Issue #2:

Evaluating Untrusted Scripts

John Ousterhout

Computer Science Division
Department of EECS

University of California at Berkeley

Introduction

Goal: use Tcl scripts as a general-purpose method of interchange:

- Among applications on a display.
- Active e-mail messages (e.g. surveys).

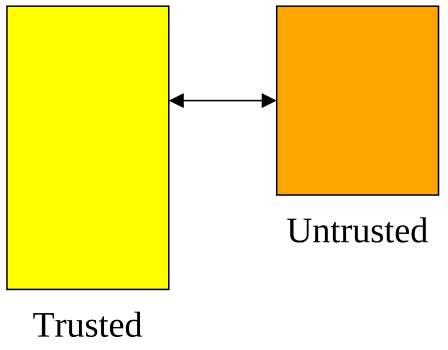
Security problems:

- Tcl is powerful (can access files, etc.).
- Evil scripts can potentially do great harm.

Solution:

- Twin interpreters (like user-space/system-space).
- Protected calls between them (like system calls).

Twin Interpreters



- Trusted interpreter: used by receiving application/user: has access to all Tcl commands.
- Untrusted interpreter: used for incoming (suspicious) scripts: all dangerous commands removed.
- New commands in trusted interpreter:

```
set evil [safetcl create]
$evil eval $script
```

- Untrusted interpreter won’t be able to do much that’s useful, though.

Safe Calls

Allow trusted interpreter to implement restricted new functions for untrusted interpreter:

- Restricted file access, sending mail, ...
- Analogous to system calls.

Mechanism: command in untrusted intepreter that causes execution of command in trusted interpreter:

- In trusted interpreter:

```
set evil [safetcl create]
$evil safecall sendmsg checksend
```

- In untrusted interpreter:

```
sendmsg $to $body
```

- Substitutions occur in untrusted interpreter.
- `Checksend` executed in trusted interpreter with fully-substituted arguments.
- Result/error returned to untrusted interpreter.

Safe Calls, cont’d

Procedures that implement safe calls must be very careful:

- Never evaluate argument as Tcl script or Tcl expression.
- Check file names before reading or writing files.
- Never execute shell commands specified in arguments without careful checks first.
- When in doubt, ask user for permission.

Result: safe calls hard to write and certify.

But, for maximum power want lots of safe calls.

Need mechanism for certifying and distributing safe calls.

Certifying Safe Calls

Use encryption techniques (digital signatures):

- Central, trusted, network authority writes new safe calls, certifies them with digital signature, distributes publically.
- Anyone can fetch certified safe calls, check signature, install locally without fear.
- Active e-mail message (untrusted) can contain new safe calls as part of the untrusted script.
- Untrusted script invokes existing safe call to make new safe call.
- In trusted interpreter, verify signature of incoming safe call before installing.

Can extend mechanism to have local certification authorities as well as global.

Other Applications

Safe call mechanism suitable for many other things besides active e-mail messages:

- Restrict incoming **send** commands in Tk.
- In commercial product, restrict access by customers to internal commands.
- In device control applications, don't allow users total control over devices (could be dangerous for some devices).