I need to prove that a person knew a file with a publicly known blake3 hash. The way I'd like to do it is to provide the file as a private parameter, the blake3 hash as a public parameter, and a signed message as another public parameter (so we know the person signed the message and provided it to the zero-knowledge system along with the file).

How do I deal with hashing a variable-length file? I tried to feel the blake3 function a slice, but it didn't work.

```
fn main() { let mut slice: [u8] = &[0; 10]; let hash = std::hash::blake3(slice); }
```

The error is:

```
ori@CryptoDocLaptop:~/noir/hello_world$ nargo execute warning: unused variable hash ┌── src/main.nr:7:9 | 7 | let hash = std::hash::blake3(slice); | ---- unused variable |

error: Expected type [u8; _], found type [u8] ┌── src/main.nr:7:34 | 7 | let hash = std::hash::blake3(slice); | ----- |

Aborting due to 1 previous error ori@CryptoDocLaptop:~/noir/hello_world$
```

All the tricks I tried to get it to work also failed.

Is there a way to do it, or does the application need to compile a new circuit for every file size?