1 *Overture* SYNTAX AND SEMANTICS

$$v \in \mathbb{F}_p, \ w \in \text{String}, \ \iota \in \text{Clients} \subset \mathbb{N}$$

$$
\begin{aligned}
\varepsilon \quad ::= \quad & \texttt{r[w]} \mid \texttt{s[w]} \mid \texttt{m[w]} \mid \texttt{p[w]} \mid && \textit{expressions} \\
& v \mid \varepsilon - \varepsilon \mid \varepsilon + \varepsilon \mid \varepsilon * \varepsilon \\[4pt]
x \quad ::= \quad & \texttt{r[w]}@\iota \mid \texttt{s[w]}@\iota \mid \texttt{m[w]}@\iota \mid \texttt{p[w]} \mid \texttt{out}@\iota && \textit{variables} \\[4pt]
\pi \quad ::= \quad & \texttt{m[w]}@\iota := \varepsilon@\iota \mid \texttt{p[w]} := e@\iota \mid \texttt{out}@\iota := \varepsilon@\iota \mid \pi; \pi && \textit{protocols}
\end{aligned}
$$

$$
\begin{aligned}
[\![\sigma, v]\!]_\iota &= v \\
[\![\sigma, \varepsilon_1 + \varepsilon_2]\!]_\iota &= [\![[\![\sigma, \varepsilon_1]\!]_\iota + [\![\sigma, \varepsilon_2]\!]_\iota]\!] \\
[\![\sigma, \varepsilon_1 - \varepsilon_2]\!]_\iota &= [\![[\![\sigma, \varepsilon_1]\!]_\iota - [\![\sigma, \varepsilon_2]\!]_\iota]\!] \\
[\![\sigma, \varepsilon_1 * \varepsilon_2]\!]_\iota &= [\![[\![\sigma, \varepsilon_1]\!]_\iota * [\![\sigma, \varepsilon_2]\!]_\iota]\!] \\
[\![\sigma, \texttt{r[w]}]\!]_\iota &= \sigma(\texttt{r[w]}@\iota) \\
[\![\sigma, \texttt{s[w]}]\!]_\iota &= \sigma(\texttt{s[w]}@\iota) \\
[\![\sigma, \texttt{m[w]}]\!]_\iota &= \sigma(\texttt{m[w]}@\iota) \\
[\![\sigma, \texttt{p[w]}]\!]_\iota &= \sigma(\texttt{p[w]})
\end{aligned}
$$

$$(\sigma, x := \varepsilon@\iota) \Rightarrow \sigma\{x \mapsto [\![\sigma, \varepsilon]\!]_\iota\} \qquad \dfrac{(\sigma_1, \pi_1) \Rightarrow \sigma_2 \qquad (\sigma_2, \pi_2) \Rightarrow \sigma_3}{(\sigma_1, \pi_1; \pi_2) \Rightarrow \sigma_3}$$

2 *Overture* ADVERSARIAL SEMANTICS

$$
\begin{aligned}
(\sigma, x := \varepsilon@\iota) \quad &\Rightarrow_{\mathcal{A}} \quad \sigma\{x \mapsto [\![\sigma, \varepsilon]\!]_\iota\} && \iota \in H \\
(\sigma, x := \varepsilon@\iota) \quad &\Rightarrow_{\mathcal{A}} \quad \sigma\{x \mapsto [\![\textit{rewrite}_{\mathcal{A}}(\sigma_C, \varepsilon)]\!]_\iota\} && \iota \in C
\end{aligned}
$$

$$
\begin{aligned}
(\sigma, \texttt{assert}(\varepsilon_1 = \varepsilon_2)@\iota) \quad &\Rightarrow_{\mathcal{A}} \quad \sigma && \text{if } [\![\sigma, \varepsilon_1]\!]_\iota = [\![\sigma, \varepsilon_2]\!]_\iota \text{ or } \iota \in C \\
(\sigma, \texttt{assert}(\phi(\varepsilon))@\iota) \quad &\Rightarrow_{\mathcal{A}} \quad \bot && \text{if } \neg\phi(\sigma, [\![\sigma, \varepsilon]\!]_\iota)
\end{aligned}
$$

$$\dfrac{(\sigma_1, \pi_1) \Rightarrow_{\mathcal{A}} \sigma_2 \qquad (\sigma_2, \pi_2) \Rightarrow_{\mathcal{A}} \sigma_3}{(\sigma_1, \pi_1; \pi_2) \Rightarrow_{\mathcal{A}} \sigma_3} \qquad\qquad \dfrac{(\sigma_1, \pi_1) \Rightarrow_{\mathcal{A}} \bot}{(\sigma_1, \pi_1; \pi_2) \Rightarrow_{\mathcal{A}} \bot}$$

$$\dfrac{(\sigma_1, \pi_1) \Rightarrow_{\mathcal{A}} \sigma_2 \qquad (\sigma_2, \pi_2) \Rightarrow_{\mathcal{A}} \bot}{(\sigma_1, \pi_1; \pi_2) \Rightarrow_{\mathcal{A}} \bot}$$

3   *Overture* CONSTRAINT TYPING

VALUE   SECRET   RANDO
$\Gamma, \varnothing, E \vdash_\iota v : \varnothing \cdot \text{High}$ $\Gamma, \varnothing, E \vdash_\iota \mathsf{s}[w] : \{\mathsf{s}[w]@\iota\} \cdot \mathcal{L}(\iota)$ $\Gamma, \varnothing, E \vdash_\iota \mathsf{r}[w] : \{\mathsf{r}[w]@\iota\} \cdot \mathcal{L}(\iota)$

MESG   PUBM   INTEGRITYWEAKEN
$\Gamma, \varnothing, E \vdash_\iota \mathsf{m}[w] : \Gamma(\mathsf{m}[w]@\iota)$ $\Gamma, \varnothing, E \vdash_\iota \mathsf{p}[w] : \Gamma(\mathsf{p}[w])$ $\dfrac{\Gamma, R, E \vdash_\iota \varepsilon : T \cdot \varsigma_1 \qquad \varsigma_1 \preceq \varsigma_2}{\Gamma, R, E \vdash_\iota \varepsilon : T \cdot \varsigma_2}$

RANDODEDUCE
$$\frac{\Gamma, \varnothing, E \vdash_\iota \varepsilon : T \cdot \varsigma \qquad E \models \lfloor \varepsilon@\iota \rfloor = \mathsf{r}[w]@\iota'}{\Gamma, \varnothing, E \vdash_\iota \varepsilon : \{\mathsf{r}[w]@\iota\} \cdot \varsigma}$$

ENCODE
$$\frac{\Gamma, R_1, E \vdash_\iota \varepsilon_1 : T \cdot \varsigma \qquad \Gamma, R_2, E \vdash_\iota \varepsilon_2 : \{\mathsf{r}[w]@\iota\} \cdot \varsigma \qquad \oplus \in \{+, -\}}{\Gamma, R1; R_2; \mathsf{r}[w]@\iota, E \vdash_\iota \varepsilon_1 \oplus \varepsilon_2 : \{c(\mathsf{r}[w]@\iota, T)\} \cdot \varsigma}$$

BINOP
$$\frac{\Gamma, R_1, E \vdash_\iota \varepsilon_1 : T_1 \cdot \varsigma \qquad \Gamma, R_2, E \vdash_\iota \varepsilon_2 : T_2 \cdot \varsigma \qquad \oplus \in \{+, -, *\}}{\Gamma, R_1; R_2, E \vdash_\iota \varepsilon_1 \oplus \varepsilon_2 : T_1 \cup T_2 \cdot \varsigma}$$

SEND   ASSERT
$$\frac{\Gamma, R, E \vdash_\iota \varepsilon : T \cdot \mathcal{L}(\iota)}{\Gamma, R, E \vdash x := \varepsilon@\iota : \Gamma; x : T \cdot \mathcal{L}(\iota), E \wedge x = \lfloor \varepsilon@\iota \rfloor}$$ $$\frac{E \models \lfloor \varepsilon_1@\iota \rfloor = \lfloor \varepsilon_2@\iota \rfloor}{\Gamma, R, E \vdash \mathsf{assert}(\varepsilon_1 = \varepsilon_2)@\iota : \Gamma, E}$$

SEQ   CONSTRAINT
$$\frac{\Gamma_1, R_1, E_1 \vdash \pi_1 : \Gamma_2, E_2 \qquad \Gamma_2, R_2, E_2 \vdash \pi_2 : \Gamma_3, E_3}{\Gamma_1, R_1; R_2, E_1 \vdash \pi_1; \pi_2 : \Gamma_3, E_3}$$ $$\frac{\Gamma_1, R, E_1 \vdash \pi : \Gamma_2, E_2 \qquad E_1' \models E_1 \qquad E_2 \models E_2'}{\Gamma_1, R, E_1' \vdash \pi : \Gamma_2, E_2'}$$

MAC
$$\frac{E \models \mathsf{m}[wm]@\iota = \mathsf{m}[wk]@\iota + (\mathsf{m}[delta]@\iota * \mathsf{m}[ws]@\iota) \qquad \Gamma(\mathsf{m}[ws]@\iota) = T \cdot \varsigma}{\Gamma, R, E \vdash \mathsf{assert}(\mathsf{m}[wm] = \mathsf{m}[wk] + (\mathsf{m}[delta] * \mathsf{m}[ws]))@\iota : \Gamma; \mathsf{m}[ws]@\iota : T \cdot \text{High}, E}$$

4   *Prelude* SYNTAX AND SEMANTICS

$$\ell \in \text{Field}, \ y \in \text{EVar}, \ f \in \text{FName}$$

$$
\begin{array}{rcl}
e & ::= & v \mid \mathsf{r}[e] \mid \mathsf{s}[e] \mid \mathsf{m}[e] \mid \mathsf{p}[e] \mid e \ binop \ e \mid \mathsf{let} \ y = e \ \mathsf{in} \ e \mid \\
  &   & f(e, \ldots, e) \mid \{\ell = e; \ldots; \ell = e\} \mid e.\ell \\
\mathbf{c} & ::= & \mathsf{m}[e]@e := e@e \mid \mathsf{p}[e] := e@e \mid \mathsf{out}@e := e@e \mid \mathsf{assert}(e = e)@e \mid \\
  &   & f(e, \ldots, e) \mid \mathbf{c}; \mathbf{c} \mid \mathsf{pre}(E) \mid \mathsf{post}(E) \\
binop & ::= & + \mid - \mid * \mid \mathbin{++} \\
v & ::= & w \mid \iota \mid \varepsilon \mid \{\ell = v; \ldots; \ell = v\} \\
fn & ::= & f(y, \ldots, y)\{e\} \mid f(y, \ldots, y)\{\mathbf{c}\} \\
\phi & ::= & \mathsf{r}[e]@e \mid \mathsf{s}[e]@e \mid \mathsf{m}[e]@e \mid \mathsf{p}[e] \mid \mathsf{out}@e \mid \phi + \phi \mid \phi - \phi \mid \phi * \phi \\
E & ::= & \phi = \phi \mid E \wedge E
\end{array}
$$

$$\frac{e[v/y] \Rightarrow v'}{\text{let } y = v \text{ in } e \Rightarrow v'}$$

$$\frac{C(f) = y_1, \dots, y_n, e \qquad e_1 \Rightarrow v_1 \cdots e_n \Rightarrow v_n \qquad e[v_1/y_1] \cdots [v_n/y_n] \Rightarrow v}{f(e_1, \dots, e_n) \Rightarrow v}$$

$$\frac{e_1 \Rightarrow v_1 \cdots e_n \Rightarrow v_n}{\{\ell_1 = e_1; \dots; \ell_n = e_n\} \Rightarrow \{\ell_1 = v_1; \dots; \ell_n = v_n\}} \qquad \frac{e \Rightarrow \{\dots; \ell = v; \dots\}}{e.\ell \Rightarrow v} \qquad \frac{e_1 \Rightarrow w_1 \qquad e_2 \Rightarrow w_2}{e_1 \text{++} e_2 \Rightarrow w_1 w_2}$$

$$\frac{e_1 \Rightarrow \varepsilon_1 \qquad e_2 \Rightarrow \varepsilon_2 \qquad e \Rightarrow \iota}{(\pi, (E_1, E_2), \mathsf{on}, \mathsf{assert}(e_1 = e_2)@e) \Rightarrow (\pi; \mathsf{assert}(\varepsilon_1 = \varepsilon_2)@\iota, (E_1, E_2 \wedge \lfloor \varepsilon_1 @ \iota \rfloor = \lfloor \varepsilon_2 @ \iota \rfloor), \mathsf{on})}$$

$$\frac{e_1 \Rightarrow \varepsilon_1 \qquad e_2 \Rightarrow \varepsilon_2 \qquad e \Rightarrow \iota}{(\pi, (E_1, E_2), \mathsf{off}, \mathsf{assert}(e_1 = e_2)@e) \Rightarrow (\pi; \mathsf{assert}(\varepsilon_1 = \varepsilon_2)@\iota, (E_1, E_2, \mathsf{off})}$$

$$\frac{e_1 \Rightarrow w \qquad e_2 \Rightarrow \iota_1 \qquad e_3 \Rightarrow \varepsilon \qquad e_4 \Rightarrow \iota_2}{(\pi, (E_1, E_2), \mathsf{on}, \mathsf{m}[e_1]@e_2 := e_3@e_4) \Rightarrow (\pi; \mathsf{m}[w]@\iota_1 := \varepsilon@\iota_2, (E_1 \wedge \mathsf{m}[w]@\iota_1 = \lfloor \varepsilon@\iota_2 \rfloor, E_2), \mathsf{on})}$$

$$\frac{e_1 \Rightarrow w \qquad e_2 \Rightarrow \iota_1 \qquad e_3 \Rightarrow \varepsilon \qquad e_4 \Rightarrow \iota_2}{(\pi, (E_1, E_2), \mathsf{off}, \mathsf{m}[e_1]@e_2 := e_3@e_4) \Rightarrow (\pi; \mathsf{m}[w]@\iota_1 := \varepsilon@\iota_2, (E_1, E_1), \mathsf{off})}$$

$$(\pi, (E_1, E_2), \mathsf{on}, \mathsf{pre}(E)) \Rightarrow (\pi, E_1, E_2 \wedge E, \mathsf{off})$$

$$(\pi, (E_1, E_2), \mathsf{off}, \mathsf{post}(E)) \Rightarrow (\pi, (E_1 \wedge E, E_2), \mathsf{on})$$

$$\frac{(\pi_1, (E_{11}, E_{12}), sw_1, \mathbf{c}_1) \Rightarrow (\pi_2, (E_{21}, E_{22}), sw_2) \qquad (\pi_2, (E_{21}, E_{22}), sw_2, \mathbf{c}_2) \Rightarrow (\pi_3, (E_{31}, E_{32}), sw_3)}{(\pi_1, (E_{11}, E_{12}), sw_1, \mathbf{c}_1; \mathbf{c}_2) \Rightarrow (\pi_3, (E_{31}, E_{32}), sw_3)}$$

$$\frac{C(f) = y_1, \dots, y_n, \mathbf{c}}{e_1 \Rightarrow v_1 \cdots e_n \Rightarrow v_n \qquad (\pi_1, (E_{11}, E_{12}), sw_1, \mathbf{c}[v_1/y_1,] \cdots [v_n/y_n]) \Rightarrow (\pi_2, (E_{21}, E_{22}), sw_2)}{(\pi_1, (E_{11}, E_{12}), sw_1, f(e_1, \dots, e_n)) \Rightarrow (\pi_2, (E_{21}, E_{22}), sw_2)}$$

5 EXAMPLES

```
encodegmw(in, i1, i2) {
  m[in]@i2 := (s[in] xor r[in])@i2;
  m[in]@i1 := r[in]@i2
}

andtablegmw(b1, b2, r) {
  let r11 = r xor (b1 xor true) and (b2 xor true) in
  let r10 = r xor (b1 xor true) and (b2 xor false) in
  let r01 = r xor (b1 xor false) and (b2 xor true) in
  let r00 = r xor (bl xor false) and (b2 xor false) in
  { row1 = r11; row2 = r10; row3 = r01; row4 = r00 }
```

```
 99        }
100
101        andgmw(z, x, y) {
102          pre();
103          let r = r[z] in
104          let table = andtablegmw(m[x],m[y],r) in
105          m[z]@2 := OT4(m[x],m[y],table,2,1);
106          m[z]@1 := r@1;
107          post(m[z]@1 xor m[z]@2 == (m[x]@1 xor m[x]@2) and (m[y]@1 xor m[y]@2))
108        }
109
110        xorgmw(z, x, y) {
111          m[z]@1 := (m[x] xor m[y])@1; m[z]@2 := (m[x] xor m[y])@2;
112        }
113
114        decodegmw(z) {
115          p["1"] := m[z]@1; p["2"] := m[z]@2;
116          out@1 := (p["1"] xor p["2"])@1;
117          out@2 := (p["1"] xor p["2"])@2
118        }
119
120        encodegmw("x",2,1);
121        encodegmw("y",2,1);
122        encodegmw("z",1,2);
123        andgmw("g1","x","z");
124        xorgmw("g2","g1","y");
125        decodegmw("g2")
126        pre();
127        post(out@1 == (s["x"]@1 and s["z"]@2) xor s["y"]@1)
128
129
130      secopen(w1,w2,w3,i1,i2) {
131          pre(m[w1++"m"]@i2 == m[w1++"k"]@i1 + (m["delta"]@i1 * m[w1++"s"]@i2 /\
132              m[w1++"m"]@i2 == m[w1++"k"]@i1 + (m["delta"]@i1 * m[w1++"s"]@i2));
133          let locsum =  macsum(macshare(w1), macshare(w2)) in
134          m[w3++"s"]@i1 := (locsum.share)@i2;
135          m[w3++"m"]@i1 := (locsum.mac)@i2;
136          auth(m[w3++"s"],m[w3++"m"],mack(w1) + mack(w2),i1);
137          m[w3]@i1 := (m[w3++"s"] + (locsum.share))@i1
138      }
139
140
141      _open(x,i1,i2){
142        m[x++"exts"]@i1 := m[x++"s"]@i2;
143        m[x++"extm"]@i1 := m[x++"m"]@i2;
144        assert(m[x++"extm"] == m[x++"k"] + (m["delta"] * m[x++"exts"]));
145        m[x]@i1 := (m[x++"exts"] + m[x++"s"])@i2
146      }`
147
```

```
148
149    _sum(z, x, y,i1,i2) {
150        pre(m[x++"m"]@i2 == m[x++"k"]@i1 + (m["delta"]@i1 * m[x++"s"]@i2 /\
151            m[y++"m"]@i2 == m[y++"k"]@i1 + (m["delta"]@i1 * m[y++"s"]@i2));
152        m[z++"s"]@i2 := (m[x++"s"] + m[y++"s"])@i2;
153        m[z++"m"]@i2 := (m[x++"m"] + m[y++"m"])@i2;
154        m[z++"k"]@i1 := (m[x++"k"] + m[y++"k"])@i1;
155        post(m[z++"m"]@i2 == m[z++"k"]@i1 + (m["delta"]@i1 * m[z++"s"]@i2)
156    }
157
158    sum(z,x,y) { _sum(z,x,y,1,2);_sum(z,x,y,2,1) }
159
160    open(x) { _open(x,1,2); _open(x,2,1) }
161
162
163    sum("a","x","d");
164    open("d");
165    sum("b","y","e");
166    open("e");
167    let xys =
168        macsum(macctimes(macshare("b"), m["d"]),
169                macsum(macctimes(macshare("a"), m["e"]),
170                        macshare("c")))
171    let xyk = mack("b") * m["d"] + mack("a") * m["e"] + mack("c")
172
173    secopen("a","x","d",1,2);
174      secopen("a","x","d",2,1);
175      secopen("b","y","e",1,2);
176      secopen("b","y","e",2,1);
177      let xys =
178        macsum(macctimes(macshare("b"), m["d"]),
179                macsum(macctimes(macshare("a"), m["e"]),
180                        macshare("c")))
181      in
182      let xyk = mack("b") * m["d"] + mack("d") * m["d"] + mack("c")
183      in
184      secreveal(xys,xyk,"1",1,2);
185      secreveal(maccsum(xys,m["d"] * m["e"]),
186                xyk - m["d"] * m["e"],
187                "2",2,1);
188      out@1 := (p[1] + p[2])@1;
189      out@2 := (p[1] + p[2])@2;
190
191
192
193
194
195
196
```