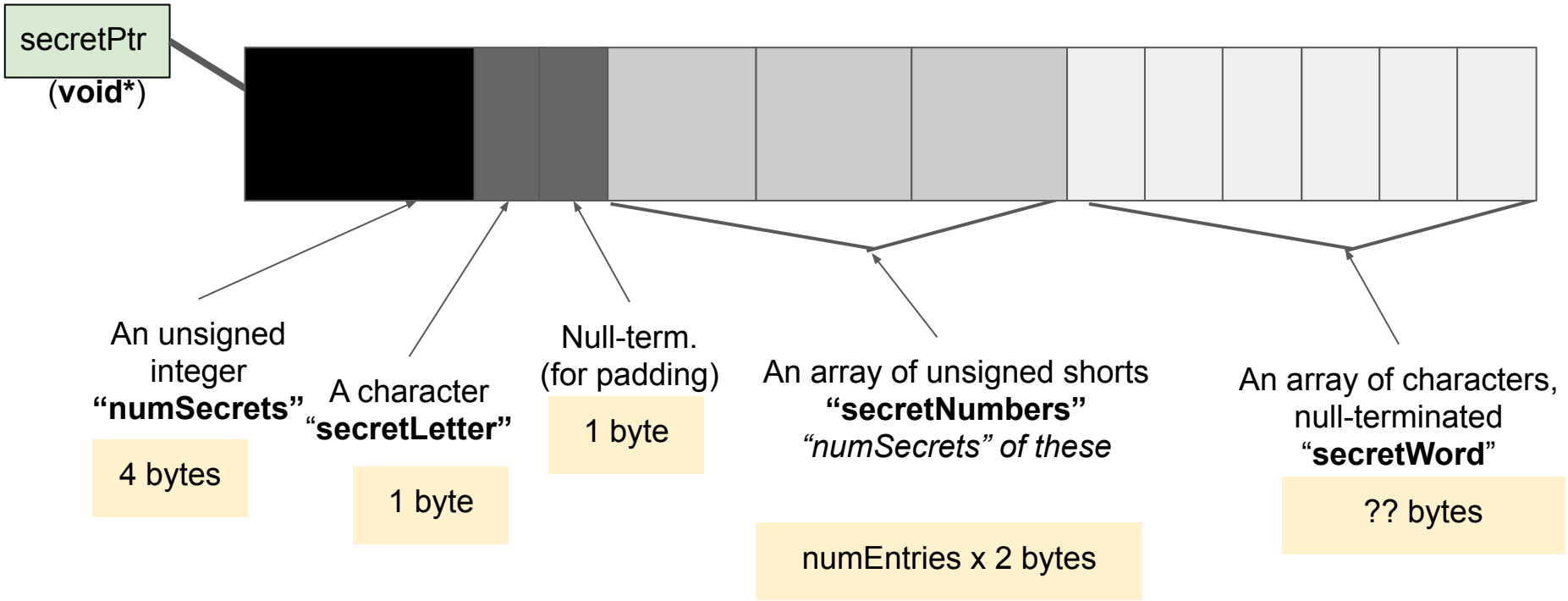
A binary data file (e.g., a bunch of data mushed together), with the following format:



How to: get numSecrets?

secretPtr

(**void***)

An

“nu

4

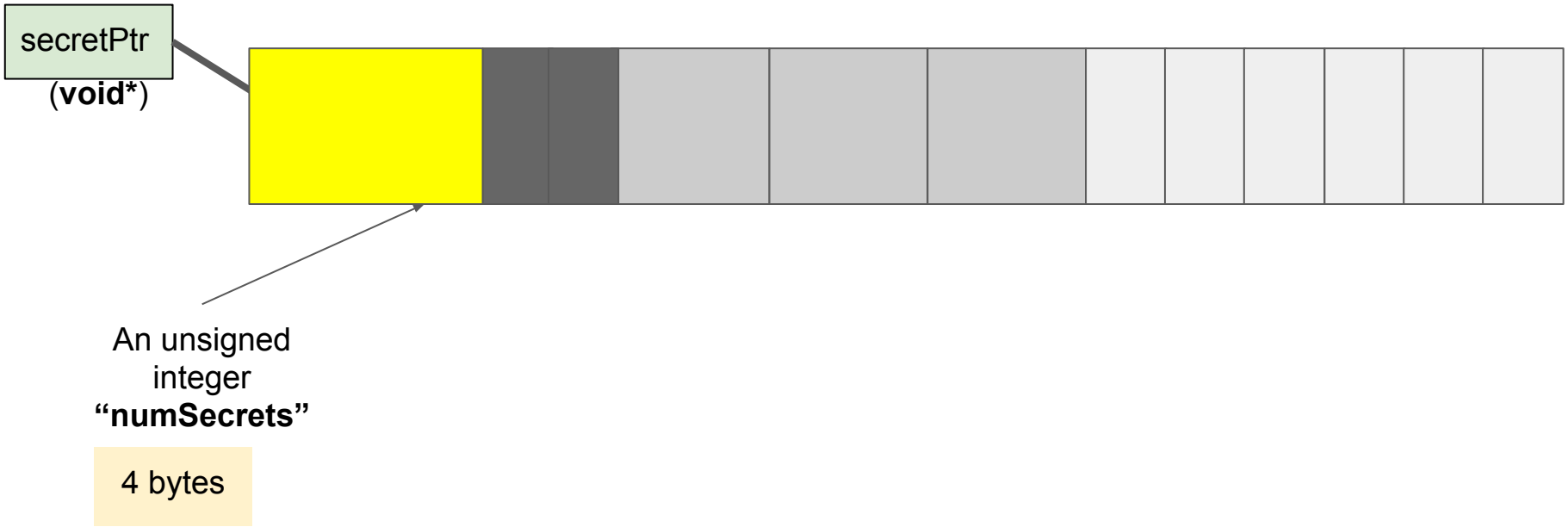
racters,

ated

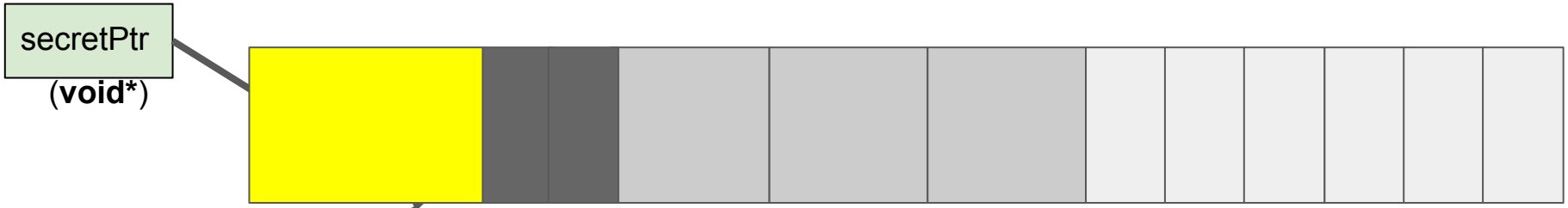
rd”

es

numEntries x 2 bytes


secretPtr
(**void***)
An unsigned integer **“numSecrets”**
4 bytes


```
*(unsigned int *)secretPtr
```

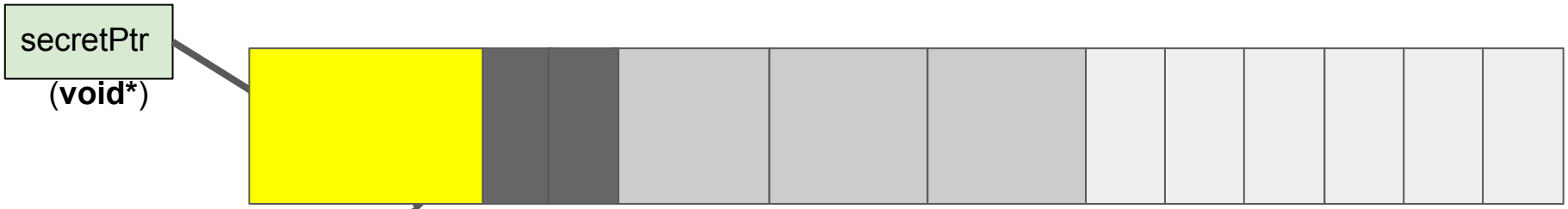


An unsigned integer **“numSecrets”**

4 bytes

```
*(unsigned int *)secretPtr
```

“cast” the pointer: “interpret this as as an **`unsigned int`**



An unsigned integer **“numSecrets”**

4 bytes

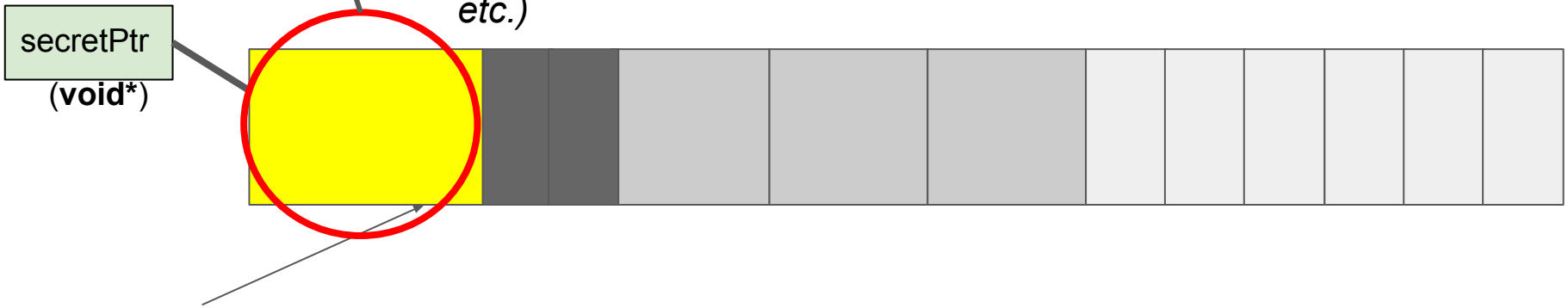
“dereference”: get the data stored at memory address

```
*(unsigned int *)secretPtr
```

because of casting: “get exactly `sizeof(unsigned int)` (4) bytes, and interpret their contents as one unsigned integer.”

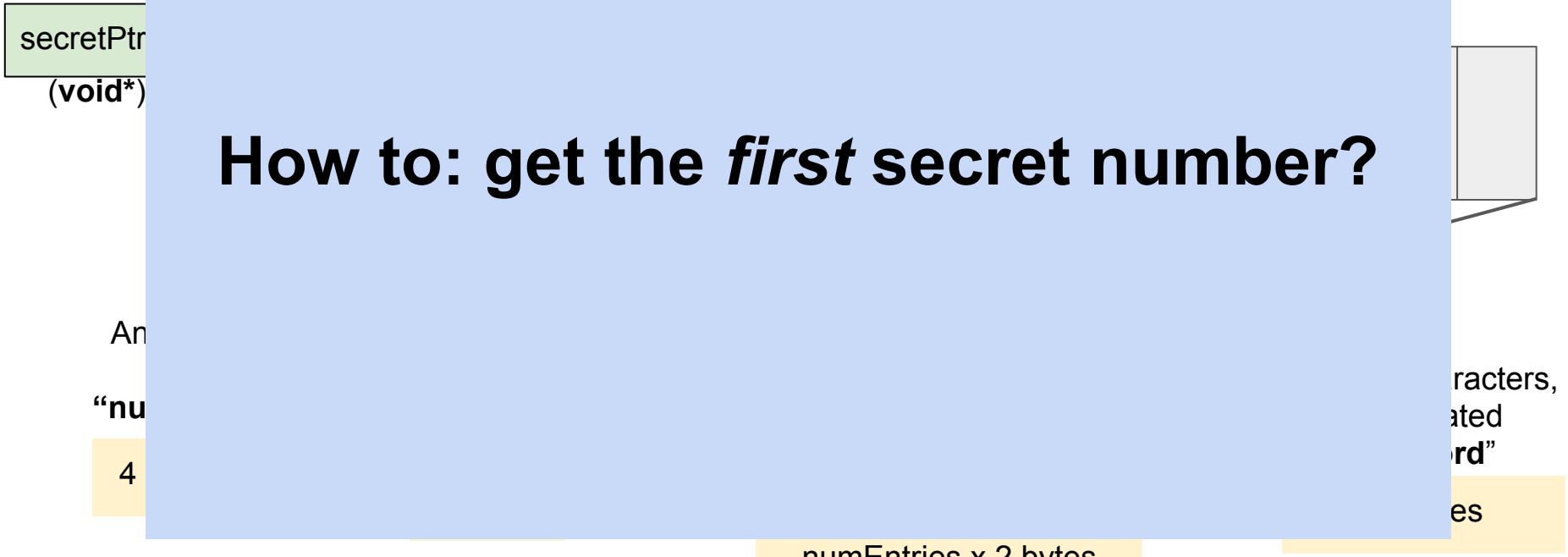
All together - end up with the unsigned integer stored here.

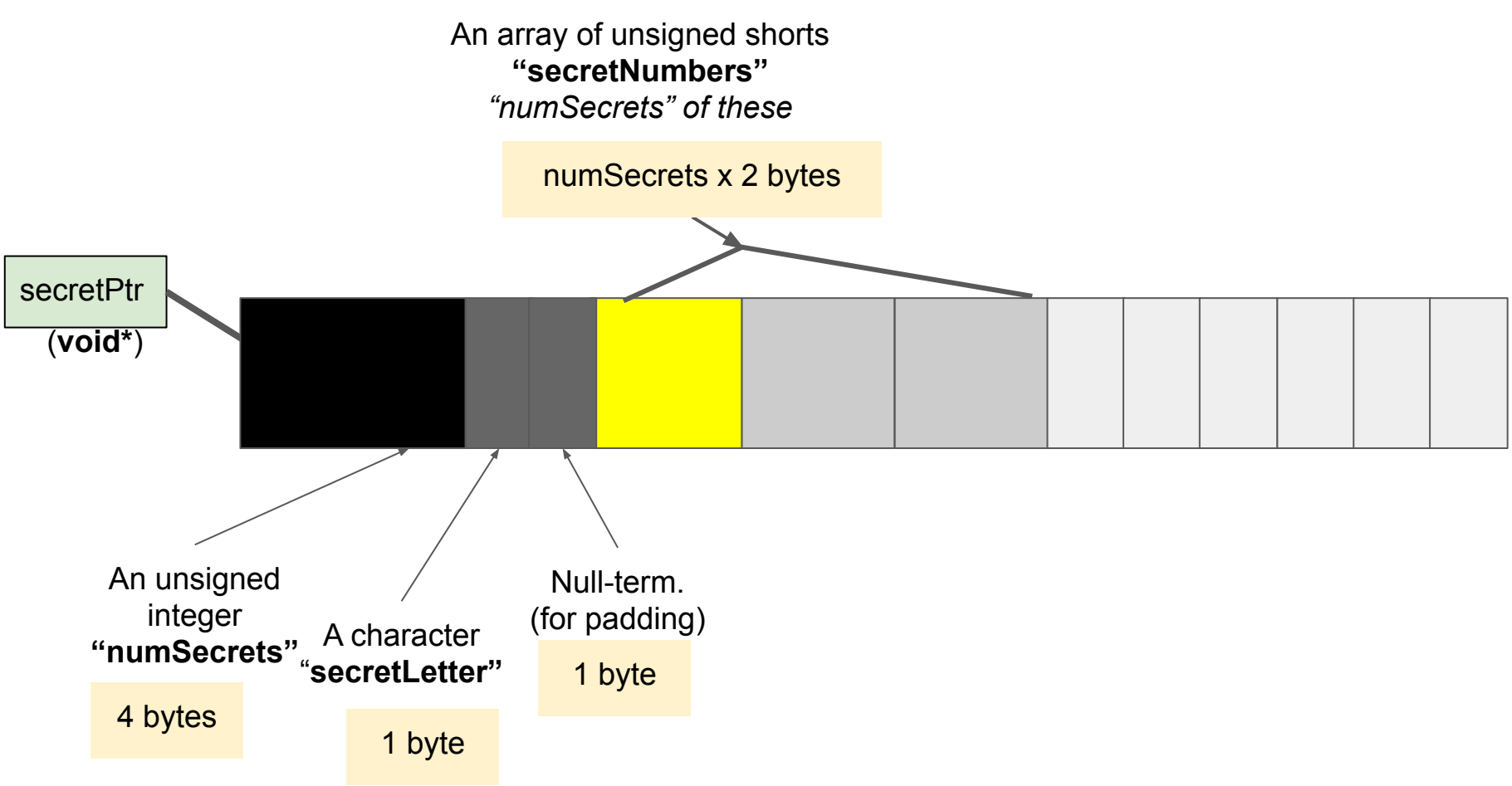
(Can then store a copy in a new variable, use it for calculations, etc.)

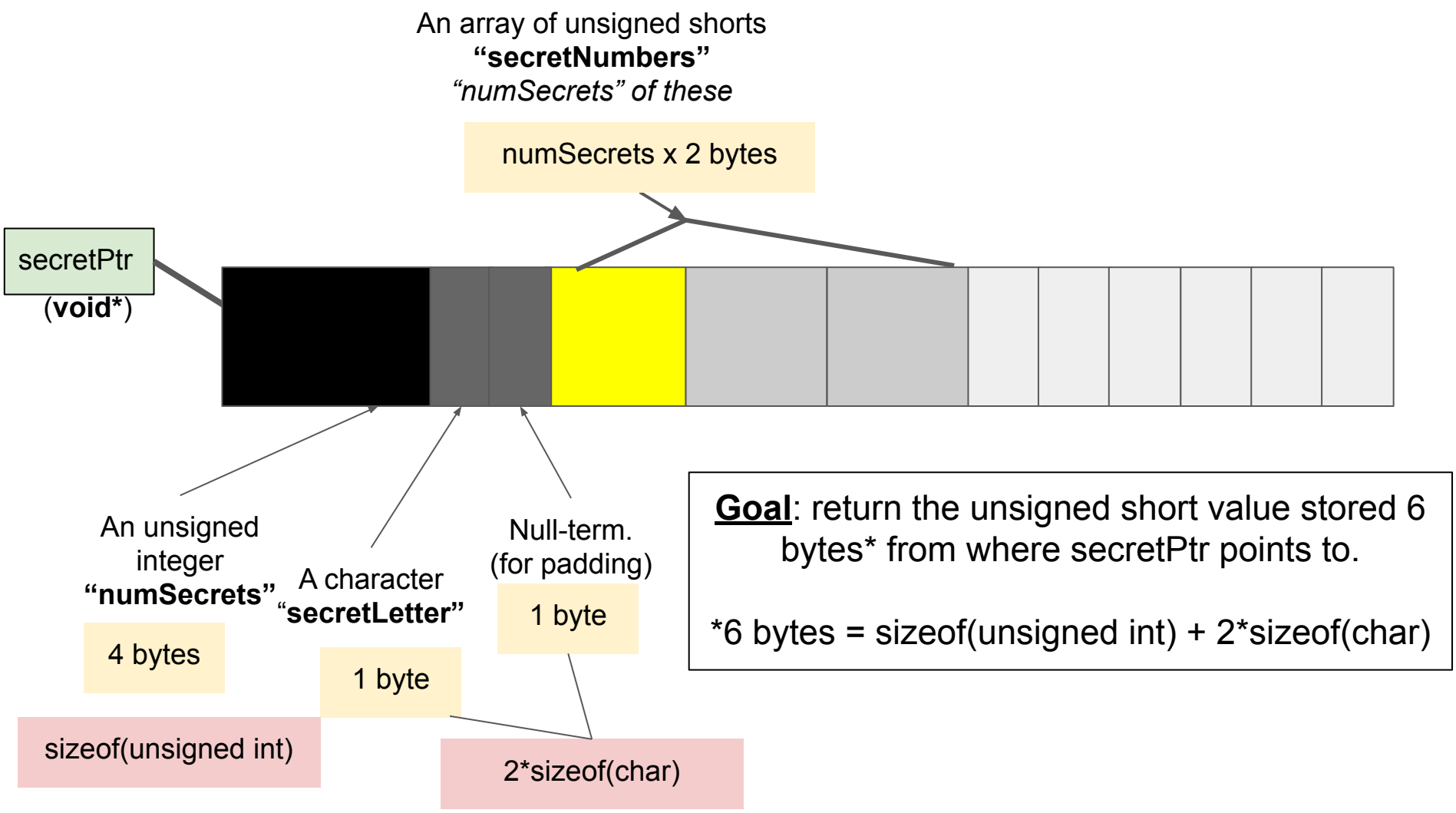


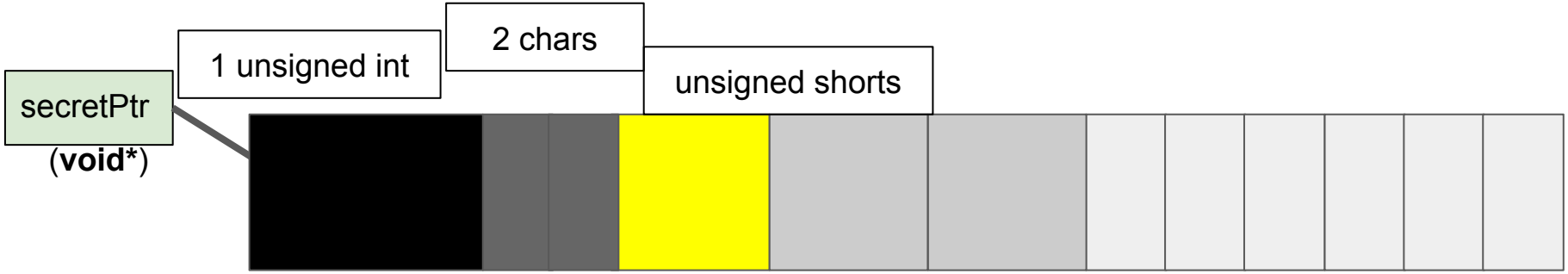

```
*(unsigned int *)secretPtr
```

How to: get the *first* secret number?

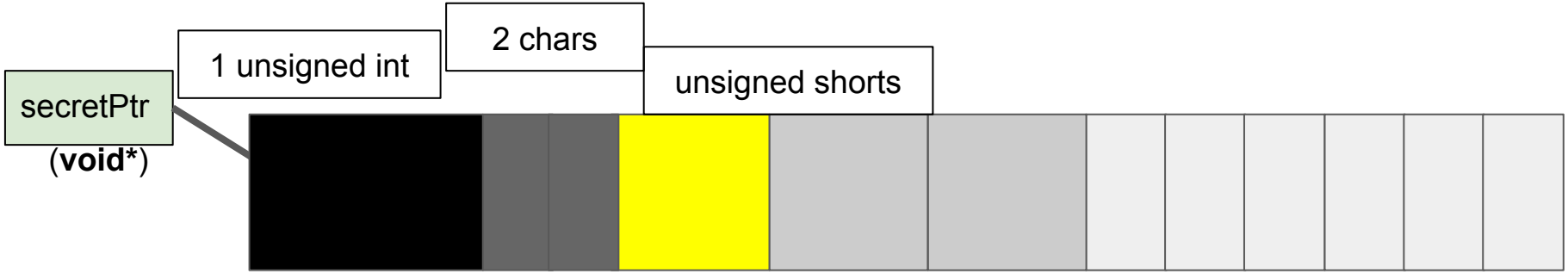



An array of unsigned shorts
“secretNumbers”
“numSecrets” of these
numSecrets x 2 bytes
secretPtr
(**void***)
An unsigned integer **“numSecrets”**
4 bytes
A character **“secretLetter”**
1 byte
Null-term. (for padding)
1 byte


An array of unsigned shorts
“secretNumbers”
“numSecrets” of these
numSecrets x 2 bytes
secretPtr
(**void***)
An unsigned integer **“numSecrets”**
4 bytes
sizeof(unsigned int)
A character **“secretLetter”**
1 byte
Null-term. (for padding)
1 byte
2*sizeof(char)
Goal: return the unsigned short value stored 6 bytes* from where secretPtr points to.
*6 bytes = sizeof(unsigned int) + 2*sizeof(char)



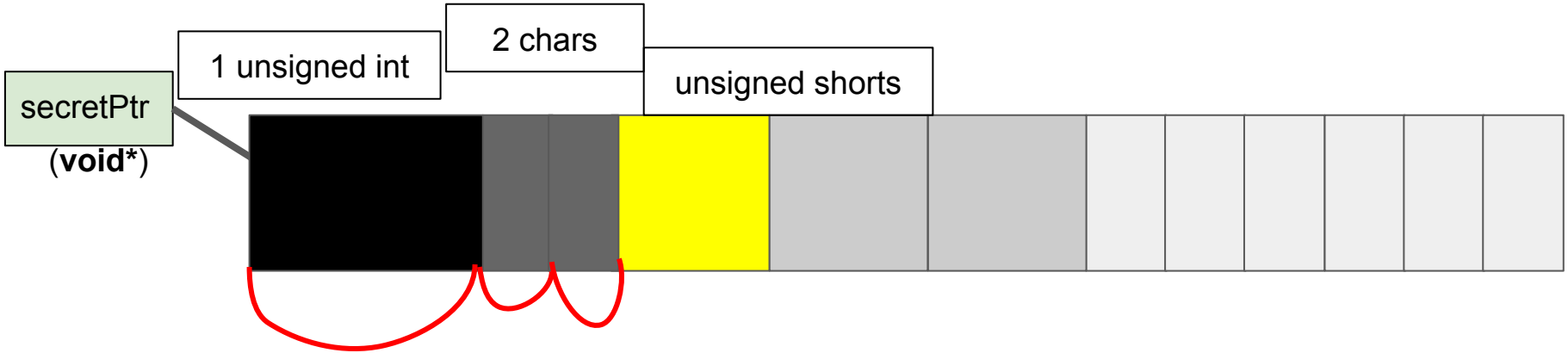

```
unsigned short *secretNumPtr =
   (unsigned short *)((char *)secretPtr +
         sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```



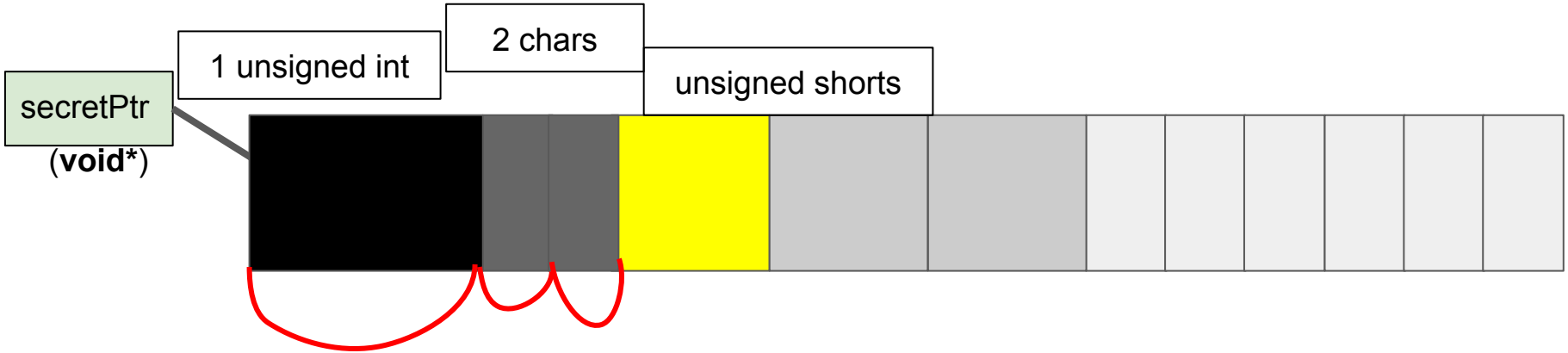

```
unsigned short *secretNumPtr =
   (unsigned short *)((char *)secretPtr +
         sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```

"cast" the pointer: "interpret this as as a **`char *`**
and scale all pointer arithmetic by **`sizeof(char)`** -- aka, one byte"

Key point: pointer arithmetic scales by the type being pointed to.
(example: **`(short *)ptr + 1`** → a pointer to a short, two bytes away from ptr)

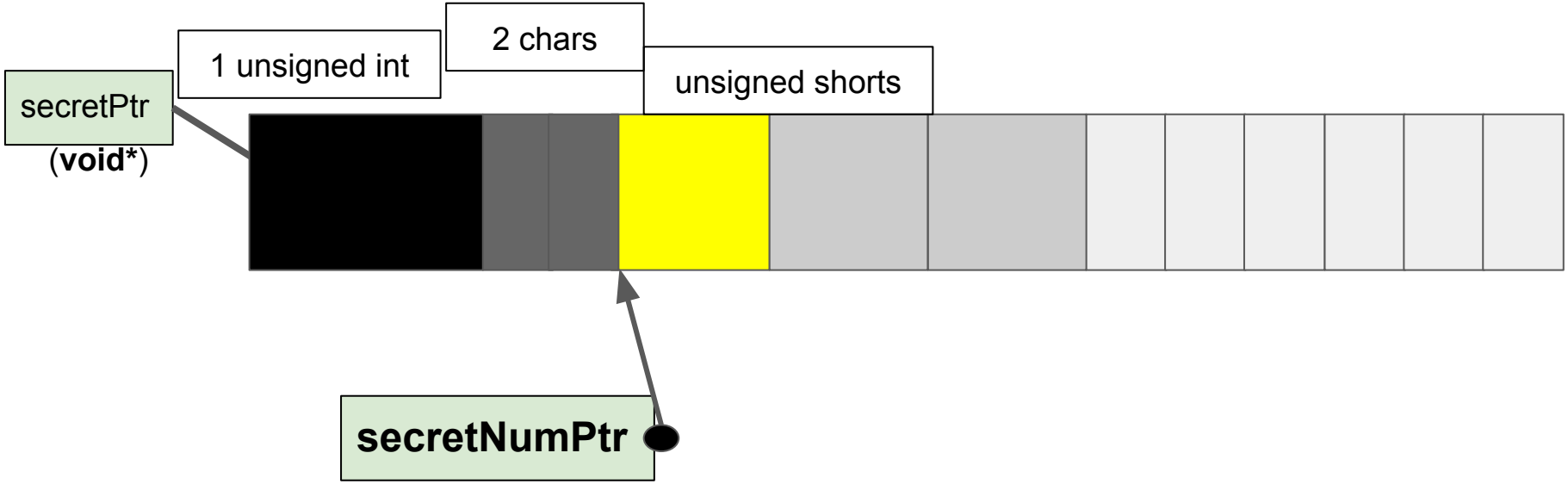



```
unsigned short *secretNumPtr =
    (unsigned short *)((char *)secretPtr +
          sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```

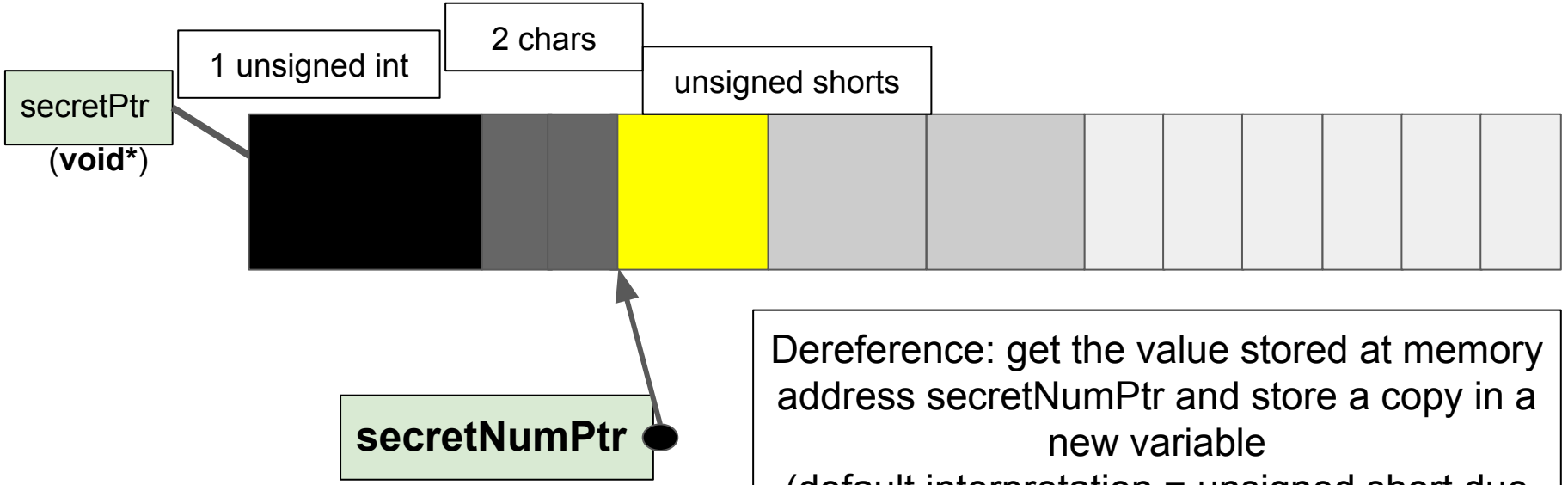


Cast result to **`unsigned short *`**

```
unsigned short *secretNumPtr =
    (unsigned short *)((char *)secretPtr +
          sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```




```
unsigned short *secretNumPtr =
    (unsigned short *)((char *)secretPtr +
            sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```



Dereference: get the value stored at memory address secretNumPtr and store a copy in a new variable
(default interpretation = unsigned short due to type of secretNumPtr)

```
unsigned short *secretNumPtr =
   (unsigned short *)((char *)secretPtr +
         sizeof(unsigned int) + 2*sizeof(char));
unsigned short secretNum = *secretNumPtr;
```

secretPtr

(**void***)

How to: get the *last* secret number?

An

“nu

4

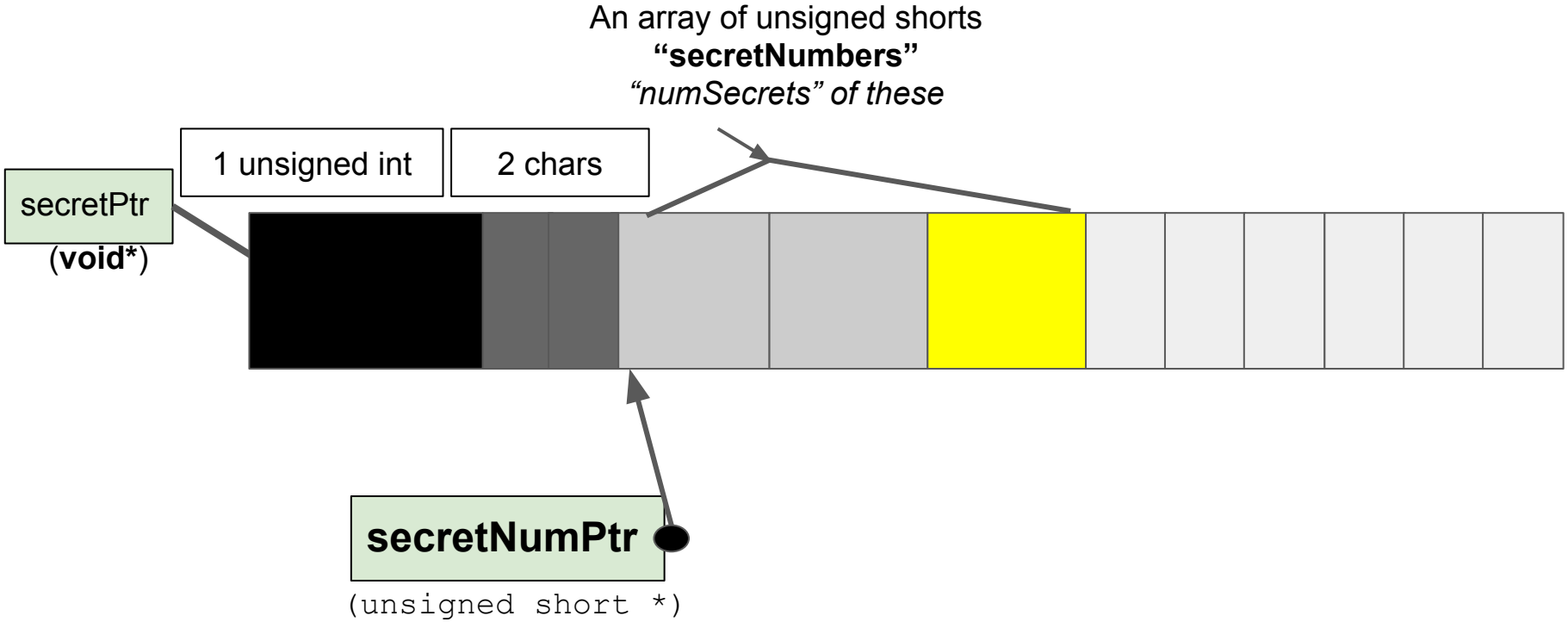
racters,

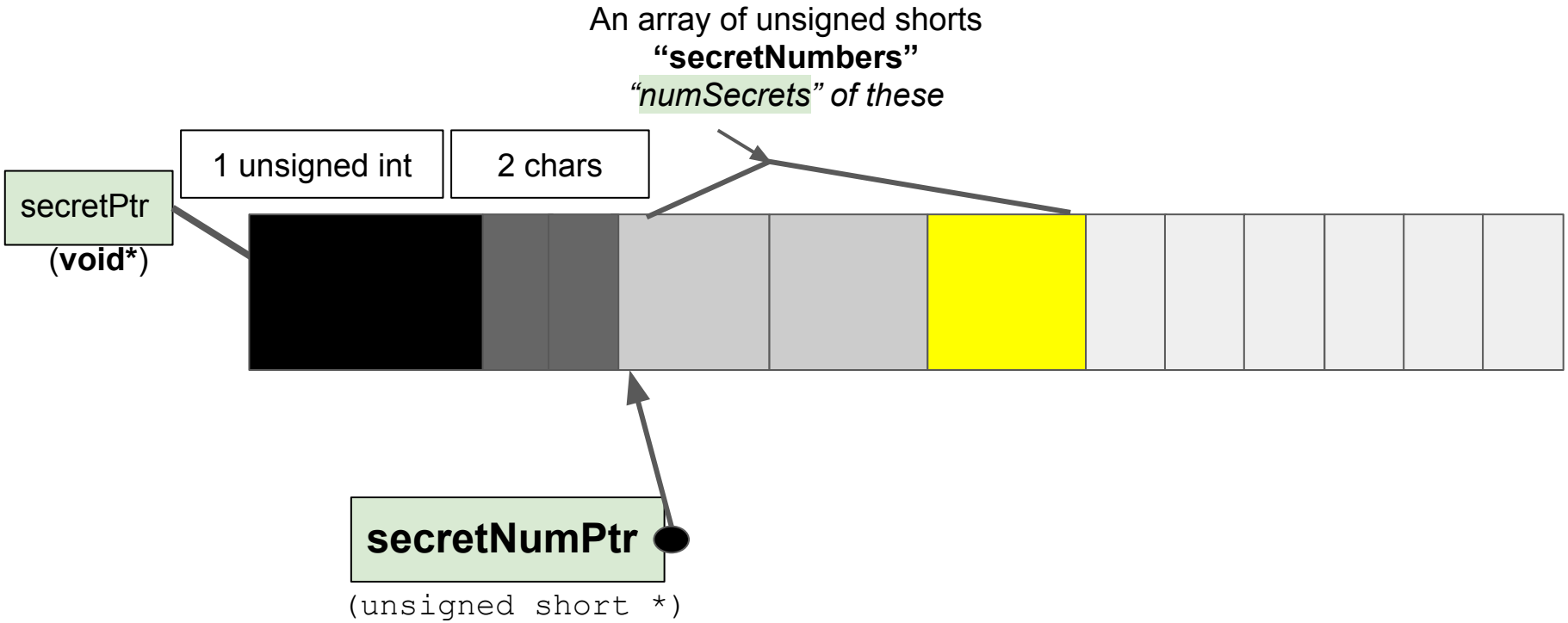
ated

rd”

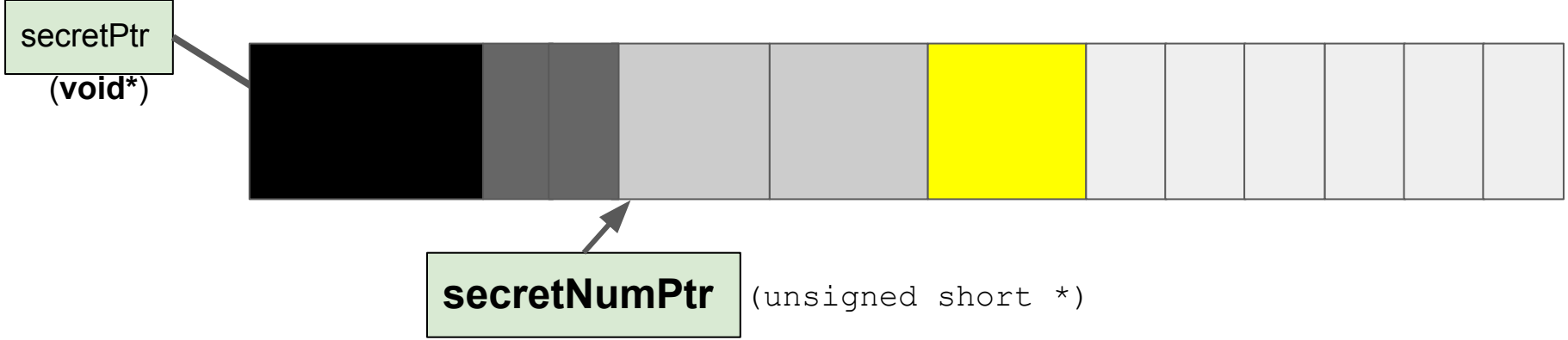
es

numEntries x 2 bytes


An array of unsigned shorts
“secretNumbers”
“numSecrets” of these
1 unsigned int
2 chars
secretPtr
(void*)
secretNumPtr
(unsigned short *)




```
// use secretNumPtr and numSecrets to get last secret number
```




```
unsigned short lastSecretNum =
```

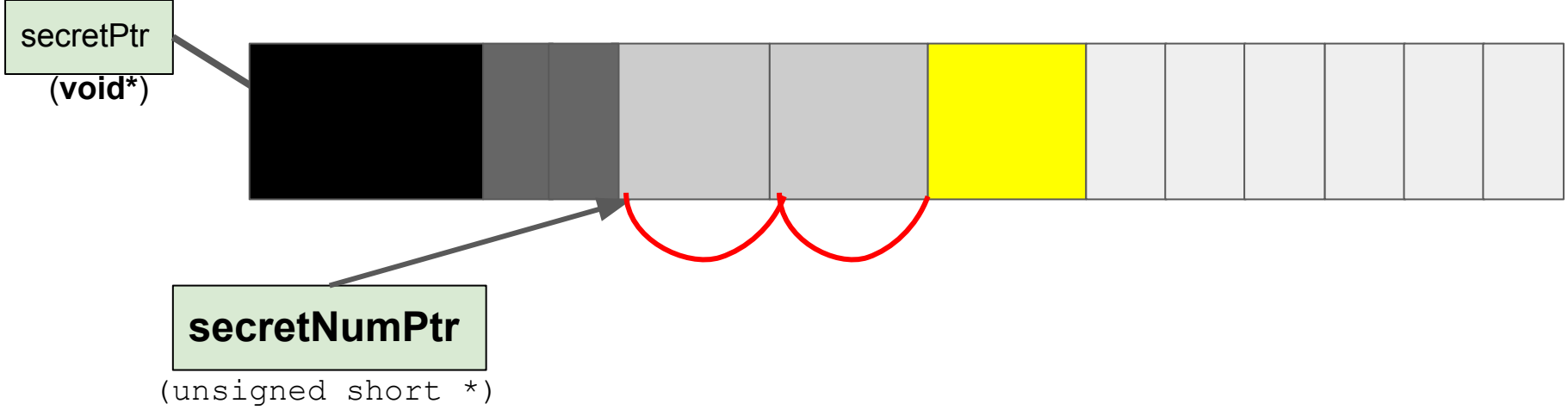
```
*(secretNumPtr + numSecrets - 1);
```

pointer arithmetic approach

OR

```
secretNumPtr[numSecrets - 1];
```

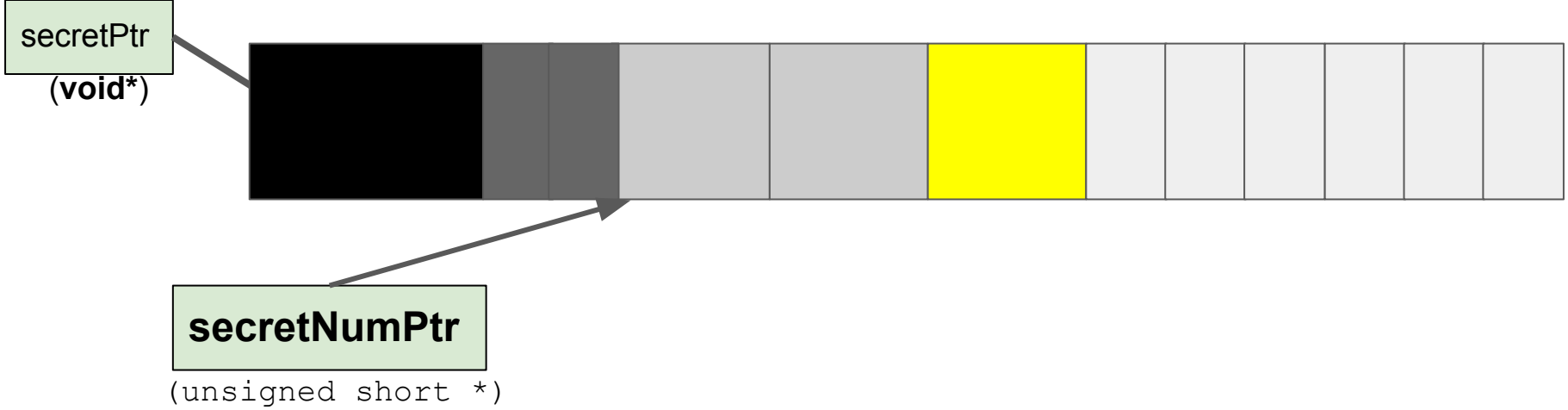
array notation approach



Pointer arithmetic scales by the type being pointed to.
`secretNumPtr` points to type **`unsigned short`**
All arithmetic is in units of **`sizeof(unsigned short)`**

```
*(secretNumPtr + numSecrets - 1);
```

pointer arithmetic approach



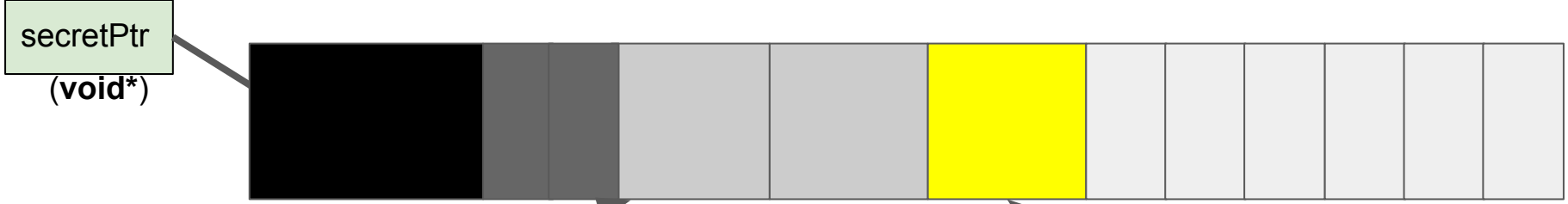
Treat secretNumPtr as a pointer to the first element of an array of unsigned shorts

Array indexing also dereferences

(And indexing is by size of type being pointed to.)

```
secretNumPtr[numSecrets - 1];
```

array notation approach



secretNumPtr `(unsigned short *)`

Both of these get the value stored here; we can then copy it into a new variable, use for calculations, etc.

```
unsigned short lastSecretNum =
```

```
*(secretNumPtr + numSecrets - 1);
```

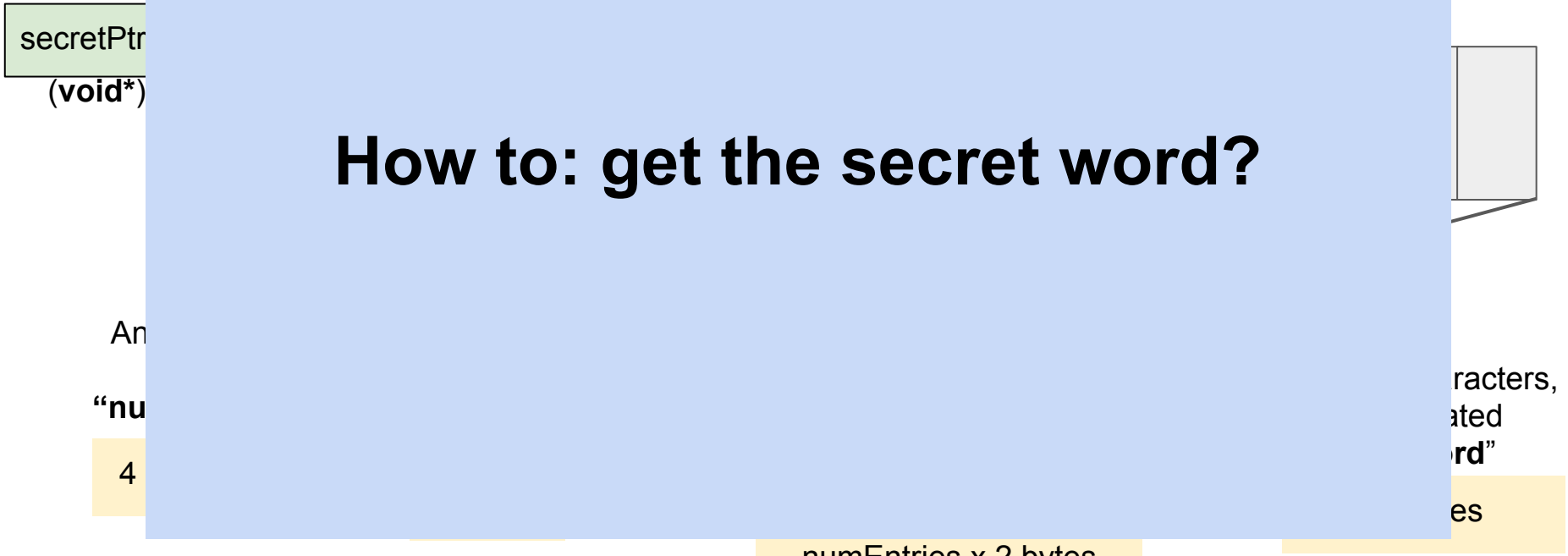
pointer arithmetic approach

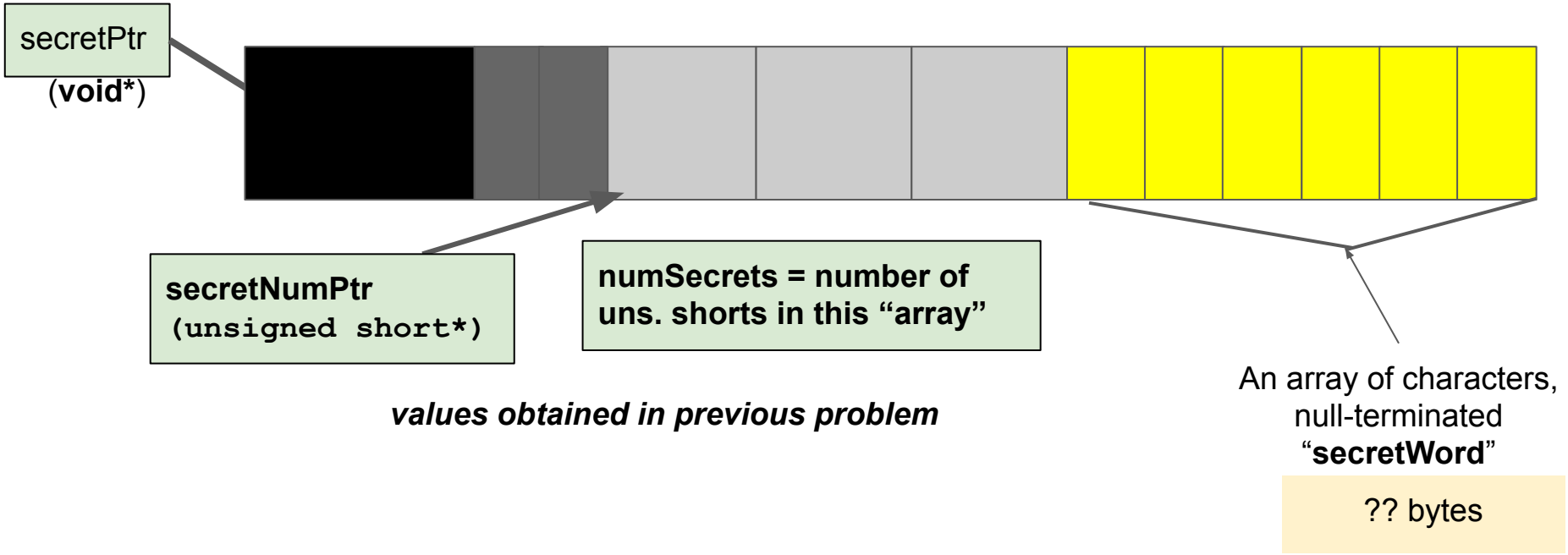
OR

```
secretNumPtr[numSecrets - 1];
```

array notation approach

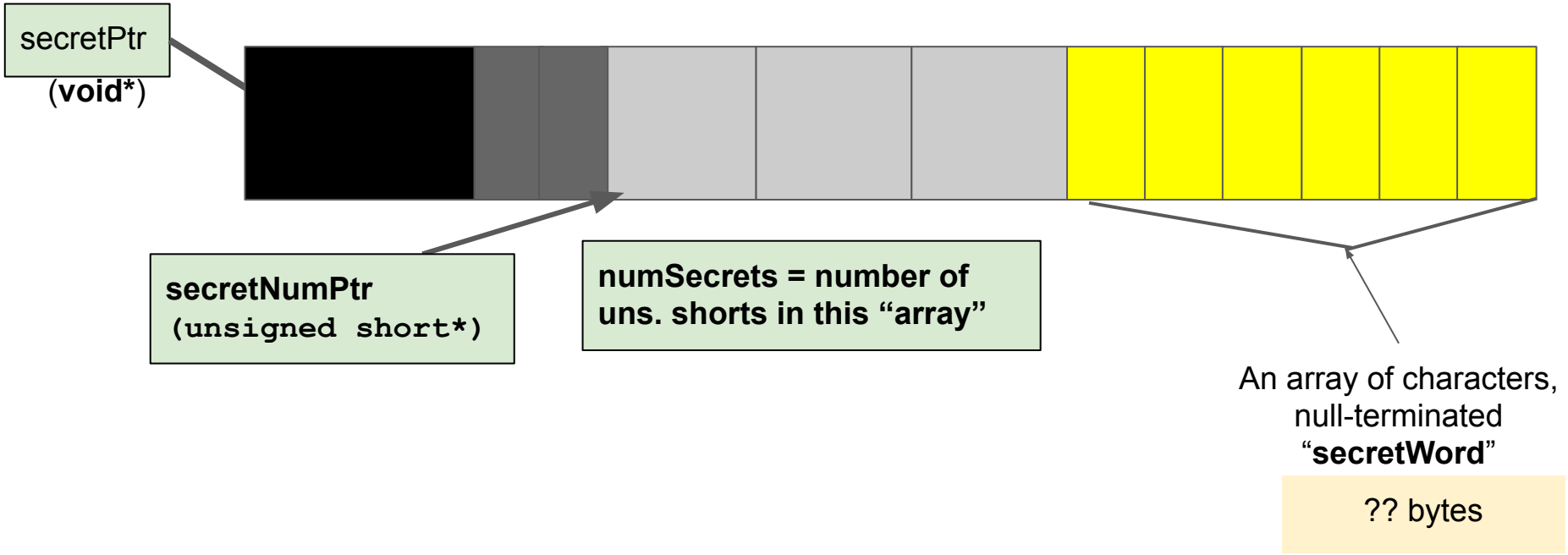
How to: get the secret word?

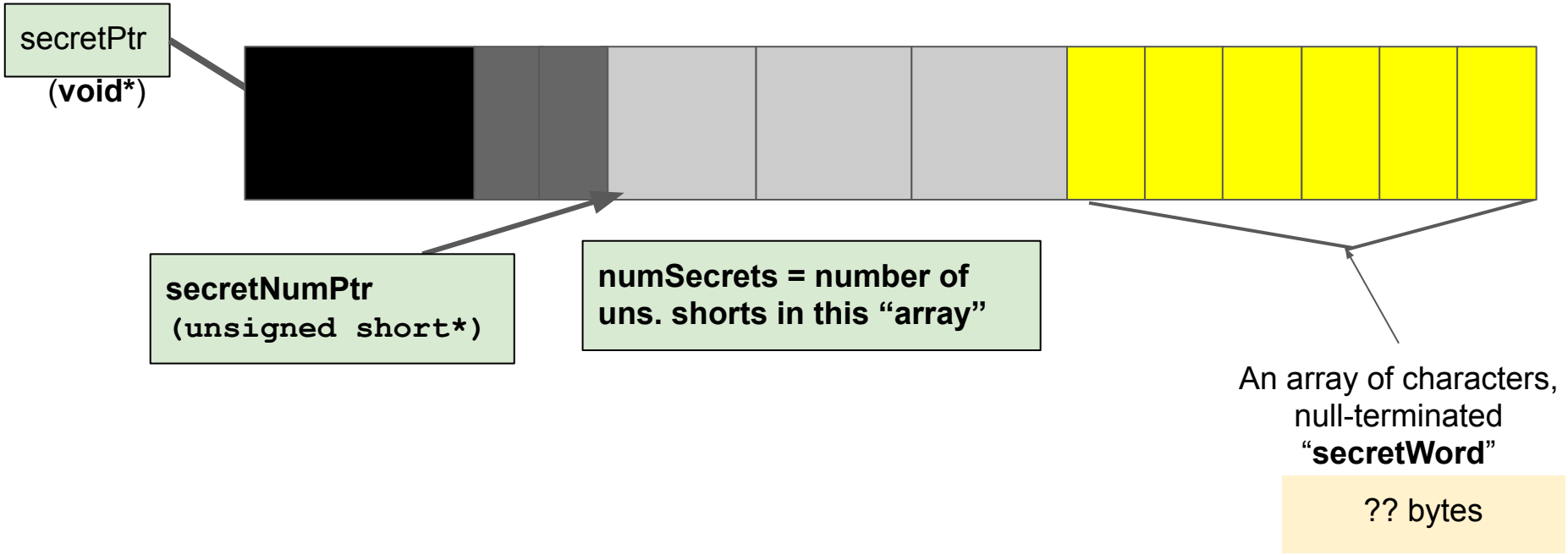




values obtained in previous problem

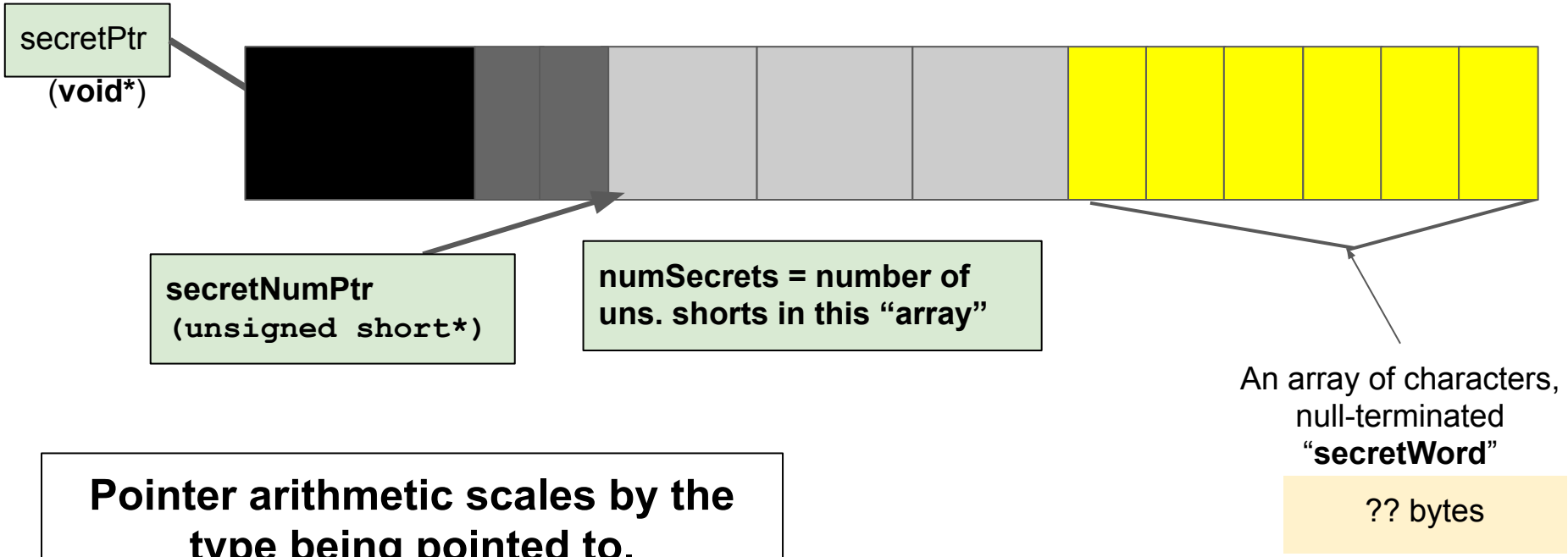
Ideas?


secretPtr
(void*)
secretNumPtr
(unsigned short*)
numSecrets = number of uns. shorts in this “array”
An array of characters, null-terminated “secretWord”
?? bytes



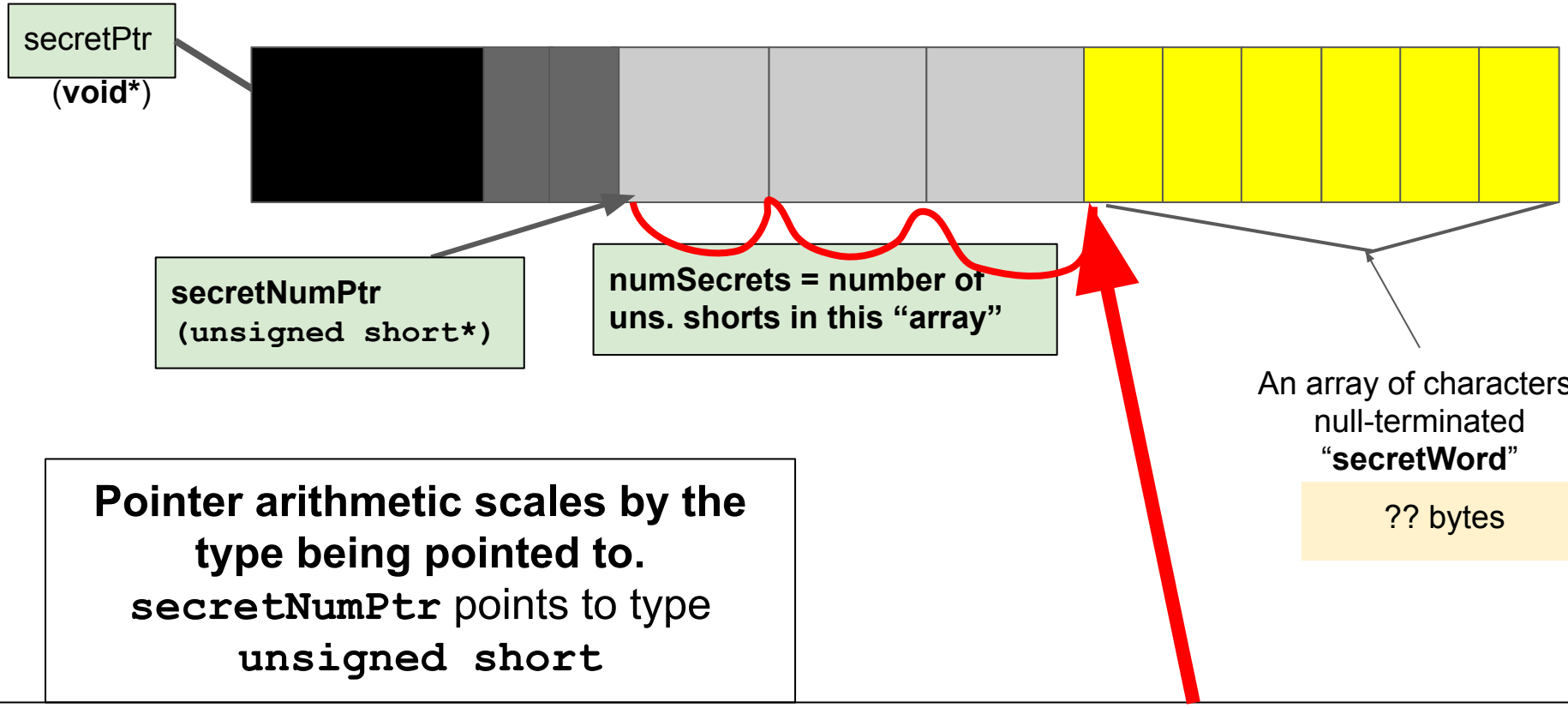
possible approach:

```
string secretWord = (char *)(secretNumPtr + numSecrets)
```



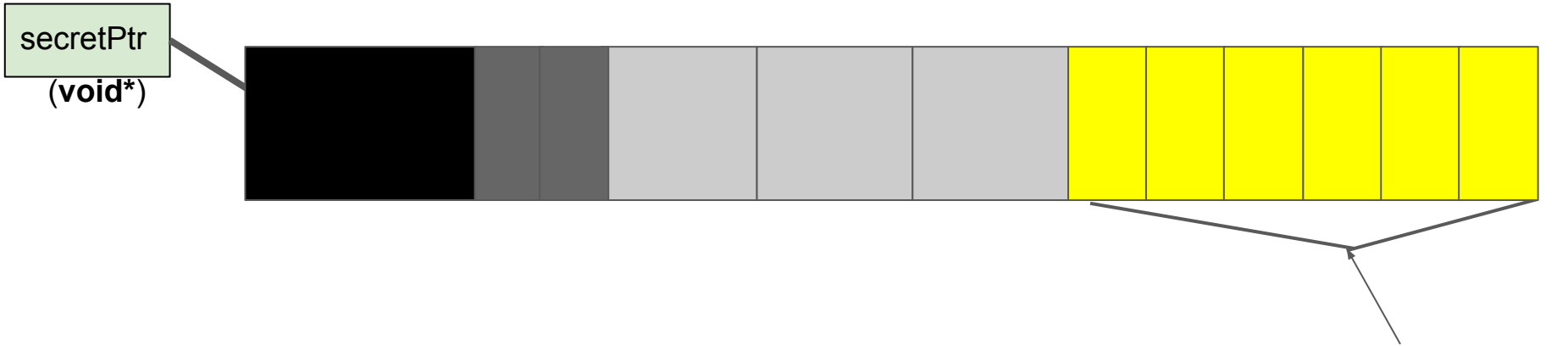
Pointer arithmetic scales by the type being pointed to.
`secretNumPtr` points to type `unsigned short`

```
string secretWord = (char *)(secretNumPtr + numSecrets)
```



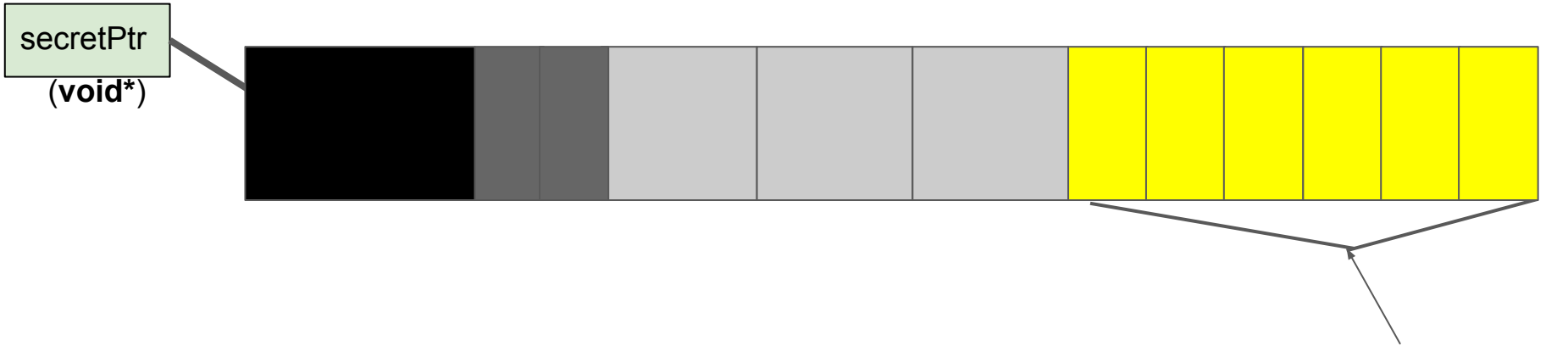
Pointer arithmetic scales by the type being pointed to.
`secretNumPtr` points to type `unsigned short`

```
string secretWord = (char *)(secretNumPtr + numSecrets)
```



Cast to **`char *`**
(otherwise, this would be interpreted as **`unsigned short *`**)

```
string secretWord = (char *)(secretNumPtr + numSecrets)
```



Storing a C-string in a C++ string variable -- automatically converts

```
string secretWord = (char *)(secretNumPtr + numSecrets)
```

Let’s try these out in gdb!