Exercise: Architecture

1 Evaluate the attack surface of MVM

Recall the equations for determining attack surface metrics based on the *damage-effort ratio* an attack surface, i.e.

$$attack\ surface\ metric = \langle \sum_{m \in M} DER_m(m), \sum_{c \in C} DER_c(c), \sum_{i \in I} DER_i(i) \rangle$$

where $DER_m = \frac{privilege}{access\ rights}$, $DER_c = \frac{protocol}{access\ rights}$, and $DER_i = \frac{data\ type}{access\ rights}$.

It's necessary to agree scores for possible values of $privilege$, $protocol$, $data\ type$, such that the higher the number, the more attack surface is exposed. Scores also need to be agreed for $access\ rights$ but, in this case, we want this value to be as high as possible.

For $privilege$ values, it might be useful to consider components that are assigned to run at 'super-user' levels of privilege (10), at a level suitable for an authenticated user (5), and a level suitable for 'guests' (1).

For $protocol$ values, you may want to think about of possible values between open, unstructured channels (10), and those which are strictly defined (1).

For $data\ type$ values, think in terms of data types that impose few (if any) restrictions on content such as binary data or text that can be interpreted as code (10), and data types that are constrained and typed, e.g. XML with a complementary schema or DTD (1).

The one consistent value in the denominator of each metric is the $access\ rights$ value. Values might include superuser and/or access is only available locally (10), access only to those who are authenticated,

Unfortunately, it appears to be comparatively easy to convince the server side processes you are some arbitrary user identity, so we can assume that access rights necessary for each DER value is at the guest level (i.e. 1).

For any values which appears to be hidden or ambiguous, fear the worse and assign these a value of 10 for $privilege$, $protocol$, $data\ type$, or 1 for $access\ right$.

Armed with this data, we can now start attempting to evaluate the DER metrics for each component (DER_m and DER_i) and connector (DER_c).

DER_m

- MVM clients runs with the authenticated privileges (5), and requires guest access rights to interact with it (1).
- MCO runs with superuser privileges (10), and requires guest access rights to interact with it (1).
- MD runs with authenticated privileges (5), and requires guest access rights to interact with it (1)

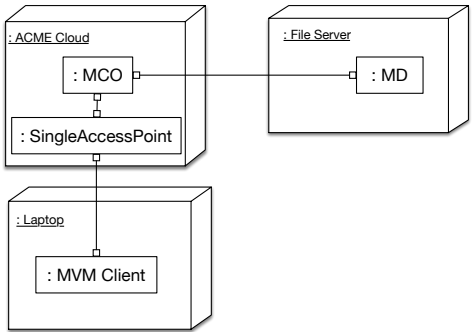

Figure 1: Revised deployment diagram

This gives us: $DER_m = \frac{5}{1} + \frac{10}{1} + \frac{5}{1} = 20$.

DER_c

- The channel connecting the MVM client with an MCO service runs over RPC (1) with guest access rights (1).
- The channel connecting the MVM client with an MD daemon runs over RPC (1) with guest access rights (1).
- The channel connecting the MCO service with an MD daemon runs over RPC (1) with guest access rights (1).

This gives us: $DER_c = \frac{1}{1} + \frac{1}{1} + \frac{1}{1} = 3$.

DER_i

- The data types used by the MVM client are undefined (10), and require guest access rights (1).
- The data types used by MCO service appear to be constrained XML (1), and require guest access rights (1).
- The data types used by the MD daemon is based on RCS, but – while standardised – it is somewhat permissive to changes leading to easy corruption of source files or change history (7). These data types also require guest access rights (1).

This gives us: $DER_i = \frac{10}{1} + \frac{1}{1} + \frac{7}{1} = 18$, and a final attack surface metric: $\langle 20, 3, 18 \rangle$.

2 Using appropriate security patterns, modify this software architecture to support authentication

Given the issues associated with the access rights, applying a SINGLE ACCESS POINT pattern, which employs a CHECKPOINT potentially has only a modest effect considering the architecture's deployment (Figure 1). Based on the original components, this would reduce the attack surface metric down to as little as $\langle 4, 0.6, 3.6 \rangle$.