Awkward example like proof.

For target language.
Assume assert loops and no flags.

We ignore malloc and pretend memory magically appears.

- We do not have a way to generate a fresh seal for the closure.
- The R relation is tailored for the call semantics, but we do not know that the adversary uses call, so when we return, we fail to use the continuation relation.

Proof

Pick $W = [L^{sta}\ ms_{link}]$
$[L^{stk}\ down(ms_{stk})]$
$[L^{adv}\ down(ms_{adv})]$
$[L^{sta}\ down(ms_{g1} \cup ms_{g2})]$ — contains g1 and R4.

where
L^{sta} is a normal region
L^{stk} is a spatial_owned region
L^{adv} is a normal region
(in particular the standard region).

For

$reg = reg_0[pc \mapsto c_{adv}, c_{data} \mapsto c_{adv,d}, r_{stk} \mapsto c_{stk}, r_1 \mapsto c_{gr}, r_1 \mapsto c_{gr,data}]$

where c_{gr} and $c_{gr,d}$ is a sealed capability pair.

c_{stk} is a linear capability for the stack.

c_{adv} is the adversary code capability and $c_{adv,d}$ is the data capability for the adversary.

Show

$(reg, ms) \in \mathcal{O}(W)$ (for ms like in the prev. lemma).

To this end we use the FTLR .... (?) to get

$(c_{adv}, c_{adv,d}) \in \mathcal{E}(W_P)$ ← No owned regions

Now it we can show

$(c_{adv}, c_{adv,d}) \in \mathcal{P}(W_P)$ ← okay because the capabilities not linear.

$(0,0) \in \mathcal{R}(W_K)$ ← nothing owned ← okay because it always fails.

(1) $reg \in \mathcal{R}(W_M)$

(2) $ms : W_M$ ← owns nothing

then we are done by the conclusion of $\mathcal{E}$.

In other words, we need to show (1) and (2).
(2) is here okay because ..., so it remains to show (1).

(1) Show

(a) $c_{stk} \in \mathcal{V}(W_{[redacted]})$

(b) $c_{g_1} \in \mathcal{V}(W')$

(c) $c_{g_1,d} \in \mathcal{V}(W')$

We can pick the owned stack region, and the rest should be ok.

where W' has no owned regions.

We need to show (b) and (c)

At this point we need a proper def. of $\mathcal{V}$ for sealed capabilities, but for arguments sake assume it amounts to the following (in this case):

Say $c_{g_1} = \overset{\text{sealed}}{(sc_{g_1}, \sigma)}$

$c_{g_1,d} = \text{sealed}(sc_{g_1,d}, \sigma)$

then we need to show

$(sc_{g_1}, sc_{g_1,d}) \in \mathcal{E}(W_1)$

for $W_1 \sqsupseteq^{priv} W'$. This means that the stack region may have been revoked in W_1.

to this end let W_R, W_M, W_K be given s.t.

$W_R \oplus W_M \oplus W_K \oplus W_1$ is defined. Further assume

$reg_1 \in \mathcal{R}(W_R)$, $(sc_{g_1}, sc_{g_1,d}) \in \mathcal{P}(W_1)$

$ms_1 : W_M$, and

$(c_{k_1}, c_{k,d}) \in \mathcal{K}(W_K)$

and show

$(reg_1', ms_1) \in \mathcal{O}(W_R \oplus W_M \oplus W_K \oplus W_1)$

where

$reg_1' = reg_1[pc \mapsto sc_{g,1}, r_{data} \mapsto sc_{g,1,d}, r_{retcode} \mapsto c_k, r_{retdata} \mapsto c_{k,d}]$

We use the anti-reduction lemma w/ the following observations

$(reg_1', ms_1) \longrightarrow^* (reg_2, ms_2)$

where

$reg_2(r_3) = c_{f4}$

$ms_2(x) = 0$

$c_x = ((RW, normal), x, x, x)$

$reg_2(r_1) = sealed\ (c_x, \sigma)$

$reg_2(r_2) = sealed\ (c_{f4}, \sigma)$

$c_{f4} = ((RX, normal), \_, \_, \_)$ ← Points to f4 code.

$reg_2(pc) = sc_k$

$reg_2(r_{data}) = sc_{k,d}$

otherwise like reg_1

} unsealed c_k and $c_{k,d}$ respectively.

Now show

$(reg_2, ms_2) \in \mathcal{O}(W_2)$ where $W_2 = W_1[L_x]$

! normal region w/ transition system

To this end, we need to use

$(c_k, c_{k,d}) \in \mathcal{K}(W_K)$

As $W_K \sqsubseteq^{pub} W_K[L_x]$, we need to show

$ms_2 : W_M[L_x]$ — Hopefully follows from previous steps

* $reg_2 \in \mathcal{R}(W_R[L_x])$

$c_k, c_{k,d} \in \mathcal{P}(W_P)$

* Remains to be shown.

Here, the most interesting case is showing
sealed($c_{fw} \in \mathcal{V}(W_2[\ldots])$) w/ no owned regions.
which (like previously) hopefully amounts to

$$(c_{fw}, c_x) \in \mathcal{E}(W_2) \quad \text{for} \quad W_2 \sqsupseteq^{priv} W_1$$

To this end let W_{2R}, W_{2M}, W_{2K} be given s.t.
$W_{2R} \oplus W_{2M} \oplus W_{2K} \oplus W_2$ is defined. Further, assume

$reg_3 \in R(W_{2R})$

$(c_k', c_{k,d}') \in R(W_{2K})$

$ms_3 : W_{2M}$

and show

$$(reg_3[pc \mapsto c_{fw}, r_{data} \mapsto c_3, r_{retcode} \mapsto c_k', r_{retdata} \mapsto c_{k,d}'],\ ms_3) \in \mathcal{O}(W_{2R} \oplus W_{2M} \oplus W_{2K}\ W_2)$$

the anti-reduction. Execution goes as follows:

$$(reg_3[pc \mapsto \ldots], ms_3) \longrightarrow^* (reg_4, ms_4)$$

where
$reg_4(r_{stk}) = ((ow, -), b_{stk}, \ldots, -)$ ← base of stack checked

$ms_4(x) = 0$ ms_4 otherwise has a changed stack (see next page) – otherwise unchanged.

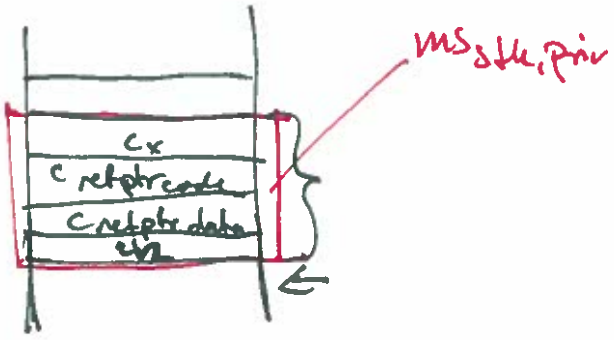
$reg_4(pc) = SC_{cb}$
$reg_4(r_{data}) = SC_{cb,d}$
} these capabilities come from the callback given to us.

$reg_4(r_{retcode}) = c_{fw,ret}$ ← sealed capability that points to the appropriate place in memory.

$reg_4(r_{retdata}) = c_{fw,data}$ ← sealed capability that contains the linear capability for the private part of the stack.

$reg_4(r_{temp1 \ldots}) = 0$. The rest of reg_4 as before.

The private stack :



Now argue

$(reg_4, ms_4) \in \mathcal{O}(W_3)$

where $W_3 = revokeRegion(i, W_2)[\ldots]$ — the region for the stack; spatial owned

$[\ldots]$ spatial owned std. region.

$W_3 \overset{?}{\ldots} W_2$ ← does this even matter?

To this end pick $W_{3R}^{scb} = revokeRegion(i, W_{2R})$

$[\ldots]$ Not owned.

Now notice $W_{2R}^{scb} \subseteq^{priv} W_{3R}^{scb}$ as:

$W_{2R}(i)$ is a spatial region as it must be owned by W_{2R}. It is thus de to revoke it. Further, we add new regions, so all in all we are good.

This gives us that

$(sc_{cb}, sc_{cb.d}) \in \mathcal{E}(W_{stack\,3R}^{scb})$

and show

$ms_4 : W_{4M}$ for a world region that owns none of the new spatial regions.

(i) $reg_4 \; W_{4R}$ for a world that owns the region for ~~the stack~~ the unsaved stacks

(ii) $(c_{fn,ret}\, c_{fn,data,ret}) \in K(W_{4R})$ for a region that owns the region for the private stacks.

To show (i) the most interesting case is rsth for which we use the stack region which is spatially-owned in W_{4R}.

To show (ii) let $W_{5K} \overset{2^{nd}}{\sqsupseteq} W_{4K}$ and let W_{5R}, W_{5M} be given s.t. $W_{5K} \circledast W_{5R} \circledast W_{5M}$ is defined and assume

$(c_{fn,ret}, c_{cnt,ret}) \in P(W_{5K})$

$reg_5 \in R(W_{5R})$

$ms_5 \Vdash : W_{5M}$

Here I used the unary K from local cap. work. so no $\triangleright$.

and show

$(regs[pc \mapsto sc_{fn,ret}, rdata \mapsto sc_{fn,data,ret}]\, regs, ms_5) \in \mathcal{O}(\ldots)$

$W_{5K} \circledast W_{5R} \circledast W_{5M}$

To this end use anti-red lemma.

Based on the program, the execution either fails or step to a configuration (reg_6, ms_6) s.t.

$reg_6(r_{stk}) = reg_3(r_{stk})$ ← i.e. the old stack pointer.

$reg_6(r_{pc}) =$ [redacted]

sc'_k

$reg_6(r_{data}) = sc'_{k+1}$ ← the unsealed versions of the return capability pair (circled on p. 5).

We skip the second call and return

In order to use the continuation/return pointer we need to take a look at the $R(W_{2K})$ relation. Here we need to use a world W_{6K} where $W_{6K} \sqsupseteq^{Pub} W_{2K}$. At the same time, we need to construct W_{6K} based on W_{5K}, so we can revoke regions that are no longer needed.

W_{6K} contains an <u>owned</u> spatial region for the private stack. We need to be able to revoke this in order to give up the stack.

Not this one, but one of the other W_5 regions. We want to make sure $W_6 \sqsupseteq^{priv} W_{5K}$ and $W_6 \sqsupseteq^{priv} W_{5M}$, so we can argue about the validity of reg_6 and ms_6.

Ignoring this, revoke the two stack regions and add the old stack region from W_{2K}.

Is the problem perhaps in R? Should the P be on before it.

world?