Ex 1:

1. check $e(H(m), y) = e(\sigma, g)$
2. assume there is such an adversary $\mathcal{A}$.
   - $\mathcal{A}$ obtains access to public key, O_H (programmable random oracle), and O_σ (signing oracle)
   - build CDH adv:
     - obtain $A = g^a$, $B = g^b$, goal: compute g^{ab} using $\mathcal{A}$
     - set public key $y = A$ and send to $\mathcal{A}$.
     - draw some $i^* \leftarrow [1, Q]$, where Q is an upper bound on $\# O_H$ - queries
     - simulate O_H:
       - $O_H(q_i)$:
         if $i = i^*$: output B
         else g^{r_i} for $r_i \leftarrow \mathbb{Z}_p$

         (note: distribution is random, i^* is hidden from $\mathcal{A}$)
     - simulate $O_\sigma(m_j)$:
       - if $m_j = q_{i^*}$: abort
         else output A^{r_j} where m_j is the j-th oracle query
     - at end, if no abort, obtain $\bar{\sigma}, \bar{m}$ for unqueried $\bar{m}$
       and $e(H(\bar{m}), A) = e(\bar{\sigma}, g)$
       $\Rightarrow \bar{\sigma} = H(\bar{m})^a$
     - if $\bar{m}$ is the i^*-th O_H-query, then $H(\bar{m}) = B$
       ↳ output CDH solution $\bar{\sigma}$
     - as i^* hidden from $\mathcal{A}$, prob. of that is $\frac{1}{Q}$
       (note that in that case $\bar{m}$ never queried to O_σ and we never abort)

   with prob. $\frac{1}{Q} \cdot \varepsilon$ we can break CDH, where ε is success prob. of $\mathcal{A}$
   as Q is poly-bounded, ε has to be $\underbrace{\text{"tiny"}}_{\text{negligible}}$, as we assume CDH is hard

3. Signer: y, g, x s.t. $y = g^x$ — User: y, g

   User: $c = H(m) \cdot g^r$ for random r; User $\xrightarrow{c}$ Signer

   Signer: $d = c^x$; Signer $\xrightarrow{d}$ User

   User: $\sigma = d \cdot y^{-r}$

   then: $\sigma = c^x \cdot g^{xr} = H(m)^x \cdot g^{rx - xr} = H(m)^x$

4. first message is uniformly random in $\mathbb{G}_1$, and σ is uniquely determined by y and $H(m)$ → both cases identically distributed
5. similar to 2. but:
   - simulate signing oracle O_σ via the $(\cdot)^x$ oracle
   - simulate O_H with the second oracle: $O_H(q_i) = h_i$

   as a BLS adversary gives a $Q+1$-th h_i^x, it cannot exist
6. - it is interactive (here: assuming one-more CDH is hard is equivalent to assume that BLS is secure)
     ↳ strong assumption
   - but: can analyze one-more CDH in the generic group model (similar to Dlog analysis of previous TD)
     ↳ for generic adversaries it is a reasonable assumption

Ex 2:

1. similar to semantic security for encryption schemes, intuitively:
- given Extract oracle, should be hard to distinguish ciphertexts for unqueried sk_{id} from random
- at least: should be hard to compute sk_{id^*}, even if sk_{id_i} is known
  (otherwise colluding with other users might allow to compute sk_{macron} for example)

2. · KeyGen(1^λ): run $mk, msk \leftarrow$ IBE.Setup(1^λ)
   output $pk = mk$, $sk = msk$
   Sign(sk, m): output $\sigma \leftarrow$ IBE.Extract(msk, m)
   Verify(pk, m, σ_m): $c \leftarrow$ IBE.Encrypt(mpk, m^*) for random m^*
   $m' \leftarrow$ IBE.Decrypt($\underbrace{\sigma_m}_{\text{acts as decryption key}}, c$)

   check $m = m'$

   Note: sk_{id} should be hard to compute (even if other sk_{id_i} are known)
   verification checks whether the decryption key works

3. we obtain BLS
4. CDH
5. both identically distributed
6. - it can be checked that with $\sigma_1 = A^{(m_i - m^*)r_i} \cdot B^{-\frac{\delta}{m_i - m^*}} \cdot g^{\delta \cdot r_i}$
   for $r_i \leftarrow \mathbb{Z}_p$, then (σ_1, σ_2) is a valid signature for m_i
   (note that for $m_i = m^*$ this is not possible but as the adversary declares m^* before we need to setup the pk, this is fine)
   - with $pk = (A, A^{-m^*} \cdot g^{\delta})$ and the above signing mechanism,
   we can simulate the challenger
   - it's not hard to check that a signature σ^* for m^* allows to compute $g^{ab} = \sigma_1^* / \sigma_2^{*\delta}$
7. no, because the adversaries usually see the public key of the signer before they decide for which message to forge a signature (but depends on setting of course)
8. given $(\sigma_1, \sigma_2) = (sk \cdot (u^m h)^r, g^r)$, then
   $(\sigma_1', \sigma_2') = (\sigma_1 \cdot (u^m h)^{\Delta r}, \sigma_2 \cdot g^{\Delta r})$ is a signature with randomness $r' = \Delta r + r$
9. sign $H(m)$ instead of m
   (provably secure by guessing which oracle query corresponds to the forgery, similar to in Ex 1.1)

Ex 3:

1.
$$s^T c = (-\bar{s}^T | 1)\left(\begin{bmatrix} \bar{A} \\ \bar{s}^T \bar{A} + e^T \end{bmatrix} \cdot r + \begin{bmatrix} 0^{n-1} \\ \lfloor q/2 \rfloor \cdot \mu \end{bmatrix} \right)$$
$$= (-\bar{s}^T \bar{A} + \bar{s}^T \bar{A} + e^T) \cdot r + \lfloor q/2 \rfloor \cdot \mu$$
$$= \underbrace{\underbrace{e^T r}_{\text{small}} + \lfloor q/2 \rfloor \mu}_{\substack{\text{close to } 0 \text{ if } \mu = 0 \\ \text{close to } \lfloor q/2 \rfloor \text{ if } \mu = 1}}$$

2. $\bar{s}^T \bar{A} + e^T$ looks random under LWE, thus hint yields that Encrypt(μ) is uniformly distr.
3. simple calculation (note: error increases slightly)
4. no, a non-small term appears in the product
5. add Enc(sk) into pk, add / multiply once, then decrypt homomorphically using Enc(sk)
   ↳ yields "almost fresh" ciphertext (which allows for at least one more add/multiply)