[redacted]

# GRIMPLATE

## First Steps Toward Identifying Adversarial Use of BitTorrent

[redacted]
NSA/R4

Derived From: NSA/CSSM 1-52
Dated: 20070108
Declassify On: 20370117

**The overall briefing is classified TOP SECRET//COMINT//REL FVEY**

# Agenda

- Motivation
- BitTorrent's TCP and UDP layers
- DHT overview
- What does it mean to crawl DHT?
- Pilot implementation
- Collaboration

# GRIMPLATE Motivation

- BitTorrent sessions are seen on a daily basis between NIPRnet hosts and adversary space (PRC, RU, etc.)
- NTOC has no way of knowing if this is innocuous file sharing or malicious activity.
- Peer-to-Peer (P2P) is not allowed on NIPRnet, but most commands do not see it as harmful.
- If we can glean some indication of the type of data that's leaving NIPRnet, we can build a case for shutting this activity down.
- Interest is not limited to NIPRnet scenario

# BitTorrent's TCP and UDP Layers

- TCP
  - Used to exchange pieces of files amongst peers
- UDP
  - Used to exchange routing messages
    - Who should I ask for file pieces?

# BitTorrent DHT

- Nodes: clients participating in DHT
- Peers: clients participating in piece exchange to share file
- DHT: distributed key, value store
- Nodes have 160 bit pseudo-random node ID
- **Keys** are 160 bit hash of .torrent file metadata - info_hash
- **Values** are list of IP addresses and ports of peers mapped to info_hash

# Mainline DHT Messages

ping Query = {"t":"aa", "y":"q", "q":"ping", "a":{"id":"abcdefghij0123456789"}}

ping Response = {"t":"aa", "y":"r", "r":" {"id":"mnopqrstuvwxyz123456"}}

find_node Query = {"t":"aa", "y":"q", "q":"find_node",

"a":{"id":"abcdefghij0123456789", "target":"mnopqrstuvwxyz123456"}}

find_node Response = {"t":"aa", "y":"r", "r": {"id":"0123456789abcdefghij", "nodes":"def456…"}}

get_peers Query = {"t":"aa", "y":"q", "q":"get_peers",

"a":{"id":"abcdefghij0123456789", "info_hash":"mnopqrstuvwxyz123456"}}

get_peers Response, with peers = {"t":"aa", "y":"r", "r": {"id":"0123456789abcdefghij",
"token":"aoeusnth", "values": ["axje.u", "idhtnm"]}}

get_peers Response, with closest nodes = {"t":"aa", "y":"r", "r": {"id":"0123456789abcdefghij",
"token":"aoeusnth", "nodes":"def456…"}}

Announce peer = {"t":"aa", "y":"q", "q":"announce_peer",

"a":{"id":"abcdefghij0123456789", "info_hash":"mnopqrstuvwxyz123456", "port" : 6881,
"token" : "aoeusnth"}}

Response = {"t":"aa", "y":"r", "r": {"id":"0123456789abcdefghij"}}

# What's it mean to crawl DHT?

- Goal: Harvest complete node list for entire DHT and peer list for info_hashes found in NIPRNET defensive tools or SIGINT

- Regular client node lookup is iterative process
  - O (log n) search
  - routing table is starting point

- Approach:
  - spray find_node messages across DHT and store responses
  - query for peers of info_hashes of interest

# What does DHT crawler collect?

- For each node in the DHT:
  - 160 bit node ID
  - IP address
  - Port
- For targeted info_hashes:
  - List of the node ID, IP address, and port of nodes sharing targeted file
  - Entries may be stale

# What value is the data?

- Use “community detection” algorithms to identify swarms that are likely to be malicious
- Download files being shared by likely malicious swarms
- Build BitTorrent mitigation case for NIPRnet
- General SIGINT reporting

- File download without identification of likely malicious swarms impractical

# Pilot on PACKAGEGOODS Server

- Deploy modification of existing crawler – dedicated PG server
- Run analytics on “swarm” metadata to determine malicious activity
- Experiment with subnet range and ID space and message interval to determine server processing and bandwidth requirements
- Test if crawler catches info_hashes we see from target in XKS
- Must we proactively collect peers to address “SIGINT lag”?

# SIGINT Lag

- BitTorrent “swarm” may be inactive by the time target info_hash reported by SIGINT system
- May require preemptive collection of peers
  - DHT has on the order of 8 active million nodes
  - info_hash/DHT address space: 2^160

# Next Steps

- Enhanced analytics
  - Community discovery
- Distributed crawler
- Peer pre-fetch
- Target file download
  - avoid lending “utility”

# Prior Work

GCHQ - SEBACIUM

POC: [redacted]

CES – XKS schema/micro-plugin

Prototype analytics

POC: [redacted]

TAO-ROC – OGC approval for operational tests

PACKAGEGOODS connection

POC: [redacted]

# GRIMPLATE Collaboration

CES - Digital Network Exploitation Applications

[redacted]

NTOC

- V25 - Malicious Activity Discovery-Characterization
- V45/47 – Technology Development
- V46 – Technology Planning and Assessment

S2B – Office of China and Korea, CNE Access Development Branch

S2H – AP Russia Production Center, Russia SIGINT Development Division

TAO-ROC - Production Operations Division

“go grimplate”

# Questions