감기한 t_6이 끝난다.

⑬ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + t_6 + 1$: K가 자체로 종결된다.

⑭ $\mathrm{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + t_6 + 2$: CP가 자체로 종결된다.

단계 7과 12에서의 ITL보기를 그림 3과 4에 보여주었다.

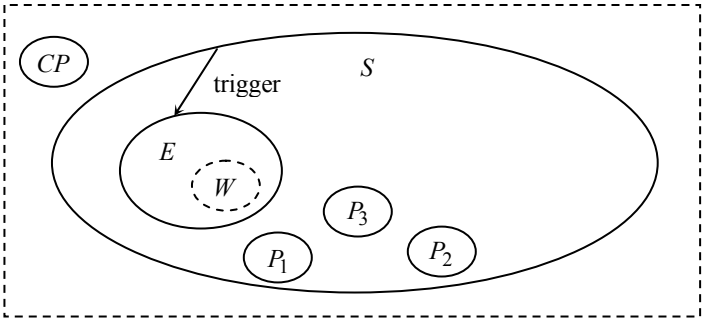

그림 3. 단계 7에서의 ITL보기

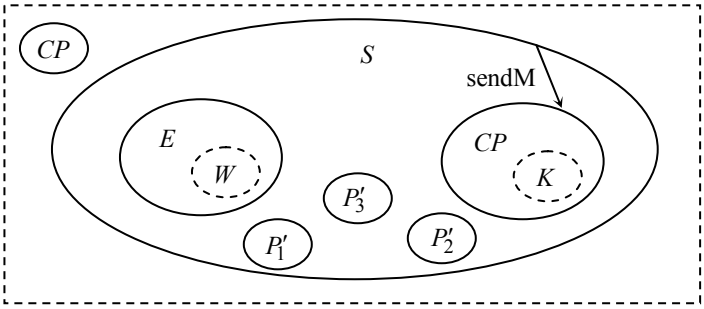

그림 4. 단계 12에서의 ITL보기

실행될 때 매 이동들은 ITL보기의 구성을 변경시킨다.

실례로 그림 3은 실행단계 7에서(이때 $\text{Time} = t_1 + t_2 + 1 + a$) trigger를 얻는 때의 실례에 대한 ITL보기이며 그것의 이행은 다음과 같이 수행된다.

$$S[E[\overline{W}] \| P_1 \| P_2 \| P_3] \| CP \xrightarrow{\tau} S[E[\overline{W}] \| P_1 \| P_2 \| P_3] \| CP$$

그림 4는 실행단계 12에서($\text{Time} = t_1 + t_2 + 1 + a + t_3 + 1 + t_4 + (t_5 + 2) \times n + b$) 요청한 보상금을 받는 때의 실례의 ITL보기이다.

모든 처리들의 암호는 변경되였으며 그 이행은 다음과 같이 수행된다.

$$S[E[\overline{W}] \| P_1' \| P_2' \| P_3' \| CP[\overline{K}]] \xrightarrow{\tau} S[E[\overline{W}] \| P_1' \| P_2' \| P_3' \| CP[\overline{K}]]$$

이 실례는 체계 S에서 일어나게 되는 안전한 업무거래에 대한 표준을 보여준다. 만일 어떤 처리가 그 거래에 간섭하거나 새치기하려고 한다면 반드시 S에 들어가서 거래할 때 그안에 물리적으로 머물러있어야 한다. 다시말하여 S를 업무거래에 대하여 일종의 안전지역으로 볼수 있다. S가 CryptoLocker와 같은 그 어떤 침입으로부터도 안전하게 하려면 침입자가 그 지역 즉 S의 밖에 있거나 거래할 때에 그 지역안에 있지 말아야 한다.

3. 효과성분석

선행연구들에서 연구된 π 계산론리는 분산이동형실시간환경에서의 업무처리들을 모형화하는데서 효과적이지만 이동들이 값통과통신의 개념으로 표현되기때문에 처리들의 실제한 이동들을 표현하는데 적합하지 않다. 특히 분산환경에서 업무응용의 보안은 그자

체에 의하여 완전히 담보될수는 없지만 알맞는 외부환경에 의해서 유지될수 있다는 측면에서 볼 때 선행한 처리대수들은 목표체계 그자체의 요구되는 특성들을 명세화할수 있지만 체계와 그 환경사이의 관계를 충분히 규정하는데서는 일정한 제한성이 있다. 그러나 δ 계산론리에서는 이러한 제한들을 환경으로부터의 동기성과 우선권, 마감기한의 속성들을 가지고 그 관계들을 명세화하고 분석함으로써 극복할수 있다. 즉 δ 계산론리는 처리들을 직접적으로 명백히 표현할수 있을뿐아니라 처리들을 본문적으로 또 그라프적방법으로도 다 표시할수 있다. 따라서 이동에 관한 업무처리들의 동작을 리해하기가 훨씬 쉽다. 또한 환경으로부터 특히 업무보안측면에서 독립적인 체계의 안전한 상태로부터 외부환경에 의하여 초래되는 목표체계의 불안전한 상태를 구별하는데 유리하다.

맺 는 말

론문에서는 새로운 δ 계산론리를 제안하고 세계적으로 널리 알려진 ransomware의 한 형태인 CryptoLocker체계를 δ 계산론리로 모형화하는 과정을 통하여 δ 계산론리가 분산이동형실시간환경에서 업무응용들에 대한 안정성과 보안문제 혹은 규약들의 다양성을 명세화하고 분석하는데 리용될수 있다는것을 론증하였다. 또한 외부환경으로부터 다른 처리들의 출입으로 하여 초래되는 안전이 보장되지 않는 수많은 업무상태들을 검출하고 방지하는데 리용될수 있다는것을 확인하였다.

참 고 문 헌

[1] **김일성**종합대학학보(자연과학), 63, 10, 32, 주체106(2017).

[2] Edmund M. Clarke et al.; Communications of the ACM, 52, 11, 74, 2009.

[3] A.Whitmore et al.; The Internet of Things, 17, 2, 261, 2015.

주체108(2019)년 8월 5일 원고접수

Study on Modeling Secure Movement of Distributed Mobile Processes Using Delta-Calculus

Pak Ji Hye, *Kim Yong Sok*

In this paper we proposed δ −Calculus to model secure movements of distributed mobile processes and illustrated the usability of this calculus by modeling the CryptoLocker system.

Key words: Delta-Calculus, distributed mobile process, security