Part 2
Ambient Calculus

Luca Cardelli
Andy Gordon

Approach

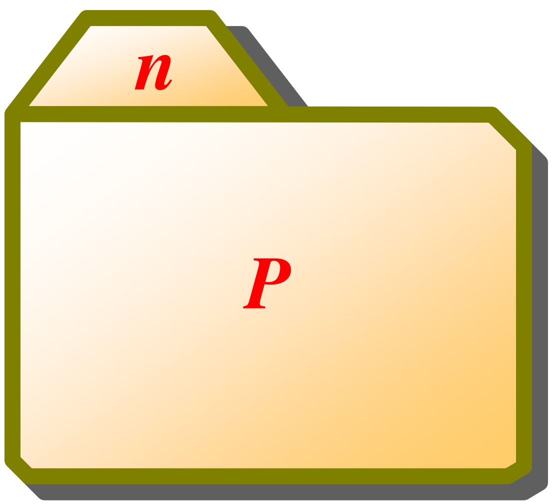
- We want to capture in an abstract way, notions of locality, of mobility, and of ability to cross barriers.
- An *ambient* is a place, delimited by a boundary, where computation happens.
- Ambients have a name, a collection of local processes, and a collection of subambients.
- Ambients can move in an out of other ambients, subject to capabilities that are associated with ambient names.
- Ambient names are unforgeable (as in π and spi).

Basic Assumptions

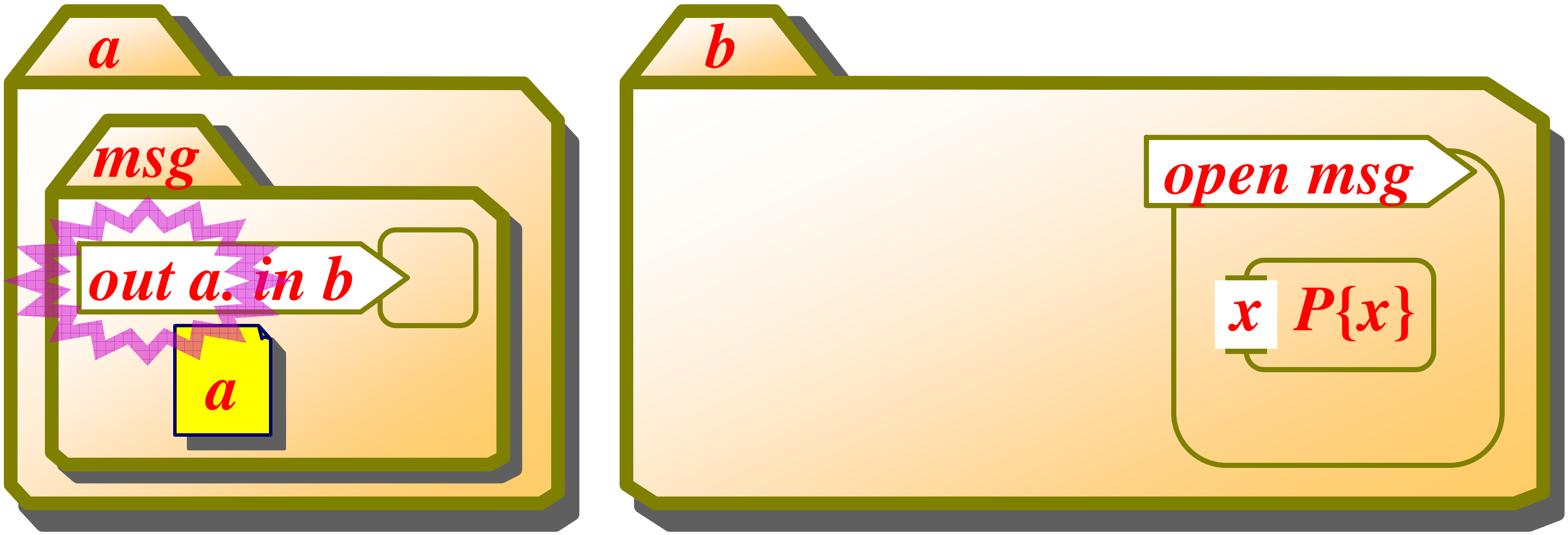
- Mobile processes are not data. *They* move, they are not moved.
  - (It might be temping to move processes by sending them over channels.)
- Mobile computation is the
  dynamic local rearrangement of labeled trees.
  - (*Cf.*: in π, it is dynamic propagation of channel names.)
- The choice of primitives for tree rearrangement depends strongly on the *design principles* one adopts.
  - Are these trees in-memory? (No, they are distributed)
  - Are they just passive data that gets globally transformed?
    (No, they are full of active local processes with a will of their own.)
  - Do mobile processes have any guarantees?
    - Can they get killed, robbed, poisoned, kidnapped? (In *Classical Ambients*, only if they are stupid: talk too much, eat bad food, step in dark alleys.)
    - Can they get infected? (Not in *Safe Ambients*, if they are careful.)
    - How do they talk to each other? (Richer options in *Boxed Ambients*.)

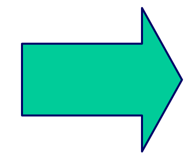
Folder Metaphor

- An ambient can be graphically represented as a folder:
  - Consisting of a folder name *n*,
  - And active contents *P*, including:
    - Hierarchical data, and computations ("gremlins").
    - Primitives for mobility and communication.

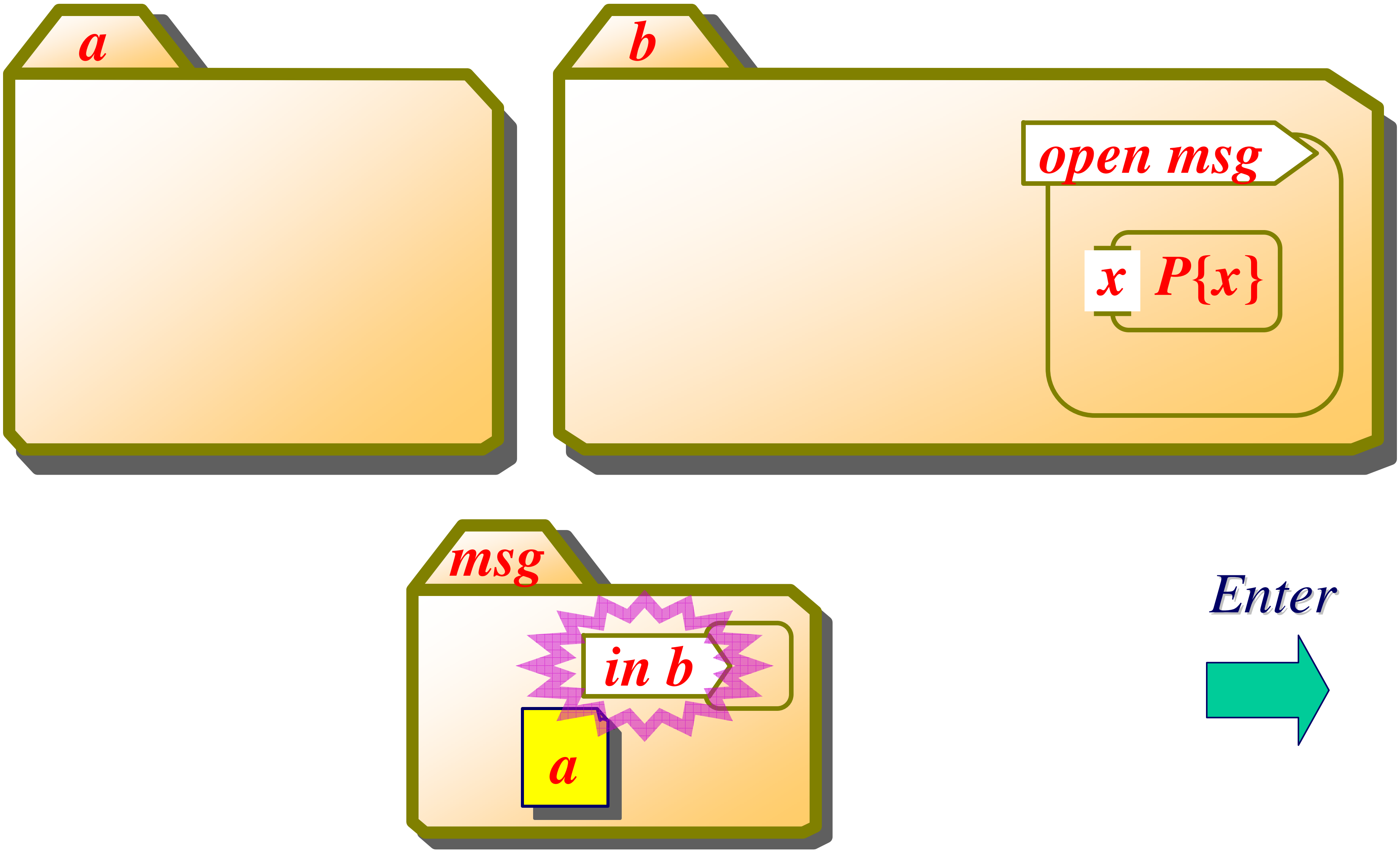


Example: Message from *a* to *b*

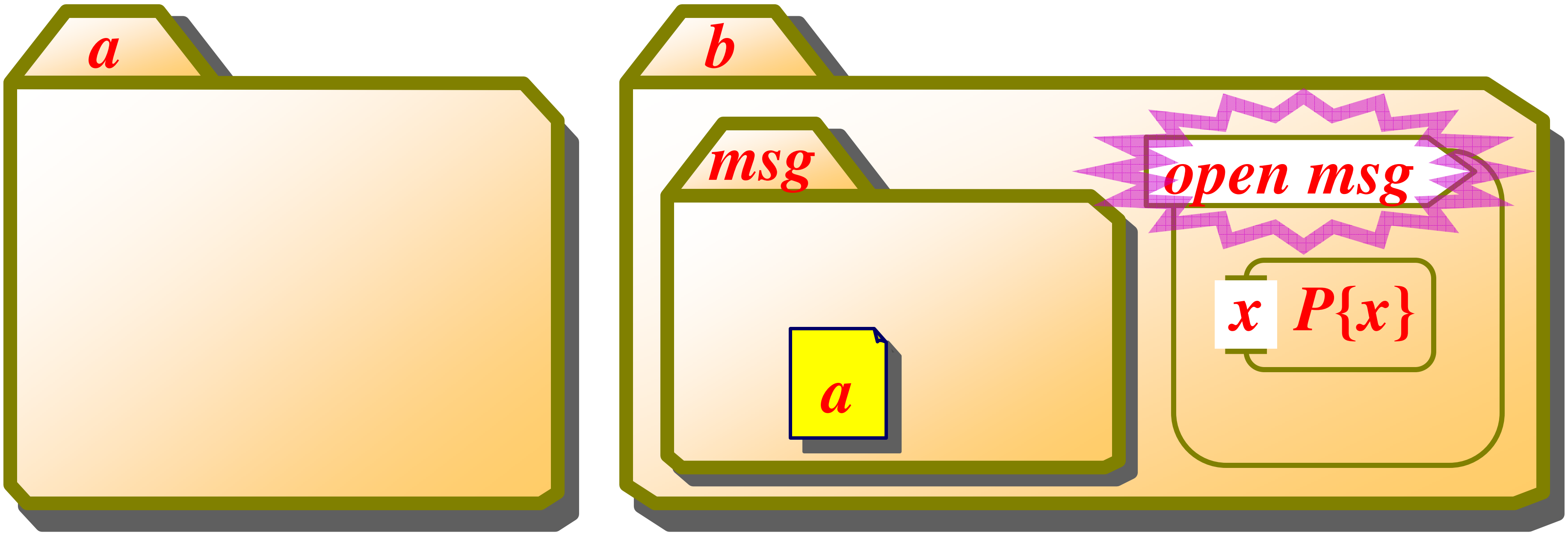


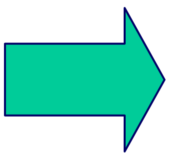


Example: Message from *a* to *b*

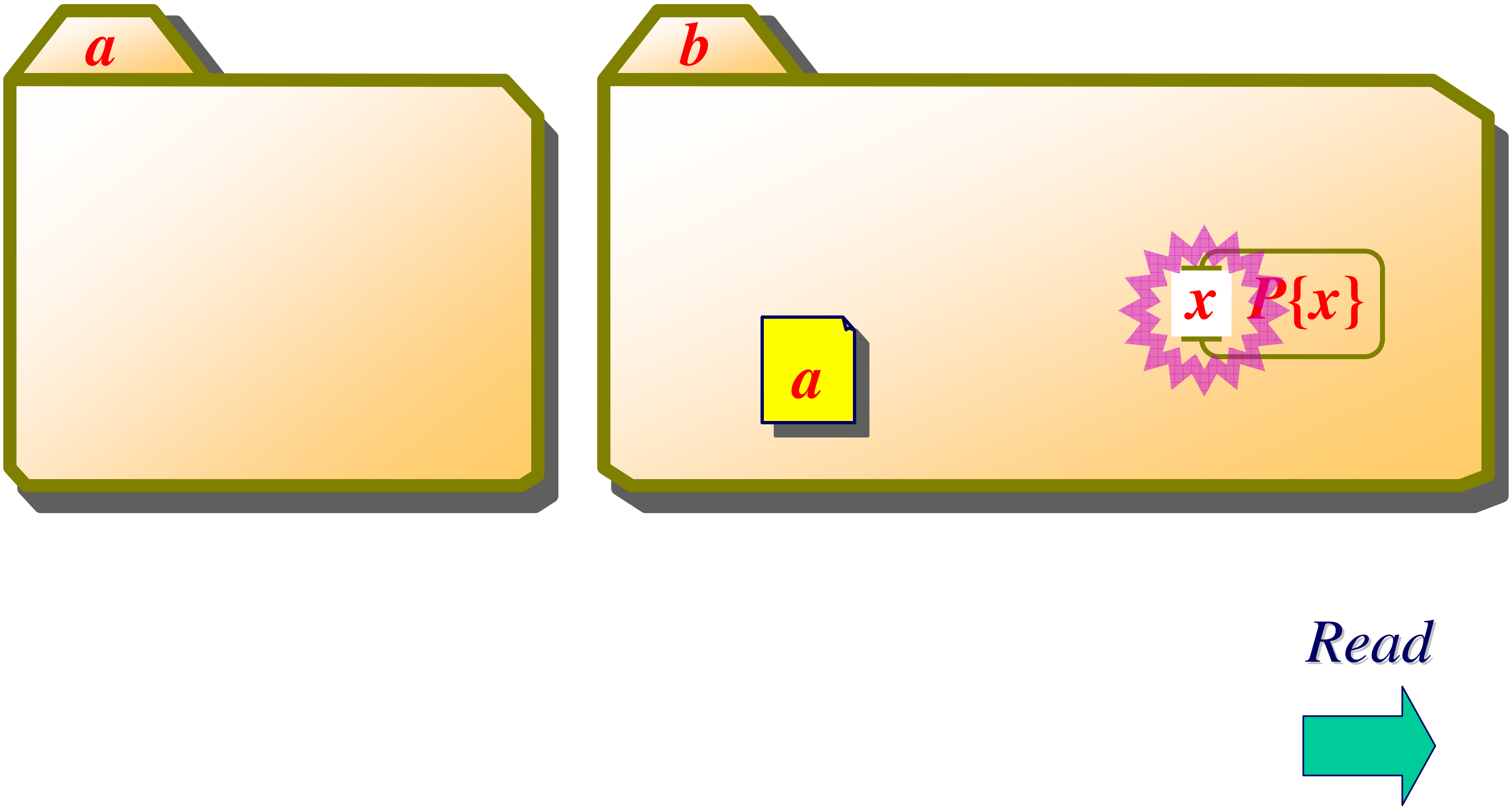


Example: Message from *a* to *b*

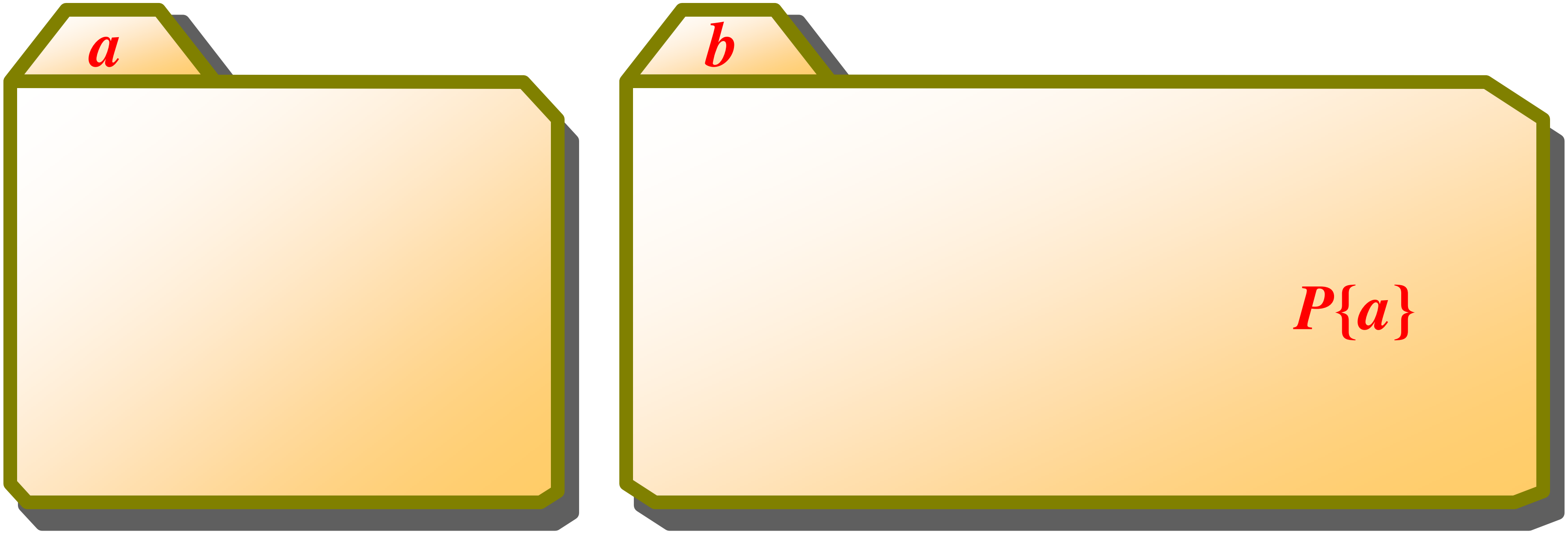




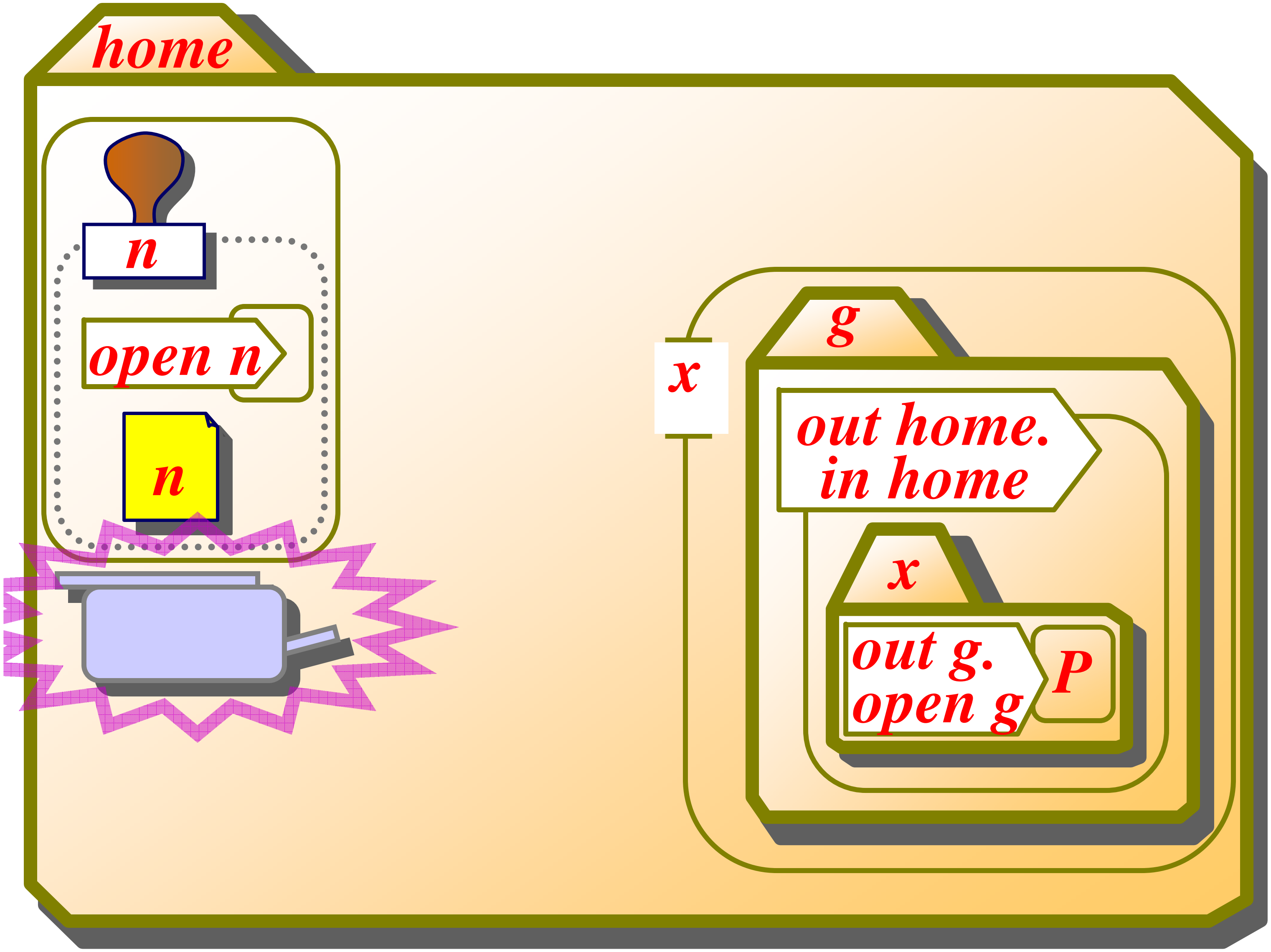
Example: Message from a to b



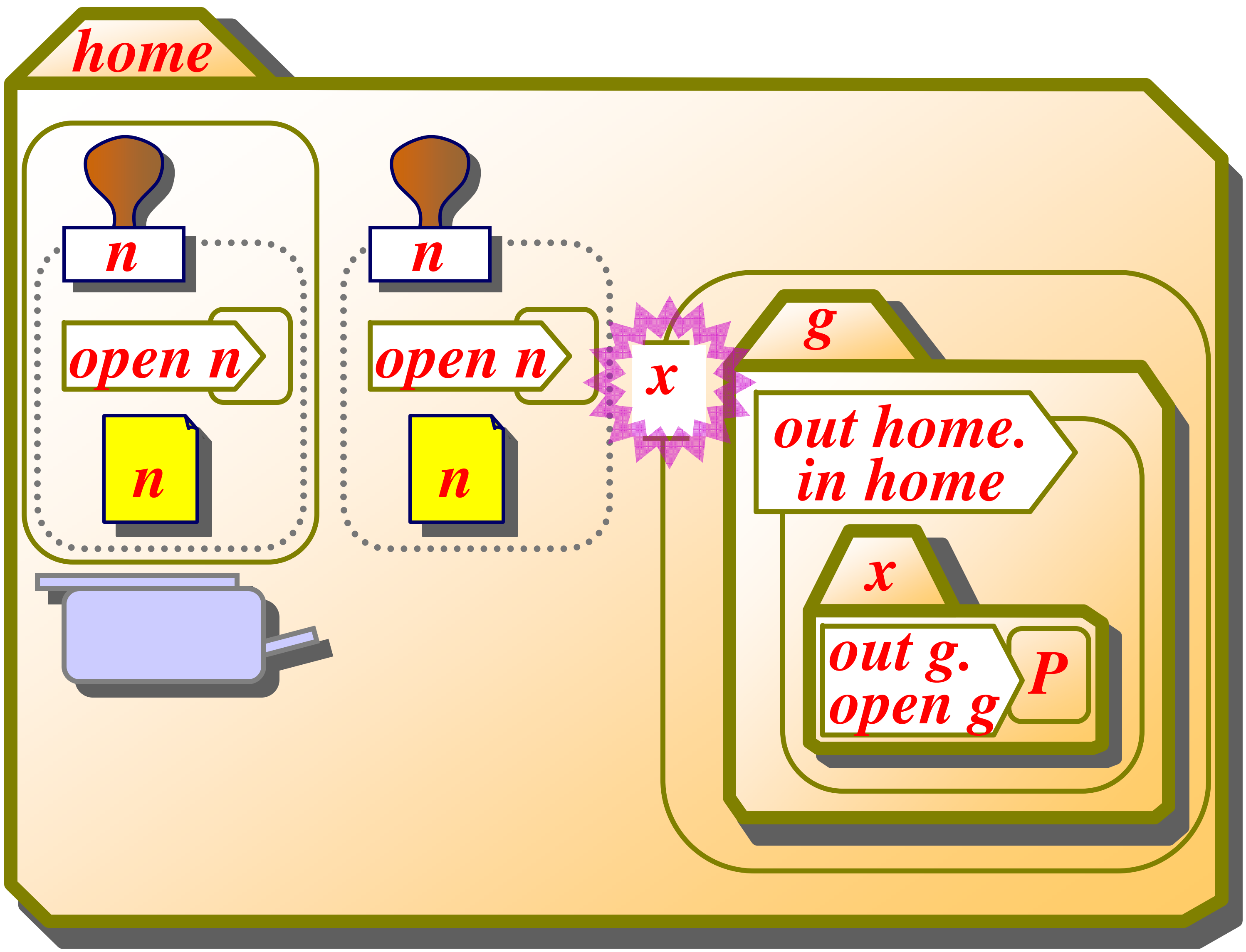
Example: Message from a to b



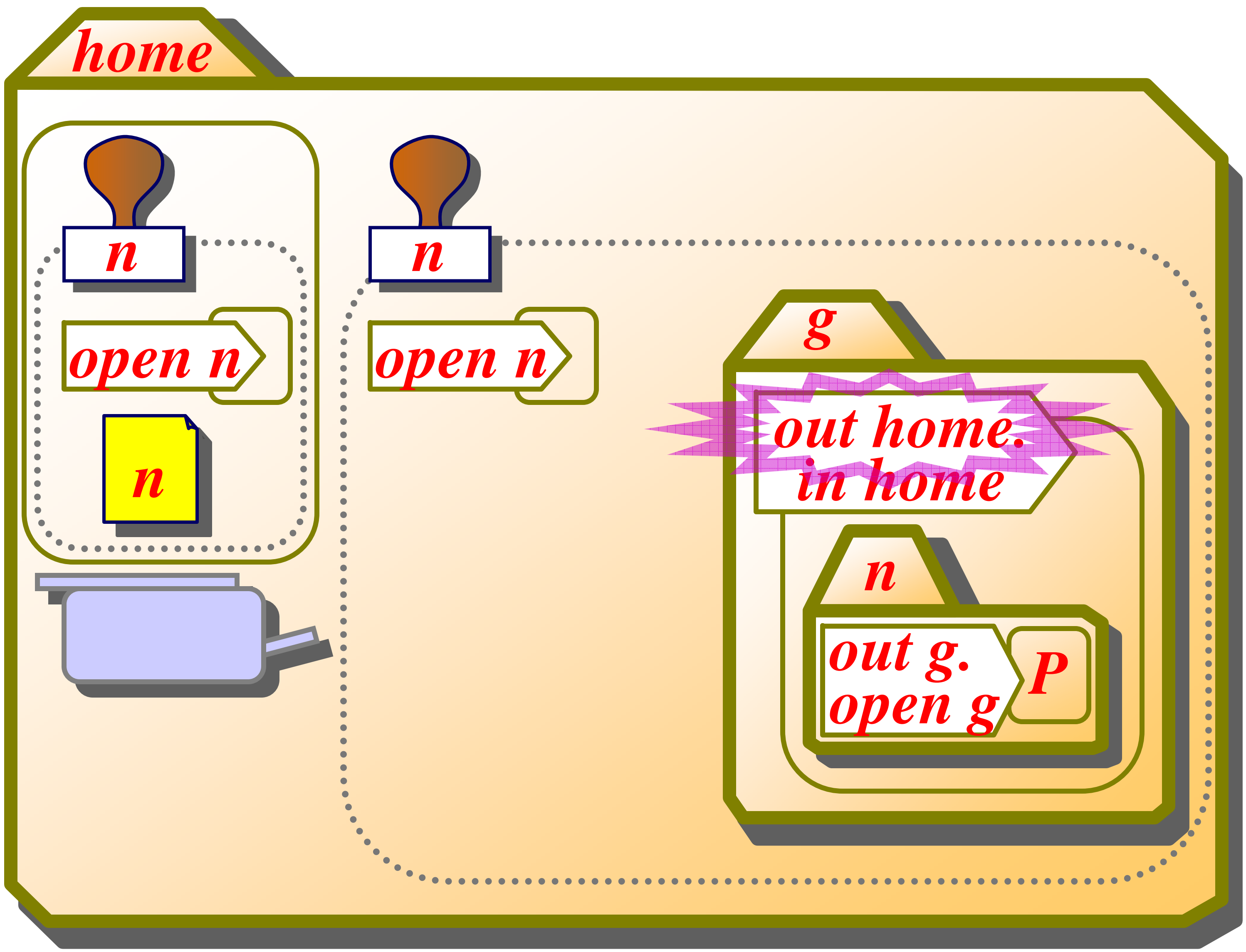
Example: Agent Authentication



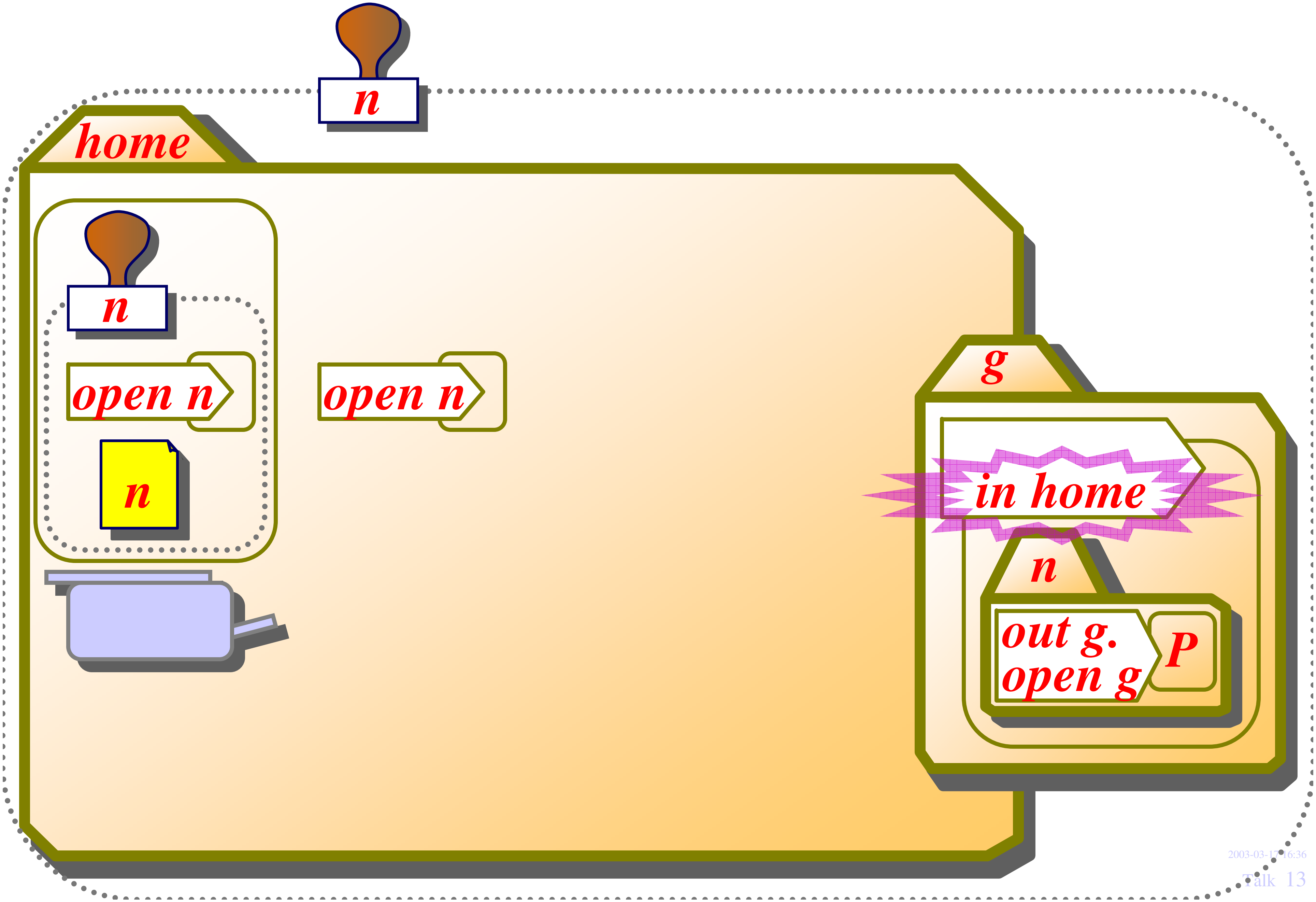
Example: Agent Authentication



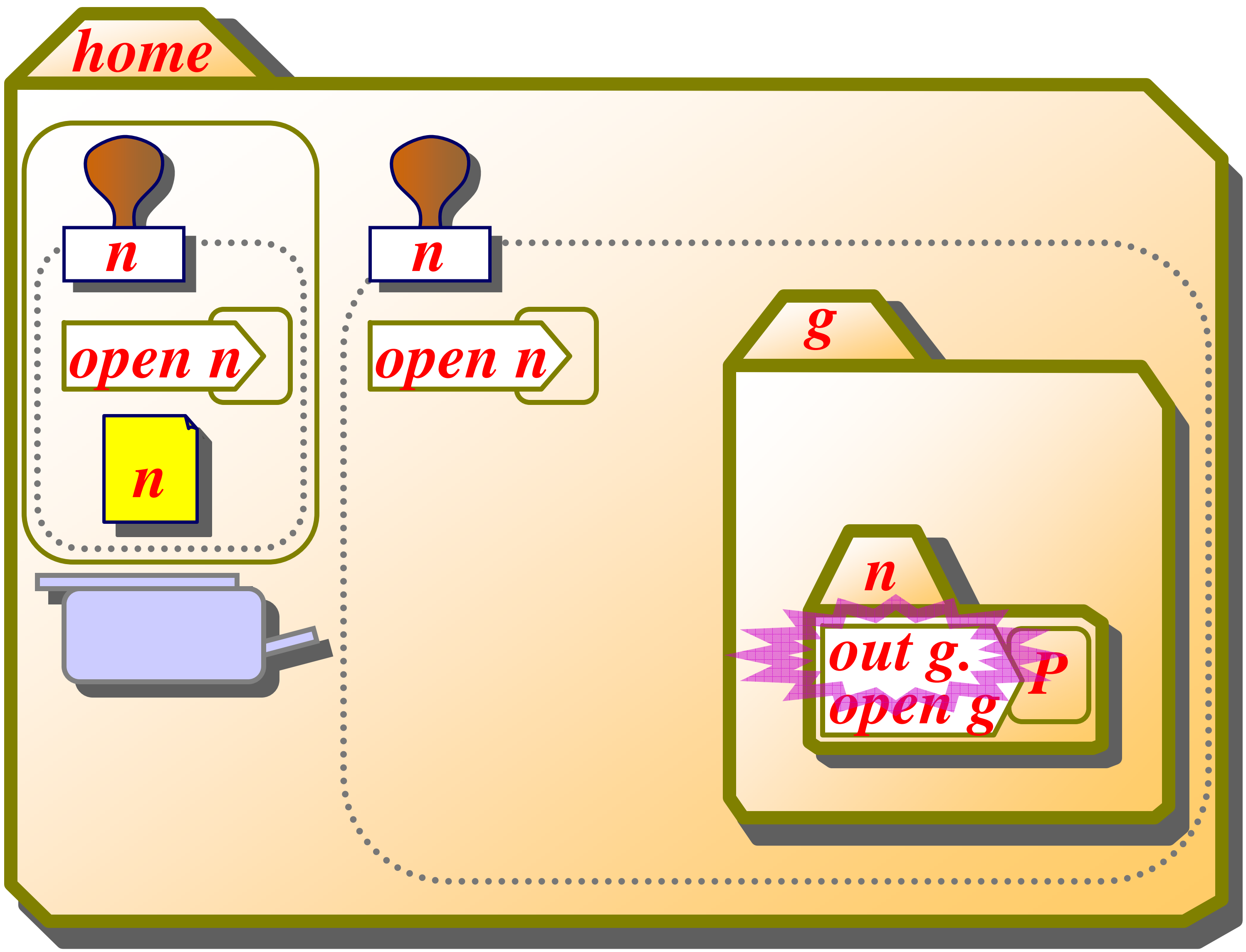
Example: Agent Authentication



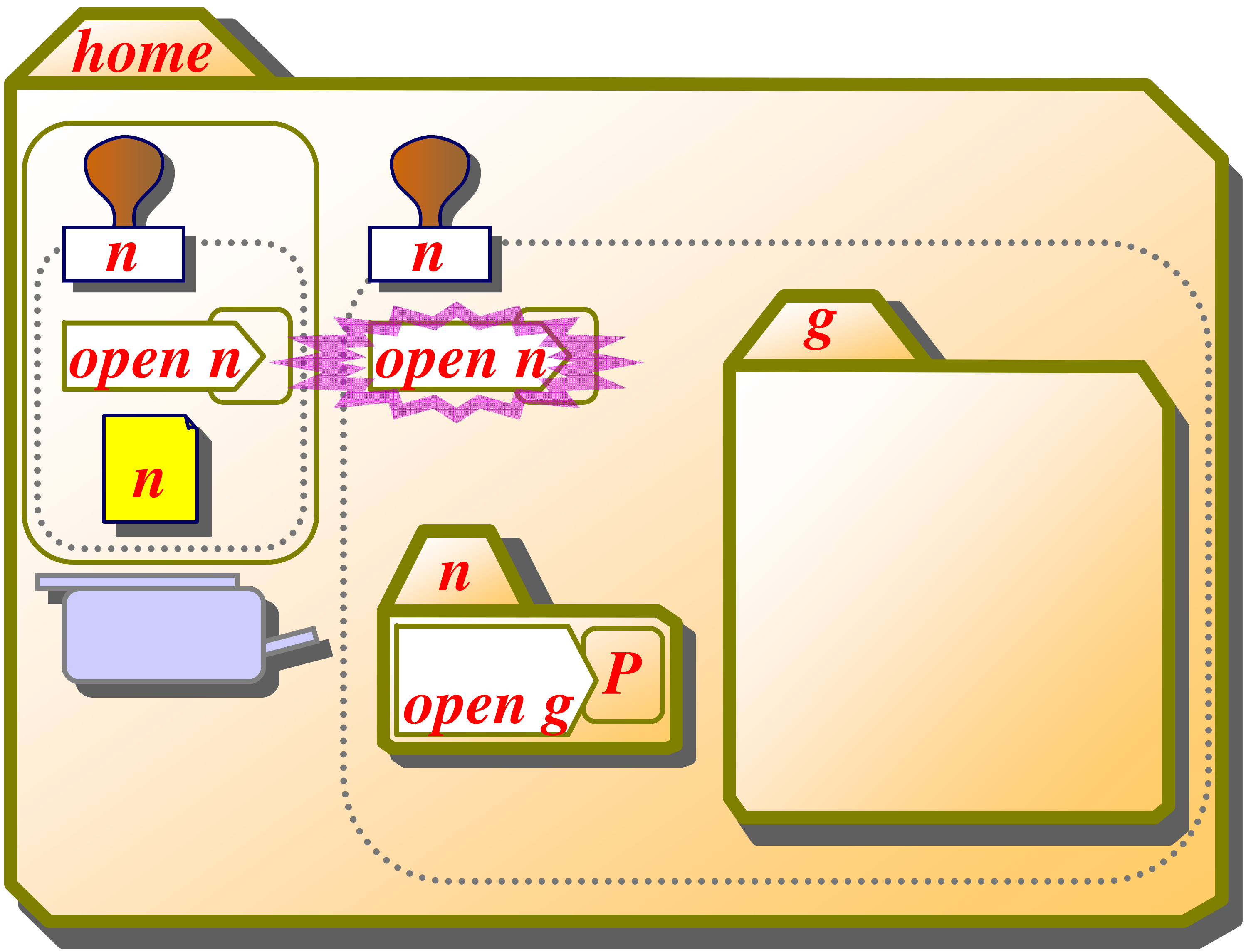
Example: Agent Authentication



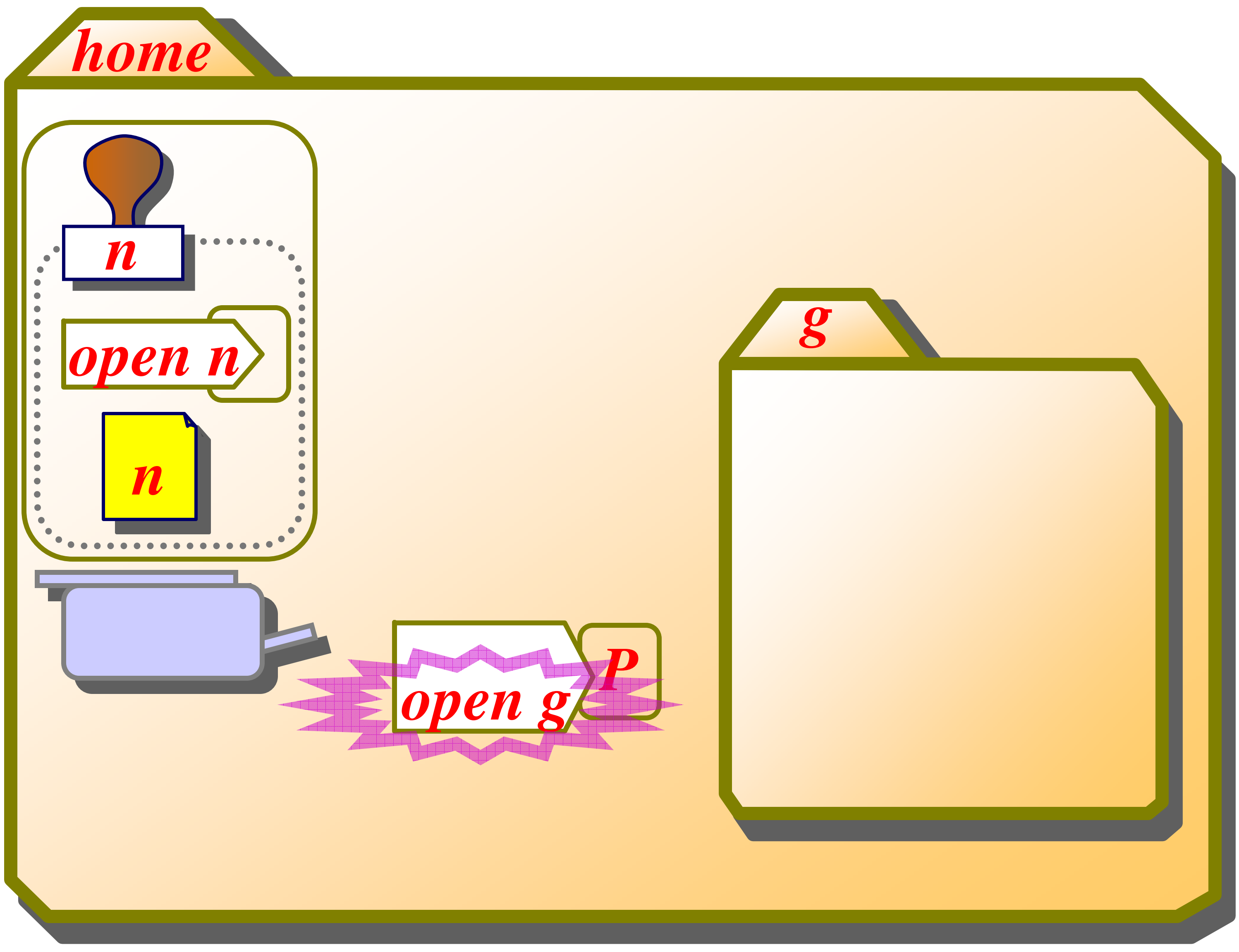
Example: Agent Authentication



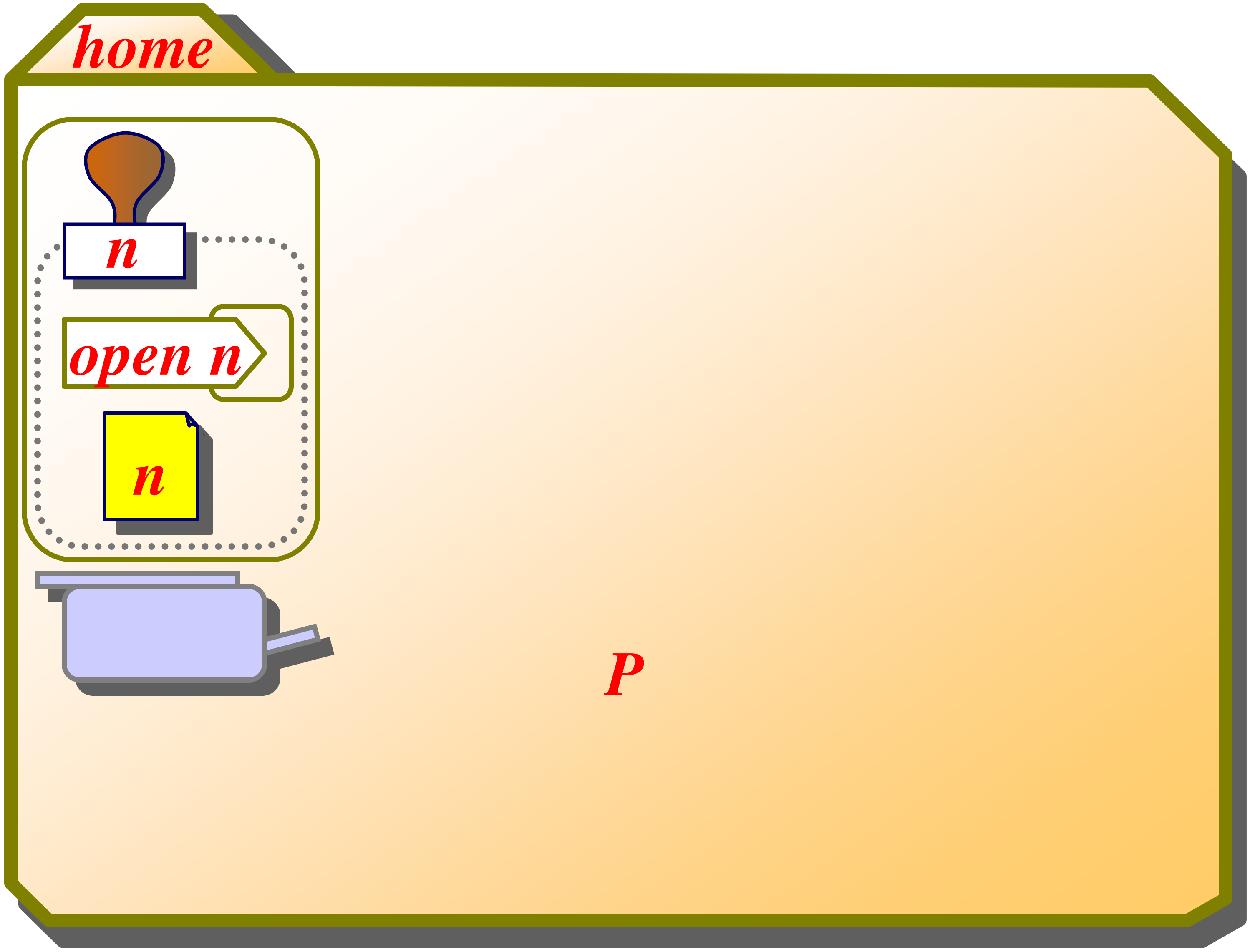
Example: Agent Authentication



Example: Agent Authentication



Example: Agent Authentication



The Ambient Calculus

$P \in \Pi$ ::=	Processes		M ::=	Messages	
$(\nu n)P$	restriction	Location Trees *Spatial*	n	name	
$\mathbf{0}$	inactivity		$in\ M$	entry capability	
$P \mid P'$	parallel		$out\ M$	exit capability	
$M[P]$	ambient		$open\ M$	open capability	
$!P$	replication		ε	empty path	
$M.P$	exercise a capability	Actions *Temporal*	$M.M'$	composite path	
$(n).P$	input locally, bind to n				
$\langle M \rangle$	output locally (async)				

$n[] \triangleq n[\mathbf{0}]$

$M \triangleq M.\mathbf{0}$ (where appropriate)

Reduction Semantics

- A structural congruence relation $P \equiv Q$:
  - On spatial expressions, $P \equiv Q$ iff P and Q denote the same tree. So, the syntax modulo $\equiv$ is a notation for spatial trees.
  - On full ambient expressions, $P \equiv Q$ if in addition the respective threads are "trivially equivalent".
  - Prominent in the definition of the logic.
- A reduction relation $P \longrightarrow^* Q$:
  - Defining the meaning of mobility and communication actions.
  - Closed up to structural congruence:

$$P \equiv P', \; P' \longrightarrow^* Q', \; Q' \equiv Q \quad \Rightarrow \quad P \longrightarrow^* Q$$

Composition

- Parallel execution is denoted by a binary operator:

$$P \mid Q$$

- It is commutative and associative:

$$P \mid Q \;\equiv\; Q \mid P$$

$$(P \mid Q) \mid R \;\equiv\; P \mid (Q \mid R)$$

- It obeys the reduction rule:

$$P \longrightarrow Q \;\Rightarrow\; P \mid R \longrightarrow Q \mid R$$

Replication

- Replication is a technically convenient way of representing iteration and recursion.

$$!P$$

- It denotes the unbounded replication of a process P.

$$!P \quad \equiv \quad P \mid !P$$

$$!(P \mid Q) \equiv !P \mid !Q$$

$$!\mathbf{0} \equiv \mathbf{0}$$

$$!P \equiv !!P$$

- There are no reduction rules for $!P$; in particular, the process P under $!$ cannot begin to reduce until it is expanded out as $P \mid !P$.

Restriction

- The restriction operator creates a new (forever unique) ambient name n within a scope P.

$$(\nu n)P$$

- As in the π-calculus, the (νn) binder can float as necessary to extend or restrict the scope of a name. E.g.:

$$(\nu n)(P \mid Q) \;\equiv\; P \mid (\nu n)Q \quad \text{if } n \notin fn(P)$$

$$(\nu n)m[P] \;\equiv\; m[(\nu n)P] \quad \text{if } n \neq m$$

- Reduction rule:

$$P \longrightarrow Q \;\Rightarrow\; (\nu n)P \longrightarrow (\nu n)Q$$

Inaction

- The process that does nothing:

$$\mathbf{0}$$

- Some garbage-collection equivalences:

$$P \mid \mathbf{0} \equiv P$$

$$!\mathbf{0} \equiv \mathbf{0}$$

$$(\nu n)\mathbf{0} \equiv \mathbf{0}$$

- This process does not reduce.

Ambients

- An ambient is written as follows, where n is the name of the ambient, and P is the process running inside of it.

$$n[P]$$

- In $n[P]$, it is understood that P is actively running:

$$P \longrightarrow Q \quad \Rightarrow \quad n[P] \longrightarrow n[Q]$$

- Multiple ambients may have the same name, (e.g., replicated servers).

Actions and Capabilities

- Operations that change the hierarchical structure of ambients are sensitive. They can be interpreted as the crossing of firewalls or the decoding of ciphertexts.
- Hence these operations are restricted by *capabilities*.

$$M.\, P$$

  - This executes an action regulated by the capability M, and then continues as the process P.
  - The reduction rules for $M.\, P$ depend on M.

Entry Capability

- An entry capability, *in m*, can be used in the action:

$$in\ m.\ P$$

- The reduction rule (non-deterministic and blocking) is:

$$n[in\ m.\ P \mid Q] \mid m[R] \quad \longrightarrow \quad m[n[P \mid Q] \mid R]$$

Exit Capability

- An exit capability, *out m*, can be used in the action:

$$out\ m.\ P$$

- The reduction rule (non-deterministic and blocking) is:

$$m[n[out\ m.\ P \mid Q] \mid R] \;\longrightarrow\; n[P \mid Q] \mid m[R]$$

Open Capability

- An opening capability, *open m*, can be used in the action:

$$\mathit{open}\ n.\ P$$

- The reduction rule (non-deterministic and blocking) is:

$$\mathit{open}\ n.\ P \mid n[Q] \;\longrightarrow\; P \mid Q$$

- An *open* operation may be upsetting to both P and Q above.
  - From the point of view of P, there is no telling in general what Q might do when unleashed.
  - From the point of view of Q, its environment is being ripped open.
- Still, this operation is relatively well-behaved because:
  - The dissolution is initiated by the agent *open n*. P, so that the appearance of Q at the same level as P is not totally unexpected;
  - *open n* is a capability that is given out by n, so $n[Q]$ cannot be dissolved if it does not wish to be.

Design Principle

- An ambient should not get killed or trapped unless:
  - It talks too much. (Making its capabilities public.)
  - It poisons itself. (Opening an untrusted intruder.)
  - Doesn't look where it's going. (Entering an untrusted ambient.)
- Some natural primitives violate this principle. E.g.:

$$n[\mathit{burst}\ n.\ P \mid Q] \ \longrightarrow\ P \mid Q$$

  - Then a mere *in* capability gives a kidnapping ability:

$$\mathit{entrap}(M) \ \triangleq\ (\nu\ k\ m)\ (m[M.\ \mathit{burst}\ m.\ \mathit{in}\ k] \mid k[])$$

$$\mathit{entrap}(\mathit{in}\ n) \mid n[P] \ \longrightarrow^*\ (\nu k)\ (n[\mathit{in}\ k \mid P] \mid k[])$$

$$\longrightarrow^*\ (\nu k)\ k[n[P]]$$

  - One can imagine lots of different mobility primitives, but one must think hard about the "security" implications of combinations of these primitives.

Ambient I/O

- Local anonymous communication within an ambient:

  $(x).\,P$ input action

  $\langle M \rangle$ async output action

- We have the reduction:

$$(x).\,P \mid \langle M \rangle \longrightarrow P\{x \leftarrow M\}$$

- This mechanism fits well with the ambient intuitions.
  - Long-range communication, like long-range movement, should not happen automatically because messages may have to cross firewalls and other obstacles.
  - Still, this is sufficient to emulate communication over named channels, etc.

Reduction

$n[in\ m.\ P \mid Q] \mid m[R]$	$\longrightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[out\ m.\ P \mid Q] \mid R]$	$\longrightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$open\ m.\ P \mid m[Q]$	$\longrightarrow P \mid Q$	(Red Open)
$(n).P \mid \langle M \rangle$	$\longrightarrow P\{n \leftarrow M\}$	(Red Comm)

$P \longrightarrow Q \;\Rightarrow\; (\nu n)P \longrightarrow (\nu n)Q$		(Red Res)
$P \longrightarrow Q \;\Rightarrow\; n[P] \longrightarrow n[Q]$		(Red Amb)
$P \longrightarrow Q \;\Rightarrow\; P \mid R \longrightarrow Q \mid R$		(Red Par)
$P' \equiv P,\ P \longrightarrow Q,\ Q \equiv Q' \;\Rightarrow\; P' \longrightarrow Q'$		(Red $\equiv$)

$\longrightarrow^*$ is the reflexive-transitive closure of $\longrightarrow$

Structural Congruence

$P \equiv P$	(Struct Refl)
$P \equiv Q \Rightarrow Q \equiv P$	(Struct Symm)
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$	(Struct Trans)
$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q$	(Struct Res)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Struct Par)
$P \equiv Q \Rightarrow !P \equiv !Q$	(Struct Repl)
$P \equiv Q \Rightarrow M[P] \equiv M[Q]$	(Struct Amb)
$P \equiv Q \Rightarrow M.P \equiv M.Q$	(Struct Action)
$P \equiv Q \Rightarrow (n).P \equiv (n).Q$	(Struct Input)
$\varepsilon.P \equiv P$	(Struct ε)
$(M.M').P \equiv M.M'.P$	(Struct .)

$(\nu n)\mathbf{0} \equiv \mathbf{0}$	(Struct Res Zero)
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	(Struct Res Res)
$(\nu n)(P \mid Q) \equiv P \mid (\nu n)Q \quad \text{if } n \notin fn(P)$	(Struct Res Par)
$(\nu n)(m[P]) \equiv m[(\nu n)P] \quad \text{if } n \neq m$	(Struct Res Amb)
$P \mid Q \equiv Q \mid P$	(Struct Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Struct Par Assoc)
$P \mid \mathbf{0} \equiv P$	(Struct Par Zero)
$!(P \mid Q) \equiv !P \mid !Q$	(Struct Repl Par)
$!\mathbf{0} \equiv \mathbf{0}$	(Struct Repl Zero)
$!P \equiv P \mid !P$	(Struct Repl Copy)
$!P \equiv !!P$	(Struct Repl Repl)

- These axioms (particularly the ones for **!**) are sound and complete with respect to equality of spatial trees: edge-labeled finite-depth unordered trees, with infinite-branching but finitely many distinct labels under each node.

Ambient Calculus: Example

$$\overbrace{a[msg[\langle M\rangle \mid out\ a.\ in\ b]]}^{\textit{location } a}\ \mid\ \overbrace{b[open\ msg.\ (n).\ P]}^{\textit{location } b}$$

$$\underbrace{a[msg[\langle M\rangle \mid out\ a.\ in\ b]]}_{\textit{send } M \textit{ from } a \textit{ to } b}\ \mid\ \underbrace{b[open\ msg.\ (n).\ P]}_{\textit{receive } n;\ \textit{do } P}$$

The packet *msg* moves from *a* to *b*, mediated by the capabilities *out a* (to exit *a*), *in b* (to enter *b*), and *open msg* (to open the *msg* envelope).

$$
\begin{array}{lll}
 & a[msg[\langle M\rangle \mid out\ a.\ in\ b]] & \mid b[open\ msg.\ (n).\ P] \\
\text{(exit)} \longrightarrow & a[] \quad \mid msg[\langle M\rangle \mid in\ b] & \mid b[open\ msg.\ (n).\ P] \\
\text{(enter)} \longrightarrow & a[] & \mid b[msg[\langle M\rangle] \mid open\ msg.\ (n).\ P] \\
\text{(open)} \longrightarrow & a[] & \mid b[\langle M\rangle \mid (n).\ P] \\
\text{(read)} \longrightarrow & a[] & \mid b[P\{n\leftarrow M\}]
\end{array}
$$

Noticeable Inequivalences

- Replication creates new names:

$$!(\nu n)P \quad \not\equiv \quad (\nu n)!P$$

- Multiple n ambients have separate identity:

$$n[P] \mid n[Q] \quad \not\equiv \quad n[P \mid Q]$$

Safe Ambients [Levi, Sangiorgi]

- "*Each action has an equal and opposite coaction.*"
- In Ambient Calculus it is difficult to count reliably the number of visitors to an ambient. The fix:

$n[\mathit{in}\ m.\ P \mid Q] \mid m[\underline{\mathit{in}}\ m.\ R \mid S]$	$\longrightarrow$	$m[n[P \mid Q] \mid R \mid S]$	(In)
$m[n[\mathit{out}\ m.\ P \mid Q] \mid \underline{\mathit{out}}\ m.\ R \mid S]$	$\longrightarrow$	$n[P \mid Q] \mid m[R \mid S]$	(Out)
$\mathit{open}\ n.\ P \mid n[\underline{\mathit{open}}\ n.Q \mid R]$	$\longrightarrow$	$P \mid Q \mid R$	(Open)
$(m).P \mid \langle M \rangle.Q$	$\longrightarrow$	$P\{m \leftarrow M\} \mid Q$	(Comm)

- The Ambient Calculus is recovered by sprinkling $!\underline{\mathit{in}}\ n$, $!\underline{\mathit{out}}\ n$, $!\underline{\mathit{open}}\ n$ appropriately.

Channeled Ambients [Pericas-Geertsen]

- Each ambient contains a list of channels c that are used for named communication within the ambient. They are restricted as usual.

$n[D, c; c\langle M\rangle.P \mid c(m).Q \mid R]$ (Send)

$\quad\longrightarrow\quad n[D, c; P \mid Q\{m\leftarrow M\} \mid R]$

$n[D; in\ m.\ P \mid Q] \mid m[E; R] \longrightarrow m[E; n[D; P \mid Q] \mid R]$ (In)

$m[E; n[D; out\ m.\ P \mid Q] \mid R] \longrightarrow n[D; P \mid Q] \mid m[E; R]$ (Out)

$m[D; open\ n.\ P \mid n[E; Q] \mid R] \longrightarrow m[D; P \mid Q \mid R]$ (Open)

Boxed Ambients [Bugliesi, Castagna, Crafa]

- I/O to parents/children is tricky to encode reliably in Ambient Calculus, but is a very natural basic primitive.
- Boxed Ambients provide it directly (simplifying Seal):

$n[in\ m.\ P \mid Q] \mid m[R]$	$\longrightarrow m[n[P \mid Q] \mid R]$	(In)
$m[n[out\ m.\ P \mid Q] \mid R]$	$\longrightarrow n[P \mid Q] \mid m[R]$	(Out)
		no (Open)
$(m).P \mid \langle M\rangle.Q$	$\longrightarrow P\{m{\leftarrow}M\} \mid Q$	(Local)
$(m)^n.P \mid n[\langle M\rangle.Q \mid R]$	$\longrightarrow P\{m{\leftarrow}M\} \mid n[Q \mid R]$	(Input n)
$\langle M\rangle^n.P \mid n[(m).Q \mid R]$	$\longrightarrow P \mid n[Q\{m{\leftarrow}M\} \mid R]$	(Output n)
$\langle M\rangle.P \mid n[(m)^{\uparrow}.Q \mid R]$	$\longrightarrow P \mid n[Q\{m{\leftarrow}M\} \mid R]$	(Input $\uparrow$)
$(m).P \mid n[\langle M\rangle^{\uparrow}.Q \mid R]$	$\longrightarrow P\{m{\leftarrow}M\} \mid n[Q \mid R]$	(Output $\uparrow$)

Ambjects [Bugliesi, Castagna]

- [CG] Ambient Calculus + [AC] Object Calculus =

$n.a(M).P \mid n[D;\, a(m).Q;\, R]$		(Send)
$\longrightarrow \quad P \mid Q\{m\leftarrow M,\, self\leftarrow n\} \mid n[D;\, a(m).Q;\, R]$		
$n[D;\, in\ m.\ P \mid Q] \mid m[E;\, R]$	$\longrightarrow\ m[E;\, n[D;\, P \mid Q] \mid R]$	(In)
$m[E;\, n[D;\, out\ m.\ P \mid Q] \mid R]$	$\longrightarrow\ n[D;\, P \mid Q] \mid m[E;\, R]$	(Out)
$m[E;\, open\ n.\ P \mid n[D;\, Q] \mid R]$	$\longrightarrow\ m[E;\, D;\, P \mid Q \mid R]$	(Open)

Joinbients [Anonymous]

- Ambient Calculus + Join Calculus =

??? $n[D; P]$			(Join)
$n[D; in\ m.\ P \mid Q] \mid m[E; R]$	$\longrightarrow$	$m[E; n[D; P \mid Q] \mid R]$	(In)
$m[E; n[D; out\ m.\ P \mid Q] \mid R]$	$\longrightarrow$	$n[D; P \mid Q] \mid m[E; R]$	(Out)
$m[E; open\ n.\ P \mid n[D; Q]]$	$\longrightarrow$	$m[E; D; P \mid Q]$	(Open)

Expressiveness: Encoding Old Concepts

- Synchronization and communication mechanisms.
- Turing machines. (Natural encoding, no I/O required.)
- Arithmetic. (Tricky, no I/O required.)
- Data structures.

  π-calculus. (Easy: channels are ambients.)

  λ-calculus. (Hard: different than encoding λ in π.)
- Spi-calculus concepts. (?)

Expressiveness: Encoding New Concepts

- Named machines and services on complex networks.
- Agents, applets, RPC.
- Encrypted data and firewalls.
- Data packets, routing, active networks.
- Dynamically linked libraries, plug-ins.
- Mobile devices.
- Public transportation.

Expressiveness: New Challenges

- The combination of mobility and security in the same formal framework is novel and intriguing.
- E.g., we can represent both mobility and security aspects of “crossing a firewall”.
- The combination of mobility and local communication raises questions about suitable synchronization models and programming constructs.

Ambients as Locks

- We can use *open* to encode locks:

$$\mathit{release}\ n.\ P \triangleq n[\,] \mid P$$

$$\mathit{acquire}\ n.\ P \triangleq \mathit{open}\ n.\ P$$

- This way, two processes can "shake hands" before proceeding with their execution:

$$\mathit{acquire}\ n.\ \mathit{release}\ m.\ P \mid \mathit{release}\ n.\ \mathit{acquire}\ m.\ Q$$

Turing Machines

$end[extendLft \mid S_0 \mid$

$square[S_1 \mid$

$square[S_2 \mid$

...

$square[S_i \mid head \mid$

...

$square[S_n\text{-}1 \mid$

$square[S_n \mid extendRht]] \ldots] \ldots]]]$

- Exercise: code up *extendLft*, *extendRht*, and (an example of) *head*. You will probably need to use restriction.

Random Access Machines [Busi]

- A finite set of registers: they can hold arbitrary natural numbers.
- A program is a sequence of numbered operations:
  - $succ(r_j)$: add 1 to the contents of register r_j and continue.
  - $decjmp(r_j, s)$: if the contents of r_j is non-zero, decrease it by 1 and continue, otherwise jump to instruction s.
  - To stop: jump to nowhere; answer is the content of registers.

$[\![r_i = 0]\!] = z_i[\ldots]$ … = some clever code

$[\![r_i = n{+}1]\!] = s_i[\ldots \mid [\![r_i = n]\!]]$

$[\![i : succ(r_j)]\!] = !p_i[inc\text{-}req_j[!in\ s_i \mid in\ z_i.\ \ldots] \mid open\ inc\text{-}ack_j.\ open\ p_{i+1}]$

$[\![i : decjmp(r_j, s)]\!] = !p_i[dec\text{-}req_j[in\ s_i] \mid zero\text{-}req_j[in\ z_i] \mid \ldots open\ ok\text{-}dec_j.\ \ldots\ open\ p_{i+1} \mid \ldots open\ ok\text{-}zero_j.\ \ldots\ open\ p_s]$

To start the program: *open* p_1

- Turing-completeness even without restriction and I/O.

Ambients as Mobile Processes

$tourist \triangleq (x).\ joe[x.\ enjoy]$

$ticket\text{-}desk \triangleq\ !\langle in\ AF81SFO.\ out\ AF81CDG\rangle$

$SFO[ticket\text{-}desk \mid tourist \mid AF81SFO[route]]$

$\longrightarrow^* \quad SFO[ticket\text{-}desk \mid$

$\qquad joe[in\ AF81SFO.\ out\ AF81CDG.\ enjoy] \mid$

$\qquad\quad AF81SFO[route]]$

$\longrightarrow^* \quad SFO[ticket\text{-}desk \mid$

$\qquad AF81SFO[route \mid joe[out\ AF81CDG.\ enjoy]]]$

Firewall Crossing (buggy)

- Assume that the shared key k is already known to the firewall and the client, and that w is the secret name of the firewall.

$$Wally \triangleq (\nu\, w\, r)\, (\langle in\ r\rangle \mid r[open\ k.\ in\ w] \mid w[open\ r.\ P])$$

$$Cleo \triangleq (x).\ k[x.\ C]$$

$$
\begin{aligned}
& Cleo \mid Wally \\
& \quad \longrightarrow^{*} (\nu\, w\, r)\, (\, (x).\ k[x.\ C] \mid \langle in\ r\rangle \mid r[open\ k.\ in\ w] \mid w[open\ r.\ P]\,) \\
& \quad \longrightarrow^{*} (\nu\, w\, r)\, (\, k[in\ r.\ C] \mid r[open\ k.\ in\ w] \mid w[open\ r.\ P]\,) \\
& \quad \longrightarrow^{*} (\nu\, w\, r)\, (\, r[k[C] \mid open\ k.\ in\ w] \mid w[open\ r.\ P]\,) \\
& \quad \longrightarrow^{*} (\nu\, w\, r)\, (\, r[C \mid in\ w] \mid w[open\ r.\ P]\,) \\
& \quad \longrightarrow^{*} (\nu\, w\, r)\, (\, w[r[C] \mid open\ r.\ P]\,) \\
& \quad \longrightarrow^{*} (\nu\, w)\ \ (\, w[C \mid P]\,)
\end{aligned}
$$

- Prone to a “stowaway attack”.

Firewall Crossing

- Assume that the shared key k is already known to the firewall and the client, and that w is the secret name of the firewall.

$$Wally \triangleq (\nu w)\,(k[in\ k.\ in\ w] \mid w[open\ k.\ P])$$

$$Cleo \triangleq k[open\ k.\ C]$$

$$
\begin{aligned}
&Cleo \mid Wally\\
&\quad\longrightarrow^* (\nu w)\,(\,k[open\ k.\ C] \mid k[in\ k.\ in\ w] \mid w[open\ k.\ P]\,)\\
&\quad\longrightarrow^* (\nu w)\,(\,k[k[in\ w] \mid open\ k.\ C] \mid w[open\ k.\ P]\,)\\
&\quad\longrightarrow^* (\nu w)\,(\,k[in\ w \mid C] \mid w[open\ k.\ P]\,)\\
&\quad\longrightarrow^* (\nu w)\quad w[k[C] \mid open\ k.\ P]\\
&\quad\longrightarrow^* (\nu w)\quad w[C \mid P]
\end{aligned}
$$

The Asynchronous π-Calculus

- A named channel is represented by an ambient.
  - The name of the channel is the name of the ambient.
  - Communication on a channel is becomes local I/O inside a channel-ambient.
  - A conventional name, *io*, is used to transport I/O requests into the channel.

$$
\begin{array}{lcl}
(ch\ n)P & \triangleq & (\nu n)\ (n[!open\ io] \mid P) \\
n(x).P & \triangleq & (\nu p)\ (io[in\ n.\ (x).\ p[out\ n.\ P]] \mid open\ p) \\
n\langle M\rangle & \triangleq & io[in\ n.\ \langle M\rangle]
\end{array}
$$

- These definitions satisfy the expected reduction in presence of a channel for *n:*

$$n(x).P \mid n\langle m\rangle \ \longrightarrow^* \ P\{x\leftarrow m\}$$

- Full translation

$$\langle\!\langle (\nu n)P \rangle\!\rangle \triangleq (\nu n)\ (n[!open\ io] \mid \langle\!\langle P \rangle\!\rangle)$$

$$\langle\!\langle n(x).P \rangle\!\rangle \triangleq (\nu p)\ (io[in\ n.\ (x).\ p[out\ n.\ \langle\!\langle P \rangle\!\rangle]] \mid open\ p)$$

$$\langle\!\langle n\langle m\rangle \rangle\!\rangle \triangleq io[in\ n.\ \langle m\rangle]$$

$$\langle\!\langle P \mid Q \rangle\!\rangle \triangleq \langle\!\langle P \rangle\!\rangle \mid \langle\!\langle Q \rangle\!\rangle$$

$$\langle\!\langle !P \rangle\!\rangle \triangleq !\langle\!\langle P \rangle\!\rangle$$

- The choice-free synchronous π-calculus, can be encoded within the asynchronous π-calculus.
- The λ-calculus can be encoded within the asynchronous π-calculus.

"Bigger"

- Ambients is certainly "bigger" than π.
- We initially strived for the smallest possible set of primitives, compatibly with our design principles. *in-out-open* are Turing-complete (even without I/O). Hard to find a smaller such set for tree operations.
- Several new versions of the Ambient Calculus primitives have been proposed:
  - They each have their merits in terms of design principles that the original Ambient Calculus does not capture or enforce.
  - They lead to even "bigger" calculi. But the features provided by Safe Ambients and Boxed Ambients (and probably more) are certainly needed in a programming language.
  - Nobody has proposed a variation that is "smaller" than the original Ambient Calculus.

The Tram Protocol

- Example:
  - A tram goes back and forth along a line with several stops.
  - A tram leaves a stop whenever it feels like.
  - A passenger can jump on any available tram.
  - A passenger cannot enter or leave a tram between stations.
- Exercise:
  - Code this in the Ambient Calculus.

The Golf Cart Protocol

- Example:
  - A golf cart carries at most one passenger. When empty, it moves randomly between “holes”.
  - A passenger can hail a golf cart. An empty golf cart will not ignore a passenger.
  - The passenger can then tell the golf cart where to go. The golf cart will then go there (without leaving the passenger behind).
  - The passenger cannot exit the golf cart until the destination.
  - The golf cart cannot leave again until the passenger has disembarked.
- Exercise:
  - Try coding this example in Ambients, Safe Ambients, and Boxed Ambients.

Think!

- To what extent is the Ambient Calculus (or its variations) WAN-sound and WAN-complete?