Appendix

IB-security: First we show, in Figures 23 and 24, how ID- and CP-difference can be trivially extended to partitioned streams.

$$\frac{s_1 \neq_l s_2}{s_1 ::_l \mathsf{S}_1 \dot{\sim}_l s_2 ::_l \mathsf{S}_2} \quad \text{(ID1)}$$

$$\frac{s_1 =_l s_2 \qquad \mathsf{S}_1' \dot{\sim}_l \mathsf{S}_2'}{s_1 ::_l \mathsf{S}_1 \dot{\sim}_l s_2 ::_l \mathsf{S}_2} \quad \text{(ID2)}$$

$$\frac{\mathsf{sil}_l(\mathsf{S}_2) \qquad \mathsf{fin}(\mathsf{S}_2)}{s ::_l \mathsf{S}_1 \dot{\sim}_l \mathsf{S}_2}(*) \quad \text{(ID3)}$$

$$\frac{\mathsf{S}_1 \dot{\sim}_l^{\mathsf{p}} \mathsf{S}_2}{* ::_l \mathsf{S}_1 \dot{\sim}_l^{\mathsf{p}} \mathsf{S}_2}(*) \quad \text{(ID4)}$$

Figure 23: ID-difference of partitioned streams

$$\frac{s_1 \neq_l s_2}{s_1 ::_l \mathsf{S}_1 \dot{\simeq}_l s_2 ::_l \mathsf{S}_2} \quad \text{(CP1)}$$

$$\frac{s_1 =_l s_2 \qquad \mathsf{S}_1' \dot{\simeq}_l \mathsf{S}_2'}{s_1 ::_l \mathsf{S}_1 \dot{\simeq}_l s_2 ::_l \mathsf{S}_2} \quad \text{(CP2)}$$

$$\frac{\mathsf{sil}_l(\mathsf{S}_2)}{s ::_l \mathsf{S}_1 \dot{\simeq}_l \mathsf{S}_2}(*) \quad \text{(CP3)}$$

$$\frac{\mathsf{S}_1 \dot{\simeq}_l^{\mathsf{p}} \mathsf{S}_2}{* ::_l \mathsf{S}_1 \dot{\simeq}_l^{\mathsf{p}} \mathsf{S}_2}(*) \quad \text{(CP4)}$$

Figure 24: CP-difference of partitioned streams

The proofs of the following lemmas are trivial.

Lemma A.1. *${S_1}_l^{\mathsf{p}} \dot{\sim}_l^{\mathsf{p}} {S_2}_l^{\mathsf{p}}$ iff $S_1 \dot{\sim}_l S_2$.*

Lemma A.2. *${S_1}_l^{\mathsf{p}} \dot{\simeq}_l^{\mathsf{p}} {S_2}_l^{\mathsf{p}}$ iff $S_1 \dot{\simeq}_l^{\mathsf{p}} S_2$.*

For instance, to obtain a proof $S_1 \dot{\sim}_l S_2$ from a proof of ${S_1}_l^{\mathsf{p}} \dot{\sim}_l^{\mathsf{p}} {S_2}_l^{\mathsf{p}}$, you simply remove each (ID4) node in the derivation tree of $S_1 \dot{\sim}_l S_2$, and "glue" the tree back together, that is, setting the child of the parent of the (ID4) node, to the child of the (ID4) node.

We add labels to the rules defining IB-difference in Figure 25.

We are now ready to prove the propositions which position IB-security between ID-security and CP-security.

Proposition 3.1. *If q is IB-secure, then q is ID-secure.*

Proof. Assume $I_1 \approx_l I_2$, and thus $(q(I_1))_{\mathsf{o}}^{\mathsf{p},l} \approx_l (q(I_2))_{\mathsf{o}}^{\mathsf{p},l}$. We must show that then either $I_1 \dot{\sim}_l I_2$, or that $(q(I_1))_{\mathsf{o}} \sim_l (q(I_2))_{\mathsf{o}}$ when $I_1 \sim_l I_2$. Since $\sim_l$ and $\approx_l$ coincide on finite streams, we have $I_1 \sim_l I_2$, so we must show that $(q(I_1))_{\mathsf{o}} \sim_l (q(I_2))_{\mathsf{o}}$. Proving this amounts to proving

$$ {S_1}_{\mathsf{o}}^{\mathsf{p},l} \approx_l {S_2}_{\mathsf{o}}^{\mathsf{p},l} \implies {S_1}_{\mathsf{o}} \sim_l {S_2}_{\mathsf{o}} $$

$$\frac{\mathsf{s}_1 \neq_l \mathsf{s}_2}{\mathsf{s}_1 ::_l \mathsf{S}_1 \dot{\approx}_l^k \mathsf{s}_2 ::_l \mathsf{S}_2} \quad \text{(IB1)}$$

$$\frac{\mathsf{S}_1 \dot{\approx}_l^0 \mathsf{S}_2}{* ::_l \mathsf{S}_1 \dot{\approx}_l^k * ::_l \mathsf{S}_2} \quad \text{(IB2)}$$

$$\frac{s_1 =_l s_2 \qquad \mathsf{S}_1 \dot{\approx}_l^1 \mathsf{S}_2}{s_1 ::_l \mathsf{S}_1 \dot{\approx}_l^k s_2 ::_l \mathsf{S}_2} \quad \text{(IB3)}$$

$$\frac{\mathsf{sil}_l(S_2) \qquad \mathsf{fin}(S_2)}{\mathsf{s} ::_l \mathsf{S}_1 \dot{\approx}_l^0 \mathsf{S}_2}(*) \quad \text{(IB4)}$$

$$\frac{\mathsf{sil}_l(S_2)}{\mathsf{s} ::_l \mathsf{S}_1 \dot{\approx}_l^1 \mathsf{S}_2}(*) \quad \text{(IB5)}$$

Figure 25: IB-difference of partitioned streams. $\dot{\approx}_l \stackrel{\text{def}}{=} \dot{\approx}_l^0$

which you get from proving

$$ \neg({S_1}_{\mathsf{o}}^{\mathsf{p},l} \dot{\approx}_l^0 {S_2}_{\mathsf{o}}^{\mathsf{p},l}) \implies \neg({S_1}_{\mathsf{o}} \dot{\sim}_l {S_2}_{\mathsf{o}}) $$

which you get from proving

$$ {S_1}_{\mathsf{o}}^{\mathsf{p},l} \dot{\approx}_l^0 {S_2}_{\mathsf{o}}^{\mathsf{p},l} \impliedby {S_1}_{\mathsf{o}} \dot{\sim}_l {S_2}_{\mathsf{o}} $$

which you, by Lemma 7, get from proving

$$ {S_1}_{\mathsf{o}}^{\mathsf{p},l} \dot{\approx}_l^0 {S_2}_{\mathsf{o}}^{\mathsf{p},l} \impliedby {S_1}_{\mathsf{o}}^{\mathsf{p},l} \dot{\sim}_l^{\mathsf{p}} {S_2}_{\mathsf{o}}^{\mathsf{p},l}. $$

We prove this last implication now. Let $\mathsf{O}_j = q(I_j)_{\mathsf{o}}^{\mathsf{p},l}$, and assume (A) that $\mathsf{O}_1 \dot{\sim}_l^{\mathsf{p}} \mathsf{O}_2$. We must show that $\mathsf{O}_1 \dot{\approx}_l \mathsf{O}_2$. We do this by strong induction in the height k of the derivation of $\mathsf{O}_1 \dot{\sim}_l^{\mathsf{p}} \mathsf{O}_2$, to prove the stronger property $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2 \wedge \mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.

$k = 1$: Two cases.

- **(ID1):** Here, $k = 1$. By (IB1), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$ and $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.
- **(ID3):** Here, $k = 1$. By (IB4), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$. By (IB5), $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.

$k + 1$, given $\leq k$: (IH) ="$\mathsf{O}_1' \dot{\sim}_l^{\mathsf{p}} \mathsf{O}_2' \implies \mathsf{O}_1' \dot{\approx}_l^0 \mathsf{O}_2' \wedge \mathsf{O}_1' \dot{\approx}_l^1 \mathsf{O}_2'$, for all $\mathsf{O}_1, \mathsf{O}_2$ with $\mathsf{O}_1' \dot{\sim}_l^{\mathsf{p}} \mathsf{O}_2'$ derivation $\leq k$" is our induction hypothesis.

- **(ID2):** By (IB3), (A), (IH), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$ and $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.
- **(ID4):** Assume wlg. that $\mathsf{O}_1 \triangleright * :: \mathsf{O}_1'$. Case on O_2.
  - $\mathsf{O}_2 \equiv []$: By (IB4), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$. By (IB5), $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.
  - $\mathsf{O}_2 \triangleright o :: \mathsf{O}_2'$: By (IB1), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$. By (IB1), $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.
  - $\mathsf{O}_2 \triangleright * :: \mathsf{O}_2'$: By (IB2), (A), (IH), $\mathsf{O}_1 \dot{\approx}_l^0 \mathsf{O}_2$ and $\mathsf{O}_1 \dot{\approx}_l^1 \mathsf{O}_2$.

□

Proposition 3.2. *If q is CP-secure, then q is IB-secure.*

Proof. Strategy is the same as in the above proposition. □

Quantitative Guarantee: First we need to establish a lemma stating that when reacting to unobservable messages, no observables are emitted. The proof uses concatenated streams, which we re-interpret as streams in Figure 26.

$$\frac{-}{[](s :: S_2) \triangleright s :: S_2} \qquad \frac{-}{(s :: S_1)S_2 \triangleright s :: (S_1 S_2)}$$

Figure 26: Concatenation of streams

Lemma A.3. *If q is IB-secure, then for any l, we have for each i with $\neg\mathsf{obs}_l(i)$ in any I, if q finishes handling i while running on I, then for any o produced while handling i, $\neg\mathsf{obs}_l(o)$ holds.*

Proof. Assume the opposite. You can construct $I_1 = II'$ and $I_2 = I[i]I'$ such that $I_{1l}^{!} \approx_l I_{2l}^{!}$ but $q(I_1) \not\approx_l q(I_2)$, meaning q is not IB-secure, a contradiction. □

Theorem 3.1. *If q is IB-secure, then q is at most $\log_2(n+1)$-bit secure, where n is the nr. of observables in q's input.*

Proof. From Lemma A.3, we get that $(*)$ q never produces observables when handling a message i in a high part of its input I, whether q terminates on i or not (the latter follows from the assumption that q is IB-secure). We proceed by induction in n, the number of observables in I.

- $n = 0$**:** By $(*)$ there is only one equivalence class for outputs for the case where nothing is observed. Thus q is $\log_2(1)$-bit secure, as desired.
- $n + 1$**, given the theorem holds for** n**:** We have for any I with $n+1$ l-observables that for some I^H and i^L, $I = I'[i^L]I^H$. I' has n observables. If q diverges on i^L, the observer cannot know whether the program really diverged on i^L or the last high part in I'. Also, if q terminates on i^L, by $(*)$, it makes no difference to the number of equivalence classes whether q terminates or diverges on I^H. So, there is only 1 more equivalence class, representing that q finished handling i^L. By induction hypothesis, the greatest number of equivalence classes for I' is $n+1$. This totals to $n+2$ equivalence classes. So q is $\log_2(n+2)$-bit secure for I.

We are done. □

Buffering Improves Security: Let $\mathsf{S}_1 \sim_l^{\mathsf{p}} \mathsf{S}_2 \stackrel{\text{def}}{=} \neg(\mathsf{S}_1 \dot{\not\sim}_l^{\mathsf{p}} \mathsf{S}_2)$. By Lemma A.1, $S_1 \sim_l S_2 \iff S_{1l}^{\mathsf{p}} \sim_l^{\mathsf{p}} S_{2l}^{\mathsf{p}}$.

Theorem 3.2. *If q is ID-secure, then q_{B} is IB-secure.*

Proof. Let $I_1 \sim_l I_2$. Then $I_1 \approx_l I_2$ as $\sim_l$ and $\approx_l$ coincide on finite streams. Also, $(q(I_1))_{\mathsf{o}} \sim_l (q(I_2))_{\mathsf{o}}$ by the definition of ID-security. We show that

$$(q(I_1))_{\mathsf{o}}^{\mathsf{p},l} \sim_l^{\mathsf{p}} (q(I_2))_{\mathsf{o}}^{\mathsf{p},l} \implies (q(I_1))_{\mathsf{o}}^{\mathsf{p},l} \approx_l (q(I_2))_{\mathsf{o}}^{\mathsf{p},l}.$$

Let $I_j = I_{j_1}^{\mathsf{p}} \cdots I_{j_{n+1}}^{\mathsf{p}}$ and $I_j^k = I_{j_1}^{\mathsf{p}} \cdots I_{j_k}^{\mathsf{p}}$. I_1 and I_2 must have the same number of observables; otherwise $I_1 \sim_l I_2$ cannot hold. Let n be the number of observables in I_1 and I_2. I_1 and I_2 will therefore both have $n+1$ phases. $I_{j\,n+1}^{\mathsf{p}}$ both contain only unobservables, while $I_{j\,k}^{\mathsf{p}}$ have an observable as last element, and all other elements unobservable. So, $I_1^k \sim_l I_2^k$; otherwise $I_1 \sim_l I_2$ cannot hold. So $(q(I_1^k))_{\mathsf{o}}^{\mathsf{p},l} \sim_l^{\mathsf{p}} (q(I_2^k))_{\mathsf{o}}^{\mathsf{p},l}$ by the definition of ID-security. If $q(I_1^k)$ both terminate, then $q(I_1^k)$ will be finite streams. Then $(q(I_1^k))_{\mathsf{o}}^{\mathsf{p},l} \approx_l (q(I_2^k))_{\mathsf{o}}^{\mathsf{p},l}$ since $\sim_l$ and $\approx_l$ coincide on finite streams.

If $q(I_j)$ is diverging, then the divergence occurs in some phase. Assume wlg. that $q(I_1)$ diverges (and if $q(I_2)$ also diverges, that $q(I_1)$ diverges after consuming at most as many inputs as $q(I_2)$). Let k be smallest such that $q(I_1^k)$ diverges. By a corresponding Lemma A.3 for ID-security (which proof is near-identical), $q(I_1^k)$ outputs no observables when handling the unobservable inputs in I_{1k}^{p}, provided $q(I_1^k)$ terminates while doing so. Eventually, $q(I_1^k)$ diverges while handling some i where $I_{1k}^{\mathsf{p}} = I[i]I'$. Regardless of whether i is observable or not, the outputs emitted while $q(I_1^k)$ reacts to i are buffered, so $q(I_1^k)$ remains silent for the rest of I_{1k}^{p}. Since $q(I_1^k)$ reacted silently to I as well, all of I_{1k}^{p} is reacted to silently. This rules out all rules for distinguishing $(q(I_j))_{\mathsf{o}}^{\mathsf{p},l}$ by $\dot{\approx}_l$. So $(q(I_1))_{\mathsf{o}}^{\mathsf{p},l} \approx_l (q(I_2))_{\mathsf{o}}^{\mathsf{p},l}$. □

Type Soundness: Let a range over $\mathbb{I} = \mathbb{C}^{\mathsf{e}} \cup \mathbb{C}^{\mathsf{c}}$, $\mathsf{lbl}_{\mathsf{e}}(ch) = \mathsf{lbl}(ch^{\mathsf{e}})$, $\mathsf{lbl}_{\mathsf{c}}(ch) = \mathsf{lbl}(ch^{\mathsf{c}})$ and $\mathsf{lbl}_{\Gamma}(a) = \bigsqcup_{a' \in \Gamma(a)} \mathsf{lbl}(a')$.

Definition A.4. lbl *is* consistent *with Γ, written $\mathsf{ct}(\Gamma)$, iff $\mathsf{lbl}_{\Gamma}(a) \sqsubseteq \mathsf{lbl}(a)$, $\forall a$.*

Definition A.5. *p is* well-typed*, written $\vdash p$, iff $\Gamma \vdash p$ and $\mathsf{ct}(\Gamma)$, for some Γ.*

Theorem 5.1. *If $\vdash p$, then p is ID-secure.*

Theorem 5.1 is the type soundness theorem we wish to prove. The proof relies on the following auxiliary definitions.

Definition A.6. lbl *is consistent with the typing of c under Γ and pc, written $\mathsf{ct}(c, \Gamma, pc)$, iff, $pc \vdash \Gamma \cdot \{c\}\ \mathsf{p}\ \Gamma$ (for some p) and $\mathsf{ct}(\Gamma)$.*

Definition A.7. *μ_1 and μ_2 l-agree on x under Γ, written $\mu_1 =_{l,x}^{\Gamma} \mu_2$, iff, $\mathsf{lbl}_{\Gamma}(x) \sqsubseteq l \implies \mu_1(x) = \mu_2(x)$.*

Definition A.8. *μ_1 and μ_2 are l-equivalent under Γ, written $\mu_1 =_l^{\Gamma} \mu_2$, iff, $\mu_1 =_{l,x}^{\Gamma} \mu_2$, $\forall x$.*

Definition A.9. *c is secure under Γ and pc, written $\mathsf{sc}(c, \Gamma, pc)$, iff, for all l,*

i) If $l_{\mathrm{pc}} \sqsubseteq l$, then for all μ_1, μ_2, if $\mu_1 =_l^{\Gamma} \mu_2$, then
   a) $(\mu_1, c) \sim_l (\mu_2, c)$ and
   b) if $(\mu_1, c) \Downarrow \mu_1'$ and $(\mu_2, c) \Downarrow \mu_2'$ then $\mu_1' =_l^{\Gamma} \mu_2'$.

ii) If $l_{\mathrm{pc}} \not\sqsubseteq l$, then for all μ,
   a) $\mathsf{sil}_l((\mu, c))$ and
   b) if $(\mu, c) \Downarrow \mu'$, then $\mu =_l^{\Gamma} \mu'$.

where $pc \vdash \Gamma \cdot \{c\}\ \mathsf{p}\ \Gamma'$ *(for some* Δ*) and* $l_{\mathrm{pc}} = \mathsf{lbl}_\Gamma(pc)$.

Here, (μ, c) is the *run* of c in μ (defined in the obvious way). We write $(\mu, c) \Downarrow \mu'$ when $(\mu, c) \xrightarrow{O}{}^{*} (\mu', \mathtt{skip})$. μ does not represent the full system state; for that we would also need a p that gets updated when `new ...` statements in c are executed. However, Definition A.9 only concerns outputs emitted and changes on μ during a (μ, c) run, that is, a single handler execution. So we omit p in these runs.

For concatenated streams, the following useful lemma holds.

Lemma A.10. $S_1 \sim_l S_2 \wedge S_1' \sim_l S_2' \implies S_1 S_1' \sim_l S_2 S_2'$.

We are ready to prove the key lemma used in the proof of Theorem 5.1.

Lemma A.11. *For all* c, Γ, pc, $\mathsf{ct}(c, \Gamma, pc) \implies \mathsf{sc}(c, \Gamma, pc)$.

Proof. Assume $\mathsf{ct}(c, \Gamma, pc)$. Let

$$pc \vdash \Gamma \cdot \{c\}\ \mathsf{p}\ \Gamma \qquad (\mu, c) \Downarrow \mu'$$
$$\Gamma \vdash e : T \qquad (\mu_i, c) \Downarrow \mu_i'$$
$$l_{\mathrm{pc}} = \mathsf{lbl}_\Gamma(pc)$$

We prove, by induction in c, that $\mathsf{sc}(c, \Gamma, pc)$ must then hold.

Base cases

$c \stackrel{\text{def}}{=} \mathtt{skip}$**:** $\mathsf{sil}_l((\mu_i, c))$, thus $(\mu_1, c) \sim_l (\mu_2, c)$, and $\mathsf{sil}_l((\mu, c))$, for all l. Also, $\mu' = \mu$ and $\mu_i' = \mu_i$. So $\mathsf{sc}(c, \Gamma, pc)$ is a tautology.

$c \stackrel{\text{def}}{=} x := e$**:** $\mathsf{sil}_l((\mu_i, c))$, thus $(\mu_1, c) \sim_l (\mu_2, c)$, and $\mathsf{sil}_l((\mu, c))$, for all l. Also, $\mu' = \mu[x \mapsto v]$ and $\mu_i' = \mu_i[x \mapsto v_i]$, where $\mu \vdash e \Downarrow v$ and $\mu_i \vdash e \Downarrow v_i$. There are two cases to consider.

$\mathsf{lbl}_\Gamma(x) \not\sqsubseteq l$**:** Then $\mu' =_l^\Gamma \mu$ and $\mu_i' =_l^\Gamma \mu_i$. By transitivity, $\mu_1' =_l^\Gamma \mu_2'$. So $\mathsf{sc}(c, \Gamma, pc)$ holds in this case.

$\mathsf{lbl}_\Gamma(x) \sqsubseteq l$**:** Then $\mathsf{lbl}_\Gamma(T \sqcup pc) \sqsubseteq l$. Thus $\mathsf{lbl}_\Gamma(pc) \sqsubseteq l$ and $l_{pc} \sqsubseteq l$ (so $\mu =_l^\Gamma \mu'$ is not required). Also, $\mathsf{lbl}_\Gamma(T) \sqsubseteq l$, and therefore $\mathsf{lbl}_\Gamma(x') \sqsubseteq l$, $\forall x' \in T$. Thus $v_1 = v_2$, and therefore, $\mu_1' =_l^\Gamma \mu_2'$. So $\mathsf{sc}(c, \Gamma, pc)$ holds in this case.

$c \stackrel{\text{def}}{=} \mathtt{out}\ \ ch(e)$**:** Then $\mu' =_l^\Gamma \mu$ and $\mu_i' =_l^\Gamma \mu_i$. By transitivity, $\mu_1' =_l^\Gamma \mu_2'$. There are three cases to consider.

$\mathsf{lbl}_\mathsf{e}(ch) \not\sqsubseteq l$**:** Then $\mathsf{sil}_l((\mu_i, c))$, thus $(\mu_1, c) \sim_l (\mu_2, c)$, and $\mathsf{sil}_l((\mu, c))$. So $\mathsf{sc}(c, \Gamma, pc)$ holds in this case regardless of whether $l_{pc} \sqsubseteq l$ or not.

To prove the other two cases we need to show that $\mathsf{lbl}_\Gamma(pc) \sqsubseteq \mathsf{lbl}_\mathsf{e}(ch)$. We have $pc \subseteq \Gamma(ch^\mathsf{e})$ by the type rule for output and the weakening rule. Now assume $\mathsf{lbl}_\Gamma(pc) \not\sqsubseteq \mathsf{lbl}_\mathsf{e}(ch)$. Then, for some a, $a \in \Gamma(ch^\mathsf{e})$ and $\mathsf{lbl}(a) \not\sqsubseteq \mathsf{lbl}_\mathsf{e}(ch)$. But this contradicts our $\mathsf{ct}(c, \Gamma, pc)$ assumption. Likewise we have $pc \cup T \subseteq \Gamma(ch^\mathsf{c})$ and, by the same argument, $\mathsf{lbl}_\Gamma(pc \cup T) \sqsubseteq \mathsf{lbl}_\mathsf{c}(ch)$.

$\mathsf{lbl}_\mathsf{e}(ch) \sqsubseteq l$**,** $\mathsf{lbl}_\mathsf{c}(ch) \not\sqsubseteq l$**:** $\mathsf{lbl}_\Gamma(pc) \sqsubseteq l$ by transitivity of $\sqsubseteq$. Recall that $l_{pc} = \mathsf{lbl}_\Gamma(pc)$. Since $l_{pc} \sqsubseteq l$, $(\mu_1, c) \sim_l (\mu_2, c)$ must hold for $\mathsf{sc}(c, \Gamma, pc)$ to hold. Indeed, we have $(\mu_i, c) \triangleright (o_i, (\mu_i, \mathtt{skip}))$ and $\mathsf{obs}_l(o_1) = \mathsf{obs}_l(o_2) = ch(\cdot)$. So $(\mu_1, c) \sim_l (\mu_2, c)$, and therefore $\mathsf{sc}(c, \Gamma, pc)$ holds in this case.

$\mathsf{lbl}_\mathsf{c}(ch) \sqsubseteq l$**:** Again, $l_{pc} \sqsubseteq l$. Also, $\mathsf{lbl}_\Gamma(T \cup pc) \sqsubseteq l$ by similar argument, and thus $\mathsf{lbl}_\Gamma(T) \sqsubseteq l$. Then $\mathsf{lbl}_\Gamma(x') \sqsubseteq l$, $\forall x' \in T$. Thus $v_1 = v_2$, where $\mu_i \vdash e \Downarrow v_i$, so $o_1 = o_2$, where $(\mu_i, c) \triangleright (o_i, (\mu_i, \mathtt{skip}))$. Thus $(\mu_1, c) \sim_l (\mu_2, c)$, and therefore, $\mathsf{sc}(c, \Gamma, pc)$ holds in this case.

$c \stackrel{\text{def}}{=} \mathtt{new}\ \ ha$**:** As we here only care about output emitions and μ updates, the proof for this case becomes the same as that of the $c \stackrel{\text{def}}{=} \mathtt{skip}$ case.

Inductive step Induction hypothesis (IH): "for any c_j structurally smaller than c, then for any Γ_j and pc_j, $\mathsf{ct}(c_j, \Gamma_j, pc_j) \implies \mathsf{sc}(c_j, \Gamma_j, pc_j)$". Let

$$pc_j \vdash \Gamma_j\ \{c_j\}\ \mathsf{p}_j\ \Gamma_j \qquad (\mu_j, c_j) \Downarrow \mu_j'$$
$$l_{\mathrm{pc_j}} = \mathsf{lbl}_\Gamma(pc_j) \qquad (\mu_{i_j}, c_j) \Downarrow \mu_{i_j}',$$

where c_j is structurally smaller than c. Then, for instance, if $\mathsf{ct}(c_j, \Gamma_j, pc_j)$, then $\mu_{1_j} =_l^{\Gamma_j} \mu_{2_j} \implies \mu_{1_j}' =_l^{\Gamma_j} \mu_{2_j}'$ by IH since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ holds.

$c \stackrel{\text{def}}{=} \mathtt{if}\ \ e\ \{c_1\}\ \{c_2\}$**:** Let $\Gamma_j = \Gamma$ and $pc_j = pc \cup T$. Then, by the $\mathsf{ct}(c, \Gamma, pc)$ assumption, $\mathsf{ct}(\Gamma)$, and thus $\mathsf{ct}(\Gamma_j)$. Since c is well-typed, $pc_j \vdash \Gamma_j \cdot \{c_j\}\ \mathsf{p}_j\ \Gamma_j$ (for some p_j). So $\mathsf{ct}(c_j, \Gamma_j, pc_j)$, and thus $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ by IH. Here, $\mathsf{p} = p_1 p_2$. It remains to be shown that $\mathsf{sc}(c, \Gamma, pc)$. Now, (μ_i, c) either take

i) different branches, or
ii) the same branch,

in c. There are two cases to consider.

$\mathsf{lbl}_\Gamma(T) \not\sqsubseteq l$**:** Then $l_{pc_j} \not\sqsubseteq l$. Either i) or ii). We consider each case in turn.

i): Assume wlg. that (μ_j, c) takes branch j, executing c_j. Then $\mu_j = \mu_{j_j}$. For all x, if x is assigned to in (μ_j, c), then $\mathsf{lbl}_{\Gamma_j}(x) \not\sqsubseteq l$ since $T \subseteq \Gamma_j(x)$. Then $T \subseteq \Gamma_j(x)$ since $\Gamma_j = \Gamma$. Thus $\mathsf{lbl}_\Gamma(x) \not\sqsubseteq l$ and therefore $\mu_j =_{l,x}^\Gamma \mu_j'$. By transitivity, $\mu_1' =_l^\Gamma \mu_2'$.

ii): Here, $\mu_i = \mu_{i_j}$ and $\mu_i' = \mu_{i_j}'$ where j is the branch taken in both runs. Since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ and $l_{pc} \sqsubseteq l_{pc_j}$, $\mu_1' =_l^\Gamma \mu_2'$.

In both cases, since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ and $l_{pc_j} \not\sqsubseteq l$, $\mathsf{sil}_l((\mu_j, c))$. So $\mathsf{sc}(c, \Gamma, pc)$.

$\mathsf{lbl}_\Gamma(T) \sqsubseteq l$**:** Then $l_{pc_j} \sqsubseteq l$. Thus $\mu_1(e) = \mu_2(e)$.

i): Impossible.

ii): Same argument as in case $\mathsf{lbl}_\Gamma(T) \not\sqsubseteq l$.

Since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$, $pc_j = pc \cup T$, and since (μ_1, c) and (μ_2, c) take the same branch, $(\mu_1, c) \sim_l (\mu_2, c)$. So $\mathsf{sc}(c, \Gamma, pc)$.

$c \stackrel{\text{def}}{=} c_1\, ;\, c_2$**:** Let $\Gamma_j = \Gamma$ and $pc_j = pc$. Like in the `if ...` case, since $\mathsf{ct}(c, \Gamma, pc)$, $\mathsf{ct}(c_j, \Gamma_j, pc_j)$, and thus $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ by IH. It remains to be shown that $\mathsf{sc}(c, \Gamma, pc)$. Since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$, by setting $\mu_i = \mu_{1_i}$, $\mu_{2_i} = \mu'_{1_i}$ and $\mu'_i = \mu'_{2_i}$ we have $\mu_1 =^\Gamma_l \mu_2 \implies \mu'_1 =^\Gamma_l \mu'_2$, if $(\mu_{1_i}, c_1) \Downarrow \mu'_{1_i}$ and $(\mu_{2_i}, c_2) \Downarrow \mu'_{2_i}$, as then $(\mu_i, c_1 ;\ c_2) \Downarrow \mu'_i$. We get $(\mu_1, c) \sim_l (\mu_2, c)$ by $\mathsf{sc}(c_j, \Gamma_j, pc_j)$, the fact that all silent streams are equivalent under $\sim_l$, and by Lemma A.10.

$c \stackrel{\text{def}}{=} \texttt{while}\ e\ \{c_1\}$**:** Let $\Gamma_1 = \Gamma$ and $pc_1 = pc \cup T$. Like in the `if` ... case, since $\mathsf{ct}(c, \Gamma, pc)$, $\mathsf{ct}(c_j, \Gamma_j, pc_j)$, and thus $\mathsf{sc}(c_j, \Gamma_j, pc_j)$ by IH. It remains to be shown that $\mathsf{sc}(c, \Gamma, pc)$. There are two cases to consider.

$\mathsf{lbl}_\Gamma(T) \not\sqsubseteq l$**:** Then $l_{pc_1} \not\sqsubseteq l$. Since $\mathsf{sc}(c_1, \Gamma_1, pc_1)$, $\mathsf{sil}_l((\mu_i, c_1))$, and thus $\mathsf{sil}_l((\mu_i, c))$, for any μ_i. Likewise, by transitivity of $=^\Gamma_l$, if $(\mu_i, c) \Downarrow \mu'_i$ and $\mu_1 =^\Gamma_l \mu_2$, then $\mu'_1 =^\Gamma_l \mu'_2$.

$\mathsf{lbl}_\Gamma(T) \sqsubseteq l$**:** Let μ_i^n be μ_i after n iterations of c_1. So $\mu_i^0 = \mu_i$ and $(\mu_i^n, c_1) \Downarrow \mu_i^{n+1}$. Since $\mathsf{sc}(c_j, \Gamma_j, pc_j)$, if $\mu_1 =^\Gamma_l \mu_2$, then $(\mu_1^n, c) \sim_l (\mu_2^n, c)$ and $\mu_1^{n+1} =^\Gamma_l \mu_2^{n+1}$, for all n, up to k (possibly nonexisting), where one of two things happens:

$\mu_i^k(e) = 0$**;** μ_i^k **defined:** Then $\mu'_i = \mu_i^k$, $\mu'_1 =^\Gamma_l \mu'_2$, $(\mu_i, c) = (\mu_i^0, c_1) \cdots (\mu_i^{k-1}, c_1)$, and $(\mu_1, c) \sim_l (\mu_2, c)$.

μ_2^k **undefined:** Then (μ_2^{k-1}, c_1) diverged (so no constraints are placed on memories for establishing $\mathsf{sc}(c, \Gamma, pc)$). Still, $(\mu_1^{k-1}, c_1) \sim_l (\mu_2^{k-1}, c_1)$. Now, Lemma 7 can be used to prove that if $S \sim_l S^\infty$ and S^∞ is infinite, then for any S', $SS' \sim_l S^\infty$. From this it follows that $(\mu_1, c) \sim_l (\mu_2, c)$, where $(\mu_1, c) = (\mu_1^0, c_1) \cdots (\mu_1^k, c_1) \cdots$ and $(\mu_2, c) = (\mu_2^0, c_1) \cdots (\mu_2^{k-1}, c_1)$.

Thus $\mathsf{sc}(c, \Gamma, pc)$.

□

Obtain a list

$$\langle pc'_1, ch_1, c_1\rangle ; \cdots ; \langle pc'_n, ch_n, c_n\rangle ; \cdot \stackrel{\text{def}}{=} \hat{\Delta}$$

when typing p by adding a side-effect $\hat{\Delta} := \langle pc, ch, c\rangle\hat{\Delta}$ as a premise in the third rule in Figure 21. Let $ha_i = ch_i(\mathtt{z})\{c_i\}$. Each ha_i is then a possibly active handler, and pc_i is the context in which ha_i was activated in. Let $\Gamma_i = \Gamma[\mathtt{z} \mapsto \check{ch}^{\mathsf{c}}]$ and $pc_i = pc'_i \sqcup \check{ch}^{\mathsf{e}}$.

Lemma A.12. *If* $\vdash p$*, then* $\mathsf{ct}(c_i, \Gamma_i, pc_i)$*,* $\forall i$*.*

Proof. Since $\vdash p$, by typing, $pc_i \vdash \Gamma_i\ \{c_i\}\ \mathsf{p}_i\ \Gamma_i$ (for some p_i), $\forall i$. Also, $\mathsf{ct}(\Gamma) \implies \mathsf{ct}(\Gamma_i)$ since $\Gamma_i = \Gamma[\mathtt{z} \mapsto \check{ch}^{\mathsf{c}}]$ and $\mathtt{z} \notin \mathbb{I}$. □

Lemma A.13. *If* $\mathsf{sc}(c_i, \Gamma_i, pc_i)$*,* $\forall i$*, then* p *is ID-secure.*

Proof. Let I_1, I_2 and l s.t. $I_1 \sim_l I_2$ be given. Let $O_1 = (q_0(I_1))_\circ$ and $O_2 = (q_0(I_2))_\circ$. We show that $O_1 \sim_l O_2$. Let μ_j denote the current memory of O_j (initially μ_0). So initially $\mu_1 =^\Gamma_l \mu_2$. As O_j processes unobservables, O_j remains silent by Definition A.9 $ii)$. Also, all μ_j are l-equivalent under Γ. If either O_j diverges when handling an unobservable, we are done. Otherwise O_j both eventually start processing observables i_j. $\mathsf{obs}_l(i_1) = \mathsf{obs}_l(i_2) = ch(w)$ for some ch and $w \in \mathbb{V} \cup \{\cdot\}$, since $I_1 \sim_l I_2$. At that time, $\mu_1 =^\Gamma_l \mu_2$. So by Definition A.9 $i)$, $O_1 \sim_l O_2$ while handling i_1 and i_2, and if O_1 and O_2 finish doing so, the resulting μ_1, μ_2 satisfy $\mu_1 =^\Gamma_l \mu_2$, and this argument repeats for the rest of the input streams, until one diverges, or they are both exhausted. Thus $O_1 \sim_l O_2$. □

Theorem 5.1 now follows from Lemmas A.11, A.12 and A.13.

Theorem 5.2. *If* p *is ID-secure, then* $\mathsf{buff}(p)$ *is IB-secure.*

Proof. Let $I_1 \sim_l I_2$. Then $q_0(I_1) \sim_l q_0(I_2)$ by assumption. Let $I_j = I^{\mathsf{p}}_{j_1} \cdots I^{\mathsf{p}}_{j_{n+1}}$ and $I^k_j = I^{\mathsf{p}}_{j_1} \cdots I^{\mathsf{p}}_{j_k}$. Then $q_0(I^k_1) \sim_l q_0(I^k_2)$, $\forall 1 \le k \le n+1$ (else we get a contradiction since $I_{1_m} \sim_l I_{1_m}$, for all m). In particular, if C_0 terminates on I^k_1 and I^k_2, then $C_0(I^k_1)$ and $C_0(I^k_2)$ will match observables exactly. Thus $C_0(I_1) \approx_l C_0(I_1)$. Since $\mathsf{buff}(p)$ has the same input-output behavior in this case, $\mathsf{buff}(p)$ is IB-secure for those inputs.

Nonterminating reactions are yet to be considered. Since p is ID-secure, by a corresponding Lemma A.3 for ID-security (which proof is near-identical), we have that p never produces observables when handling a message i in a high part of the input stream, when p terminates on i. Same holds for $\mathsf{buff}(p)$. If p diverges on some i, then $\mathsf{buff}(p)$ outputs nothing while diverging on i. Thus, if $\mathsf{buff}(p)$ diverges on a part, high or low, no outputs emerge. We are done. □