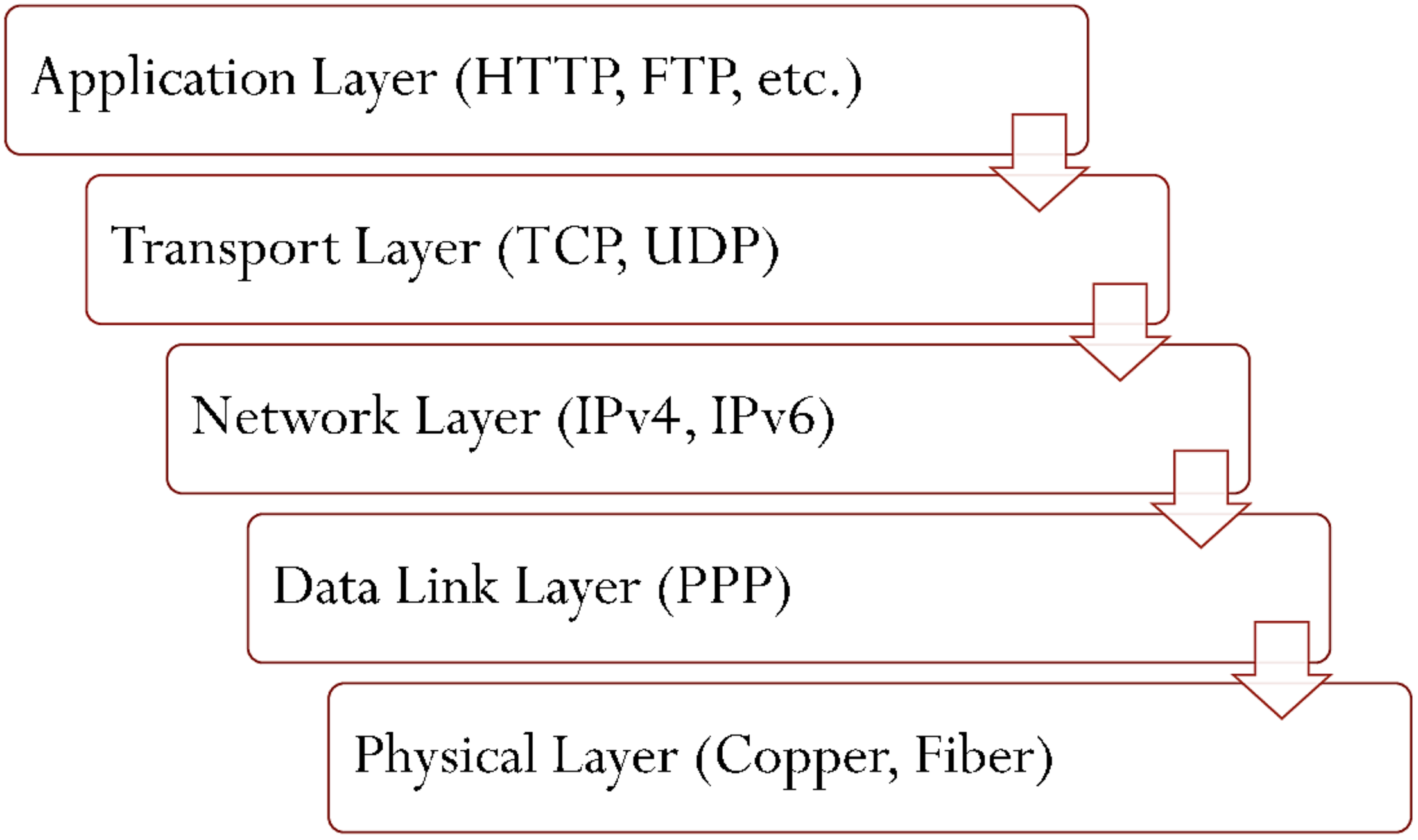
# KEYSCORE, Cipher Detection, and You!

[redacted]

Mathematics Research Group

21 August 2008

# The Protocol Stack



# Data Sources

- FORNSAT (downlink)
- Overhead (uplink)
- Special Source
- Tailored Access
- F6
- FISA (limited)
- 3rd party

# Front-end Processing

Demodulate

Demultiplex

WEALTHYCLUSTER2/
TURMOIL

Packetize

Sessionize

Decrypt?

# What does  do ?


- Selection of tasked CADENCE/UTT terms.
  - Send hits to PINWALE/PRESSUREWAVE.
- Tipping to TRAFFICTHIEF.
- Fingerprinting.
- SIGINT development using two rolling buffers:
  - Metadata
  - Content (data)

# Retrospective Searching


- All data are stored, not just hits.
- Queries are distributed to entire network of sites.

| Metadata Buffer | Content Buffer |
|---|---|
| ~30 days | ~7 days |
| Searchable | Retrievable |
| MySQL database | Archived on disk |

# Fingerprinting

- Pattern matching against the data.
- Session is marked, but not sent to PINWALE.
  - Fingerprint stored as metadata.
  - Have to search for it.
- Rich set of patterns
  - Strings have a minimum of three* anchors (fixed bytes).

    [Exception: Two bytes at the beginning of a session]
  - Regular expressions allowed (require nonoptional string of three* bytes within regex)
  - Context-dependent terms.

*XKS reserves the right to increase this to four.

# Examples

- `fingerprint('encryption/helixstronghold', 7.0) = 'helix stronghold encrypted file';`

- `fingerprint('encryption/wharfrat', 3.0) = '\xd6\x56\x34\xb7\x80\x05\xfe\x8b'c and '\xaf\x52\x72\x60\xdd\xfe\x72\xc2'c and (port(443) or port(80));`

- `fingerprint('encryption/the_algorithm', 3.0)=`
  `/-XYZ-.{0,30}mp[eg]/;`

# Syntax Features

- Case Sensitivity

```
fingerprint('certificate/digital_id') =
'-BEGIN CERTIFICATE-'c;
```

- Full Boolean logic
  - Grouping with parentheses
  - Operators: **and**, **or**, **not**
- Variables

```
$udp = protocol('udp');
fingerprint('vpn/openvpn/x509/wera') =
$udp and 'openvpn_wera'c;
```

# Available Functions

- **port**
- **first**
- **hex**

```
fingerprint('encryption/kryptel') =
hex('E8E2454300040004635C4EE9A2F9D111A
489E498F70C0B43404F4BFA50F2D111A4898E6
30458E285');
```

- **pos**

```
fingerprint('encryption/cipherpad') =
pos('CPAD1'c) < 4000;
```

- Distance (similar to pos, but for distance between tokens)
- Lpos

  Spop_basic = lpos('+OK 'c) or '\nQUIT';
- First

  appid('mail/smtp/...) = first('ehlo') and ... ;
- Last (similar to first)
- Follows (one token after another)
- Between (one token between two others)
- Order

# Other Features

- Fingerprint definitions updated hourly throughout the entire enterprise.
- Workflows
  - Submit through user interface.
  - Standing queries that run like cron jobs.
  - Limited follow-on processing.
- User interface for fingerprint submission (coming soon).
  - Currently done by XKS personnel.

# Plug-ins

- Full power of C++ for when pattern matching does not suffice.
- Usually limited to certain file types
  - Huge JPEG volume from web surfing
- Current steg/encryption plugins that fingerprint sessions:
  - PHOSPHORESSENCE library of steg detectors
  - SHELLLOCK steg detection
  - SEDENA indigenous encryption software
- Drawback: Must wait for site upgrade to deploy.

# Trade-off

- Fingerprints easily deployed, but limited to pattern matching.
- Plug-ins slow to deploy, but allow for complex testing.
- New compromise:
  - Snippets of C++ code in fingerprint
  - Deployed hourly like fingerprint with most of the flexibility of a full plug-in.
  - Very complicated tests probably still need to be plug-ins.
  - Currently stood up at only a few sites.

# Example

```
fingerprint('encryption/archive/rar') =
        '\x52\x61\x72\x21\x1a\x07\x00'c
        : c++ {{
          const uint8_t *ptr =
            find_first("\x52\x61\x72\x21\x1a\x07\x00");
          if (ptr == NULL)
            return false;
          if (end()-ptr < 64)
            return false;
          if ((ptr[23]&0x04) != 0x04)
            return false;
          if ((ptr[10]&0x80) != 0x80)
            return false;
          return true;
        }};
```

# Advanced Feature

- Follow-on check with anchorless regexes:

```
%dhcp_check = regex {{
            ^[\x01\x02][\x01- ]\x06.*c\x82sc
          }};

appid('netmanagement/dhcp/client_to_server',
  3.0) =
          from_port(68) and to_port(67)
          : %dhcp_check;
```

# Releasability Issues

- Nearly all XKS personnel have PICARESQUE!
  - Those that don't have PRIVAC.
- XKS distribution comes in two flavors
  - 1st & 2nd party
  - 3rd party
  - No NOFORN capabilities permitted.
  - Special dispensation from Fred Mann for some capabilities to SMOKYSINK.
- Can keep PICARESQUE code running on R1's rednet if absolutely necessary.