Simple 2-out-of-2 secret-sharing scheme:

Σ:		
$\mathcal{M} = \{0,1\}^\ell$ $t = 2$ $n = 2$	$\underline{\text{Share}(m):}$ $s_1 \leftarrow \{0,1\}^\ell$ $s_2 := s_1 \oplus m$ return (s_1, s_2)	$\underline{\text{Reconstruct}(s_1, s_2):}$ return $s_1 \oplus s_2$

Claim:

Σ is a secure 2-out-of-2 secret-sharing scheme. That is,

$$\mathcal{L}^{\Sigma}_{\text{tsss-L}} \equiv \mathcal{L}^{\Sigma}_{\text{tsss-R}}.$$

We will **use** the fact that one-time pad has one-time security ($\mathcal{L}^{\text{OTP}}_{\text{ots-L}} \equiv \mathcal{L}^{\text{OTP}}_{\text{ots-R}}$).

Security proof

$\mathcal{L}^{\Sigma}_{\text{tsss-L}}$

$\underline{\textsc{query}(m_L, m_R, U):}$
  if $|U| \geq 2$: return `err`
  $\boldsymbol{s} \leftarrow \Sigma.\mathsf{Share}(m_L)$
  return $(s_i)_{i \in U}$

Starting point is $\mathcal{L}^{\Sigma}_{\text{tsss-L}}$.

$$\mathcal{L}^{\Sigma}_{\text{tsss-L}}$$

$\underline{\text{QUERY}(m_L, m_R, U):}$
  if $|U| \geq 2$: return err
  $\boldsymbol{s} \leftarrow \Sigma.\text{Share}(m_L)$
  return $(s_i)_{i \in U}$

Starting point is $\mathcal{L}^{\Sigma}_{\text{tsss-L}}$. Fill in details of Σ

$\mathcal{L}^{\Sigma}_{\text{tsss-L}}$

$\underline{\text{QUERY}(m_L, m_R, U):}$

if $|U| \geq 2$: return `err`
$s_1 \leftarrow \{0,1\}^\ell$
$s_2 := s_1 \oplus m_L$
return $(s_i)_{i \in U}$

Details of Σ filled in.

$$\mathcal{L}^{\Sigma}_{\text{tsss-L}}$$

$\underline{\textsc{query}(m_L, m_R, U):}$

if $|U| \geq 2$: return err

$s_1 \leftarrow \{0,1\}^{\ell}$

$s_2 := s_1 \oplus m_L$

return $(s_i)_{i \in U}$

Details of Σ filled in.

Security proof

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

Duplicate body for the 3 possible authorized sets: $\{1\}, \{2\}, \emptyset$.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

Duplicate body for the 3 possible authorized sets: $\{1\}, \{2\}, \emptyset$.

Security proof

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

s_2 not used in this branch.

Security proof

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

s_2 not used in this branch, so we can change how it is assigned.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

s_2 not used in this branch, so we can change how it is assigned.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_L
    return s_2
  else return null
```

Recognize s_2 as OTP encryption of m_L.

Security proof

$\textsc{query}(m_L, m_R, U)$:
- if $|U| \geq 2$: return $\texttt{err}$
- if $U = \{1\}$:
  - $s_1 \leftarrow \{0,1\}^\ell$
  - $s_2 := s_1 \oplus m_R$
  - return s_1
- elsif $U = \{2\}$:
  - $s_2 \leftarrow \textsc{query}'(m_L, m_R)$
  - return s_2
- else return $\texttt{null}$

$\diamond$

$\mathcal{L}^{\text{OTP}}_{\text{ots-L}}$

$\textsc{query}'(m_L, m_R)$:
- $k \leftarrow \{0,1\}^\ell$
- $c := k \oplus m_L$
- return c

Write it in terms of the “left” OTP security library.

$$\begin{array}{|l|}\hline \underline{\textsc{query}(m_L, m_R, U):} \\ \quad \text{if } |U| \geq 2\text{: return } \mathtt{err} \\ \quad \text{if } U = \{1\}: \\ \qquad s_1 \leftarrow \{0,1\}^\ell \\ \qquad s_2 := s_1 \oplus m_R \\ \qquad \text{return } s_1 \\ \quad \text{elsif } U = \{2\}: \\ \qquad s_2 \leftarrow \textsc{query}'(m_L, m_R) \\ \qquad \text{return } s_2 \\ \quad \text{else return } \mathtt{null} \\ \hline \end{array} \diamond \begin{array}{|l|}\hline \mathcal{L}^{\text{OTP}}_{\text{ots-L}} \\ \hline \underline{\textsc{query}'(m_L, m_R):} \\ \quad k \leftarrow \{0,1\}^\ell \\ \quad c := k \oplus m_L \\ \quad \text{return } c \\ \hline \end{array}$$

Write it in terms of the “left” OTP security library.

Security proof

$$\begin{array}{|l|}\hline \underline{\textsc{query}(m_L, m_R, U):} \\ \quad \text{if } |U| \geq 2\text{: return } \texttt{err} \\ \quad \text{if } U = \{1\}: \\ \qquad s_1 \leftarrow \{0,1\}^\ell \\ \qquad s_2 := s_1 \oplus m_R \\ \qquad \text{return } s_1 \\ \quad \text{elsif } U = \{2\}: \\ \qquad s_2 \leftarrow \textsc{query}'(m_L, m_R) \\ \qquad \text{return } s_2 \\ \quad \text{else return } \texttt{null} \\ \hline \end{array} \diamond \begin{array}{|l|}\hline \mathcal{L}^{\text{OTP}}_{\text{ots-R}} \\ \hline \underline{\textsc{query}'(m_L, m_R):} \\ \quad k \leftarrow \{0,1\}^\ell \\ \quad c := k \oplus m_R \\ \quad \text{return } c \\ \hline \end{array}$$

OTP security says we can replace $\mathcal{L}_{\text{ots-L}}$ with $\mathcal{L}_{\text{ots-R}}$.

Security proof

$$\begin{array}{l}
\underline{\textsc{query}(m_L, m_R, U):} \\
\quad \text{if } |U| \geq 2\text{: return } \texttt{err} \\
\quad \text{if } U = \{1\}\text{:} \\
\qquad s_1 \leftarrow \{0,1\}^\ell \\
\qquad s_2 := s_1 \oplus m_R \\
\qquad \text{return } s_1 \\
\quad \text{elsif } U = \{2\}\text{:} \\
\qquad s_2 \leftarrow \textsc{query}'(m_L, m_R) \\
\qquad \text{return } s_2 \\
\quad \text{else return } \texttt{null}
\end{array}
\diamond
\begin{array}{l}
\mathcal{L}^{\text{OTP}}_{\text{ots-R}} \\
\hline
\underline{\textsc{query}'(m_L, m_R):} \\
\quad k \leftarrow \{0,1\}^\ell \\
\quad c := k \oplus m_R \\
\quad \text{return } c
\end{array}$$

OTP security says we can replace $\mathcal{L}_{\text{ots-L}}$ with $\mathcal{L}_{\text{ots-R}}$.

$$\begin{array}{|l|}\hline \underline{\textsc{query}(m_L, m_R, U):} \\ \quad \text{if } |U| \geq 2\text{: return } \mathtt{err} \\ \quad \text{if } U = \{1\}: \\ \qquad s_1 \leftarrow \{0,1\}^\ell \\ \qquad s_2 := s_1 \oplus m_R \\ \qquad \text{return } s_1 \\ \quad \text{elsif } U = \{2\}: \\ \qquad s_2 \leftarrow \textsc{query}'(m_L, m_R) \\ \qquad \text{return } s_2 \\ \quad \text{else return } \mathtt{null} \\ \hline \end{array} \diamond \begin{array}{|l|}\hline \mathcal{L}^{\text{OTP}}_{\text{ots-R}} \\ \hline \underline{\textsc{query}'(m_L, m_R):} \\ \quad k \leftarrow \{0,1\}^\ell \\ \quad c := k \oplus m_R \\ \quad \text{return } c \\ \hline \end{array}$$

Inline the subroutine call.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_2
  else return null
```

Inline the subroutine call.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_2
  else return null
```

Inline the subroutine call.

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  if U = {1}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_1
  elsif U = {2}:
    s_1 ← {0,1}^ℓ
    s_2 := s_1 ⊕ m_R
    return s_2
  else return null
```

Three branches of if-statement can be unified.

$\underline{\textsc{query}(m_L, m_R, U):}$

if $|U| \geq 2$: return err

$s_1 \leftarrow \{0,1\}^\ell$

$s_2 := s_1 \oplus m_R$

return $(s_i)_{i \in U}$

Three branches of if-statement can be unified..

$\underline{\text{QUERY}(m_L, m_R, U):}$

if $|U| \geq 2$: return err
$s_1 \leftarrow \{0,1\}^\ell$
$s_2 := s_1 \oplus m_R$
return $(s_i)_{i \in U}$

Three branches of if-statement can be unified..

```
QUERY(m_L, m_R, U):
  if |U| ≥ 2: return err
  s_1 ← {0,1}^ℓ
  s_2 := s_1 ⊕ m_R
  return (s_i)_{i∈U}
```

This happens to be $\mathcal{L}^{\Sigma}_{\text{tsss-R}}$.

$\mathcal{L}^{\Sigma}_{\text{tsss-R}}$
$\underline{\text{QUERY}(m_L, m_R, U)}$:
if $\lvert U \rvert \geq 2$: return `err`
$\boldsymbol{s} \leftarrow \Sigma.\text{Share}(m_R)$
return $(s_i)_{i \in U}$

This happens to be $\mathcal{L}^{\Sigma}_{\text{tsss-R}}$.

Security proof

$\mathcal{L}^{\Sigma}_{\text{tsss-R}}$
<u>QUERY(m_L, m_R, U):</u> if $\|U\| \geq 2$: return `err` $\boldsymbol{s} \leftarrow \Sigma.\text{Share}(m_R)$ return $(s_i)_{i \in U}$

This happens to be $\mathcal{L}^{\Sigma}_{\text{tsss-R}}$.

Summary

We showed:

$$\begin{array}{|l|}\hline \qquad\qquad \mathcal{L}^{\Sigma}_{\text{tsss-L}} \\ \hline \underline{\textsc{query}(m_L, m_R, U):} \\ \quad \text{if } |U| \geq 2\text{: return } \texttt{err} \\ \quad \boldsymbol{s} \leftarrow \Sigma.\text{Share}(m_L) \\ \quad \text{return } (s_i)_{i \in U} \\ \hline \end{array} \equiv \cdots \equiv \begin{array}{|l|}\hline \qquad\qquad \mathcal{L}^{\Sigma}_{\text{tsss-R}} \\ \hline \underline{\textsc{query}(m_L, m_R, U):} \\ \quad \text{if } |U| \geq 2\text{: return } \texttt{err} \\ \quad \boldsymbol{s} \leftarrow \Sigma.\text{Share}(m_R) \\ \quad \text{return } (s_i)_{i \in U} \\ \hline \end{array}$$

So Σ is a secure 2-out-of-2 secret-sharing scheme.