How to determine change between two arrays

Recently, I was building a historical data tracker, which would track any changes in settings across a large application. All of the settings would be pulled from their MySQL tables and stored in a PHP array. Storing those settings (just short of 25,000 of them) would have been unfeasible on the MySQL database. At the time, we didn't have a Mongo instance running either.

So I serialized each array and stored it as its own record. Every minute, a cron would kick off the script, which would pull the latest record, deserialize it, and check each value recursively to determine if the value changed. If the value did change, then it would store the new array as a new record. It would then kick off an email saying a setting was changed.

The interesting bit is how the arrays were checked against each other, recursively:

```
function get_array_changes($array_1, $array_2, $differences = [], $old_ke
y = '') {      if($old_key != ''){ $old_key .= ' > '; }           foreac
h($array_1 as $key => $value){          if(is_array($value)){
  return get_array_changes($value, $array_2[$key], $differences, $old_key
 . $key);          } else {              if(strtolower(trim($value)) != s
trtolower(trim($array_2[$key]))){                  $differences[] = [
                'key' => $old_key . $key,                     'old_val
ue' => $value,                     'new_value' => $array_2[$key]
         ];             }         }      }      return $differences;
}
```

Calling it would be a simple ordeal:

```
$array_1 = ['a' => ['c' => ['d' => 1]], 'b' => 2];  $array_2 = ['a' => ['
c' => ['d' => 3]], 'b' => 2];  $x = get_array_changes($array_1, $array_2)
;  print_r($x);
```

The result would then be whatever was changed, including a breadcrumb key:

```
[      [0] =>           [              'key' => 'a > c > d'
'old_value' => 1              'new_value' => 3          ]  ]
```