УДК 681.3

В.А. Артамонов, Е.В. Артамонова

ФОРМАЛЬНАЯ МОДЕЛЬ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ КРИТИЧНЫХ ОБЪЕКТОВ ИНФОРМАТИЗАЦИИ

Определения

В модели используются следующие определения.

Классификация – обозначение, накладываемое на информацию, отражающее ущерб, который может быть причинен неавторизованным доступом, включающее уровни: ***TOP SECRET***, ***SECRET*** и т.д. и множество меток (***"CRYPTO", "NUCLEAR"*** и т.д.). Множество классификаций и отношение между ними образуют решетку.

Степень доверия пользователю – уровень благонадежности персоны. Каждый пользователь имеет степень доверия, и операции, производимые системой для данного пользователя, могут проверить степень доверия пользователю и классификацию объектов, с которыми он оперирует.

Пользовательский идентификатор – строка символов, используемая для того, чтобы отметить пользователя системы. Для использования системы пользователь должен предъявить ей пользовательский идентификатор, и система должна провести аутентификацию пользователя. Данная процедура называется ***login***. Каждый пользователь должен иметь уникальный идентификатор.

Пользователь – персона, уполномоченная для использования системы.

Роль – уполномоченная работа, выполняемая пользователем (например, удаление, распространение или понижение классификации объектов).

Объект – одноуровневый блок информации. Это минимальный блок информации в системе, который имеет классификацию. Объект не содержит других объектов, он не многоуровневый.

Контейнер – многоуровневая информационная структура. Имеет классификацию и может содержать объекты (каждый со своей классификацией) и (или) другие контейнеры. Файл – это ***контейнер.*** Некоторые структуры файла могут быть *контейнерами*. Различие между *объектом* и *контейнером* базируется на типе, а не на текущем содержимом: если один из файлов данного типа является контейнером, то все остальные файлы данного типа являются контейнерами, даже если некоторые из них содержат только объекты или пусты. Устройства такие, как диски, принтеры, ленты, сетевые интерфейсы и пользовательские терминалы –

контейнеры.

Сущность – любая именованная (т.е. имеющая имя) составляющая системы ИТ, например о*бъект* или *контейнер*.

Требование степени доверия контейнеров – атрибут некоторых *контейнеров*. Для некоторых *контейнеров* важно требовать минимум степени доверия, то есть пользователь, не имеющий соответствующего уровня благонадежности, не может просматривать содержимое *контейнера*. Такие *контейнеры* помечаются соответствующим атрибутом – **CCR.** Например, пользователь, имеющий степень доверия ***CONFIDENTAL***, не может просматривать ***CONFIDENTAL*** параграф сообщения, помеченного ***TOP SECRET***, если оно содержится в **CCR -** *контейнере*.

Идентификатор (ID) - имя сущности без ссылки на другие сущности, например, имя файла есть идентификатор этого файла. Обычно все сущности имеют идентификатор.

Ссылка на сущность ***прямая****,* если это идентификатор *сущности*.

Ссылка на сущность ***косвенная****,* если это последовательность двух или более имен *сущностей* (из которых только первая - *идентификатор*). Пример: "текущее сообщение, первый абзац, вторая строка".

Операция - функция, которая может быть применена к сущности. Она может позволять просматривать или модифицировать сущность. Некоторые операции могут использовать более одной сущности (пример - операция копирования).

Множество доступа - множество троек ***(Пользовательский Идентификатор или Роль, Операция, Индекс операнда)***, которые связаны с *сущностью*. Операция, которая может быть специфицирована для особых сущностей, зависит от типа данной сущности. Если операция требует более одного операнда, индекс операнда специфицирует позицию, на которой ссылка на данный операнд может появиться в операции.

Сообщение является *контейнером*. *Сообщение* включает поля *куда, откуда, время, предмет, текст, автор*. Чертежные сообщения включают поле чертежа.

Предположения

Мы предполагаем существование множества возможных пользователей и множества возможных сущностей. С помощью них мы определяем состояние системы и безопасное состояние. Далее мы определяем систему и ее историю и вводим ограничения на переход из одного состояния в другое. Система, все переходы которой удовлетворяют данным ограничениям, безопасна для переходов. В заключение определяется безопасная история и безопасность системы.

Структура формальной модели спроектирована для упрощения определения предусловий и постусловий для операций системы.

Состояние системы

Определим состояние системы и то, что необходимо для того, чтобы

система была безопасной.

OP – множество операций.

L – множество уровней безопасности; ≥ – частичный порядок на L такой, что **(L, ≥)** – решетка.

UI – множество идентификаторов пользователей.

RL – множество ролей пользователей.

US – множество пользователей. $\forall u \in US$, **CU(u)** $\in$ **L** – степень доверия к пользователю **u,** **R(u)** $\subseteq$ **RL** – множество ролей, для которых авторизован пользователь **u** и **RO(u)** $\subseteq$ **RL** – текущая роль пользователя **u**.

RF – множество ссылок. Данное множество подразделяется на множества **DR** – прямых ссылок и **IR** – множество косвенных ссылок. Хотя истинная природа ссылок неважна, мы предполагаем, что прямые ссылки могут быть пронумерованы целыми числами. В модели мы рассматриваем каждую ссылку как последовательность, состоящую из простых чисел; для прямых ссылок – одно число, например **<17>,** для косвенных – конечная последовательность из двух или более целых чисел, например $<n_1,...,n_m>$, где n_1 – прямая ссылка.

VS – множество строк (символьных или битовых). Данные строки обозначают значение сущности (например содержимое файла).

TY – множество типов данных в системе, включающие **"DM"** для чертежных сообщений и **"RM"** для освобожденных сообщений.

ES – множество сущностей системы. $\forall \in$**ES,** **CE(e)** $\in$ **L** – классификация **e**.

AS(e) $\subseteq$ **(UI**$\cup$**RL)**×**OP**×**N** – множество троек, которые составляют множество доступа к **e**. **(u, op, k)** $\in$ **AS(e),** если **u** – идентификатор пользователя или его роль, в которой он авторизован для того, чтобы исполнить операцию **op** со ссылкой к **e** как **k**-параметру операции **op**.

T(e) $\in$ **TY** – тип сущности **e**. **V(e)** $\in$ **VS** – значение сущности **e**. Если **T(e) = DM** или **T(e) = RM**, то **V(e)** включает поле освобождения **RE(e),** которое, если оно не пусто, содержит идентификатор пользователя.

ES содержит подмножество контейнеров. Для любой сущности **e** в данном множестве контейнеров **H(e)** $= <e_1,...,e_n>$, где e_i - сущность, содержащаяся в **e**. **CCR(e)** - истинно, если **e** помечено **CCR**, иначе - ложно. Если $T(e_1) = T(e_2)$, то e_1 и e_2 – оба контейнеры или объекты.

Множество **O** устройств вывода – подмножество множества контейнеров. Элемент **o** $\in$ **O** рассматривается как домен из двух функций, рассматриваемых далее. **D(o)** – множество упорядоченных пар $[(x_1,y_1), (x_2,y_2),..., (x_n,y_n)]$, где:

– каждое y_i отображается на **o**;

– каждое x_i – пользователь или сущность и соответствующее y_i - ссылка, пользовательский идентификатор или результат применения описанных выше функций к x_i. Требуется, чтобы $(x,V(x)) \in D(o) \rightarrow x \in H(o)$.

CD(o) дает максимум классификации информации, которая может быть отображена на **o**. Это позволяет использовать **CE(o)** как текущую верхнюю

границу классификации для информации, которая может быть отображена на устройстве вывода так, что пользователи могут ограничивать классификацию вывода до уровня, меньшего чем максимально разрешенный.

Состояние отображает подмножество пользовательских идентификаторов и ссылок в элементы **US** и **ES**, которые представляют соответствующие свойства. Состояние также отображает подмножество пользовательских идентификаторов, которые "существуют" в ссылках к устройствам вывода. Определим три отображения:

– **id** функция **U** – отображение один к одному от подмножества **UI** к подмножеству **US**.

– *функция ссылок* **E** – отображение подмножества **RF** к **ES** такое, что $\forall n \geq 2$ $E(<i_1,...,i_n>) = e$, если $E(<i_1,...,i_{n-1}>) = e^*$, где e^* – контейнер, так что **e** – i_n элемент $H(e)$. Для некоторой ссылки **r**, если $E(r)=e$, мы говорим, что **r** – ссылка к **e** (относительно **E**).

– **login** функция, **LO** – отображение один к одному из подмножества **UI** в **RF**.

В соответствии со ссылочной функцией **E**, каждая ссылка в форме $<n_0,...,n_m>$ к сущности **e** соответствует пути из сущностей $<e_0,...,e_m>$ таких, что каждая $e_i \in rng(E)$, e_0 – прямая ссылка к $<n_0>$, и для всех положительных целых $i, m \geq i$, $e_i - n_i$ сущность в контейнере e_{i-1}. О таких косвенных ссылках будем говорить, что они базируются на сущности e_j, $m > j \geq 0$.

Определение 1. Состояние системы **s** – упорядоченная тройка **(U, E, LO)**, где **U** – **id** функция, **E** – функция ссылок, **LO** – **login** функция такие, что $dom(LO) \subseteq dom(U)$ и $rng(LO) \subseteq dom(E \cap (RF \times O))$. Требуется: если $o \in rng(E) \cap O$ и $(x,y) \in D(o)$, то $x \in rng(E) \cup rng(U)$ для гарантии того, что только информация о пользователях или сущностях, которая существует в данном состоянии может быть просмотрена для некоторой ссылки **r**, $(x,r) \in D(o) \rightarrow E(r)=x$. В заключение требуется $E(LO(u_1)) = E(LO(u_2)) \rightarrow u_1 = u_2$ для предотвращения вхождения двух пользователей с одного терминала.

Для системного состояния $s=(U, E, LO)$ определим $E(r)$ как r_s, $U(u)$ как u_s и $E(LO(u))$ как u^s.

Определение 2. Состояние s безопасно, если $\forall x,y \in rng(E)$, $Vo \in O \cap rng(E)$, $\forall w \in dom(LO)$, $H \forall u \in rng(U)$:

$x \in H(y) \rightarrow CE(y) \geq CE(x)$,

$x \in H(w_s) \rightarrow CE(w_s) \geq CE(x)$,

$(x, V(x)) \in D(o) \rightarrow (x, CE(x)) \in D(o)$,

$RO(u) \subseteq R(u)$,

$CD(o) \geq CE(o)$.

Безопасность системы

Дадим определение, что такое система и что нужно для того, чтобы она была безопасна в информационном отношении.

Определение 3. Система **Σ** – четверка (I, S, s_0, T), где

I – множество правильно сформированных системных запросов, в котором $\forall$ **i** $\in$ **I** – форма **<op,$x_ь$ x_2,... x_n>, x_J** $\in$ **RF** $\cup$**UI** $\cup$**VS** и **op** $\in$ **OP**;

S – множество возможных состояний системы;

s_0 – отмечает специальное состояние, называемое начальным;

T – переход системы, функция из **UI** ×**I** ×**S** в **S**.

Определение 4. История **П** – функция из множества положительных целых **N** в **UI** × **I** ×**S** такая, что третий элемент **П s_0** и $\forall$ **n** $\in$ **N**, если **П(n) = (u,i,s) и П(n+1) = (u*, i*, s*),** то **T(u, i, s) = s*.**

До определения операции, возможно модифицирующей сущность, отметим, что ссылочная функция **E** и состояние s индуцируют множество функций на ссылках, которые являются двойниками ко множеству функций, определенных выше на сущностях. Например, существует функция **V_s** такая, что **V_s(r) = V(r_s).** Также определим предикат **H_s** такой, что **H(r_s) = <r_{s1},...,r_{sn}>.** Каждый двойник является видимой для пользователя соответствующей функции сущностей. Назовём их ссылочными двойниками и используем для определения того, что означает быть эквивалентными для двух состояний, исключая множество ссылок.

Состояния **s = (U, E, LO)** и **s* = (U*, E*, LO*)** эквивалентны, исключая некоторое множество ссылок ρ, если:

1) **U=U*,**
2) **LO=LO*,**
3) **dom(E)=dom(E*),**
4) для некоторой функции сущности **F**, исключая **V**, **F=F_s*** , и
5) для некоторой ссылки **r** $\in$ **dom(E) ~ ρ, V_s(r) = V_s*(r).**

Определим потенциальную модификацию : **u, i, s** потенциально модифицируют **r**, если **$\exists s_1 s_1$*:s_1** эквивалентно **s**, исключая возможное множество ссылок и **T(u_i,s_1)=s_1*** и для некоторой функции сущностей **F**, **F(r_{s1})≠F(r_s1*).**

Назовем **y** вкладывающим фактором в том случае, если **y=r** или $\exists$ **s_1**, в том же смысле, что и выше и **s2, s2* : s_1** и **s_2** эквивалентны, исключая [**y**] и **T(u, i, s_1)=s_2* и F(r_{s2} V F(r_sI*).**

То есть **u, i, s** потенциально модифицируют **r**, если существует некоторое (второе) состояние, которое может отличаться от **s** значениями некоторых сущностей, и **T** отображает **u, i** и это состояние в третье состояние, в котором некоторая функция сущности **F** (значение, контейнер, множество доступа и т.д.) на **r** отличается от второго состояния. Вкладывающими факторами являются **r** и те сущности, чьи значения воздействуют на конечное **F(r).**

Для каждого соответствующего двойника и каждой функции, определенной на пользователях, определяется уникальная операция, которая изменяет существующую сущность или пользователя в соответствии с этой функцией. Изменение областей определения **E** или **U** (создание или уничтожение сущностей или ссылок) также предполагается происходящим

при определенных запросах. Формальное определение операции освобождения, определенное ниже – единственное исключение в этом предположении; оно изменяет тип **r** и, возможно, поле освободителя **r**.
Истинная природа этих операций неважна, так как предположения включены единственно для простоты представления. Их назначение не управлять реализацией команд, которые изменяют различные части сущностей, а устранять проблему неспецифицированных эффектов в формальной модели (например разрешение просматривать сообщения, помеченные **CCR** не есть разрешение очищать отметку **CCR**). Реализация команд, изменяющих более чем одну часть отдельной сущности, соответствует последовательности формальных операций. Для данной реализации это соответствие определяется семантикой реализации командного языка. Для однажды определенного соответствия такого, что относящиеся к безопасности эффекты для каждой пользовательской команды ясны, можно изменить множество команд реализации с соответствующе измененным множеством доступа. Несмотря на это, благоразумие диктует то, что модификация, которая может быть сделана только офицером безопасности (изменяющим степень доверия к пользователю), должна быть ограничена так, чтобы в любой реализации существовала единственная команда, выполняющая эту операцию.

Следующие ограничения на переходы системы ведут к определению истории безопасности и безопасности системы. В нижеследующих выражениях там, где не отмечена квантификация, предполагается квантор универсальности.

Определение 5. Переход системы **T** - безопасен по доступу, если $\forall$ **u**, **i, s, s*, [(op** $\in$ **I** $\cap$ **OP** и $r_k \in$ **I** $\cap$ **RF)** $\rightarrow$ **((u,op,k)** $\in$ **AS(E(**r_k**))** или $\exists$**I** $\in$ **RO(**u_s**)** и **(I,op,k)** $\in$ **AS(E(**r_k**)))]** или **s = s*.**

Определение 6. Переход **T** безопасен по копированию, если $\forall$ **u, i, s, s*** : **T(u,i,s)=s* x** – потенциально модифицируется через вкладывающий фактор **y** $\rightarrow$ **CE(**x_s**)** $\geq$ **CE(**y_s**).**

Определение 7. Переход **T** безопасен по **CCR** $\forall$**u, i, s, s*:T(u,i,s)=s*,**

r $\in$ **I** $\cap$ **IR** базируется на **y** и **CCR(y)** и **z** потенциально модифицируется через вкладывающий фактор **r** $\rightarrow$ **CE(**u_s**)** $\geq$ **CE(y).**

Определение 8. Переход **T** безопасен по трансляции, если $\forall$ **u, i, s, s*: T(u,i,s)=s*, x** $\in$ **DR** и **(**x_S***, x)** $\in$ **D(**u_s***)** $\rightarrow$ $\exists$ **r** $\in$ **I** $\cap$ **RF,** r_s**=**x_s и (**r** базируется на **z** и $\rightarrow$ **CE(**u_s**)**$\geq$ **CE(z)).**

Определение 9. Переход T безопасен для множества, если $\forall$ **u, i, s, s*: T(u,i,s)=s*,**

(a) $\exists$ **o** $\in$ **dom(E** $\cap$ **(RFxO)), CD(**o_s**)** $\neq$ **CD(**o_s***)** или $\exists$ **x** $\in$ **dom(U), CU(**x_s**)** $\neq$ **CU(**x_s***)** или **R(**x_s**)**$\neq$ **R(**x_s***),**

(b) $\exists$ **x** $\in$ **dom(U)** и **R(**x_s**)** $\neq$ **R(**x_s***)** $\rightarrow$ u_s=x_s или **security_officer** $\in$ **RO(**u_s**).**

Определение 10. Переход **T** безопасен по снижению уровня, если $\forall$**u**,**i,s,s*: T(u,i,s)=s*, x** $\in$ **dom(E~(RF**$\times${**us**}**))** и **CE(**x_s**)** > **CE(**x_s***)** $\rightarrow$ **downgrader** $\in$ **RO(**U_S**).**

Определение 11. Переход **T** безопасен по освобождению, если $\forall$ **u, i, s**, **s*: T(u,i,s)=s*, (T(x_s)=RM→T(x_s*)=RM** и **RE(x_s)=RE(x_s*))** и **(T(xs) ≠RM** и **T(x_s*) = RM → RE(x_s*)=u, ∃r: r_s=x_s, i** – операция *освободить* <***releaser, r***>, ***releaser*∈*RO*(U_S)** и **T(X_S) = DM**).

Определение 12. Переход является безопасным, если он безопасен по доступу, копированию, по **CCR**, трансляции, множеству, снижению уровня и освобождению.

Определение 13. История безопасна, если все ее состояния и переходы безопасны.

Определение 14. Система безопасна, если все ее истории безопасны.

Обсуждение

Основной идеей проведенной формализации является взгляд на систему информационных технологий как на взаимоотношения между состояниями системы и самой системой. В данном контексте состояние системы состоит из сущностей и их отношений, и система добавляет к данным отношениям пользователей пользовательские операции над сущностями. Следовательно, все ограничения на свойства пользователей (в частности, ограничение, что для любого **u RO(u) ⊆ R(u))** включены в определение безопасности системы. Данный взгляд разделяет состояние системы и систему в терминах статики в противоположность динамическим свойствам.

Статические свойства – свойства, которые сохраняются для всех состояний системы, и, следовательно, могут быть проверены для изолированного состояния системы.

Динамические свойства состояния – те состояния, которые нуждаются в проверке взаимоотношений между состояниями безопасности системы, и, следовательно, могут быть проверены только исследованием двух или более состояний. В определение безопасности состояния включены только статические свойства.

Принципиальной трудностью, возникающей при формализации модели, является представление "копирования", "просмотра", "вывода" и авторизованной операции. Семантика для копирования, включенная в определения "потенциальной модификации" и "вкладывающего фактора", базируется на граничной интерпретации копирования. Информация считается копируемой не только тогда, когда она непосредственно переносится из одной сущности в другую, но и когда она дает потенциальный вклад в другую сущность. Например, если операция сканирует файл сообщений **А** и копирует сообщения, выбранные фильтром **Ф** в файл сообщений **Б**, то и **А**, и **Ф** являются потенциальным вкладом в модификацию **Б** (и, следовательно, субъектом для ограничений, вызванных безопасностью копирования и **CCR** – безопасностью), даже если и **А** и **Ф** – пусты. Семантика для просмотра проста: сущность может быть просмотрена, если операция делает ее членом выходного контейнера.

В формализации вывод системы интерпретируется как множество

контейнеров, другие сущности, части сущностей, ссылки и классификации, которые видны пользователю и интерпретируются как копирующиеся в контейнер выхода. Ссылки ясно включены как часть выхода, так как одинаковые операции, примененные к одинаковым сущностям, могут вызвать разные результаты, в зависимости от того, как они выводятся – напрямую или косвенно. Для реализации этого ограничения система должна распознавать ссылки как особую часть вывода.

Формализация концепции авторизованного вывода трудна, так как семантика авторизованных операций неспецифицирована. Определение безопасности доступа требует того, что если операция изменяет состояние системы (кроме вывода сообщения об ошибке), то для каждой сущности во множестве операций пользователь или роль, операция и индекс операнда присутствовали во множестве доступа. Неавторизованные операции не должны изменять состояние системы, исключая сообщения об ошибке.

Основная теорема безопасности для формальной модели

Теорема 1. Любое состояние системы $\sum$ безопасно, если безопасно $\mathbf{s_0}$ и **T** удовлетворяет следующим условиям для всех $\mathbf{u, i, s, s^*: T(u, i, s)=s^*}$ и для всех $\mathbf{x,y \in RF, w \in US:}$

1. $\mathbf{x_s \notin H(y_s)}$ и $\mathbf{x_s^* \in H(y_s^*) \rightarrow CE(y_s^*) \geq CE(x_s^*).}$

2. $\mathbf{x_s \in H(y_s)}$ и $\mathbf{CE(y_s^*) < CE(x_s^*) \rightarrow x_s \notin H(y_s^*).}$

3. $\mathbf{x_s \notin H(w_s)}$ и $\mathbf{x_s^* \in H(w_s^*) \rightarrow CU(y_s^*) \geq CE(x_s^*).}$

4. $\mathbf{x_s \in H(w_s)}$ и $\mathbf{CU(w_s^*)\ \ CE(x_s^*) \rightarrow x_s \notin H(w_s^*).}$

5. $\mathbf{(x_s, V(x_s)) \notin w_s}$ и $\mathbf{(x_s^*, V(x_s^*)) \in w_s^* \rightarrow (x_s^*, CE(x_s^*)) \in w_s^*.}$

6. $\mathbf{(x_s, V(x_s)) \in w_s}$ и $\mathbf{(x_s^*, CE(x_s^*)) \notin w_s^* \rightarrow (x_s^*, V(x_s^*)) \in w_s^*.}$

7. $\mathbf{R(w_s) \neq R(w_s^*)}$ или $\mathbf{RO(w_s) \neq RO(w_s^*) \rightarrow RO(w_s^*) \subseteq R(w_s)\ .}$

8. $\mathbf{CE(w_s) \neq CE(w_s^*)}$ или $\mathbf{CD(w_s) \neq CD(w_s^*) \rightarrow CD(w_s^*) \geq CE(w_s^*)\ .}$

Условия (1) – (8) – являются необходимыми и достаточными для того, чтобы система была безопасна в любом состоянии, достижимом из $\mathbf{S_0}$.

Данная абстрактная модель может быть интерпретирована для доказательства безопасности других систем.

Формальная модель безопасности может является базисом для спецификации и реализации критичных систем информационных технологий.

БНТУ, г. Минск