CS 526 Information Security: Assignment 4

John Ross Wallrabenstein (jwallrab)

October 1, 2010

[redacted]

2 Problem 2

2.1 (a)

True.

2.2 (b)

We assume that *"in order to support verification of"* refers to the ability to ascertain whether or not the integrity and confidentiality of the system still hold. We are *not* addressing whether or not availability is more important than integrity or confidentiality, as that would depend on the particular system under consideration. Additionally, we are *not* addressing whether or not integrity or confidentiality can be supported without availability, as the question specifically pertains to the *verification* of integrity or confidentiality. Finally, we do not consider "verification" to include the assumption that if there is no availability whatsoever, and the system's integrity/confidentiality was not violated prior to the loss of all availability, you can "verify" they are still in tact because no one could access

them. We consider "verification" to explicitly mean that the data is accessed in some way to verify neither integrity nor confidentiality have been compromised.
This statement, under the above assumptions, is clearly true. That is, it is impossible to verify the integrity or confidentiality of a system if the system is unavailable. That is, some degree of availability must be present, or no user can access the system to verify that its integrity or confidentiality has not been compromised.

3 Problem 3

3.1 Release-Write

Definition The *release-write* rule enables a subject s to request to release the right to write an object o. Represent this request as $r = (release, s, o, \underline{w}) \in R^{(1)}$, and let the current state of the system be $v = (b, m, f, h)$. Then $release - write$ is the rule $\rho_1(r, v)$:

Algorithm 1 Release-Write

if $r \notin \Delta(\rho_1)$ **then**
  $\rho_1(r, v) = (\underline{i}, v)$;
else
  $\rho_1(r, v) = (\underline{y}, (b - (s, o, \underline{w}), m, f, h))$;
end if

Theorem 3.1 *The release-write rule ρ_1 preserves the simple security condition, the *-property, and the ds-property.*

Lemma 3.2 *The release-write rule ρ_1 preserves the simple security condition.*

Proof Let v satisfy the simple security condition, and let $\rho_1(r, v) = (d, v^{`})$. Either $v^{`} = v$ or $v^{`} = (b - (s, o, \underline{w}), m, f, h)$, by the *release-write* rule. When $v^{`} = v$, we have that $v^{`}$ satisfies the simple security condition because $v$ does. In the latter case, we have $v^{`} = (b - (s, o, \underline{w}), m, f, h)$. For either choice of $v^{`}$, we have that $b^{`} - b = \varnothing$. That is, if $v^{`} = v$ then $b^{`} - b = \varnothing$ and the simple security condition is satisfied because v satisfies the condition. If $b^{`} \neq b$, then $\{(s, o, \underline{w})\} \notin b^{`}$ and we have that $b^{`} \subseteq b$. As $v$ satisfies the simple security condition, all rules in $b$ must also satisfy the simple security condition. Thus, $b^{`} \subseteq b$ and $f^{`} = f$ implies that $v^{`}$ will also satisfy the simple security condition.

Lemma 3.3 *The release-write rule ρ_1 preserves the *-property.*

Proof Let v satisfy the *-property, and let $\rho_1(r, v) = (d, v^{`})$. Either $v^{`} = v$ or $v^{`} = (b - (s, o, \underline{w}), m, f, h)$, by the *release-write* rule. When $v^{`} = v$, we have that $v^{`}$ satisfies the *-property because $v$ does. In the latter case, we have $v^{`} = (b - (s, o, \underline{w}), m, f, h)$. For either choice of $v^{`}$, we have that $b^{`} - b = \varnothing$. That is, if $v^{`} = v$ then $b^{`} - b = \varnothing$ and the *-property is satisfied because v satisfies the property. If $b^{`} \neq b$, then $\{(s, o, \underline{w})\} \notin b^{`}$ and we have that $b^{`} \subseteq b$. As $v$ satisfies the *-property, all rules in $b$ must also satisfy the *-property. Thus, $b^{`} \subseteq b$ and $f^{`} = f$ implies that $v^{`}$ will also satisfy the *-property.

Lemma 3.4 *The release-write rule ρ_1 preserves the ds-property.*

Proof Let v satisfy the ds-property, and let $\rho_1(r, v) = (d, v^{`})$. Either $v^{`} = v$ or $v^{`} = (b - (s, o, \underline{w}), m, f, h)$, by the *release-write* rule. When $v^{`} = v$, we have that $v^{`}$ satisfies the ds-property because $v$ does. In the latter case, we have $v^{`} = (b - (s, o, \underline{w}), m, f, h)$. For either choice of $v^{`}$, we have that $b^{`} - b = \varnothing$. That is, if $v^{`} = v$ then $b^{`} - b = \varnothing$ and the ds-property is satisfied because v satisfies the property. If $b^{`} \neq b$, then $\{(s, o, \underline{w})\} \notin b^{`}$ and we have that $b^{`} \subseteq b$. As $v$ satisfies the ds-property, all rules in $b$ must also satisfy the ds-property. That is, we have $m[s, o] \subseteq m^{`}[s, o] \forall s \in S, o \in O$. Thus, $v^{`}$ will also satisfy the ds-property.

As we have shown that *release-write* preserves the simple security condition, *-property and ds-property, we have proven Theorem 3.1.

3.2 Rescind-Execute

Definition The *rescind-execute* rule enables a subject s_α to request to rescind subject s_β's right to execute an object o. Represent this request as $r = (s_\alpha, r, s_\beta, o, \underline{e})$, and let the current state of the system be $v = (b, m, f, h)$. Then $rescind-execute$ is the rule $\rho_2(r, v)$:

Algorithm 2 Rescind-Execute

if $r \notin \Delta(\rho_2)$ **then**
  $\rho_2(r, v) = (\underline{i}, v)$;
else if $r \in \Delta(\rho_2) \wedge ((o \neq root(o) \wedge parent(o) \neq root(o) \wedge parent(o) \in b(s_\alpha : \underline{w})) \vee (parent(o) = root(o) \wedge canrescind(s_\alpha, o, v)) \vee (o = root(o) \wedge canrescind(s_\alpha, root(o), v)))$ **then**
  $\rho_2(r, v) = (\underline{y}, (b - (s_\beta, o, \underline{e}), m \wedge m[s_\beta, o] - \underline{e}, f, h))$;
else
  $\rho_2(r, v) = (\underline{n}, v)$
end if

Theorem 3.5 *The rescind-execute rule ρ_2 preserves the simple security condition, the *-property, and the ds-property.*

Lemma 3.6 *The rescind-execute rule ρ_2 preserves the simple security condition.*

Proof Let v satisfy the simple security condition, and let $\rho_2(r, v) = (d, v^‘)$. Either $v^‘ = v$ or $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$, by the *rescind-execute* rule. When $v^‘ = v$, we have that $v^‘$ satisfies the *-property because v does. In the latter case, we have $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$. Either $b - b^‘ = \varnothing$ or $b - b^‘ = \{(s_\beta, o, \underline{e})\}$. If $b - b^‘ = \varnothing$, then $b^‘ = b$ and $f' = f$ so $v^‘$ satisfies the simple security condition because v satisfies the condition. If $b^‘ \neq b$, then $b - b^‘ = \{(s_\beta, o, \underline{e})\}$. This implies that $b^‘ \subseteq b$ and $f^‘ = f$ so $v^‘$ satisfies the simple security condition because v satisfies the condition.

Lemma 3.7 *The rescind-execute rule ρ_2 preserves the *-property.*

Proof Let v satisfy the *-property, and let $\rho_2(r, v) = (d, v^‘)$. Either $v^‘ = v$ or $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$, by the *rescind-execute* rule. When $v^‘ = v$, we have that $v^‘$ satisfies the *-property because v does. In the latter case, we have $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$. Either $b - b^‘ = \varnothing$ or $b - b^‘ = \{(s_\beta, o, \underline{e})\}$. If $b - b^‘ = \varnothing$, then $b^‘ = b$ and $f' = f$ so $v^‘$ satisfies the *-property because v satisfies the property. If $b^‘ \neq b$, then $b - b^‘ = \{(s_\beta, o, \underline{e})\}$. This implies that $b^‘ \subseteq b$ and $f^‘ = f$ so $v^‘$ satisfies the *-property because v satisfies the property.

Lemma 3.8 *The rescind-execute rule ρ_2 preserves the ds-property.*

Proof Let v satisfy the ds-property, and let $\rho_2(r, v) = (d, v^‘)$. Either $v^‘ = v$ or $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$, by the *rescind-execute* rule. When $v^‘ = v$, we have that $v^‘$ satisfies the ds-property because v does. In the latter case, we have $v^‘ = (b - \{s_\beta, o, \underline{e}\}, m[s_\beta, o] - \underline{e}, f, h)$. For either choice of $v^‘$, we have that $b^‘ - b = \varnothing$. That is, if $v^‘ = v$ then $b^‘ - b = \varnothing$ and the ds-property is satisfied because v satisfies the property. If $b^‘ \neq b$, then $\{(s_\beta, o, \underline{e})\} \notin b^‘$ and we have that $b^‘ \subseteq b$. As v satisfies the ds-property, all rules in b must also satisfy the ds-property. That is, we have $m[s, o] \subseteq m^‘[s, o] \forall s \in S, o \in O$. Thus, $v^‘$ will also satisfy the ds-property.

As we have shown that *rescind-execute* preserves the simple security condition, *-property and ds-property, we have proven Theorem 3.5.

3.3 Write Rules

The *get-write* rule would only require that $f_o(o)$ *dom* $f_s(s)$ in addition to the requirements of the *get-append* rule. All of the other rules (*release*,*give*,*rescind*) could easily be generalized to accept any right $r \in \{read, append, execute, write\}$. It is trivial to see why *release* and *rescind* can be generalized, as the type of right being deleted does not affect the rule. We have already addressed the necessary restrictions for a *get-write* rule derived from the *get-append* rule, and the book clearly states that the *give* rule can be generalized to accept an arbitrary right.

4 Problem 4

[redacted] [redacted]

[redacted] [redacted]

[redacted]

[redacted] [redacted]

[redacted]

[redacted] [redacted]

[redacted]

[redacted] [redacted]

[redacted]

4.2 (b)

Consider a stock price ticker system T where any state that prevents the system from returning the correct price of the stock is unauthorized. That is, the only authorized state is R, where the system reports the current and correct price of a given stock. Letting S represent the set of all states that T may enter, the unauthorized states are $U = S - R$. Here, the time window w is useful in that it provides time for the system to aggregate new buy/sell offers and compute the updated price. At the beginning and end of each time window w, the policy requires that T be in state R. Unauthorized states may be entered during the middle of the window to compute the value of the stock for the upcoming R state.